TAMPEREEN TEKNILLINEN YLIOPISTO
TAMPERE UNIVERSITY OF TECHNOLOGY

Sajjad Nouri

Power and Energy Aware Heterogeneous Computing Platform

Julkaisu 1594 • Publication 1594

Tampere 2018

Tampereen teknillinen yliopisto. Julkaisu 1594
Tampere University of Technology. Publication 1594

Sajjad Nouri

Power and Energy Aware Heterogeneous Computing Platform

Thesis for the degree of Doctor of Science in Technology to be presented with due permission for public examination and criticism in Tietotalo Building, Auditorium TC133 at Tampere University of Technology, on the 20th of November 2018, at 12 noon.

Tampereen teknillinen yliopisto - Tampere University of Technology
Tampere 2018

Doctoral candidate: Sajjad Nouri
Laboratory of Electronics and Communications Engineering
Faculty of Computing and Electrical Engineering
Tampere University of Technology
Finland

Supervisor: Jari Nurmi, Professor
Laboratory of Electronics and Communications Engineering
Faculty of Computing and Electrical Engineering
Tampere University of Technology
Finland

Pre-examiners: Dr.-Ing. Michael Hübner, Professor
Chair for Embedded Systems in Information Technique
Ruhr-University Bochum
Germany

Peeter Ellervee, Professor
Department of Computer Systems
Tallinn University of Technology
Estonia

Opponent: Peeter Ellervee, Professor
Department of Computer Systems
Tallinn University of Technology
Estonia

Pasi Liljeberg, Professor
Embedded Computing Architectures
Department of Future Technologies
University of Turku
Finland

ISBN 978-952-15-4253-4 (printed)
ISBN 978-952-15-4259-6 (PDF)
ISSN 1459-2045

Abstract

During the last decade, wireless technologies have experienced significant development, most notably in the form of mobile cellular radio evolution from GSM to UMTS/HSPA and thereon to Long-Term Evolution (LTE) for increasing the capacity and speed of wireless data networks. Considering the real-time constraints of the new wireless standards and their demands for parallel processing, reconfigurable architectures and in particular, multicore platforms are part of the most successful platforms due to providing high computational parallelism and throughput. In addition to that, by moving toward Internet-of-Things (IoT), the number of wireless sensors and IP-based high throughput network routers is growing at a rapid pace. Despite all the progression in IoT, due to power and energy consumption, a single chip platform for providing multiple communication standards and a large processing bandwidth is still missing. The strong demand for performing different sets of operations by the embedded systems and increasing the computational performance has led to the use of heterogeneous multicore architectures with the help of accelerators for computationally-intensive data-parallel tasks acting as coprocessors. Currently, highly heterogeneous systems are the most power-area efficient solution for performing complex signal processing systems. Additionally, the importance of IoT has increased significantly the need for heterogeneous and reconfigurable platforms.

On the other hand, subsequent to the breakdown of the Dennardian scaling and due to the enormous heat dissipation, the performance of a single chip was obstructed by the utilization wall since all cores cannot be clocked at their maximum operating frequency. Therefore, a thermal melt-down might be happened as a result of high instantaneous power dissipation. In this context, a large fraction of the chip, which is switched-off (Dark) or operated at a very low frequency (Dim) is called Dark Silicon. The Dark Silicon issue is a constraint for the performance of computers, especially when the up-coming IoT scenario will demand a very high performance level with high energy efficiency. Among the suggested solution to combat the problem of Dark-Silicon, the use of application-specific accelerators and in particular Coarse-Grained Reconfigurable Arrays (CGRAs) are the main motivation of this thesis work.

This thesis deals with design and implementation of Software Defined Radio (SDR) as well as High Efficiency Video Coding (HEVC) application-specific accelerators for computationally intensive kernels and data-parallel tasks. One of the most important data transmission schemes in SDR due to its ability of providing high data rates is Orthogonal Frequency Division Multiplexing (OFDM). This research work focuses on the evaluation of Heterogeneous Accelerator-Rich Platform (HARP) by implementing OFDM receiver blocks as designs for proof-of-concept. The HARP template allows the designer to instantiate a heterogeneous reconfigurable platform with a very large amount of custom-tailored computational resources while delivering a high performance

in terms of many high-level metrics. The availability of this platform lays an excellent foundation to investigate techniques and methods to replace the Dark or Dim part of chip with high-performance silicon dissipating very low power and energy. Furthermore, this research work is also addressing the power and energy issues of the embedded computing systems by tailoring the HARP for self-aware and energy-aware computing models. In this context, the instantaneous power dissipation and therefore the heat dissipation of HARP are mitigated on FPGA/ASIC by using Dynamic Voltage and Frequency Scaling (DVFS) to minimize the dark/dim part of the chip. Upgraded HARP for self-aware and energy-aware computing can be utilized as an energy-efficient general-purpose transceiver platform that is cognitive to many radio standards and can provide high throughput while consuming as little energy as possible. The evaluation of HARP has shown promising results, which makes it a suitable platform for avoiding Dark Silicon in embedded computing platforms and also for diverse needs of IoT communications.

In this thesis, the author designed the blocks of OFDM receiver by crafting template-based CGRA devices and then attached them to HARP's Network-on-Chip (NoC) nodes. The performance of application-specific accelerators generated from template-based CGRAs, the performance of the entire platform subsequent to integrating the CGRA nodes on HARP and the NoC traffic are recorded in terms of several high-level performance metrics. In evaluating HARP on FPGA prototype, it delivers a performance of 0.012 GOPS/mW. Because of the scalability and regularity in HARP, the author considered its value as architectural constant. In addition to showing the gain and the benefits of maximizing the number of reconfigurable processing resources on a platform in comparison to the scaled performance of several state-of-the-art platforms, HARP's architectural constant ensures application-independent figure of merit. HARP is further evaluated by implementing various sizes of Discrete Cosine transform (DCT) and Discrete Sine Transform (DST) dedicated for HEVC standard, which showed its ability to sustain Full HD 1080p format at 30 fps on FPGA. The author also integrated self-aware computing model in HARP to mitigate the power dissipation of an OFDM receiver. In the case of FPGA implementation, the total power dissipation of the platform showed 16.8% reduction due to employing the Feedback Control System (FCS) technique with Dynamic Frequency Scaling (DFS). Furthermore, by moving to ASIC technology and scaling both frequency and voltage simultaneously, significant dynamic power reduction (up to 82.98%) was achieved, which proved the DFS/DVFS techniques as one step forward to mitigate the Dark Silicon issue.

Preface

This thesis is based on the research work carried out during the years 2015-2018 in the Laboratory of Electronics and Communications Engineering at Tampere University of Technology (TUT), Tampere, Finland. I would like to gratefully acknowledge the financial support I received from the TUT Graduate School (during 2017-2018), Tuula and Yrjö Neuvo fund, HPY Research Foundation, Tekniikan Edistämissäätiö (TES) Foundation, DELTA doctoral training network, HiPEAC collaboration grant and Nokia Scholarship. I also wish to acknowledge the funding received from the Academy of Finland under contract # 258506 (DEFT: Design of a Highly-parallel Heterogeneous MP-SoC Architecture for Future Wireless Technologies).

First and foremost, I wish to express my deepest gratitude to my supervisor Prof. Jari Nurmi, who welcomed me into his research group during my M.Sc. degree, always believed in my abilities, provided finance throughput my research in TUT and abroad and made possible the accomplishment of this research work. I must also thank Dr. Waqar Hussain who has been a truly inspiring instructor and always gave me precious suggestions and feedbacks and helped me to find my own way in the research world. I would like to acknowledge Prof. Dr.-Ing. Diana Göhringer and Prof. Davide Rossi for being my supervisors at Ruhr University Bochum (RUB), Bochum, Germany and University of Bologna, Bologna, Italy, respectively. I thank Jens Rettkowski for being helpful colleague collaborating for joint research work at RUB. I would like to express all my deepest acknowledgments to Prof. Markku Renfors for his support and guidance, who granted me useful feedback whenever I asked. I am also thankful to Jyrki Hyrsylä, R&D Line Manager at Nokia Networks, who welcomed me into his team recently.

I would also like to express my thanks to the reviewers of this thesis: Prof. Dr.-Ing. Michael Hübner as well as Prof. Peeter Ellervee for their invaluable comments to improve the quality of this thesis.

I have had the good fortune to meet many special individuals over the past 5 years in Tampere, many of whom have inspired me to be the person I am today. I would like to express my unutterable gratitude specially to Mohsen Shahshahan, Orod Raeesi and Mohammad Ali Pourabed who have been side by side with me through thick and thin, particularly at times when it was even hard for me to stand myself. I also must thank Ramin Ghaznavi, Mitra Akbari, Mona Aghababaee, Farid Mehrabkhani, Nader Daneshfar and Shervin Safineh for their warm friendship and nice moments we have shared together and also for their long support. I must acknowledge my dear friend, Ritayan Biswas, who has helped me in reviewing this thesis. I am also grateful to my dear friends from Iran, Mohammad Tavoli, MohammadAli Tizhoush Taban, Mehdi Joafshani, Keyvan Abdi, Ata Meshkinzar, Leila Nikouei and Makan Aghakhani. There are also so many friends whose names I have not mentioned here but whom I also must

thank.

I wish to express my deepest gratitude and respect to my mother Tayyebeh Sadeghi Moghadam and my father Majid Nouri for their patience, enormous love and invaluable support in all moments throughout my entire life to bring me up to this stage. I owe all my achievements to them and without their support, none of this would have been possible. I also thank my faithful Lucy, for making a story of love and devotion between a dog and a man and being beside me to share the joys and happy times once I started to lose them.

Finally, I wish to thank the one dearest to me, Zohre Bijani. **Zohre**, The moment it hit me, I looked and looked at you, and I knew, as clearly as day and night, that I love you more than anything I had ever seen or imagined on earth. It is like the entire universe conspired to help me find you. My love, I have been fortunate to have you on my side to provide hope and motivation throughout this journey. You are the one person that made me risk everything for a future worth having. If my love for you was an ocean, there would be no more land. With you, I finally understood what true love means.

Tampere, October 2018
Sajjad Nouri

Contents

Acronyms

Well-Known Acronyms

ASIC	Application-Specific Integrated Circuit
ADC	Analog to Digital Conversion
ALM	Adaptive Logic Module
ALU	Arithmetic and Logic Unit
ALUT	Adaptive Look-Up Table
CC	Clock Cycle
CGRA	Coarse-Grained Reconfigurable Array
DA	Distributed Arithmetic
DFS	Dynamic Frequency Scaling
DFT	Discrete Fourier Transform
DMA	Direct Memory Access
DSP	Digital Signal Processing
DVFS	Dynamic Voltage Frequency Scaling
FCS	Feedback Control System
FFT	Fast Fourier Transform
FPGA	Field Programmable Gate Array
FU	Functional Unit
GOPS	Giga Operations Per Second
GPP	General Purpose Processor
GUI	Graphical User Interface
HEVC	High Efficiency Video Coding
HLS	High-Level Synthesis
I/O	Input/Output
ILP	Instruction-Level Parallelism
LUT	Lookup-Tables
MIMO	Multiple-Input, Multiple-Output
MOPS	Millions of Operations Per Second
MPEG	Moving Picture Experts Group
MPSoC	Multi-Processor System-on-Chip
NoC	Network-on-Chip
OFDM	Orthogonal Frequency-Division Multiplexing
RAW	Reconfigurable Architecture Workstation
RISC	Reduced Instruction-Set Computing
SDR	Software Defined Radio
SIMD	Single Instruction Multiple Data
VHDL	Very high-speed integrated circuit Hardware Description Language
VLIW	Very Long Instruction Word

Specific Acronyms For This Thesis

BB	Body Biasing
CE	Channel Estimation
CFO	Carrier Frequency Offset
COFFEE	COre For FrEE
CP	Cyclic Prefix
CREMA	Coarse-grained REconfigurable array with Mapping Adaptiveness
DCT	Discrete Cosine Transform
DRF	Data Register File
DST	Discrete Sine Transform
EXTRA	Exploiting eXascale Technology with Reconfigurable Architectures
FB	Frame Buffer
FBB	Forward Body Biasing
FOE	Frequency Offset Estimation
FUV	Future Video Coding
HARP	Heterogeneous Accelerator-Rich Platform
HMA	Heterogeneous Multicore Architecture
HPC	High Performance Computing
HRE	Heterogeneous Reconfigurable Engine
IDCT	Inverse Discrete Cosine Transform
IDST	Inverse Discrete Sine Transform
LLC	Last-Level Caches
LRF	Local Register File
MFC	Multi-Function logic Cells
MVM	Matrix-Vector Multiplication
PAE	Processing Array Element
PE	Processing Element
PID	Proportional Integral Derivative
PU	Processing Units
RBB	Reverse Body Biasing
RC	Reconfigurable Cell
RD	Rate-Distortion
SEEC	SElf-awarE Computing
SM	Square Modulus
TS	Time Synchronization
URF	Unregistered-Feed Through
UTBB FD-SOI	Ultra-Thin Body and Buried oxide Fully-Depleted Silicon-On-Insulator
VCD	Value Change Dump
VCEG	Video Coding Experts Group

List of Publications

This thesis is mainly based on the following publications in which the author of the thesis is the main author and has the major contribution. The co-authors contributed to reviewing and providing the feedback. In the manuscript the publications are referred to as [P.♯]. The publications are appended at the end of the thesis.

I S. Nouri, W. Hussain, and J. Nurmi, "Implementation of IEEE-802.11a/g Receiver Blocks on a Coarse- Grained Reconfigurable Array," *International Conference on Design & Architectures for Signal & Image Processing (DASIP)*, Cracow, Poland, pp. 23-25., Sep. 2015.

II S. Nouri, W. Hussain, and J. Nurmi, "Design and evaluation of correlation accelerator in IEEE-802.11a/g receiver using a template-based Coarse-Grained Reconfigurable Array," *Nordic Circuits and Systems Conference (NORCAS): NORCHIP & International Symposium on System-on-Chip (SoC)*, Oslo, Norway, pp. 1-6., Oct. 2015.

III S. Nouri, J. Rettkowski, D. Göhringer, and J. Nurmi, "HW/SW Co-design of an IEEE 802.11a/g Receiver on Xilinx Zynq SoC using High-Level Synthesis," *In Proceedings of the 8th International Symposium on Highly Efficient Accelerators and Reconfigurable Technologies (HEART2017)*, Bochum, Germany, no 15, June 2017.

IV S. Nouri, W. Hussain, and J. Nurmi, "Evaluation of a Heterogeneous Multicore Architecture by Design and Test of an OFDM Receiver," *in IEEE Transactions on Parallel and Distributed Systems*, vol. 28, no. 11, pp. 3171-3187, Nov. 2017.

V S. Nouri, D. Rossi, and J. Nurmi, "Power Mitigation of a Heterogeneous Multicore Architecture on FPGA/ASIC by DFS/DVFS Techniques," *Elsevier Journal of Microprocessors and Microsystems (MICPRO)*, Sep. 2018.

VI S. Nouri, R. Ghaznavi-Youvalari, and J. Nurmi, "Design and Implementation of Multi-Purpose DCT/DST-Specific Accelerator on Heterogeneous Multicore Architecture," *Nordic Circuits and Systems Conference (NORCAS): NORCHIP & International Symposium on System-on-Chip (SoC)*, Tallinn, Estonia, Oct. 2018.

List of Figures

List of Tables

1 Introduction

With the continuous development of wireless mobile communications and embedded systems, Internet-of-Things (IoT) as a third wave of information technology will be widely used in homes, industries and cities. Because of its huge market prospects, IoT is a hot topic today and has been paid close attention by several companies and standardization bodies all over the world. It is expected that by 2020 more than 50 billion devices will get connected with each other over a network without requiring human-to-human or human-to-computer interaction. On the other hand, with increasing demand for additional bandwidth, new wireless communication technologies namely Long-Term Evolution (LTE) and 5G (the next generation of mobile internet connectivity) should employ Dynamic Spectrum Access (DSA) [2] for efficiently utilizing the spectrum across frequency, space and time while resolving spectrum scarcity issue. In this context, the reconfigurable wireless platforms can be utilized in order to realize DSA. It can be achieved by using Software Defined Radio (SDR) technology and its intelligent version called Cognitive Radio (CR) [3]. In order to design and implement the radio systems on multi-mode multi-standard transceivers, several requirements have to be fulfilled, mainly: flexibility, scalable high computational power to meet the real-time constraints by wireless communication systems, and power efficiency to meet the allocated tight power budget in radio systems. According to the mentioned requirements, Multi-Processor Systems-on-Chip (MPSoCs) are suitable architectures for the implementation of SDR by providing a high degree of flexibility as well as high computational power. In a SDR platform, most of the kernels related to signal processing can be implemented by programmable processing technologies including General Purpose Processor (GPP), Digital Signal Processor (DSP), Field Programmable Gate Array (FPGA) and Coarse-Grained Reconfigurable Arrays (CGRAs). Different types of data transmission schemes are employed in wireless communication systems such as Orthogonal Frequency Division Multiplexing (OFDM) which is important in SDR because of its ability to provide high data rates. OFDM systems split an input high-speed data stream into several parallel streams which demand parallel processing.

The importance of meeting the real-time constraints of the new wireless standards and their requirements for parallel processing make the reconfigurable architectures as one of the most successful platforms. They can provide high computational parallelism and throughput while having low energy. Reconfigurability means modifying the functionality at run-time for several applications which offers the flexibility of software with the performance of hardware. Reconfigurable architectures can be classified into three different categories (fine-grained, middle-grained and coarse-grained) based on the level of granularity. Furthermore, considering the requirement of IoT in future, a single-chip platform for providing multiple communication standards and a large processing bandwidth is still missing. The wireless communication terminals have to

be multi-functional in order to support multiple applications. Although many Single Instruction Multiple Data (SIMD) architectures like SODA (2007) [4] and Montium (2008) [5] have been proposed for SDR applications in the past decade, an efficient cognitive radio engine based on MPSoC has not been developed yet. Currently, highly heterogeneous multicore architectures and reconfigurable platforms are one of the best candidates in terms of the power-area efficiency for performing the complex signal processing systems as well as meeting the requirements of IoT. During the recent years, a handful of research groups around the world have focused their attention on reconfigurable systems, in particular CGRAs, for wireless signal processing. An array of predefined Processing Elements (PEs) is used in CGRAs to provide high computational power as well as low energy consumption.

In 1965, Moore's Law predicted that the number of transistors per chip doubles every two years at constant cost [6]. Then Dennard in 1974 showed that the power dissipation density can be kept constant by scaling the voltage and the dimensions of the transistor which was threatening an end to Moore's Law [7]. According to the Dennard scaling, voltage and current should be proportional to the linear dimensions of a transistor. Therefore, circuits can be operated at higher frequencies at the same power as the size of the transistors shrunk and the voltage was attenuated. Even though Dennard Scaling tried to show how CMOS devices can be scaled for constant power density, it ignored the baseline of power per transistor which is established by leakage current and threshold voltage. Subsequent to putting an end to Dennard Scaling, a team at Massachusetts Institute of Technology (MIT) found a solution in multicore scaling by presenting RAW microprocessor architecture in 2002 [8]. However, in 2010, the experimental work explained in [6] showed that only 7% of a 300mm^2 chip can be operated at full frequency under a power budget of 80W which means an end also to multicore scaling. This inability to keep the power constant results to a technology-imposed *utilization wall* which affected multicore scaling. It states that in a single chip, all cores cannot be clocked at their maximum operating frequency due to a given Thermal Design Power (TDP) constraint which forced a large fraction of chip to be to be switched-off (dark) or to be operated at very low frequency (dim). The dark or dim part of the chip is called Dark- or Dim-Silicon, respectively. In 2012, Michael B. Taylor has introduced *Four Horsemen of Dark Silicon* as a four top contenders in order to deal with this issue which are characterized into four different approaches: shrink, dim, specialize and technology magic [9]. Among them, the *Specialized Horseman* suggested that the dark area of chip can be replaced by instantiating application-specific accelerators as the compute engines highly optimized for executing massively-parallel workloads of critical-priority applications. Therefore, the execution time can be decreased while the kernels operate at very low frequencies. In this regard, template-based CGRAs, employed to build up Heterogeneous Multicore Architectures (HMAs) using a Network-on-Chip (NoC), are advocated to be instantiated in the dark area of the chip. CGRAs can be used to generate special-purpose accelerators while being operated at a very low frequency. They are run-time reconfigurable and programmable with a higher level language ,i.e., C/C++ and can offer enormous performance improvements. There are a number of already existing CGRA architectures such as Pixie (2017) [10], EXTRA (2015-2016) ([11], [12]), BUTTER (2007) [13], ADRES (2003-2007) ([14], [15] [16]), Morphosys (1998-2000) ([17], [18], [19], [20]) and PACT-XPP (2002-2003) ([21], [22]). Recent outcomes suggest designing heterogeneous multicore platforms which support user-specified scalability and are composed of general-purpose and reconfigurable special-purpose cores with seamless integration capability while the communication backbone can be a NoC ([6],

[9]). On the other hand, CGRAs have potentially high transient power dissipation which requires to be mitigated by employing SElf-awarE Computing (SEEC) models and various techniques are required such as Dynamic Voltage and Frequency Scaling (DVFS) and Feedback Control System (FCS) [23]. Furthermore, in order to support IoT infrastructure, it is important that the power dissipation and energy consumption by processors reduces and the on-chip resource utilization increases. SEEC model has been introduced to dynamically control the voltage and operating frequency by using an adaptive feedback control system to mitigate the power dissipation. In this context, the instantaneous power/heat dissipation will be decreased to minimize the dark part of the chip. It should be mentioned that the frequency control system also dims that part of the chip.

1.1 Objective and Scope of Research

The primary objective of this thesis is the evaluation of Heterogeneous Accelerator-Rich Platform (HARP) by subjecting it to a stringent test to determine its potential architectural fallacies and pitfalls. HARP is already designed as a template-based architecture programmable in C language in a collaboration between the University of Chicago (UoC), IL, USA, and our research group in TUT for maximizing the number of computational resources for algorithm acceleration ([24], [25]). There are a number of already existing heterogeneous platforms with nearly similar design philosophy that are generally structured as many general-purpose and application-specific cores connected over a NoC, such as MORPHEUS (2007-2017) ([26], [27], [28]) and P2012 (2012-2014) ([29], [30]). However, the efficient utilization of on-chip resources has to be justified. It is observable that most of the on-chip resources are not utilized over the entire frame of execution-time and are over or under resourced for the applications that they are designed for [6]. The HARP template allows the designer to instantiate a heterogeneous reconfigurable platform with a very large amount of custom-tailored computational resources for high performance. The HARP template is composed of nine nodes where several template-based CGRAs ([31], [32]) of specific sizes tailored for specific applications are integrated with one or a few Reduced Instruction-Set Computing (RISC) cores over a NoC as a processor/coprocessor model. The template-based CGRAs can work in parallel to execute different tasks independently or in communication with each other in the case of data dependency.

The HARP template is first evaluated by implementing the OFDM receiver blocks as a proof-of-concept. OFDM is one of the most important data transmission schemes in SDR and also one of the candidates to be employed in 5G. An OFDM receiver is composed of some of the most computationally-intensive parallel tasks such as Fast Fourier Transform (FFT), Correlation and Complex Matrix-Vector Multiplication (MVM) which should be implemented by crafting template-based CGRAs. Furthermore, it includes some serial in nature tasks which require to be performed by GPP. Recently, a power efficient, FPGA-based, parallel-pipelined architecture for an OFDM baseband modulator has been proposed in order to support a set of OFDM numerologies for future 5G communication systems ([33], [34]). Therefore, OFDM receiver with such a fine mixture of serial and parallel algorithms is selected and implemented based on IEEE 802.11a/g standard specifications [35] in order to explore and evaluate almost all the design features of HARP. The implementation of OFDM receiver blocks has led to scale template-based CGRAs to different dimensions and establish almost all possible ways of PE interconnections based on the algebraic expressions of each particular application

while time-multiplexed patterns for data placement in the CGRA memories is explored. Due to the data dependency between the blocks of an OFDM receiver, the data must be exchanged among different crafted template-based CGRAs over the NoC by using Direct Memory Access (DMA) device for complete OFDM implementation. The instantiated few RISC cores in this design and test regime distribute the configuration streams and the data to be processed to all the integrated template-based CGRAs in addition to act as the controllers. The RISC cores are also available for performing the serial in nature algorithms and establishing synchronization among all the CGRA nodes and monitor their performance. Although the implementation of an OFDM receiver provided sufficient switching activity on all the NoC nodes and resulted in meaningful and tangible conclusions, a single case-study was not be enough to empirically assess an architecture. Therefore, the author further evaluated the HARP template by selecting various sizes of Discrete Cosine Transform (DCT) and Discrete Sine Transform (DST) dedicated for High Efficiency Video Coding (HEVC) standard. In both case-studies, the performance of each template-based CGRA, the collective performance of the entire platform and the NoC traffic are recorded in terms of the number of Clock Cycles (CC) and several high-level performance metrics such as Giga Operations Per Second per milliWatts (GOPS/mW). Furthermore, cross-technology comparisons among HARP and other state-of-the-art platforms are performed.

This thesis is also addressing the power and energy issues of the computer systems by tailoring the HARP for self-aware computing model. To support IoT infrastructure, it is important that the power dissipation and energy consumption by processors reduces and the on-chip resource utilization increases. For this purpose, the HARP template is estimated in 28nm Stratix-V FPGA device ([36], [37]) and also 28 nm Ultra-Thin Body and Buried oxide (UTBB) Fully-Depleted Silicon-On-Insulator (FD-SOI) ASIC technology in order to provide the possibility of scaling the voltage in addition to the frequency and get more benefits in terms of dynamic power dissipation reduction. The self-aware computing model applied on HARP exploits Feedback Control System (FCS) which constantly monitors the execution-time of each core. Then based on the worst execution-time, FCS dynamically scales the operating frequency and the voltage of each CGRA node of the NoC. As a result, the performance of the overall system is equalized towards a desired level besides mitigating the dynamic power dissipation. The upgraded HARP can act as an energy-efficient general-purpose transceiver platform that is cognitive to many radio standards and can provide high throughput while consuming as low energy as possible. Moreover, the Dark Silicon as a resource will be harvested and transformed into efficiently utilizable processing units. In other words, the proposed approach is utilizing Dark Silicon for performing massively-parallel workloads of critical-priority applications by employing the specialized accelerators in large-scale heterogeneous multicore architecture. HARP lays an excellent foundation to investigate techniques to replace the Dark Silicon with high-performance silicon dissipating very low power and energy.

The author also presents a HW/SW codesign of an OFDM receiver by using High-Level Synthesis (HLS) tools from Xilinx. The tool provided an embedded C/C++ application programming interface in order to develop heterogeneous embedded systems. The fine mixture of serial and parallel algorithms makes an OFDM receiver as a suitable candidate to be implemented into hardware by compiling the functions written in C/C++ code for every block separately. A general comparison among the HARP template and ZC706 evaluation board with the implemented OFDM receiver is presented.

The main results achieved in this thesis can be highlighted as follows:

- Evaluation of almost all the architectural features of HARP by design and implementation of IEEE 802.11a/g receiver blocks as well as 4/8/16/32-point DCT and 4-point DST dedicated for HEVC;
- Proof-of-concept scalable template-based HARP suitable for diverse needs of IoT as a general-purpose power- and energy-aware transceiver platform;
- Methodology for improving Dark Silicon in embedded computing platforms by applying self-aware computing model on HARP and mitigating the signal transition activity over the entire platform with reference to the worst-case performing core for power conservation;
- Demonstration of modularity and regularity in the HARP as well as exploration of an architectural constant for HARP based on GOPS/mW which is not changeable by scaling the computational resources of a platform;
- Allocation of computational resources at design-time in order to utilize most of the on-chip resources over the entire frame of execution time;
- Demonstration of the benefits of scaling the integrated template-based CGRAs to very large dimensions and maximizing the number of reconfigurable processing resources on a platform by comparing the HARP template against other state-of-the-art platforms;
- Providing important insight and guidelines for the designers in order to implement near-optimal mapping of target applications in terms of performance, resource utilization, power dissipation and execution time.

1.2 Author's Contribution to the Published Work

The work presented in this thesis has mainly been extracted from the following publications in which the author was the first author and the research area was proposed and supervised by Prof. Jari Nurmi. He has contributed to all of the publications and provided extensive feedback for the manuscript drafts.

In ***publication [P. I]***, the author designed and implemented the IEEE 802.11a/g receiver blocks by using the template-based CGRAs as application-specific accelerators to COFFEE RISC processor. The author designed all the accelerators by crafting the template-based CGRA, based on the algebraic expressions of the required algorithms. The accelerators designed are synthesized on the Stratix-V FPGA device and evaluated for performance in terms of the number of clock cycles, speed-up in comparison with the RISC software implementation, resource utilization, maximum operating frequency, power dissipation and energy consumption. Dr. Waqar Hussain and Prof. Jari Nurmi provided technical comments and supervised the entire work.

In ***publication [P. II]***, the author put his effort in optimizing the implementation of correlation accelerator used for time synchronization block required in IEEE 802.11a/g receiver by scaling-up the template-based CGRA. The accelerator is synthesized on FPGA device and the power dissipation is estimated by simulating the postfit gate-level FPGA netlist. Furthermore, the author performed cross-technology comparisons of the

platform with the other state-of-the-art platforms in terms of GOPS and GOPS/mW. Dr. Waqar Hussain and Prof. Jari Nurmi gave the final comments on the work.

In ***publication [P. III]***, the author presented an implementation of IEEE 802.11a/g receiver blocks by employing HLS tool from Xilinx called SDSoC which provides an embedded C/C++ application programming interface for developing heterogeneous embedded systems. The experimental platform in this case-study was Zynq SoC containing an ARM processor besides a FPGA. Every block of the receiver was written by the first author in C/C++ code to be realizable on the FPGA and the ARM processor and then evaluated on ZC706 board and compared against the HARP. This work was completely performed by the first author during his research visit to Ruhr University Bochum (RUB), while the second author assisted in working with HLS tool and ZC706 board. The overall work was supervised by Prof. Diana Göhringer and Prof. Jari Nurmi.

In ***publication [P. IV]***, which is an extension to the author's first and second conference papers published in DASIP 2015 and NORCAS 2015, the author evaluated the HARP by implementing IEEE 802.11a/g receiver blocks as designs for the test of functionality. In this paper, the author mapped and integrated all designed receiver blocks using the template-based CGRAs on HARP and then subjected it to a stringent test for identifying potential architectural fallacies and pitfalls. The author also assigned an architectural constant based on achieved GOPS/mW to HARP due to its scalability and regularity which ensures application-independent figure of merit. Dr. Waqar Hussain and Prof. Jari Nurmi gave technical comments and invaluable feedback and supervised the entire work.

In ***publication [P. V]***, which is an extension to the first author's conference paper (published in NORCAS 2017) with an invitation, the author presented an integrated self-aware computing model in the HARP to mitigate the power dissipation of an IEEE 802.11a/g receiver. In the proposed self-aware computing model, FCS is exploited for monitoring the execution-time of the NoC nodes constantly and scaling their operating frequency and the voltage dynamically based on the worst execution-time. The implementation is estimated on the FPGA-based prototyped platform and also in 28nm Ultra-Thin Body and Buried oxide (UTBB) Fully-Depleted Silicon-On-Insulator (FD-SOI) ASIC technology. This work was mainly done by the first author, while the second author assisted in synthesizing the whole platform on ASIC and scaling the voltage and frequency. Prof. Davide Rossi and Prof. Jari Nurmi gave invaluable feedback on overall work and helped the first author to come up with the final results.

In ***publication [P. VI]***, the author designed and implemented various sizes of DCT/DST-specific accelerators on HARP for HEVC standard by using template-based CGRAs. The final architecture consists of 4/8/16/32-point DCT and 4-point DST. The accelerators are arranged over a NoC structure along with three RISC cores. The author reported the performance of each DCT/DST-specific accelerator, the collective performance of the whole platform and the NoC traffic in terms of the number of clock cycles and several high-level performance metrics. The whole platform is synthesized on FPGA device and then, total power dissipation and energy consumption based on post placement and routing information are reported. Moreover, the architectural constant of HARP in terms of GOPS/mW is again proven by the author. The entire work was completely performed by the first author, while the writing of introduction and the equations related to DCT/DST was done by the help of the second author. This work was fully supervised by Prof. Jari Nurmi.

1.3 Thesis Outline

This manuscript is organized as follows: Chapter 2 presents some of the existing state-of-the-art platforms related to the work presented in the next chapters. Chapter 3 explains the architecture of HARP and the template-based CGRAs used for integration, the overall functionality of HARP and the internal structure of the NoC nodes [P.IV]. Chapter 4 presents the design and implementation of OFDM receiver blocks by using template-based CGRAs ([P.I], [P.II]). Evaluation of HARP by instantiating and implementing an OFDM receiver base-band processing as well as distributing the data among different nodes are discussed in Chapter 5 [P.IV]. Chapter 6 focuses on power mitigation and performance equalization of HARP on FPGA and ASIC devices by employing DFS and DVFS techniques, respectively [P.V]. Chapter 7 presents a novel HW/SW co-design of OFDM receiver on ZC706 board using the High-Level Synthesis (HLS) tool from Xilinx called Software Defined System-on-Chip (SDSoC) [P.III]. Chapter 8 concentrates on the design and implementation of multi-purpose DCT/DST-specific accelerator by using template-based CGRAs on HARP [P.VI]. Finally, in Chapter 9, concluding remarks, open issues and the future work is presented.

2 Literature Review

Application-specific accelerators in the form of processor/coprocessor model have been developed and embedded in MPSoC for accomplishing computationally intensive tasks. The most powerful class of accelerators are CGRAs, which act as coprocessor to a programmable processor to form a heterogeneous multicore platform where they can be employed simultaneously and independently of each other or even in communication with each other at run-time. Prior to creating multicore platforms in processor/coprocessor models, single core as a GPP has been in use for performing many applications. Meanwhile, some accelerators have been also assigned for some computationally intensive tasks. Later on, Very Long Instruction Word (VLIW) machines have been developed in order to support large-scale parallel applications [38]. As the time passed, VLIW architectures were combined with DSP processors for supporting also DSP applications targeting high-end mobile communication [39]. In multiocore platforms, the accelerators can be operated as coprocessors to processors either in loose or tight coupling. In loose coupling, multiple accelerators are connected to a processor with relatively low bandwidth and can work simultaneously [29]. In tight coupling, the bandwidth is higher than loose coupling for communication with the processor, which enables faster data transfer and synchronization ([40], [41]). The accelerators can be tightly-coupled to the processor by either employing coprocessor bus or integrating the accelerator in the datapath.

2.1 Reconfigurable Devices

In the recent past, reconfigurable devices become popular architectures due to their flexibility as well as because of time and cost reduction in system development. Reconfigurable devices are able to switch their functions at run-time occasionally based on the data flow specified at design-time by application developer for various purposes. Therefore, reconfigurable platforms are able to implement many different tasks. In general there are three different reconfigurable devices, which are classified according to their level of granularity: fine-grained (with granularity of less than 4-bit), middle-grained (with granularity of less than or equal to 8-bit) and coarse-grained (with granularity of more than 8-bit) [26]. Fine-grained devices have the most optimal resource utilization due to their fine level of granularity while the placement and routing is more demanding. Middle-grained devices are a compromise among fine-grained and coarse-grained devices with higher bit-width processing and simpler compilers for design and implementation in comparison with the fine-grained devices. Coarse-grained devices have the simplest compilers and the highest level of granularity while the processing of wide range of applications on the word level can get supported. In the following sections, several existing examples of fine-, middle-, and coarse-grained devices in the literature

are described briefly.

2.1.1 Fine-Grained Devices

Fine-grained devices have much more PEs than coarse-grained devices while the granularity level of the PEs are 1 to 4 bits. Nowadays most of the applications are working with 8, 16 or 32-bit operations. In comparison with the coarse-grained devices, fine-grained devices should employ more number of PEs in order to execute the same operation at the cost of more resource utilization and inefficient mapping. One of the most promising fine-grained devices are FPGA and in particular embedded FPGA (eFPGA). Since FPGAs are not suitable for run-time reconfiguration, a small FPGA block is embedded in a fixed architecture (called eFPGA) to enable run-time reconfigurablity. An FPGA is composed of Logic Elements (LEs) including a Look-Up Table (LUT), a few logic gates, 2-to-1 multiplexers and also a few Flip-Flops (FFs). In the market, Xilinx [42] and Altera [43] are the most well-known fine-grained devices. For instance, in a case-study done by [44], three eFPGAs are integrated with the NoC-based system. One of the fine-grained devices is GARP architecture, which acts as a reconfigurable coprocessor, tightly coupled with GPP and offering a low granularity by mainly 2-bit and 4-input LUTs [45]. GARP consists of PE arrays while in each row one control block and 23 logic blocks are instantiated. The size of PE arrays can be modified at design-time. In order to keep the operating frequency fixed, the connectivity on its fabric is limited [46]. Another fine-grained device is FlexEos as an eFPGA [47]. It consists of 4K Multi-Function logic Cells (MFC) according to SRAM 1-bit Lookup-Tables (LUT). FlexEOS as a reprogrammable SRAM-based scalable FPGA fabric built on high-density multi-function logic cells can be programmed by using standard description languages, i.e., VHDL and Verilog. Another instance of fine-grained reconfigurable unit is MOLEN, which acts as a tightly-coupled coprocessor to a GPP ([48], [49]). There is possibility for MOLEN to be mapped on Xilinx FPGA chip while the FPGA fabric itself acts as an accelerator. Although MOLEN is physically separated from the GPP, added special instructions can be operated on that due to the extension of Instruction-Set Architecture (ISA) of the GPP.

2.1.2 Middle-Grained Devices

Middle-grain reconfigurable devices have been introduced to support the maximum word length of eight. Therefore, the algorithm with the same processing word length can be mapped on them. It has to be mentioned that, compared to fine-grained devices, mapping the algorithms on middle-grained reconfigurable units is more difficult due to increasing the processing word length. The middle-grain paradigm could provide a good compromise among area, performance and power while different word length applications can be supported. One of the middle-grained reconfigurable devices in the literature is PiCoGA-III consisting of Reconfigurable Datapath Unit (RDU) ([50], [51]). Each RDU is composed of a 4-bit LUT, a 4-bit ALU and a 4-bit integer and Galois field multiplier. Another middle-grained reconfigurable processing engine is DART, which support 8- and 16-bit processing word length ([52], [53]).

2.1.3 Coarse-Grained Devices

CGRAs are one of the most promising platforms with the possibility to support a 8-bit, 16-bit and 32-bit arithmetic or logic operation onto a single PE. In CGRAs, an array

of predefined PEs provides a high-level of granularity, high data level parallelization, high computational power, a large bandwidth, high throughput processing and low energy consumption. They are reconfigurable and programmable with a higher level language and can yield tremendous performance improvements while being operated at a very low frequency. Due to the internal structure and the level of granularity of CGRAs, they are suitable for performing massively-parallel computationally-intensive signal processing algorithms. CGRAs are successfully evaluated for a proof-of-concept for a set of data parallel applications such as FFT [32], Correlation [54], Wideband Code Division Multiple Access (WCDMA) cell search [55], image and video processing ([56], [57]), Finite Impulse Response (FIR) filtering [58]. Against all advantages of CGRAs, they have potentially high transient power dissipation and occupy a large area of a few million gates. Furthermore, most of the CGRAs have a fixed set of PEs and interconnections, which are not optimum in terms of cost and performance. Some of the CGRA architectures found in literature are briefly described as follows.

2.1.3.1 BUTTER

BUTTER acts as a coprocessor for COFFEE RISC core. It has been developed by our research group in order to accomplish computationally-intensive tasks [13]. It is composed of a 4×8 array of PEs. However, the size of PE array can be scaled-up/down at design-time. The PEs are connected to each other in a point-to-point fashion for exchanging data. Each PE has a functional unit for arithmetic and logic operations with the fixed granularity level at 32-bit and also with the ability to support integer and single precision floating-point. The configuration data and the data to be processed can be transferred from the main memory to the CGRA with the help of a DMA device. BUTTER is characterized by run-time reconfigurability, which means the possibility of selection of connection at run-time. The entire platform is synthesized on FPGA device.

2.1.3.2 ADRES

Architecture for Dynamically Reconfigurable Embedded Systems (ADRES) as a coarse-grained reconfigurable architecture is tightly coupled with Very Long Instruction Word (VLIW) processor ([14], [15] [16]). Compared to the other CGRAs, ADRES showed many advantages such as improvement in terms of the performance, a simplified programming model, reduction of communication costs and substantial resource sharing. The VLIW processor and CGRA are integrated into a one single architecture with two virtual functional views (VLIW view and reconfigurable array view). The architecture of ADRES is composed of reconfigurable array of 8×8 elements. The elements are connected in a certain topology, which comprise mainly Functional Units (FU), Register Files (RF) and routing resources. The first row of reconfigurable arrays (VLIW view) is FUs while the rest of the rows (called reconfigurable matrix) are Reconfigurable Cells (RC), which comprise FUs and RFs and belong to the second view. The FUs, which can also be heterogeneous supporting various operation sets are connected together through one multi-port global Data Register File (DRF) with the data bus width of 32 bits. The RCs communicate through a multi-port global DRF, Local Register Files (LRF) and dedicated interconnects between the FUs. In order to store the intermediate data, the RFs can be employed in such a way that the words with the length of 16 and 64 bits should be stored in local and global RF, respectively. The routing resources are composed of wires, buses and networks. The RCs are used to accelerate the dataflow-like kernels in a highly parallel way. On the other hand, for performing the non-kernel

codes, VLIW can be used by exploiting Instruction-Level Parallelism (ILP). Since ADRES is basically a processor/co-processor model, VLIW and reconfigurable array views can share some resources and therefore, their executions will never overlap with each other. In order to generate ADRES specific intances, an XML-based architecture specification language can be utilized. ADRES is synthesized on 90 nm CMOS technology and delivers a performance of 40 MOPS/mW.

2.1.3.3 Morphosys

The Morphosys architecture as a single chip is designed to operate on 8 or 16-bit data ([17], [18], [19], [20]). It is composed of an 8×8 array of reconfigurable processing units called Reconfigurable Cell (RC) with configuration memory, tightly coupled 32-bit general-purpose processor (TinyRISC) and a high-bandwidth memory interface. The operation of the RC array is monitored by a RISC core. The RC array is partitioned into four quadrants. The data transfer can be initiated by RISC core between an external memory and RC array by using two sets of Frame Buffers (FBs), each has two memory banks. Each RC incorporates an ALU for fixed-point operations, a multiplier, a shift unit, input multiplexers and a register file. The RC array can be configured by using 32-bit context word, which can be distributed to all eight RCs in the same column or row. Special instructions were added to TinyRISC's ISA to transfer RC array related operations such as control operations, configuration and data transfer between the array and the main memory.

2.1.3.4 PACT-XPP

PACT-XPP as a self reconfigurable processing engine is based on a hierarchical array of coarse-grained architectures ([21], [22]). It is composed of 3×3 adaptive computing elements called Processing Array Elements (PAEs) and packet-oriented communication network. The strength of PACT-XPP is powerful run-time partial reconfiguration capability, which makes the PAEs independent of each other. It means that some PAEs can be reconfigured for a new functionality while the other PAEs on different parts of the array can keep computing data simultaneously. Reconfiguration is externally-triggered event-based by special event signals originating within the array. The algorithms written in C subset program can be mapped on PACT-XPP by using vectorizing C compiler XPP-VC. PACT-XPP delivers a peak performance of 57.6 GOPS at 150.0 MHz.

2.1.3.5 Pixie

Pixie as a heterogeneous Virtual CGRA (VCGRA) has been introduced for high performance image processing applications [10]. The VCGRA is composed of generic PEs and virtual channels, which are described by VHDL and optimized by using the parameterized configuration tool flow. The optimization has been resulted in resource reduction of 24% and 82% for each PE and virtual channel, respectively. The PEs are connected to each other by employing virtual connection blocks and switch blocks. The PEs of VCGRA are able to execute various arithmetic operations such as addition, subtraction, multiplication and division. In comparison with the Fine-Grained FPGAs, the compilation time has been reduced since the designers can write the code at a higher abstraction level.

2.1.3.6 EXTRA

EXTRA (Exploiting eXascale Technology with Reconfigurable Architectures) as a flexible exploration platform has been developed for developing and programming reconfigurable architectures with built-in run-time reconfiguration ([11], [12]). The platform can be employed in order to handle High Performance Computing (HPC) systems of the future by enabling the joint optimization of architecture, tools, application and reconfiguration technology. In total, three key objectives have been defined for the success of EXTRA as following; evaluation and optimization of reconfigurable architectures by combining a reconfigurable architecture description with reconfigurable design tools in an open reconfigurable technology exploration platform, development of reconfigurable architectures, reconfigurable tools and the optimization of reconfigurable HPC applications, improvement of performance, area and power efficiency by using the EXTRA ecosystem in order to perform three HPC applications. The reconfigurable HPC systems can be developed by designing inherently reconfigurable architecture, enabling efficient reconfiguration by developing new tools and identifying the most appropriate applications for exploiting this novel concept of reconfigurability.

2.2 Multicore Platforms

Multicore platforms can be either homogeneous with several RISC processors loosely coupled with each other or heterogeneous with the RISC processors and reconfigurable architectures loosely or tightly coupled with each other. In homogeneous platforms, the algorithm code, generally written in C, can be distributed equally to all the RISC cores. In heterogeneous platforms, additional effort is required to program processors and coprocessors at the data flow level by using some customized tools. In addition to the employed heterogeneous multicore platform in this manuscript, HARP, some of the state-of-the-art platforms with almost similar features are described as follows.

2.2.1 NineSilica

NineSilica has been developed by our research group as a general-purpose homogeneous MPSoC, programmable in C language [59]. It is composed of nine homogeneous cores connected over a NoC in 3×3 mesh topology where each node contains a 32-bit COFFEE RISC processor. The central node acts as a supervisor node to monitor the rest slave nodes. Each node has its own instruction and data memory. The data can be transferred among all the nodes over the NoC by using packet switching technique. In order to test the functionality of NineSilica, many SDR applications have been designed and implemented, e.g., FFT and correlations. According to the conducted experimental results, 64-point FFT can be performed by NineSilica in 10.3 μs while synthesized on a 40 nm FPGA device.

2.2.2 MORPHEUS

MORPHEUS as one of the most promising heterogeneous multiple accelerator platform is published recently ([26], [27]). It is a complex and dynamically reconfigurable SoC consisting of three different types of reconfigurable devices: fine-grained embedded FPGA, middle-grained configurable processor and coarse-grained array for exploiting dynamic frequency scaling to mitigate the power consumption. It is designed as a heterogeneous digital signal processor for dynamically reconfigurable computing based

on a 64-bit NoC [28]. The master node is ARM 926EJ-S RISC processor, which controls communication, synchronization and reconfiguration mechanisms. The overall system has a regular infrastructure including communication and memories in order to enable regularity in control between the heterogeneous accelerators. For providing an efficient use, the platform has also a specific software, which contains an operating system and design tools. The fine-grained device is FlexEOS, which is briefly explained above. The middle-grained device is DREAM, which is reconfigurable DSP core. It contains a 32-bit RISC core and PiCoGA-III reconfigurable datapath as a matrix of reconfigurable logic cells. It delivers a performance of 0.2 GOPS/mW using a 90 nm CMOS technology. The coarse-grained device is XPP-III, integrated into the datapath of a VLIW processor. It is proposed in order to provide highly parallel processing performance for streaming applications. All three reconfigurable devices beside the system modules communicate with each other over a NoC. The system modules are Heterogeneous Reconfigurable Engines (HREs), memory units and I/O peripherals. The overall MORPHEUS chip delivers a performance of 0.02 GOPS/mW while synthesized on a 90 nm CMOS technology with an average dynamic power of 700 mW. MORPHEUS delivers 120 GOPS using 90-nm technology with the size of 110 mm^2 in order to accomplish a video surveillance motion detection application with the power dissipation of 1.45 W and peak power consumption of 2.5 W.

2.2.3 P2012

P2012 is an area- and power-efficient many-core computing platform consisting of four clusters communicating with each other using a high-performance fully-asynchronous NoC [29]. The clusters are locally synchronous and globally asynchronous while each cluster is composed of 16 general-purpose processors with independent instruction streams. The interaction among hardware and software is built by using the local and global interconnection as a point-to-point stream communication. In P2012, a specialized hardware is allocated in order to perform synchronization and advanced power management. The heterogeneous extended version of P2012 is called He-P2012, which is presented in [30] and can be considered as a heterogeneous MPSoC platform. The platform shows a performance of 40 MOPS/mW using a 28 nm CMOS technology.

2.2.4 RAW

Reconfigurable Architecture Workstation (RAW) as multicore platform is composed of 16 32-bit modified MIPS2000 processors organized as an array of order 4×4 mesh over a NoC [8]. RAW allows both static scheduling (similar performance to reconfigurable arrays) and dynamic scheduling (a mechanism typical of multi-core systems) for the network transactions. In RAW microprocessor, the effect of wire-delay problem has been managed by programmable NoC and exposing the wiring channel operators to the software.

2.2.5 Fulmine

Fulmine has been developed as a heavily specialized multicore platform for IoT applications, in particular the emerging class of smart secure near-sensor data analytics for IoT end-nodes without voltage scaling [60]. It is a 65 nm SoC based on a tightly coupled multicore cluster supported with specialized blocks for computationally intensive tasks. In Fulmine, the engines are four enhanced 32-bit OpenRISC cores, which can exchange

data with the accelerators in a flexible and efficient way because of employing memory sharing mechanism. Fulmine delivers a performance of up to 25 MIPS/mW in software with the power consumption of 20 mW on average at 0.8V. It has also achieved low energy, potentially high speed and low-effort data exchange.

2.3 Power Mitigation of Multicore platforms by DVFS Techniques

As it is mentioned earlier, CGRAs have potentially high transient power dissipation, which requires additional efforts to be mitigated by employing DVFS techniques and self-aware computing model. Furthermore, in order to support IoT infrastructure, it is important that the power dissipation and energy consumption by processors reduces and the on-chip resource utilization increases. Despite all the progression in IoT, a single chip platform for providing multiple communication standards and a large processing bandwidth is still missing due to power and energy consumption. There are several case-studies in the literature, which are discussed in the following and show the effects of DVFS along with FCS methods for mitigating the power and maximizing the performance.

In a case study has been done by W. Hussain et al., FCS technique defined in the processor software was applied on a heterogeneous multicore architecture [61]. Based on the conducted experiment results, the total dynamic power dissipation has been mitigated by 20.7% subsequent to equalizing the performance. In another experimental work, the authors investigated the thermal problems of 3D multicore processors by using an adaptive DFS technique [62]. The mentioned problem can happen due to high power density, which is the result of the stacking of multiple layers vertically. According to the achieved results, they could decrease the peak temperature by up to 10.35 °C. In another research work, the authors applied a DVFS method to reduce the energy level of the shared resources in a multicore processor, i.e., NoC and Last-Level Caches (LLCs) [63]. Based on the obtained results, the energy level has been reduced by 56%. In a research work done by S.M.A.H. Jafri et al., energy aware task parallelism method has been introduced [64]. This method was targeted for CGRAs and relies on resource allocation graphs, autonomous parallelism and the algorithms for voltage and frequency selection. In comparison with the other state-of-the-art DVFS algorithms, significant reduction in energy, power and configuration memory demands has been achieved, up to 36%, 28% and 36%, respectively. In a case study, Proportional Integral Derivative (PID) controller technique has been presented as a DVFS technique for maintaining the operating temperature of the multicore platforms within the thermal design power [65]. Thus, the total power dissipation can be mitigated according to the certain power limit. The authors could enhance the system throughput up to 43% and show the effectiveness of employed method. In another case study, 8.4X energy saving has been achieved in a self-aware processor [66]. The presented processor was able to self-adapt itself by constantly monitoring the system energy consumption. In [67], our research group applied DVFS technique to NineSilica platform as a homogeneous multi-processor architecture. According to the conducted experiment results, significant energy saving has been observed by the authors while the overall system performance has been improved by a factor of 3 in comparison with clock gating technique. In one of the latest published work by D. Rossi et al. [68], a self-aware architecture exploiting Body Biasing (BB) controlled by software has been presented. BB is an advanced technique in order to shift the effective transistor threshold voltage as well as decrease the leakage power consumption by applying the voltage to the body contact of CMOS transistors. The

proposed architecture has been implemented in 28nm Ultra-Thin Body and Buried oxide (UTBB) Fully-Depleted Silicon-On-Insulator (FD-SOI) ASIC technology. The reason behind choosing such a technology was implementing the low-power modes in near-threshold processors as well as compensating the parameters, operational voltage and temperature. In this research work, the authors employed the wide range Forward BB (FBB) and Reverse BB (RBB) in order to decrease the design time margins and introduce a low-power mode and state-retentive sleep mode. According to the achieved experiment results by D. Rossi et al., design margins reduced enormously while the energy efficiency of the processor improved by 32% with a hardware cost of less than 1% and a runtime cost for software control of less than 0.01% [68]. Furthermore, 24x area reduction for the compensation loop and 21.2x better efficiency were achieved in [68] compared to their previous design. In another case study ([33], [34]), DFS technique has been employed by a parallel-pipelined architecture for an OFDM baseband modulator supporting 5G numerologies in order to adapt the clock frequency for baseband processing at run-time based on the communication throughput demands. The achieved results showed the power savings up to 62.5%, which results in improving the system's power efficiency.

2.4 Hardware Implementation of DCT/DST for HEVC

Joint Collaborative Team on Video Coding (JCT-VC) presented HEVC as the latest standard on video compression. The work has been done in joint effort among ITU-T Video Coding Experts Group (VCEG) and the ISO/IEC Moving Picture Experts Group (MPEG). The provided output by HEVC standard was 50% reduction in terms of the bitrate on a specific video quality against the Advanced Video Coding (H.264/AVC) standard [69]. Similar to the H.264/AVC standard, the HEVC has a hybrid block-based coding scheme including intra-picture and inter-picture prediction tools. From the intra-picture or inter-picture prediction process, there are resulted prediction error residuals, which require the decorrelation operation. For each N×N residual block, 2D transform coding operation should be applied in a way that N-point 1D transforms to each row and column of the residual block have to be performed separately. Various transform sizes are supported in the HEVC standard for transform coding of the prediction error residuals mainly 4×4, 8×8, 16×16 and 32×32 Discrete Cosine Transforms (DCT) as well as 4×4 Discrete Sine Transform (DST) compared to 4×4 and 8×8 DCT transforms of H.264/AVC [70]. Due to having more provided transform sizes, 5% to 7% additional bitrate reduction was achieved in comparison with the conventional transforms of the H.264/AVC. However, despite the improvements in the Rate-Distortion (RD) performance of the HEVC using such transforms, the complexity overhead has increased enormously consequently. In the last section of this manuscript, the author presents the design and implementation of 4/8/16/32-point DCT and 4-point DST dedicated for HEVC on HARP template. There are also several similar case-studies in the literature, which are briefly discussed in the following.

In a research work, 4/8/16/32-point 2D DCT and 4-point 2D DST for HEVC were implemented by the use of two 1D transforms with row-column and even-odd decomposition techniques [71]. The design was implemented on Arria II FPGA device, which showed the ability of the design to support encoding 1080p format at 60 fps and 2160P at 30 fps at the cost of 10.0 and 13.9 kALUTs and 216 and 344 DSP blocks, respectively. In another case-study, a real-time Kvazaar HEVC intra encoder including DCT and IDCT for 4K Ultra HD video streaming on Nokia AirFrame Cloud Server were presented [72]. The encoder has been synthesized on Arria 10 FPGA device, which

depicted the ability of the system to encode 4K video at 30 fps and 40 fps with one or two intra coding accelerators, respectively. The achieved results also showed up to 1.6 and 2.1 respective speed-up compared to the pure Xeon implementation. In an experimental work done by J. Nikara et al., a unified 2D 8-point DCT and IDCT has been implemented and synthesized on 0.11 μm CMOS technology [73]. The achieved maximum operating frequency for running the entire platform was equal to 253.0 MHz at the cost of 40,000 equivalent gates. The obtained results from resource utilization showed 10-15% smaller estimated area against another unified structure. The authors in a case study implemented the efficient 4/8/16/32-point 2D IDCT and 4-point IDST for HEVC by using HLS tool [74]. The final architecture was synthesized on Arria II FPGA device, which resulted to real-time 60 fps HEVC decoding up to 2160P format. Based on the achieved results, 12.4 kALUTs and 344 DSP blocks were consumed by the architecture. Moreover, 5X speed-up was also obtained in comparison with the other HLS approaches. In an experimental work, the FPGA implementation of Future Video Coding (FVC) 4/8-point 2D DCT by employing fixed DSP blocks for performing multiplication has been presented [75]. According to the conducted experiment results, energy consumption was reduced by 29% in order to process 54 8K Ultra HD (7680×4320) video frames per second in the worst case compared to other existing FVC 2D transform hardware. In another research work, 8-point DCT and IDCT have been implemented on FPGA [76]. The implementation required 512 CC for processing 8-bit words while 75,488 kilobytes of total memory were consumed. In a case study, the authors presented the efficient architecture of 4/8/16/32-point 2D DCT for HEVC in the FPGA device [77]. The achieved results showed hardware cost reduction of 31-64% and performance improvement. In an experimental work done by P. Kitsos et al., two high performance FPGA architectures have been used for implementing the 2D 8-point DCT for Ultra High Definition video coding system [78]. The implementation of the first FPGA architecture with 2 ROMs required 105 CC for 8×8 block at a 338.5 MHz. The second implemented architecture with 4 ROMs required 65 CC for the same task at 256 MHz. In a case-study done by R. Jeske et al., a 16-point 1D DCT used for performing 16-point 2D DCT dedicated for the HEVS standard has been designed by replacing the multiplications with shifts and adds [79]. It resulted to low cost and high throughput hardware design. The architecture was able to sustain 30 QFHD frames (3840×2160 pixels) per second in real time. According to the obtained results from synthesizing the design on Cyclone II and Stratix III, 73.7% and 72% reduction of hardware resources have been observed against other existing architecture with non-optimized algorithm at the clock frequency of 23.51 MHz and 87.6 MHz, respectively. Furthermore, the designed architecture proved its ability by processing 3840×2160 pixels in about 30 fps in real time. In another research work done by E. D. Kusuma et al., the hardware implementation of 2D DCT has been presented while it was combined with quantization and zig-zag process by using pipeline architecture for achieving high throughput. The synthesis results on Spartan-3E XC3S500 FPGA device depicted 84.81 MHz maximum operating frequency with the pipeline latency of 123 CC for one 8×8 input block [80]. In an experimental work done by R. Ebrahimi Atani et al., a novel Distributed Arithmetic (DA) based 8-point 2D DCT/DST coprocessor has been presented [81]. In this design, a fully parallel architecture based on row/column decomposition has been proposed. The architecture was implemented on a Virtex-IV FPGA device with the achieved maximum operating frequency of 117.1 MHz and dynamic power consumption of 393 mW.

3 Platform Architecture

The importance of IoT and its future, IoE, has increased significantly the need for heterogeneous and reconfigurable platforms. During the recent years, a handful of research groups around the world have focused their attention on reconfigurable systems, in particular CGRAs, for wireless signal processing. Additionally, CGRAs are valuable candidate in order to improve the issue of Dark Silicon. In this context, the computationally intensive kernels and data-parallel tasks can be designed as application-specific accelerators by using CGRAs specialized for exploiting thread-level and data-level parallelism. In this chapter, the internal structure of template-based CGRAs used for integration over NoC and the architecture of HARP are described.

3.1 Coarse-Grained Reconfigurable Arrays

CREMA as a template-based CGRA was first introduced by F. Garzia [31] and then developed and scaled-up by W. Hussain [32] as AVATAR. Both CREMA and AVATAR have the same architectural features with different sizes. CREMA and AVATAR are equipped with 4 rows$\times$8 columns and 4 rows$\times$16 columns PEs and two 32-bit local memories each of size 16 columns$\times$256 rows and 32 columns$\times$512 rows, respectively. The architecture of template-based CGRA is depicted in Fig. 3.1. It has to be mentioned that the number of rows of PEs can be further adjusted based on the proposed application. In order to interleave the data from the local memories to the PEs or vice-versa, two I/O buffers are embedded, which are made of 2n$\times$1 multiplexers and 2n 32-bit registers where n is equal to the number of template-based CGRA's columns (8 or 16).

As it is shown in Fig. 3.2, each PE has two input operands for performing both integer and floating-point operations in IEEE-754 format by using 32-bit Arithmetic and Logic Unit (ALU). There are also several components embedded inside the PEs as following: LUT, adder, multiplier, shifter, immediate register and floating-point logic. Each of these components can be selectivity instantiated at design-time (blocks with the dashed borders) by the designer based on the processing requirements of an application, which results to considerable resource/area utilization saving. In order to perform an application completely by employing the PEs, the suitable and most efficient interconnection among them in a point-to-point fashion has to be applied by the designer at design-time as well as specifying the type of operation of each PE. The existing possibilities for interconnections with neighboring PEs can be categorized into three parts, i.e., local connections with the neighbor PEs, interleaved, and global connections for those PEs that are far from each other.

Interconnections between the PEs and the set of instantiated operations to be implemented by each PE at any clock cycle is called context. The number of contexts to

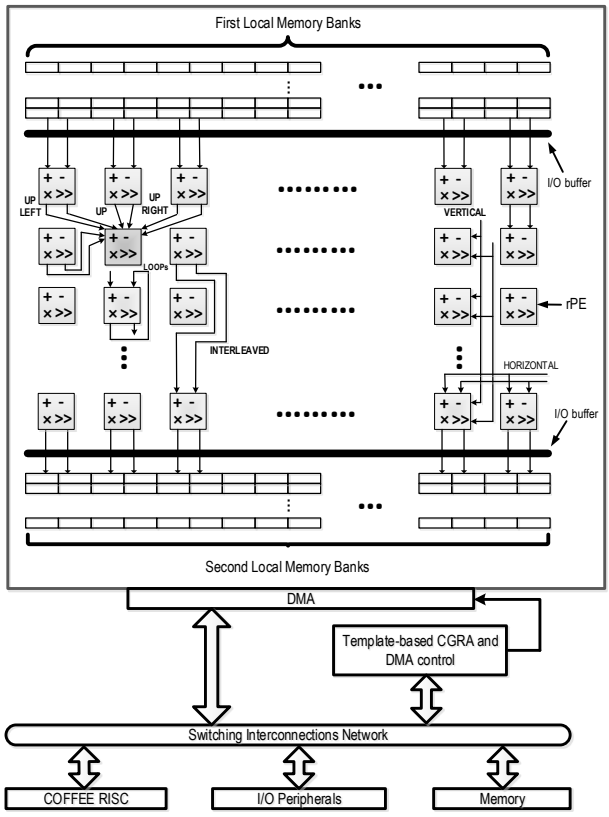

Figure 3.1: The architecture of template-based CGRA used for integration as a CGRA node on HARP.

be used for performing an application is dependent on the user and application's algebraic expressions. The contexts can be designed at the system design-time by using custom Graphical User-Interface (GUI) tool while at run-time, it can be enabled for an execution step according to the application scenario and the stored configuration data in the template-based CGRA. The configuration words generated by the GUI tool are injected sequentially and stored into the configuration memory of PE arrays by the DMA device [82] using a pipelined infrastructure [83]. In this context, reconfigurability means replacing already existing configuration words with the new set of configuration bit streams. The configuration words are composed of two parameters: an address to specify the destination and an operation field to specify the task of each PE. Subsequent to loading the configuration bit streams, the data to be processed over the PE array can be loaded and stored sequentially into the local memories by employing the DMA device. The execution flow of generated accelerators is programmable in C language, which is controlled by COFFEE RISC core by writing control words to the control registers of the CGRA accelerators. In the next step, the data can be processed over the PE array by enabling a context and then stored in the second local memory. Meanwhile, the template-based CGRA can be reconfigured by switching the context for performing an execution step. If required, a new set of data can also be loaded for processing during switching the contexts. The same phases of execution flow might be iterated until the algorithm completes its execution. At the end, the results can be sent back to

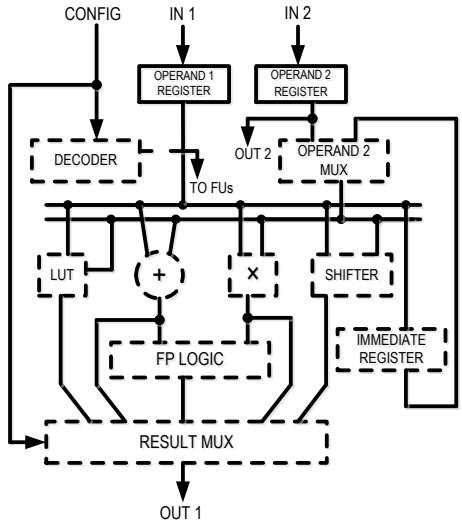

Figure 3.2: PE Core Template. [P.I]

the RISC processor or even transferred to the local memory of another template-based CGRA for further processing.

3.2 Heterogeneous Accelerator-Rich Platform

The HARP platform, shown in Fig. 3.3, is constructed over a NoC of nine loosely coupled nodes arranged in a mesh topology of three rows and three columns ([24], [25]). The hierarchically heterogeneous NoC is chosen with data width of 32 bits and address width of 16 bits, which provides local and global communication [84]. The central node is always integrated with COFFEE RISC core as a master node while the rest of the nodes can be either RISC core or a template-based CGRA, which act as slaves. The NoC nodes also contain a data memory besides a CGRA device and a DMA device with master and slave interfaces. The CGRA nodes integrated on HARP can be modified in terms of size to a specific dimension by the user or the NoC nodes can be employed as a data routing resource only.

A detailed view of both the master and the slave node is shown in Fig. 3.4. The master node contains a data and an instruction memory besides a RISC processor while the slave nodes contain a DMA device and a data memory besides a CGRA device. The DMA device in the slave nodes has a master interface and two slave interfaces. The master interface can be employed for writing to the network, transferring the data within the node, accessing the node's internal data memory and also for connecting to the master side of the DMA device. The DMA's master will be activated once the slave interface of the DMA receives data from the NoC. The slave interfaces are connected to the local memories and slave part of the DMA device, which can receive data from the NoC. The RISC core(s) can write data to the NoC for transferring data between its own data memory and the other slave nodes data memory in the form of packets. The packet has routing information in addition to the data and configuration words. The data in the form of packet can be written to the NoC through the target module and reached to its specified destination (slave node), selected by the request switch

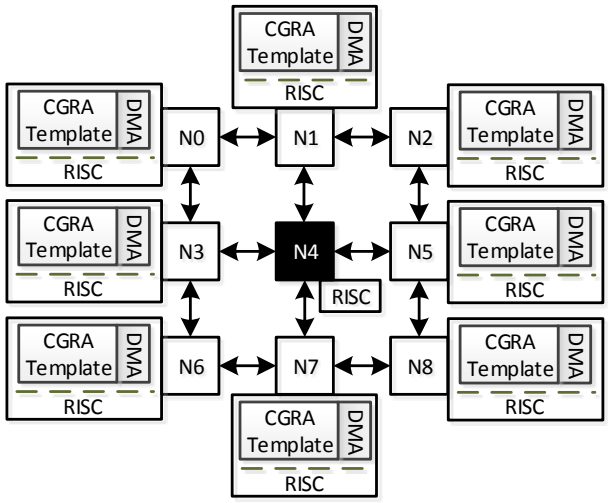

Figure 3.3: General view of HARP architecture. [P.IV]

according to the address field of the routed packet. The targeted slave node can be either the data memory of the respective node or the DMA slave. Furthermore, a shared space is embedded in the data memory of RISC core(s), which is responsible for setting and resetting 'read' and 'write' flags in the case of establishing synchronization for data transfer between two different nodes. The size of shared memory space can be specified based on the number of slave nodes. The RISC core and the data memory are integrated with each other using request/response switches, which can be activated by the arbiter. The arbiter is also responsible to establish connections among different modules. Moreover, the RISC core can read data from the instruction and data memory or write data to them by the use of request switch, which makes the connection between the RISC core and one of its local slave devices.

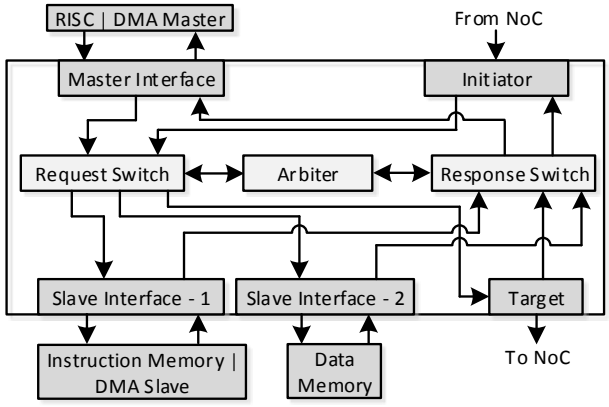

Figure 3.4: A detailed view of the Master and Slave nodes of HARP. [P.IV]

At the first stage of execution flow of HARP, the central supervisor node sends the configuration words and the data to be processed over the NoC in the form of packet transmission from its data memory to the data memories of the slave nodes during the

system start-up time. The transferred data will be received first by the initiator and then by the request switch of the destination node, which selects the targeted destination node. Then the configuration words and the data will be loaded by using the DMA device into the local memory of CGRA node. In the case of several configuration words and sets of data for the same node, they have to wait until the previous data transfer task has been completed. Although the RISC core is not allowed to send any new packet and configuration word for the same node, however, for the other nodes, the RISC core can perform the same transfer operations within this period. In order to transfer the data, which are pending in a queue and also retain the sequence of data transfers, the RISC core requires establishing synchronization between its own and slave nodes by writing '1' in the allocated shared memory location corresponding to the destination node. Once the packet arrives to the destination node by using the DMA slave, it will be read by the DMA master and then distributed to the PE's configuration memories and the local memories of a CGRA node. By completing the data transfer, DMA's master should send an acknowledgment, which results in writing '0' over the NoC by RISC core and reset the allocated shared memory space. By loading the configuration words and the data into the local memories of CGRA nodes and configuration memories of PEs, the RISC core can send the control words to the template-based CGRA for cycle-accurate processing. Based on the application requirements and data dependencies in the scenario, the CGRA nodes can work either independent of each other in parallel or in sequence by exchanging data. The data exchange among the CGRA nodes can be performed in a way that the stored results in a local memory can be fetched by the local memory of another CGRA node directly. By completing the processing of data, they will transfer back from the local memory of CGRA to the data memory of CGRA and then to the data memory of the supervisor node.

4 Design and Implementation of OFDM Receiver Blocks on CGRAs

In this chapter, the design and implementation of IEEE 802.11a/g receiver blocks on template-based CGRAs are presented and analyzed. At the market of wireless communication systems, OFDM is one of the most promising techniques to obtain higher data rates, which is also selected as a valuable candidate for 5th generation wireless systems (5G). OFDM systems split an input high-speed data stream into several parallel streams which are simultaneously transmitted across a contiguous collection of non-overlapping subcarriers [35]. OFDM has various advantages, including resilience to Inter-Symbol Interference (ISI) and frequency selective fading caused by the multipath propagation [35]. The simplified version of the IEEE 802.11a/g receiver blocks and the tasks implemented by template-based CGRAs or RISC core are shown in Fig. 4.1. OFDM receiver blocks contain computationally-intensive and time-consuming tasks such as FFT and Correlation, which can be processed in parallel. Additionally, OFDM receiver also has some tasks, which are serial in nature and required to be processed by processor software.

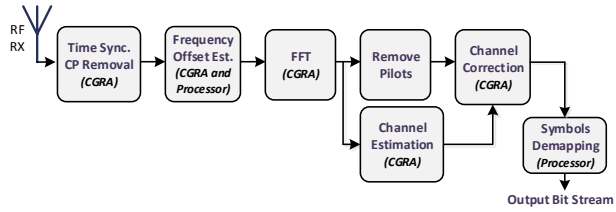

Figure 4.1: A simplified view of an OFDM receiver. [P.IV]

Subsequent to transmitting the entire OFDM symbol across the noisy channel and performing Analog to Digital Conversion (ADC), the receiver is responsible for performing several stages mainly Time Synchronization (TS), Frequency Offset Estimation (FOE), Fast Fourier Transform (FFT), Channel Estimation (CE) and finally, Symbols Demapping (SD). The received data packet at the receiver side is composed of training symbols (short and long preambles) and OFDM data symbols. The training symbols, used for synchronization purposes, are constant and known to the receiver. Short training symbols are composed of ten segments in total. The first seven segments are appointed for packet detection while the last three segments are specified for coarse FOE. Each segment has 16 subcarriers, defined at the transmitter side according to specific repetition. Regarding the long preambles, they are composed of two segments, each containing 64 identical samples. The CE and fine FOE are use cases for them. In the following section, each block of OFDM receiver is described in detail and following

that, the design and near optimal implementation of application-specific accelerator for that specific block is presented.

4.1 Time Synchronization

Subsequent to performing ADC, the first block of OFDM receiver is TS, which has to detect the arrival moment of received OFDM symbols. Moreover, the correct position of FFT window can be detected. In this research work, Cyclic Prefix (CP) correlation based method has been selected in order to implement this block [85]. CP is the copy of the end part of OFDM symbol as a prefixing of a symbol with the length of 16 and the time period of 0.8μs, which is equal to Guard Interval $(1/4) \times T_{FFT}$ (3.2μs). CP is employed in order to combat with ISI and enable the OFDM signal to operate reliably. Fig. 4.2 depicts the overall performance of TS block where the received signal y_n is correlated with its delayed version with the length of delay z^{-D} of 16 (the length of CP based on IEEE 802.11a/g standard specifications) to determine the similarity. According to the signal flow of TS process, the correlation algorithm produces outputs c_n and z_n expressed in Eq. 4.1 ($*$ stands for the complex conjugate) and Eq. 4.2, respectively.

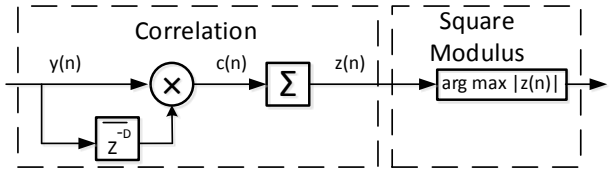

Figure 4.2: Signal flow structure of CP correlation based method for TS. [P.IV]

$$c_n = y_n y^*_{n-D} \tag{4.1}$$

$$z_n = \sum_{i=0}^{L-1} c_{i+n} \tag{4.2}$$

In order to map the above equations on the template-based CGRA with efficient placement and routing, first of all, they have to be simplified and then, the suitable size of PE array has to be selected for achieving higher level of parallelism and reducing the processing time.

$$\begin{aligned} c_n = & \underbrace{((y_{n(R)} \times y_{n-D(R)}) + (y_{n(I)} \times y_{n-D(I)}))}_{Real} \\ & + \underbrace{((y_{n(I)} \times y_{n-D(R)}) - (y_{n(R)} \times y_{n-D(I)}))}_{Imaginary} \end{aligned} \tag{4.3}$$

Due to the size of correlation algorithm (80-point) and based on the above simplified equation, the author decided to scale-up the template-based CGRA to 5×16 PE array. In Eq. 4.3, R and I stand for Real and Imaginary parts of the received signal, respectively. The mapping of Eq. 4.3 on template-based CGRA in order to accomplish 80-point correlation can be performed by using three contexts in total. The first context, which is always the same for every design, is related to loading immediate values to the PEs for performing shift operation. It is required for preventing data overflow after each

multiplication. Since all the data in this manuscript are represented in 12-bit to keep the accuracy in an acceptable level, the immediate value is also equal to 12, which means shifting the result of multiplication by 12 bits. The second and third designed contexts for performing 80-point correlation algorithm are depicted in Fig. 4.3.

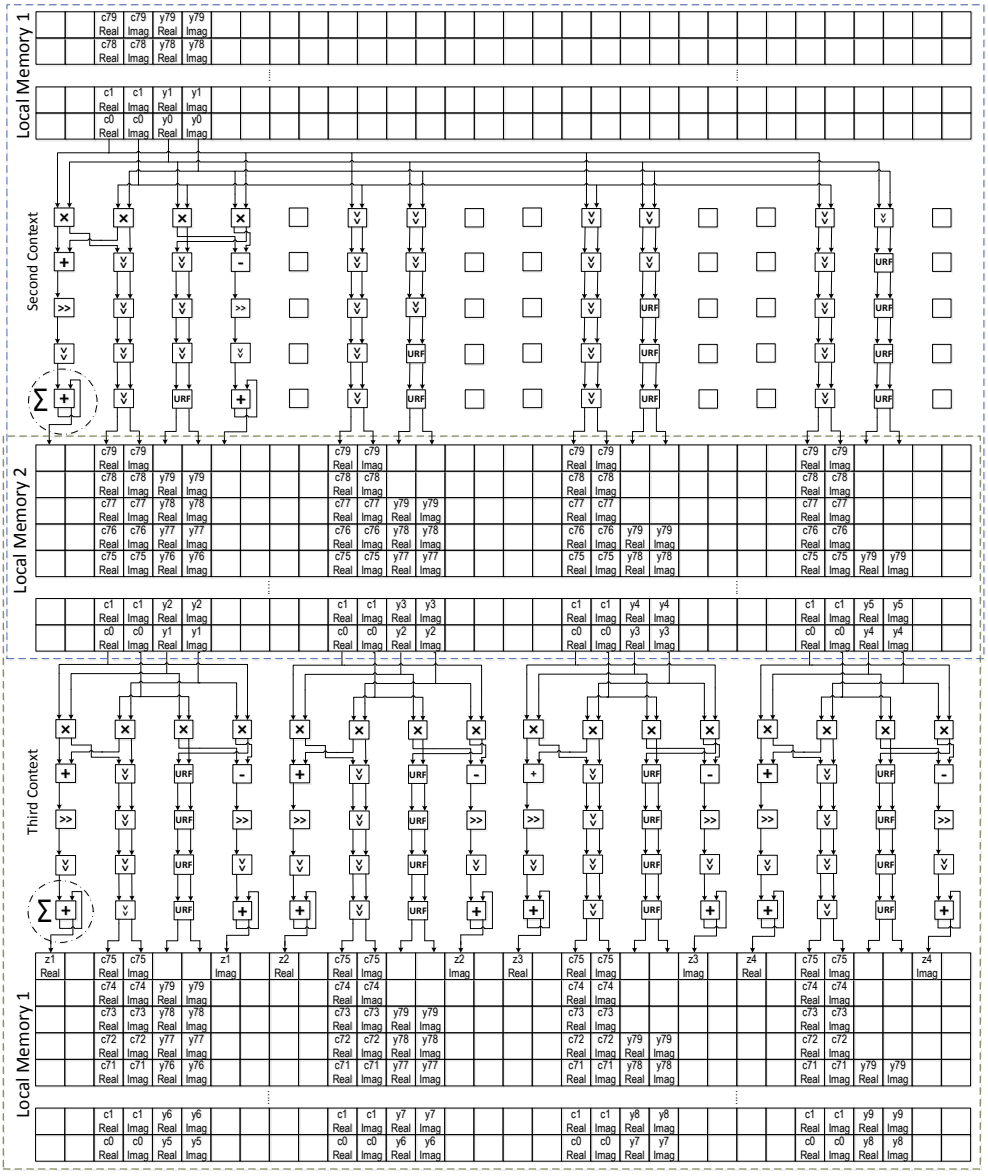

Figure 4.3: Second & Third Contexts for the Calculation of the Correlations. c_n and y_n stand for the original received signal and the complex conjugate of the delayed version of the signal, respectively. [P.II]

Subsequent to loading the data to be processed into the first local memory of second context, two sets of tasks can be performed. The first is related to the multiplication among the received data symbols and the complex conjugation of its delayed version. The second task is distributing the data to the other columns of local memory for

maximizing the parallel usage of PEs in the third context. As it can be observed from Eq. 4.3, there are four multiplications (done in the first row of PEs), one addition and one subtraction (done in the second row of PEs). The data indicated by the indexes from 0 to 79 belong to 80-point correlation, which resulted to 80 correlations in total for performing TS. The last row of PEs (in the second and the third contexts) is also employed for executing a sum-of-products of the outputs by using feedback operation for computing the final output of each correlation z_n based on Eq. 4.2. The number of iterations of feedback operation for performing accumulation can be determined at run-time based on the number of existing data symbols. Unregistered-Feed Through (URF) is also used to delay the data symbols by one or a few cycle after each correlation. For instance, four URFs are required in order to shift the delayed version of data symbols by four positions. By that way, the data can be shifted in parallel with its process in order to accomplish the next step. Due to using ping-pong memories [86] in template-based CGRAs, the stored data symbols in each of local memories can be read directly by changing the direction of data flow, specified by the user at design-time. The second context is just used once while the third context should be iterated until the correlation algorithm completes its execution. By completing all 80 correlations, the maximum value should be found, which is equivalent to the edge of FFT window. Since the results of correlation are complex numbers, the following equation can be used as the Square Modulus (SM) to calculate the magnitude of complex numbers

$$\begin{aligned} \hat{\tau_s} &= \underset{n}{\operatorname{argmax}} \mid z_n \mid \\ &= \underset{n}{\operatorname{argmax}} \mid z_{n(R)} \times z_{n(R)} + z_{n(I)} \times z_{n(I)} \mid \end{aligned} \tag{4.4}$$

where $\hat{\tau_s}$ stands for the maximum value of the performed correlations (z_n) and the index of the time offset, R and I also represent the Real and Imaginary parts. Subsequent to determining the largest value in the RISC core, it has to be returned to the supervisor node for further processing.

4.2 Frequency Offset Estimation

Although OFDM is popular for its advantages, however it suffers from sensitivity to Carrier Frequency Offset (CFO), which might be added to the signal due to the device impairments and causes ISI and also rotation of demodulated symbols in the constellation [35]. It means that after downconversion, the received baseband signal is centered at f_Δ (represents the frequency offset) instead of zero as it is shown in Fig. 4.4.

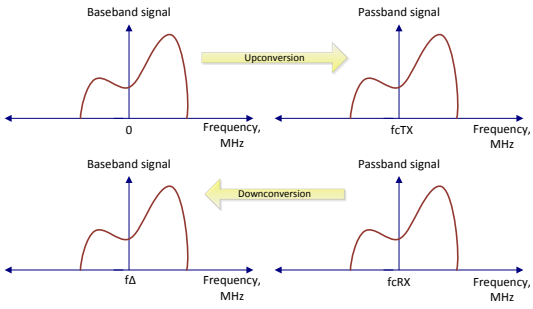

Figure 4.4: Upconversion and downconversion of signal in transceiver. [P.IV]

The amount of added CFO to the signal can be estimated by using known last three segments of the short training symbols added in the transmitter. The down-conversion of the received signal r_n can be expressed as following:

$$r_n = x_n e^{j2\pi f_\Delta n T_s} \tag{4.5}$$

where x_n is transmitted and r_n is received signal. Delay and correlation method can be employed for estimating the value of CFO which is expressed as Eq. 4.6.

$$\begin{aligned} z &= \sum_{n=0}^{L-1} r_n r_{n+D}^* \\ &= \sum_{n=0}^{L-1} (\underbrace{(((r_{n(R)} \times r_{n+D(R)}) + (r_{n(I)} \times r_{n+D(I)}))}_{Real} \\ &+ \underbrace{((r_{n(I)} \times r_{n+D(R)}) - (r_{n(R)} \times r_{n+D(I)})))}_{Imaginary} \end{aligned} \tag{4.6}$$

Here R and I equivalent to the real and imaginary parts of the signal, respectively. Moreover, delay, D, is computed performing the multiplication between the period of short training symbols 0.8 μs and the frequency space 20.0 MHz. As it is mentioned earlier, each segment of short training symbols contains 16 subcarriers, resulting to 48 predefined data symbols for three last segments. The mapping of 48 complex multiplications on CREMA based on Eq. 4.6 is depicted in Fig. 4.5.

As the next step, the amount of added frequency offset should be calculated by using Eq. 4.7 where T_s is the sampling period and $\angle$ takes the angle of outputs of the performed complex multiplication z. The phase angle can also be found by using Eq. 4.8 in which the arctangent ratio should be applied on the division among the imaginary part y and the real part x.

$$\hat{f_\Delta} = -\frac{1}{2\pi D T_s} \angle z, \tag{4.7}$$

$$\hat{\theta} = atan(\frac{y}{x}) \tag{4.8}$$

The division operation is better to be performed by using CORDIC algorithm [87] in RISC processor software instead of mapping the algorithm on template-based CGRA due to its complex algebraic equations and high number of iterations. In general, CGRAs are the suitable candidates for parallel tasks, not for complex algebraic operations. Although the implementation of CORDIC algorithm resulted to shorter execution time than template-based CGRA, however it was at the cost of more energy and power. Furthermore, the accuracy of CORDIC algorithm can be tuned by the user at design-time at the cost of execution time. The CORDIC algorithm for calculating the division operation can be written based on Algorithm 1.

By completing the division operation, the phase angle has to be calculated. According to Eq. 4.9, the calculation of the arctangent ratio in processor software can be performed by employing Taylor series where the value of N can be employed for adjusting the level of accuracy (4 in this case-study).

$$arctan\ x = \sum_{n=0}^{N} \frac{-1^n}{2n+1} x^{2n+1} \tag{4.9}$$

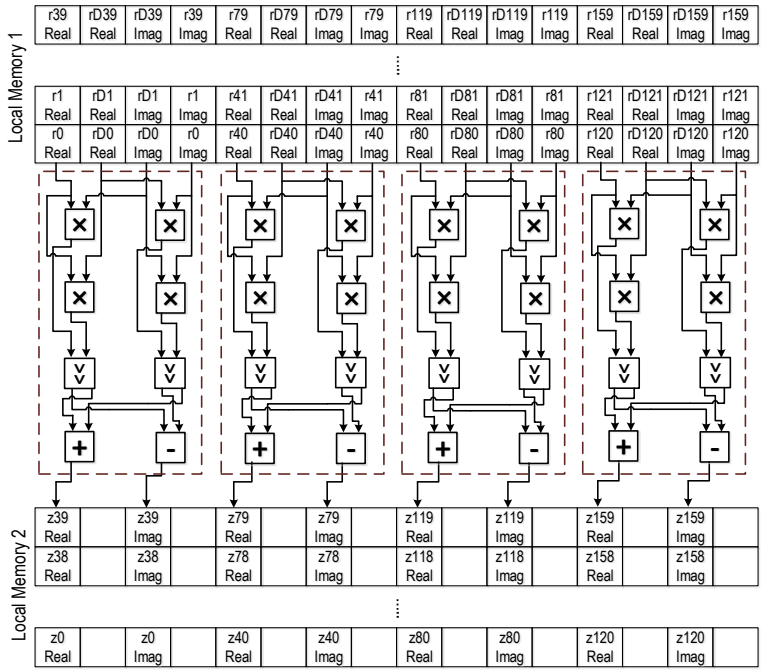

Figure 4.5: Context for the complex multiplication between a signal and its complex conjugation [1]. r and rD stand for the last three segments of short training symbols and its delayed version. [P.I]

Algorithm 1 CORDIC algorithm for calculating the division operation

```
Initialize MaxBits
for i:=0 to MaxBits step 1 do
    if (y < 0||z >= 0) then
        y = y + x × t
        z = z − t
    else
        y = y − x × t
        z = z + t
    t = t >> 1
```

Subsequent to estimating the added frequency offset, it has to be corrected based on Eq. 4.10 where the received signal should be multiplied by the estimated frequency offset.

$$r_n{}' = r_n \times e^{-j2\pi f_\Delta \frac{n}{N}} \tag{4.10}$$

Here, $r_n{}'$, n and N stand for the corrected signal, the sample index and the number of samples in a symbol, respectively. In order to compute the exponent function, Taylor series can be employed again in processor software based on Eq. 4.11. However, it can be further simplified as Eq. 4.12 due to the complex data symbols.

$$e^x = \sum_{n=0}^{\infty} \frac{x^n}{n!} \tag{4.11}$$

$$e^z = e^x(cos(y) + isin(y)) \tag{4.12}$$

Here, z is a complex value, which consists of real (x) and imaginary (y). In order to implement *cos* and *sin* functions in processor software, they have to be expanded by using Taylor series as Eq. 4.13.

$$cos\ y = \sum_{n=0}^{\infty} \frac{(-1)^n}{(2n)!} y^{2n} \quad , \quad sin\ y = \sum_{n=0}^{\infty} \frac{(-1)^n}{(2n+1)!} y^{2n+1} \tag{4.13}$$

Subsequent to performing the above-mentioned steps in processor software, the final step related to data symbols correction in terms of the frequency offset (Eq. 4.10) can be implemented by using CREMA-generated accelerator, almost the same context as shown in Fig. 4.5. Thus, the data have to be transferred back from the main memory of the supervisor node to the local memory of the template-based CGRA.

4.3 Fast Fourier Transform

FFT is required to convert the corrected data symbols in terms of the frequency offset from time domain to frequency domain. FFT, also called demodulation, is computationally intensive and time consuming tasks, which can be effectively accomplished by the use of radix-2^m structures [88], where $m \in Z^+$ and its structural unit is called a butterfly. In this manuscript, demodulation is performed by using designed AVATAR-generated accelerator [32] for processing a 64-point FFT in radix-4 scheme within 3.2 μs according to the IEEE 802.11a/g standard specifications.

4.4 Channel Estimation

As it is mentioned earlier, the transmitted data symbols have to pass through the wireless channel, which can be noisy. Therefore, because of the various impairments, the data symbols may get distorted while passing through the channel. At the receiver side, the data symbols have to be recovered by estimating the channel frequency response. The channel estimation task can be accomplished by employing pilot-assisted linear interpolation algorithm, shown in Fig. 4.6. In this method, four predefined pilots are added at the transmitter. The values and position of the pilots are known for the receiver and can be used for estimating the channel frequency response of all the subcarriers, located among the pilots at specific positions.

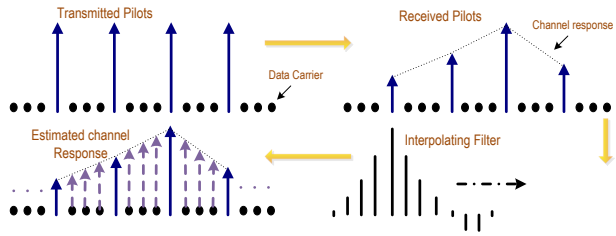

Figure 4.6: Channel Estimation based on Pilot-Assisted Linear Interpolation. [P.IV]

By considering the transmitted data symbols (X_n), channel impulse response (H_n) and additive noise (N_n), the received data symbols can be demonstrated as

$$Y_n = X_n H_n + N_n \tag{4.14}$$

where n stands for the number of subcarriers. As the first step, the H_n should be estimated and then, Y_n has to be corrected according to X_n [89]. In order to calculate the channel impulse response H_n, first the the channel impulse response of the received pilots $\tilde{H_k}$ should be computed, which is expressed as Eq. 4.15 where k is the number of pilots and P_{Rx} is the received noisy pilots, respectively. Moreover, a diagonal matrix is required to be made by transmitted pilots, expressed as Eq. 4.16.

$$\tilde{H_k} = M^{-1} P_{Rx} \tag{4.15}$$

$$M = \begin{bmatrix} M_{1,1} & 0 & \cdots & 0 \\ 0 & M_{2,2} & \cdots & 0 \\ \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & 0 & M_{k,k} \end{bmatrix} \tag{4.16}$$

The complex Matrix Vector Multiplication (MVM) between the Received Pilots (RP) and the Inverse of Transmitted Pilots (ITP) can be computed by employing AVATAR-generated accelerator, shown in Fig. 4.7. As the next step, in order to accomplish the

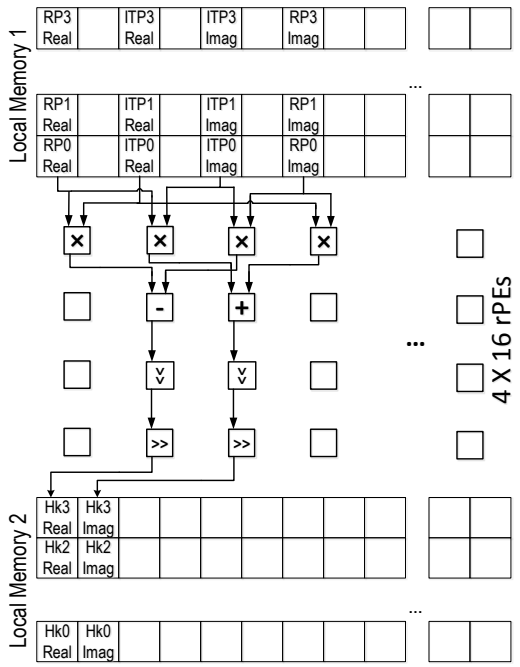

Figure 4.7: Second Context for the Channel Estimation. [P.IV]

channel equalization, the channel frequency response of the adjacent subcarriers H_n requires to be estimated by employing Linear Interpolation method as Eq. 4.17 based on the found $\tilde{H_k}$.

$$\hat{H_n} = \sum_{i=1}^{N_p-1} \sum_{j=1}^{N_s} \tilde{H_k}(i) \underbrace{+}_{3} ((\tilde{H_k}(i+1) \underbrace{-}_{1} \tilde{H_k}(i)) \underbrace{\times}_{2} \underbrace{\frac{j-1}{N_s}}_{\mu}) \tag{4.17}$$

where N_p and N_s stand for the number of pilots and samples, respectively and also μ is the step size in order to estimate the channel frequency response for other subcarriers located around pilots $\tilde{H}_n$ by expanding the channel frequency response for four received pilots $\tilde{H}_k$. Eq. 4.17 can be mapped on AVATAR as a third context shown in Fig. 4.8. Both real and imaginary parts of the pilots can be performed simultaneously by using the left and right side of the third context, respectively. The step sizes are also computed by the processor software and loaded into the local memory as 16 different fixed values. As it is mentioned in Eq. 4.17, three steps are required by the algorithm to be completed, which are also instantiated by dashed border circles in the third context.

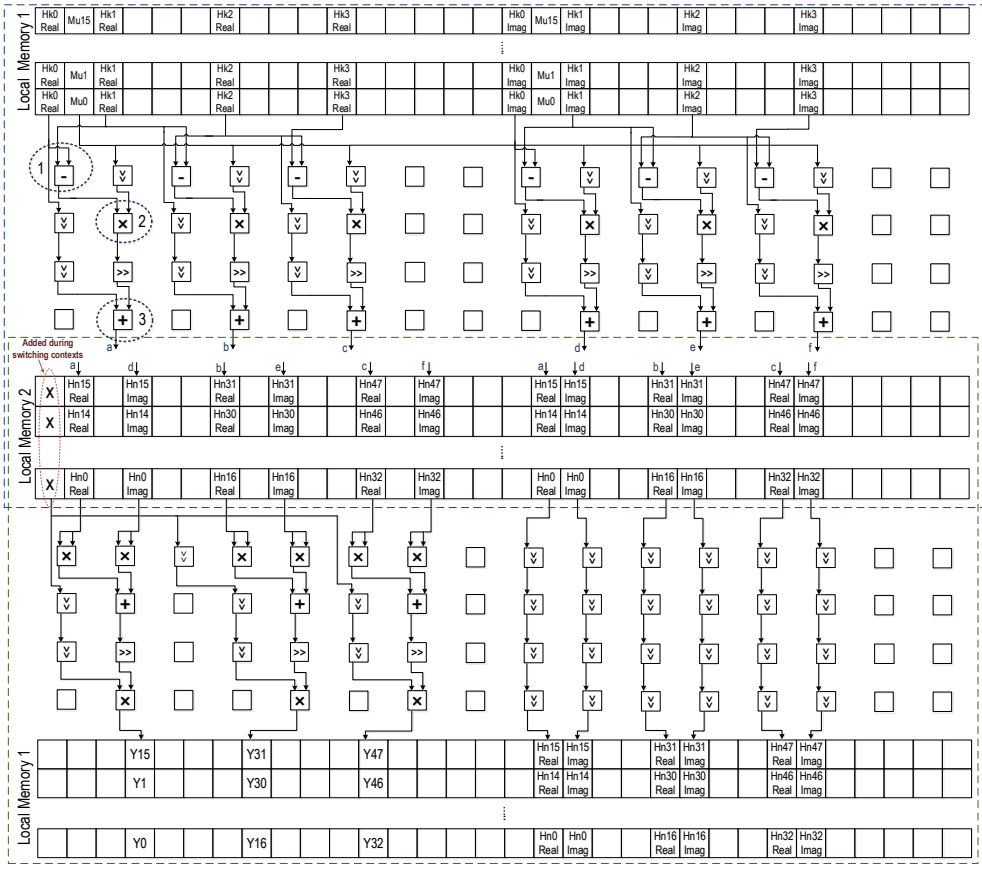

Figure 4.8: Third & Fourth Contexts for the Calculation of Linear Interpolation and Newton-Raphson Method. [P.IV]

By completing the linear interpolation, the estimated channel frequency response can be stored in the second local memory for performing the channel equalization in order to refine Y_n as close as possible to X_n. Accordingly, the received data symbols excluding the pilots should be divided by the estimated channel frequency response as following.

$$\hat{Y_n} = \frac{Y_n}{\hat{H_n}} \tag{4.18}$$

In order to perform the division operation, Newton-Raphson method [90] is employed

as an iteration algorithm for determining the root of an equation. As an example, for the given function *f(x)*, the first approximation of the root can be found by using Eq. 4.19 where n is the number of iteration, x_n is the initial guess of the root and $f^{'}(x_n)$ is derivative of a function $f(x_n)$.

$$x_{n+1} = x_n + \frac{f(x_n)}{f^{'}(x_n)}, \tag{4.19}$$

The above equation can be further modified to be used for performing division operation as Eq. 4.20. For instance, in order to calculate $\frac{1}{D}$, the function *f(x)* can be written as $f(x) = \frac{1}{x} - D$ where D and x_n stand for the denominator and initial guess, respectively.

$$\begin{aligned} x_{n+1} &= x_n + \frac{\frac{1}{x_n} - D}{-\frac{1}{x_n^2}} \\ &= x_n + (\frac{\frac{1}{x_n} - D}{-\frac{1}{x_n^2}} \times \frac{-x_n^2}{-x_n^2}) \\ &= x_n - (-x_n + Dx_n^2) \\ &= 2x_n - Dx_n^2 \\ &= x_n.(2 - Dx_n), \end{aligned} \tag{4.20}$$

According to Eq. 4.18, the denominator is a complex number since it is damaged by the noisy channel. Therefore, in order to be mapped on the template-based CGRA, it can be further simplified as

$$\frac{x + iy}{a + ib} \times \frac{a - ib}{a - ib} = \frac{(x + iy) \times (a - ib)}{a^2 + b^2} \tag{4.21}$$

where $x + iy$ is representing the outputs of FFT block and $a + ib$ and $a - ib$ stand for estimated channel response and its complex conjugate, respectively. As it is depicted in the fourth context of Fig. 4.8, the left side is allocated to perform the first step of Newton-Raphson method. As the first step, the first two rows of PEs are instantiated for computing the square values of the real and imaginary parts of the channel frequency response, which is equivalent to $(a^2 + b^2)$ (denominator). The initial guess also can be loaded into the local memory during switching the contexts. On the last row of PEs, the multiplication among the initial guess and denominator can be performed. The right part of the fourth context, which consists of delay operation is used for passing the results of linear interpolation for further processing over its following context.

The next step of Newton-Raphson method has been implemented by the left side of the fifth context where the constant 2 is loaded into the local memory along with the results of the previous part. It has to be mentioned that the reason behind loading the initial guess and the constant value in Fig. 4.8 and Fig. 4.9 is making the process of the algorithm faster because of line readability feature of local memories in the CGRAs. As it can be observed from Fig. 4.9, Eq. 4.19 gets completed by using the left side of the fifth context where the required shift operation, subtraction and multiplication are performed over the instantiated PEs. However, at the right side of the fifth context, the multiplication among the demodulated data symbols (FFT outputs) and complex

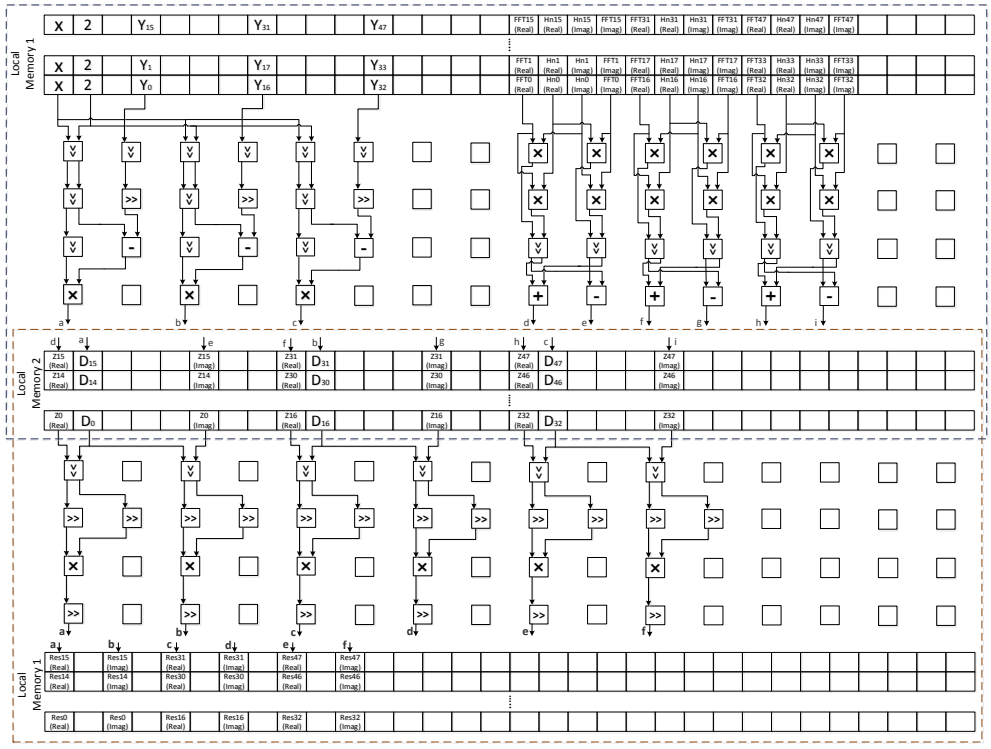

Figure 4.9: Fifth & Sixth Contexts for the Calculation of Newton-Raphson Method and Channel Equalization. [P.IV]

conjugation of estimated channel frequency response can be executed. The last stage of channel estimation is channel equalization, which is implemented by the sixth context. Here, the outputs of Newton-Raphson method (left side of the fifth context) should be multiplied by the results of calculated complex multiplication (right side of the fifth context). In Fig. 4.9, Res_i stands for the equalized received data symbols, which can be passed to the next block of an OFDM receiver, Symbols Demapping block, in order to extract data bits from data symbols.

4.5 Symbols Demapping

Finally, as the last stage of an OFDM receiver, demodulation or SD has to be performed in order to ascertain the transmitted data bits for each received data symbol. It can be done by the use of two different methods, hard-decision or soft-decision. In this research work, the author selected the hard-decision method. Once the data bits are generated at the transmitter side, they have to be modulated with one of the schemes such as 16-QAM (Quadrature Amplitude Modulation) constellation points, which comprise four bits per symbol and are used in this case-study. QAM is a signal in which both of the amplitude and phase of the carrier are changed (by 90 degrees), resulting to complex data symbols with Quadrature and In-phase carriers. In order to perform the hard-decision method at the receiver side, the complex plane should be divided into decision boundaries based on In-phase and Quadrature areas in a way that four zones are required to be shaped (in the case of 16-QAM) for each leftmost and rightmost two

bits of a data symbol. Each data symbol has real and imaginary parts that are equivalent to Quadrature and In-phase, respectively and cause to place the data symbols inside the decision boundaries. As a result, the data symbols will be mapped to the data bits based on the closest constellation point to the data symbols in each of 16 decision areas [35].

5 Evaluation of HARP by Design and Test of an OFDM Receiver

In this chapter, the author presents the evaluation of HARP by implementing the OFDM receiver blocks, which are primarily designed by crafting template-based CGRAs during the previous chapter. Due to the combination of serial and parallel algorithms in the OFDM receiver blocks, HARP is subjected to a stringent test for identifying potential architectural fallacies and pitfalls, demonstrating almost all its design features and technical capabilities and also for the proof-of-concept. Furthermore, the performance of each template-based CGRA integrated on HARP as well as the entire platform and the NoC traffic are recorded in terms of the various performance metrics. HARP with the OFDM receiver baseband processing instance is also compared against other state-of-the-art platforms by using cross-technology comparison.

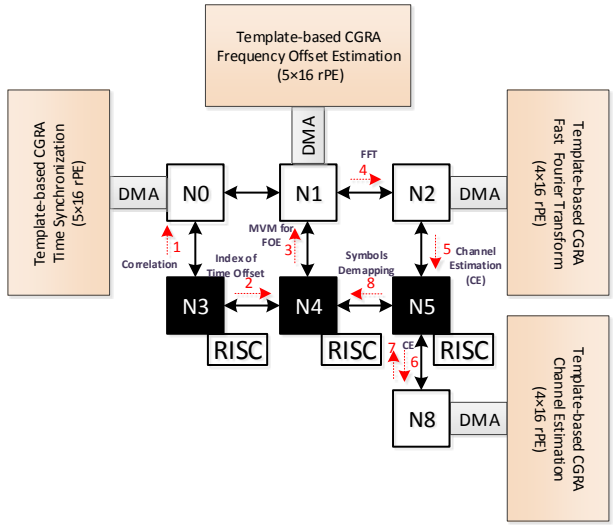

Figure 5.1: Modified HARP platform for an OFDM receiver test-case. The black colored nodes are the master and white colored nodes are the slave nodes. [P.IV]

As it can be observed from Fig. 5.1 as a general view of overall processing architecture of OFDM receiver on HARP platform, there is the bidirectional communication among master and slave nodes in order to exchange data with each other. At design-time, the designer can write the distributed control for transferring the control words and the data as well as execution by using the generated header VHDL files from GUI tool in the processor software of the three RISC cores and compile them. Moreover,

the designer can specify the order of exchanging data among the nodes based on the algorithm data-flow, which is shown in the figure with the numbers and dashed arrows. In Fig. 5.1, the red colored numbers specify the order in which the nodes should be executed. In this case, the platform is modified in a way that node N3 RISC core is the supervisor node for No CGRA, N4 RISC core is the supervisor node for N1 CGRA and N5 RISC core is the supervisor node for N2 and N8 CGRAs in order to transfer the configuration stream and data to be processed. During the system start-up time, configuration words are transferred to the data memory of the CGRA nodes by the three RISC cores. Thereafter the received data symbols are transferred from the data memory of N3 RISC core to the data memory of No CGRA node by using DMA device and then loaded into the local memory for performing TS block. The required CC for data transfer from data memory of a RISC core to the data memory of a CGRA node and from data memory of a CGRA node to the CGRA's local memory and also the execution of different tasks are shown in Table 5.1.

Table 5.1: Clock cycles required for data transfer and processing at different stages. In the table, * signs represent data transfer from CGRA to Node's data memory. [P.IV]

Node-to -Node	Data Memory to Data Memory	Data Memory to CGRA's Local Memory	Transfer Total	Execution Total
N3-No (Correlation)	1072	910	1982	1120
No-N3 (SM)	-	7590*	-	2345
N3-N4	47	-	-	-
N4-N1	1961	2703	4664	12634 (+74)
N1-N2	-	570* + 504	1074	328
N2-N5	-	529*	-	-
N5-N8	637	401	1038	250
N8-N4	-	397*	-	2253

As it can be observed from the table, 1072 CC and 910 CC are required for data transfer from data memory of N3 RISC core to the data memory of No CGRA and to local memory, respectively, which resulted to 1982 CC total data transfer. It is specified by dashed arrow number 1 in Fig. 5.1. The processing of correlation algorithm by No CGRA and SM by RISC processor software can be performed in 1,120 CC and 2,345 CC, respectively. It has to be considered that by completing the correlation algorithm, 80 results of correlation have to transferred back from the data memory of CGRA node to data memory of N3 RISC core within 7,590 CC for executing SM. Subsequent to finding the index of time offset, it can be transmitted to the data memory of N4 RISC core within 47 CC for further processing, which is depicted with dashed arrow number 2 in Fig. 5.1.

As the next step, the last three segments of the received short training symbols are loaded into the local memory of N1 in order to start the process of frequency offset estimation. As it is shown in Table 5.1, transferring the data from data memory of N4 RISC core to the data memory of N1 CGRA and then to the local memory takes 4,664 CC in total, shown with dashed arrow number 3. As it is mentioned in the previous section, the first and the last parts of this block can be performed by the crafted template-based CGRAs in 74 CC (40 CC plus 30 CC) while some parts should be performed by using RISC processor software (in 12,634 CC), which requires data exchange between N1 and N4 twice. In total, the required processes for performing frequency offset estimation

can be completed in 12,708 CC.

As it is shown with dashed arrow number 4, the corrected data symbols in terms of the added carrier frequency offset, which are located in the local memory of N1 CGRA can be fetched directly by the data memory and the local memory of N2 CGRA in 570 CC and 504 CC, respectively. Then the computation of 64-point radix-4 FFT can be processed in 328 CC.

Subsequent to implementing the FFT by N2 CGRA, the results are transferred by the DMA device from the local memory to the data memory of N5 RISC (depicted with dashed arrow number 5) in 529 CC and then to the local memory of N8 CGRA (specified with dashed arrow number 6) in 1,038 CC in order to accomplish channel estimation. The channel estimation block can be executed completely in 250 CC. As the last step, the data symbols have to be returned to the data memory of N5 RISC (depicted with dashed arrow 7) and then to the data memory of N4 RISC (depicted with dashed arrow 8) in 397 CC for performing symbols demapping by RISC processor software in 2,253 CC.

In this case-study, the nodes N6 and N7 are not instantiated with neither CGRA node nor RISC core as they are not required by our specific case study. According to the achieved CC related to execution time of the CGRA nodes, the percentage of the time for the useful execution of the N0, N1, N2 and N8 CGRA nodes is equal to 3.61%, 0.22%, 1.05% and 0.8%, respectively. Although each of CGRA nodes can be designed in way that be reconfigured to implement several tasks, however the critical path will become longer while the maximum operating frequency will become lower. Thus, the author preferred to increase the resources of highly dense FPGA device and not to compromise the speed.

The digital waveforms related to the utilization of NoC bandwidth for transmitting the configuration words and the data transfers between the data memories of the CGRA nodes are specified with red and blue lines in Fig. 5.2. Furthermore, the execution of OFDM receiver blocks are also specified with green lines in the case of template-based CGRAs and gray lines in the case of RISC processor software. First of all, it should be noticed that the activation control words of the slave nodes (N0, N1 and N2) are transferred in parallel, which demonstrates the technical capabilities of the HARP to have simultaneous execution. This resulted to speed up the execution time. However, regarding the CGRA-nodes N2 and N8, since they are supervised by the same RISC core (N5), N8 has to wait until N2 completes its data transfer. Furthermore, because of the data dependency among different blocks of an OFDM receiver, they have to be performed consecutively. The whole platform is running at 200.0 MHz operating frequency. The reason behind selecting this operating frequency will be discussed later. There are three RISC cores and for each of them, a special counter for general-purpose measurements is embedded with the signal name of `tmr_cnt_out`. Therefore, N3 RISC core is responsible for counting the number of clock cycles of N0 CGRA, N4 RISC is responsible for N1 CGRA and N5 RISC is responsible for N2 and N8 CGRAs. As it can be seen from the details shown in Fig. 5.2, the number of measured clock cycles mentioned in Table 5.1 are also written inside the waves. According to that, frequency offset estimation block is the most time-consuming task, which causes a large space of frame without any signal activity. It can be considered as the worst-case candidate and be used later (Chapter 6) for dynamically scaling the operating frequency of each of the other nodes of the NoC without compromising system throughput. Therefore, the instantaneous power dissipation can be mitigated and the computational workload can be uniformly balanced as the frequency of memory accesses will reduce

- a major factor to overall power dissipation by the system. During the next chapter, we will discuss in detail about the employed self-aware computing for identifying the worst case and dynamically monitoring the voltage and operating frequency of all the system components. Furthermore, it can be observed from the waveform that a very computationally intensive task, such as FFT and channel estimation can be computed in a very small fraction of time relatively. Therefore, a designer could employ a large number of computational resources such as AVATAR containing 5×16 PEs to a demanding algorithm like correlation and eventually find it over-performing relatively in the whole design space while it might be an expensive choice.

5.1 Measurements, Estimations, Evaluation and Comparisons

The modified HARP for performing an OFDM receiver is synthesized on a Stratix-V (5SGXEA4H1F35C1) FPGA device for prototyping purposes. The operating conditions are selected for both 0°C (low) and 85°C (high) junction temperatures and 23 mm heat sink with 200 LFPM airflow as a preset cooling solution. The achieved maximum operating frequencies after placement and routing are equal to 182.32 MHz 0°C and 170.3 MHz 85°C for slow timing model (900 mV). In the case of fast timing model (900 mV), maximum operating frequencies were equal to 258.73 MHz and 235.85 MHz at 0°C and 85°C, respectively. The entire platform works on a single clock source and is simulated at 200.0 MHz operating frequency. It has to be mentioned that the clock frequency first has been set to 170.0 MHz according to the minimum achieved operating frequency for running the simulations. Then the author increased the operating frequency up to 200.0 MHz in order to get more advantages from higher clock frequency while there is no timing error on the employed FPGA device.

5.1.1 Resource Utilization

Node-by-node breakdown of resource utilization summary for the entire prototyped platform on FGPA device is depicted in Table 5.2. The resources are categorized in terms of the number of employed Adaptive Logic Modules (ALMs), Registers, Memory Bits and DSP elements. As it can be observed, about 62% of the ALMs, 11% of the registers and 51% of the memory bits are consumed by the entire platform, including three RISC cores and four template-based CGRAs. Additionally, 90% of 18-bit DSP resources are utilized by 32-bit multipliers instantiated in the PEs. Each 32-bit multiplier requires two 18-bit DSP elements to be synthesized on the FPGA. It means that in order to perform 97 32-bit multiplications, 194 18-bit DSP resources are required by template-based CGRAs. In addition, 36 18-bit DSP resources are also required by RISC cores, which results to 230 18-bit DSP elements in total for the entire platform. The breakdown of the number of 32-bit multipliers used for each block of an OFDM receiver and the RISC cores is also given on Table 5.2.

5.1.2 Energy and Power Estimations

The platform's total power dissipation is estimated based on post placement and routing (post P&R) information, not functional simulations, using PowerPlay Power Analyzer Tool of Quartus II 15.0 at an operating frequency of 200.0 MHz and at the room temperature of 25°C. The total power dissipation is estimated by simulating the gate-level netlist of the entire platform and then generating the Value Change Dump (VCD) file by using ModelSim software [91] while it yielded 'HIGH' level of

Table 5.2: Node-by-node Breakdown of Resource Utilization Summary for Stratix-V (5SGXEA4H1F35C1) FPGA device. [P.IV]

Node	ALMs	Registers	Memory Bits	(32-bit Multipliers) DSPs
N0	22,616	9,471	2,633,472	(20) 40
N1	8,171	7,985	2,365,736	(16) 32
N2	22,809	11,583	2,635,136	(28) 56
N3	5,436	5,648	3,145,728	(6) 12
N4	5,505	5,716	3,145,728	(6) 12
N5	5,442	5,650	3,145,728	(6) 12
N8	25,908	16,399	2,633,144	(33) 66
NoC	2,842	4,371	-	-
Total	98,729 62%	66,823 11%	19,704,672 51%	(115) 230 90%

confidence. The VCD file, generated at the same time of running the whole platform, contains all the information of signal transition activity during the implementation of an OFDM receiver [36]. The total power dissipation is composed of static, dynamic and I/O power. The power of the FPGA chip is called static power, which is required to maintain the FPGA device in the ON state. The power which is due to the signal switching activity of the design for the entire run-time duration is called dynamic power. The static power is almost the same for all template-based CGRAs since a large are of the chip remains unused and accordingly adds a large offset to all the static power estimations. Although the static power is essentially characteristic to a particular FPGA chip, it might increase/decrease for a few mW by scaling up/down the size of the template-based CGRAs. According to the results obtained from the tool, the estimated values of static, dynamic and I/O power dissipation are equal to 1243.84 mW, 2623.72 mW and 27.37 mW, which resulted to 3894.93 mW as a total power dissipation. Node-by-node breakdown of dynamic power, active time and energy consumption of the NoC nodes are shown in Table 5.3. As it can be found out from the table, the dynamic power dissipation increases as the size of the template-based CGRA increases and vice-versa. For instance, AVATAR-generated accelerators integrated in N0, N2 and N8 require almost 1.5X-2X dynamic power consumption compared to the CREMA-generated accelerator used in N1. The energy consumption is the product of power dissipation and active time of each node and is calculated for every node separately.

Table 5.3: Node-by-node Breakdown of dynamic power dissipation and energy estimation. [P.IV]

Node	Accelerator Type	Dynamic Power (mW)	Active Time (μs)	Dynamic Energy (μJ)
N0	Time Synchronization	414.35	7.32	3.03
N1	Frequency Offset Estimation	272.23	0.37	0.1
N2	FFT	526.07	1.64	0.86
N3	General Purpose Processing, Synchronization, Control	114.47	141.17	16.15
N4		113.82		16.06
N5		114.52		16.16
N8	Channel Estimation	448.01	1.25	0.56
NoC	-	10.10	-	-
Integration Logic	- -	609.07	- -	-
Total	-	2623.72	-	53.16

5.1.3 Evaluation and Comparisons

As it can be seen from Fig. 5.1, the current instance of HARP, which is modified for processing an OFDM receiver, consists of 240 PEs in total. Furthermore, the entire platform is running at 200.0 MHz operating frequency. By considering the total power dissipation of 3894.93 mW, the current instance of HARP delivers a performance of 48 GOPS and 0.012 GOPS/mW for Altera Stratix-V chip in 28 nm. It has to be mentioned that GOPS is the product of the number of PEs and the operating frequency. In addition to this case-study, two more research work have been done already by [24], [25] with varying sizes of template-based CGRAs and different number of cores, which also yielded 0.012 GOPS/mW of performance. Accordingly, the author considered 0.012 GOPS/mW as an **architectural constant** for the HARP template on Stratix-V FPGA due to its scalability and regularity, which ensures application-independent figure of merit. Regarding the utilization rate of PEs, 2914 operations have been performed by 240 instantiated PEs for performing the whole OFDM receiver in 9.845 μs. It results to 295.98 MOPS. Thus, the utilization rate of PEs would be equal to 0.61% (295.98 MOPS / 48 GOPS). It shows that HARP would be an excellent candidate for alleviating Dark Silicon issues.

According to HARP's **architectural constant**, comparisons with other state-of-the-art platforms can be established as shown in Table 5.4. In some cases where various platforms have been synthesized for different technologies, i.e., FPGA and ASIC, **cross-technology comparisons** is a solution to do the comparison with acceptable accuracy. For this purpose, the process sizes have to be scaled and the performance gaps have to be analyzed. In Stratix FPGA devices, scaling from 40 nm to 28 nm and from 90 nm to 28 nm will increase the synthesis operating frequency by 20% and 40%, respectively [37]. Regarding the performance gap between two technologies, FPGAs and ASICs, a 90 nm ASIC implementation depicts a speed-up of 4X on average against a 90 nm FPGA implementation while requiring 14.0X lower dynamic power dissipation [92]. In the worst case when the static and the dynamic power dissipation have almost the same values, the factor of 14 for the total power dissipation can be reduced to almost 2 [92]. Furthermore, in order to estimate the performance gap from 90 nm ASIC to 28 nm FPGA, the platform's value in terms of GOPS or GOPS/mW should be multiplied by a scaling factor of 60% [92].

Table 5.4: Cross-technology comparisons between HARP and other state-of-the-art platforms. SU_fTfs, SD_aTfs and Ps stand for Speed-Up for FPGA to FPGA scaling, Speed-Down for ASIC to FPGA scaling and Power scaling, respectively. [P.IV]

Platform Technology	Performance Metric	Platform's Value	Scaled PV	HARP's Value	Gain
NineSilica FPGA 40 nm	FFT Execution Time (μs)	10.3	Exe. Time - Exe. Time × 20% SU_fTfs = 10.3 - 10.3 × 0.2 = 8.24	2.1	3.9×
[93] FPGA 90 nm	GOPS	19.2	PV×40% SU_fTfs = 19.2×1.4= 26.88	48	1.78×
P2012 CMOS 28 nm	GOPS/mW	0.04	PV×SD_aTfs×Ps = 0.04×1/4×1/2= 0.005	0.012	2.4×
ADRES CMOS 90 nm	GOPS/mW	0.004	(PV×SD_aTfs×Ps)×60% SU_fTfs = (0.004×1/4×1/2)×1.6= 0.0008	0.012	15×
MORPHEUS CMOS 90 nm	GOPS/mW	0.02	(PV×SD_aTfs×Ps)×60% SU_fTfs = (0.02×1/4×1/2)×1.6= 0.004	0.012	3×

As it can be observed from Table 5.4, NineSilica [59] requires 10.3 μs for performing 64-point radix-4 FFT. In this context, the same task has been performed in HARP in

2.1 μs. It shows 3.9X speed-up after performance scaling. Furthermore, regarding the resource utilization, NineSilica requires 71,679 ALUTs on Stratix-IV device while HARP requires 102,637 ALMs on Stratix-V device. Since one ALM on Stratix-V device is equivalent to two ALUTs on Stratix-IV device, HARP shows the cost of 1.5X logic resources in comparison with NineSilica.

Another homogeneous MPSoC synthesized on a 90 nm FPGA device delivers a performance of 19.2 GOPS [93]. In order to the comparison with HARP, the performance should be scaled from 90 nm FPGA to 28 nm FPGA by a factor of 40%. It is resulted to increase the value of homogeneous MPSoC to 26.88 GOPS. Against the scaled platform's value, HARP platform obtained a gain of 1.78X.

The platform P2012 has been synthesized using 28 nm CMOS technology ([29] & [30]). Considering the performance gap between two technologies as well as the worst case scenario with the equality among the static and dynamic power, the scaled performance of P2012 on a 28 nm FPGA is estimated to be 0.005 GOPS/mW. Accordingly, HARP achieved a performance gain of 2.4X.

The ADRES platform has been synthesized on 90 nm CMOS with the performance of 0.004 GOPS/mW ([14] & [15]). Subsequent to scaling the performance from 90 nm CMOS to a 90 nm FPGA and following that to a 28 nm FPGA, the ADRES platform presents the performance of 0.0008 GOPS/mW, which resulted to 15X gain by HARP.

In the case of MORPHEUS platform, it presents the 0.02 GOPS/mW at 90 nm CMOS ([26] & [27] & [28]). Subsequent to scaling the performance from 90 nm CMOS to a 90 nm FPGA and multiplying by a 60% speed-up, the HARP's performance depicts a performance gain of 3X.

Another important vision of each platform is their throughput. In this research work, by considering the implementation of an OFDM receiver at 200.0 MHz operating frequency, the throughput is equal to 17 Mbit/s. It is calculated based on the number of data bits (192) extracted from 48 data symbols at the SD block. The required throughput (data rate) for 16-QAM modulation based on IEEE 802.11a/g standard specifications is 48 Mbit/s, including the coding bits. Therefore, the required throughput has been not met due to having rather low operating frequency in the employed Stratix-V FPGA device. However, the required throughput set by the standard can be met by changing the device to one with at least 3.0X higher operating frequency, i.e., ASIC. The lowest achievable clock frequency as mentioned by the standard to define the throughput is not high. Low cost off-the-shelf commercial components can also achieve this.

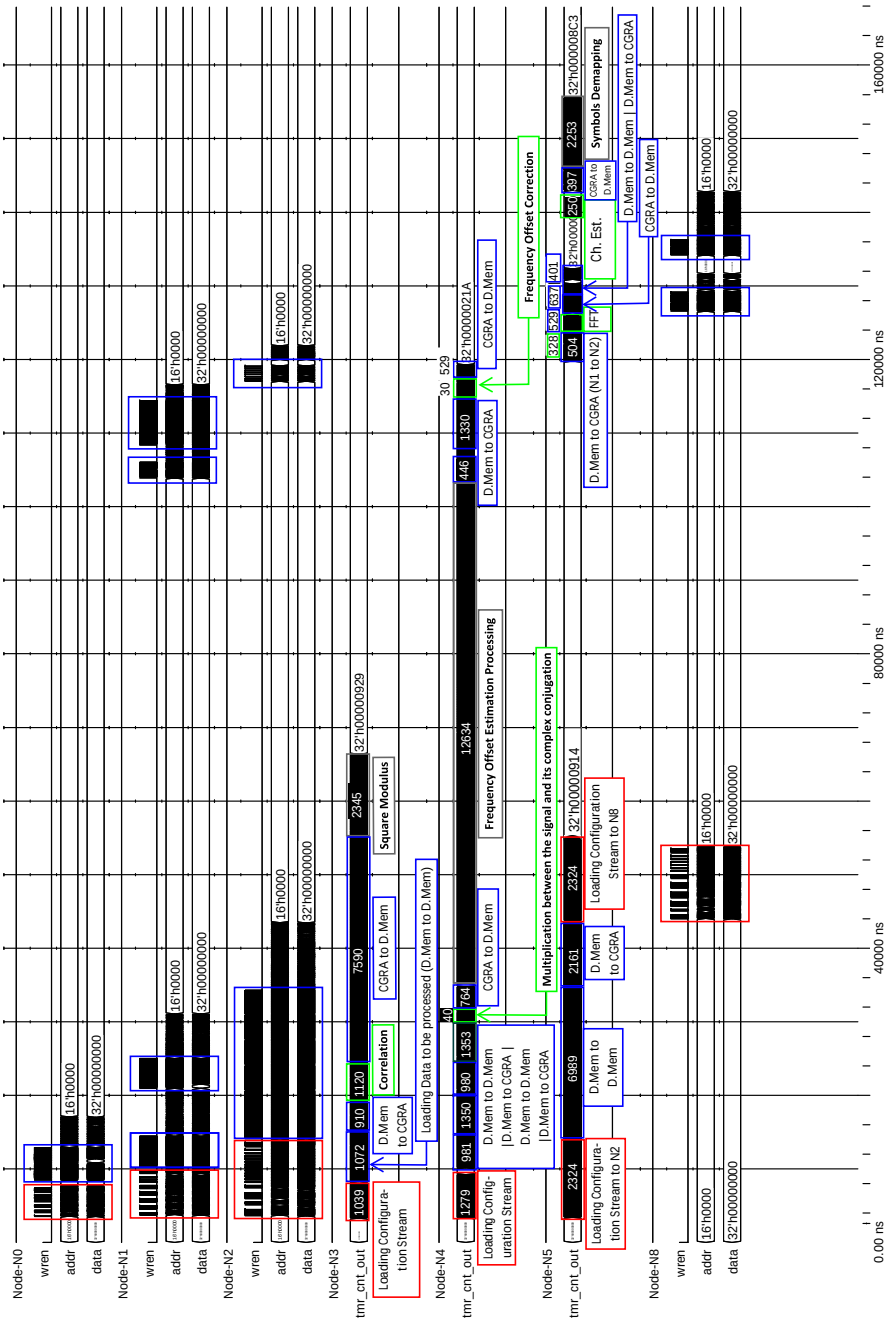

Figure 5.2: The time frame related to the utilization of NoC bandwidth in which configuration words and the data transfers between the data memories of the CGRA nodes are specified with red and blue lines, respectively. The execution of OFDM receiver blocks are also specified with green lines in the case of CGRAs and gray lines in the case of RISC processor software. The signals 'wren', 'addr' and 'tmr_cnt_out' stand for write-enable, address ports of the Data Memory and the counter for enumerating the number of CC, respectively. The numbers inside the digital waveforms of N3, N4 and N5 are extracted from the Table I. [P.IV]

6 Power Mitigation of a HARP on FPGA/ASIC by DFS/DVFS Techniques

This chapter presents an integrated SEEC model in HARP template for mitigating the dynamic power dissipation by employing an OFDM receiver as a test-case. CGRAs are popular for providing high data level parallelism and also due to the level of their granularity. However, they have potentially high power dissipation, which should be mitigated in order to maximize the performance by employing various techniques, i.e., DVFS. In this context, SEEC models are playing an important role in order to create self-adaptive computing systems for the complex extreme-scale computer systems that require maximum performance besides minimum energy consumption. Self-adaptive computing systems are able to change their behavior based on the performance requirements [23]. For this purpose, an advanced FCS technique is exploited in order to monitor the execution-time of the CGRA nodes constantly by the RISC cores and then dynamically scale the operating frequency and the voltage of the CGRA nodes based on the worst execution time to meet the desired performance level. Although FCS technique can be utilized for dynamically scaling both frequency and the voltage, however, voltage supply cannot be scaled on the FPGA-based prototype (implemented in Chapter 5). Therefore, the implementation is also estimated in 28nm UTBB FD-SOI ASIC technology in order to get more benefits in terms of the power mitigation by scaling the voltage in addition to the frequency.

6.1 Equalization of the OFDM Receiver Performance by Frequency Scaling

As it can be observed from the modified version of HARP for performing an OFDM receiver, Fig. 6.1, three RISC cores are instantiated to monitor the performance of the CGRA nodes they are responsible for. In addition to transferring the configuration stream and data to be processed by the RISC cores, they are also responsible for continuously monitoring the performance of each CGRA node, determining the worst-case execution time and then upgrading or downgrading the clock frequencies and supply voltage. As a result, the total power dissipation will be minimized while the overall performance of the system will be maximized. Once the configuration words are transferred by the RISC cores in parallel, the data should be loaded into local memories of No in order to accomplish the TS block of an OFDM receiver. Then the RISC processor starts to count the number of CC during the process of TS by using its special counter. As it is also shown in Table 5.1, total CC required for executing each block of an OFDM receiver is composed of transfers from data memory of RISC core to the data memory of CGRA node, data memory to local memory of the CGRA and vice versa and the

CGRA execution time. Due to having the data dependency among the OFDM receiver blocks, each computing node has to wait until the other one completes its execution. Therefore, synchronization has to be established by the three RISC cores. The CGRA nodes of the platform form a software-defined macro-pipeline.

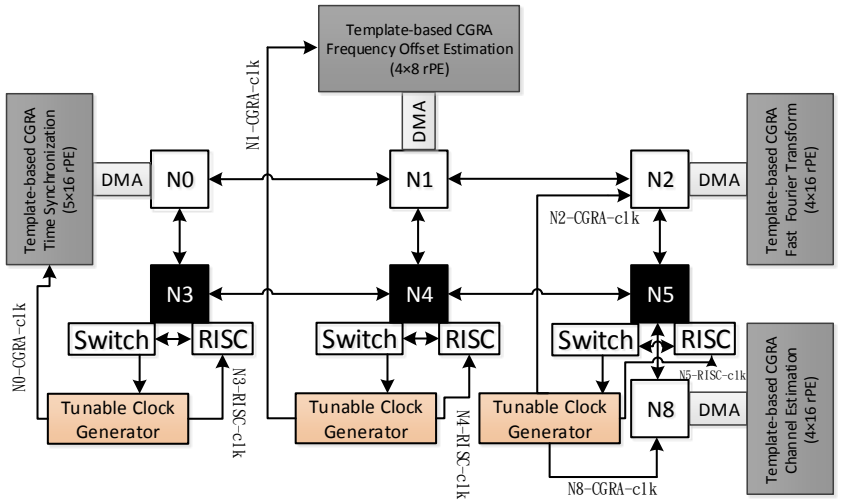

Figure 6.1: A simplified overview of the HARP architecture with three RISCs and four template-based CGRAs in a processor/coprocessor model. [P.V]

By completing the process of TS completely, the counted number of CC should be transmitted to the N4 RISC core and stored at reserved locations in its data memory. The same procedure should be also applied on the other CGRA nodes. Once the first iteration is completed, the data related to the number of CC of each CGRA node should be retrieved in order to recognize the stored most time consuming CGRA node (worst-case). As it is already reported, the total number of CC related to N0, N1, N2 and N8 CGRA nodes are equal to 9,985, 18,039, 1,931 and 1,685 CC, respectively. According to the conducted experiment results, N1 CGRA node, the one responsible for accomplishing FOE block, is the most time consuming and can be selected as the worst-case candidate. Then, N4 RISC core will notify other CGRA nodes regarding the selected worst-case candidate. From the second round of iteration, nodes N3, N4 and N5 RISC cores will tune the operating frequency of the CGRA nodes belonging to them in order to approach the defined equalization region. The FCS technique can be implemented in the processor software to perform dynamically frequency scaling. It can be performed by comparing the counted CC of the CGRA nodes and the selected worst-case execution-time and then reducing each core operating frequency to approach the performance goal. The clock frequencies of the CGRAs can be updated through a module emulating a DVFS Power Management Unit (PMU). This PMU includes a 32-bit general-purpose register, defined as allocated 4-bit field (16 bits in total for four cores) for each one when in DVFS mode (on ASIC), the corresponding supply voltage supporting the clock frequency is also selected. When the frequency is updated the PMU clock gates the part of the systems where the frequency or voltage is changed. This guarantees that the subsystem is not operating during the transitions of frequency and voltage, and that operation is safely restored once the voltage/frequency are stable.

On the FPGA prototype, each RISC core can tune the clock frequency of its associated

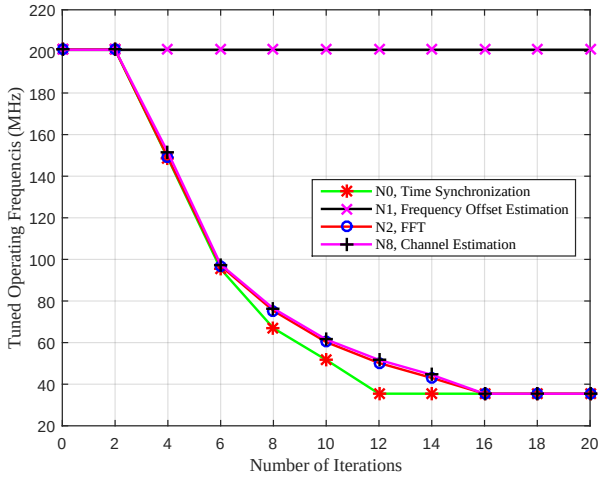

Figure 6.2: Tuning the operating frequencies in the range of ≈35.0-200.0 MHz. [P.V]

CGRA node by using this 4-bit field between 16 different frequencies within the range of 35.0-200.0 MHz, depicted in Fig. 6.2. During each of 20 iterations, the 4-bit field is added or subtracted by "0001", which results to update the current clock frequency of the CGRA node. However, the clock frequency of the RISC cores will remain the same. It can be observed from Fig. 6.2 that during the system start-up time, all the CGRA nodes are executing at the maximum operating frequency of 200.0 MHz. Subsequent to identifying the worst-case candidate, from the second iteration onwards, FCS will automatically update the target and start to tune the clock frequency of other CGRA nodes in order to approach the equalization region. The number of iterations can be specified by the user at desig-time.

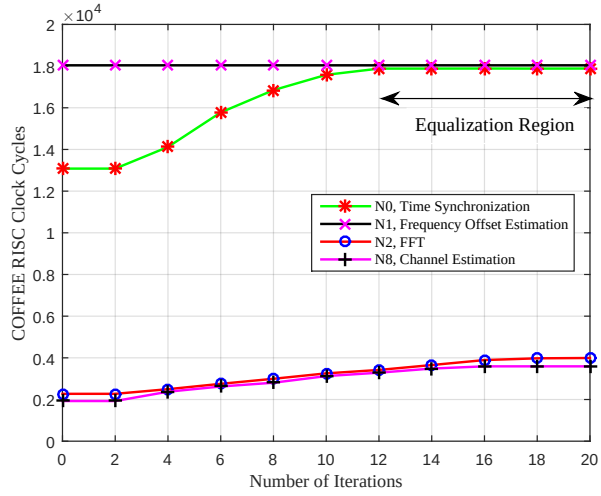

Figure 6.3: Performance Equalization of the CGRAs based on the Worst-Case Execution Time. [P.V]

Fig. 6.3 shows the performance equalization of the CGRA nodes. In the case of No (TS block), it successfully approached the equalization region by reaching it in the 12th iteration while running at 35.0 MHz operating frequency. However, for the other two computing-nodes, N2 (FFT block) and N8 (CE block), their performance could not degraded to the equalization region even with running at 35.0 MHz operating frequency by reaching the 20th iteration. The reason is the smaller workload, lower computation complexity algorithm and relatively shorter execution time.

In the next step, the entire platform was synthesized on ASIC in addition to FPGA in order to dynamically scale both frequency and the voltage within the range of 55.0-500.0 MHz and 0.5-1V, respectively. The dynamic power dissipation of the CGRA nodes are estimated first by applying DFS and keeping the voltage constant at 1V and then, by applying DVFS, which is explained in detail in the next section. Moreover, in order to have a fair comparison between FPGA and ASIC technologies in terms of the power mitigation, the maximum operating frequency of ASIC is reduced from 500.0 MHz to 200.0 MHz as the clock frequency of FPGA.

6.2 Measurements and Estimations

The entire HARP platform with the applied FCS was first synthesized on a Stratix-V (5SGXEA4H1F35C1) 28nm FPGA device for prototyping the concept and then in 28nm UTBB FD-SOI ASIC technology in order to analyze the added benefits of including also voltage scaling. Node-by-node breakdown of resource utilization on FPGA is depicted in Table 5.2. However, subsequent to applying the FCS technique, the logic utilization increased for just around 1%, which is almost negligible.

6.2.1 FPGA Evaluation

Table 6.1: Dynamic power dissipation of each node and the NoC before/after applying FCS on FPGA prototype. [P.V]

Node	Accelerator Type	Dynamic Power (mW) FCS Inactive	Dynamic Power (mW) FCS Active	Savings %
No	Time Synchronization	414.35	284.43	31.35
N1	Frequency Offset Estimation	272.23	272.83	≃0.0
N2	FFT	526.07	391.23	25.63
N3	Integration Logic e Processing, Synchronization, Control	114.47	104.13	9.03
N4		113.82	114.61	≃0.0
N5		114.52	105.07	8.25
N8	Channel Estimation	448.01	344.61	23.07
NoC	-	10.10	14.34	-
Integration Logic	- -	609.07	461.35	24.25
Total	-	2623.72	2092.6	20.24

Subsequent to synthesizing the overall platform on Stratix-V FPGA device by using Quartus II 15.0, two timing models were selected in order to measure the operating frequencies after placement and routing. The timing models used by the Quartus II software could cover worst-case voltage to the minimum and maximum supported Vdd operating conditions for Slow 900mV 85°C and Fast 900mV 0°C, respectively. By using these timing models, the timing of FPGA can be verified without the need to implement physical simulation. In this regard, the maximum achieved operating frequencies for

slow timing model at an operation voltage of 900 mV are equal to 163.61 MHz and 188.29 MHz at temperatures of 85°C and 0°C, respectively. In the case of fast timing model (900 mV), the maximum operating frequencies are equal to 246.12 MHz at 0°C and 223.51 MHz at 85°C. First of all, the clock frequency of 170.0 MHz has been used for running the simulations using the ModelSim simulator and then it is increased up to 200.0 MHz as the average of the achieved operating frequencies without any timing error on the particular FPGA instance used. As it can be observed from Table 6.1, the power measurements are performed in two states: inactive state of FCS with the fixed operating frequency of 200.0 MHz and active state of FCS with the tunable operating frequency within the range of 200.0-35.0 MHz. In the case of inactive state of FCS, the dynamic power dissipation is equal to 2623.72 mW. Once the FCS starts to tune the clock frequency of the computing nodes based on the worst-case execution time, the dynamic power dissipation will also start to get reduction. By completing all the 20 iterations, the total dynamic power dissipation can be decreased up to 20.24%. In the case of FCS active, the estimated static, dynamic, I/O and total power dissipation for the overall platform showed the value of 1121.19 mW, 2092.6 mW, 27.37 mW and 3239.2 mW, respectively. In comparison with the inactive state of FCS (analyzed in Chapter 5), the total power dissipation of the entire platform can be decreased up to 16.8%. Although it is proved that applying the FCS technique can be highly effective in order to reduce the instantaneous dynamic power dissipation and accordingly the heat dissipation and the dark part of the chip, however the energy consumption on the FPGA prototype remained approximately the same. It is due to increasing the active time of the CGRA nodes proportionally as the result of decreasing the clock frequency.

6.2.2 ASIC Evaluation

The entire platform is then synthesized on 28nm UTBB FD-SOI ASIC technology in order to apply DVFS method and get more power and energy reduction. The various blocks of the system were synthesized with Synopsys Design Compiler 2014.09 [94] on a 28nm UTBB FD-SOI RVT standard cell library. The power dissipation was estimated with Synopsys PrimeTime 2013.06 assuming 20% of switching activity. Furthermore, the libraries have been characterized down to 0.5V with 0.1V steps using Cadence Liberate for estimating the power consumption of the blocks at different voltages. The design was synthesized at 1.0V (slow-slow (ss) corner, 125°C), while the signoff was performed at different operating voltages (from 1.0V to 0.5V with steps of 0.1V), in order to provide to the software the knowledge of the maximum operating frequency and power consumption of the system at the different voltage supplies evaluated in this work. As it can be observed from the post-syntesis results depicted in Table 6.2, the maximum achieved operating frequency is 500.0 MHz at nominal voltage in the slow corner (ss, 125°C, 0.9V). The leakage and dynamic power consumption are measured in typical operating conditions (typical-typical (TT), 25°C, 1.0V). Similar to FPGA implementation, the area occupation, leakage and dynamic power consumption are also doubled as the the size of the template-based CGRAs doubled, i.e., N0, N2 and N8 against N1. The overall platform occupies 8.68 mm^2 of the area with the consumption of 0.1424 mW and 218.66 mW as the leakage and dynamic power consumption, respectively.

As the next step, both frequency and the voltage are scaled down simultaneously within five steps while at each step, the dynamic power consumption has been measured. Node-by-node breakdown of dynamic power dissipation affected by applying DVFS method at each stage is depicted in Table 6.3. However, the leakage power is ignored

Table 6.2: Power consumption and area utilization of the nodes synthesized on ASIC at the operating frequency of 500.0 MHz (0.9V, ss, 125°C) and typical conditions (tt, 25°C, 1.0V) for estimating the power numbers. [P.V]

Node	Frequency [MHz]	Area [mm2]	Leakage Power @ 1V [mW]	Dynamic Power [mW] @ 1V, 500 MHz
N0	500.0	1.78	0.0298	51
N1	500.0	0.9	0.0146	24.75
N2	500.0	1.81	0.0292	48.75
N3	500.0	0.79	0.0136	10.91
N4	500.0	0.79	0.0130	15
N5	500.0	0.79	0.0128	19.65
N8	500.0	1.82	0.0294	48.6
Total	500.0	8.68	0.1424	218.66

because of its negligible value. At the first stage, 31.42% power reduction can be obtained by scaling down the operating frequency and the voltage from 500.0 MHz and 1V to 480.0 MHz and 0.9V, respectively. By moving forward in scaling down the operating frequency and the voltage down to 55 MHz and 0.5V, the dynamic power dissipation can be decreased down to 5.23 mW, which is equivalent to 97.61% saving compared to the dynamic power dissipation at 1V and 500.0 MHz.

Table 6.3: Impact of DVFS method on the dynamic power dissipation at scaled down voltages and frequencies on ASIC prototype. DP stands for Dynamic Power. Gains are calculated against dynamic power at 1V and 500.0 MHz. [P.V]

Node	DP [mW] @ 0.9V 480.0 MHz	DP [mW] @ 0.8V 366.0 MHz	DP [mW] @ 0.7V 238.0 MHz	DP [mW] @ 0.6V 119.0 MHz	DP [mW] @ 0.5V 55.0 MHz
N0	33.58	19.15	9.65	3.5	1.17
N1	16.3	9.29	4.68	1.7	0.57
N2	32.1	18.3	9.22	3.34	1.12
N3	7.18	4.09	2.06	0.75	0.25
N4	9.88	5.63	2.84	1.03	0.35
N5	12.94	7.38	3.72	1.35	0.45
N8	32	18.25	9.2	3.34	1.12
Total	149.96	85.51	43.09	15.62	5.23
Gain %	31.42	60.89	80.29	92.86	97.61

6.2.3 Mixed Comparison Between FPGA and ASIC Technologies

In order to have a fair comparison against FPGA implementation, the maximum operating frequency should be kept at the fixed value of 200.0 MHz. Then the dynamic power dissipation is estimated in three states on ASIC: inactive state of FCS, active state of FCS with DFS and active state of FCS with DVFS. As it is depicted in Table 6.4, the dynamic power dissipation of FOE lock, N1, as the worst-case candidate is constant in all states. Regarding the rest of the nodes, subsequent to tuning the operating frequency of the cores by FCS, the dynamic power dissipation of the entire platform is reduced by 64.29%. Therefore, the equalization region specified by FCS can be approached successfully by all the cores, which are running at 55.0 MHz operating frequency. In the case of DVFS where the supply voltages are also scaled down to 0.5V in parallel with the frequency scaling, the total dynamic power dissipation can be mitigated by 82.98% in comparison with FCS inactive state.

Table 6.4: Dynamic power dissipation of each node before applying FCS, after applying FCS with dynamic frequency scaling (DFS) and also with DVFS on ASIC. [P.V]

Node	Dynamic Power [mW] FCS Inactive 200 MHz	Dynamic Power [mW] FCS Active with DFS @ 1V	Dynamic Power [mW] FCS Active with DVFS
N0	20.4	5.61	1.7
N1	9.9	9.9	9.9
N2	19.5	5.36	1.12
N3	4.36	1.2	0.25
N4	6	1.65	0.35
N5	7.86	2.16	0.45
N8	19.44	5.35	1.12
Total	87.46	31.23	14.89

Table 6.5 shows the comparison between the OFDM receiver implementation on FPGA and ASIC technologies at the condition of active state of FCS with the applied DFS technique. The maximum operating frequency for running the whole platform in both technologies is 200.0 MHz. The conducted experiment results showed the significant dynamic power mitigation for performing an OFDM receiver on ASIC against the FPGA. Figure 6.4 also shows the general comparison as a visual summary of total dynamic power dissipation of FPGA inactive FCS, FPGA active FCS with DFS, ASIC inactive FCS, ASIC active FCS with DFS and ASIC active FCS with DVFS. It can be observed that by applying FCS and DFS on FPGA, 1.25X gain can be achieved in terms of dynamic power dissipation reduction. In the case of ASIC implementation, the amount of obtained dynamic power saving with DFS is 7X, which can be further reduced by 5.97X with DVFS. This approach paves the way for self-aware systems for energy efficient OFDM receiver, mapped on HARP, by mitigating the signal transition activity over the entire platform with reference to the worst-case performing core.

Table 6.5: General comparison of the impact of DFS technique on FPGA and ASIC dynamic power dissipation at the same operating condition. [P.V]

Node	Dynamic Power [mW] FPGA FCS Active with DFS	Dynamic Power [mW] ASIC FCS Active with DFS @ 1V
N0	284.43	5.61
N1	272.83	9.9
N2	391.23	5.36
N3	104.13	1.2
N4	114.61	1.65
N5	105.07	2.16
N8	344.61	5.35
Total	2092.6	31.23

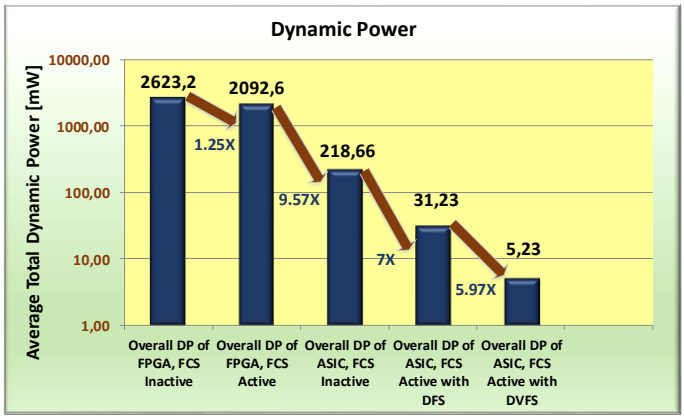

Figure 6.4: General Comparison of Total Dynamic Power of FPGA Inactive FCS, FPGA Active FCS, ASIC Inactive FCS, ASIC Active FCS with DFS and with DVFS. DP stands for Dynamic Power. [P.V]

7 HW/SW Co-design of an OFDM Receiver on Xilinx Zynq SoC using HLS

In this chapter, a novel HW/SW co-design of an OFDM receiver using Xilinx Software Defined System-on-Chip (SDSoC) as the High-Level Synthesis (HLS) tool is presented. The ZYNQ SoCs are composed of an ARM processor besides a FPGA in order to improve the power efficiency and the system performance. On a single chip, heterogeneous MPSoC can be created by employing FPGA in order to integrate numerous PEs. However, designing application-specific accelerators and programming of FPGAs by using Hardware Description Language (HDL) are expensive, time-consuming and complicated. Therefore, HLS tools such as VivadoHLS [95] and SDSoC (Software-Defined SoC) [96] have been introduced by Xilinx. Currently, the development of heterogeneous embedded systems on the Zynq MPSoC/SoC platform by using an embedded C/C++ application programming interface provided by SDSoC becomes much easier for application developer than writing HDL. It contains a C/C++ compiler in which the designer can select each thread of an application to be compiled by a processor in SW or HW. As it is explained in Chapter 4, the OFDM receiver is composed of parallel and serial nature algorithms, which require HW/SW co-design to fulfill system requirements. The blocks of an OFDM receiver are written in C/C++ code to be realizable on the ARM processor or on the FPGA. The parallel tasks can be created as accelerators based on the HW functions written in C/C++ code. The accelerators can be implemented into the HW by compiling the written HW functions. The HW functions can also be optimized by employing pragmas to be efficiently performed in the FPGA.

7.1 Implementation of an OFDM receiver on the ZC706 Evaluation Board

In this section, the implementation of an OFDM receiver on ZC706 evaluation board (Xilinx Zynq SoC) by using the HLS tool as an application for a Linux host is presented. As the first stage, the blocks of OFDM receiver are written in C++ code based on the explained algorithms in Chapter 4. The general view of the overall platform for OFDM receiver on ZC706 evaluation board is shown in Fig. 7.1.

The blocks of an OFDM receiver can be created as HW functions (application-specific accelerators) then connected to the Processing System (PS) through a data motion network. As it can be observed from the figure, the order of executing the OFDM receiver blocks as the HW accelerators and exchanging the data symbols between the different accelerators are specified with the dashed arrows and numbers. Fig. 7.2 depicts the HW/SW connectivity for transferring the data to be processed from the SW to the

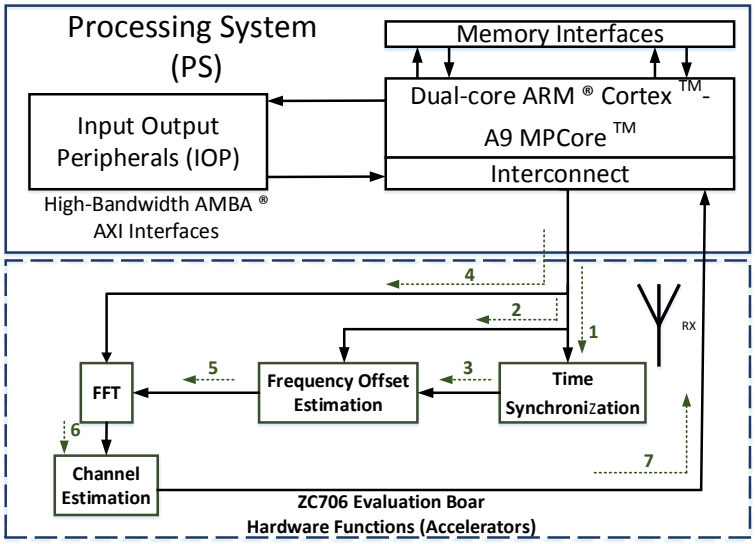

Figure 7.1: An overview of an OFDM receiver implementation on ZC706 evaluation board. [P.III]

HW or vice-versa. Moreover, the required CC for calling the HW functions, setting up the Data Mover (DM), transferring the data and executing the application-specific accelerators during the processing of the OFDM receiver are specified in the timeline.

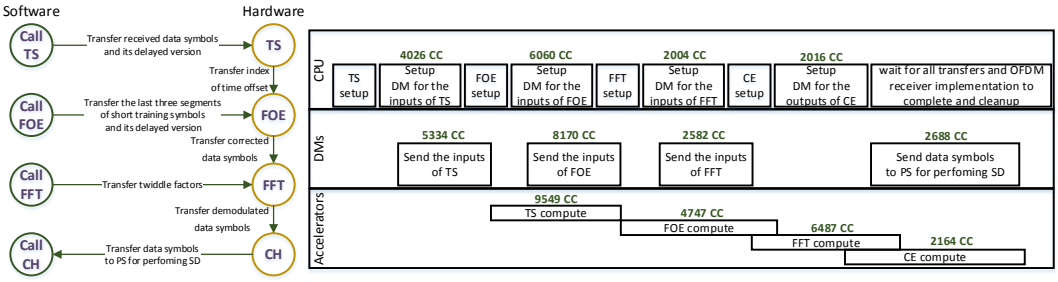

Figure 7.2: HW/SW connectivity and the timeline for seting up DM and calling HW function. [P.III]

In order to transfer the received data symbols and training symbols from the ARM processor to the FPGA HW, various DMA configurations provided by SDSoC can be employed. In this case-study, `axi_dma_sg` (scatter-gather DMA) data mover for array arguments is selected. In the SDSoC environment, the data transfer network is composed of HW interface for the application-specific accelerators and DM among the PS and the FPGA through AXI ports. As it can be observed from Fig. 7.2, the ARM processor (CPU) is responsible for establishing the HW functions and setting up DM for each function call. Subsequent to completing the function call and the data transfer, the loaded data as the inputs of HW functions will become available. Then according to the algorithm scenario monitored by the SW and written by the application developer at design-time, the accelerators can start their actual processing. Furthermore, it is obvious from Fig. 7.2 that in the case of several sets of data, setting up DM for each new array of data can be executed in parallel with the data transfer of the previous one. For instance,

setting up DM for the first TS's array of data with the size of 80 takes 1005 CC, which in the following, can be forwarded to the HW function in 1317 CC. At the same time of transmitting the data to the HW function, DM for the second TS's array of data can be established. Therefore, the whole process of transferring data from the CPU to the FPGA HW functions can be calculated as the following

$$6642CC = 5334CC + (5334CC - 4026CC) \tag{7.1}$$

where it takes 6642 CC in total. Subsequent to transferring the received data symbols from the CPU to the FPGA Time Synchronization accelerator (dashed arrow number 1), the correlation algorithm, looking for the index of the time offset and specifying the edge of FFT window can be implemented in 9549 CC. According to Eq. 4.1 and Eq. 4.2, the simplified algorithm for executing the Time Synchronization block by using CP correlation based method in SDSoC development environment is written as following pseudo code.

Algorithm 2 CP correlation based method for TS. [P.III]

```
1:  Initialize Location to 0, and Max to −1
2:  for i:=1 to 80 step 1 do
3:      #pragma HLS PIPELINE enable_flush rewind off
4:      for j:=1 to 80 step 1 do
5:          #pragma HLS PIPELINE II=1
6:          #pragma HLS unroll factor=4
7:          c ← y(j)y*_D(j + i) + c
8:      z = real(c) ∗ real(c) + imag(c) ∗ imag(c)
9:      if (z > Max) then
10:         Max = SM
11:         TimeOffsetIndex = i
```

As it can be seen from the highlighted parts, loop pipelining and loop unrolling are employed in order to have efficient implementation. By using the `pragma` *pipeline* and transforming the sequential execution of operations into parallel, the performance of the HW function can be improved by performing the loop in a concurrent manner. As the result, the clock cycles for performing all operations are reduced. However, the `pragma` *pipeline* is limited in terms of efficiency mainly by data dependency. By using the `pragma` *loop unrolling*, the performance can be improved by exploiting the parallelism among loop iterations. The `pragma` *loop unrolling* creates copies of the loop body, which are implemented concurrently. The number of created copies can be determined by the designer at design-time by inserting an unroll factor. The data dependency and available HW resources affect on the number of manufacturable copies. As it can be observed from the pseudo code, HLS unroll factor 4 is selected by the author as the most efficient factor. Moreover, due to the large size of the whole design, it was not synthesizable for the larger unroll factors.

Subsequent to performing TS block, the short training symbols along with the index of time offset are transferred from the ARM processor to the FPGA HW accelerator for executing FOE, specified with the dashed arrows number 2 and 3, respectively. The data transfer including setting up DM for the input of FOW and the process of this block can be performed in 10280 CC and 4747 CC, respectively. In parallel with the computation

of FOE, the twiddle factors that are required for performing the FFT block can be loaded from CPU to the HW accelerator in 7168 CC (dashed arrow number 4). The process of 64-point radix-4 FFT can be completed in 6487 CC in which the data symbols are fetched directly from the outputs of FOE accelerator (dashed arrow number 5). The last HW accelerator belongs to CE block. By completing the implementation of FFT, the data symbols should be transferred to the CE accelerator (dashed arrow number 6) in order to estimate the channel frequency response and accomplish the channel equalization in 2164 CC. Finally, the equalized data symbols should be transferred back to PS in 2688 CC (dashed arrow number 7) for performing symbols demapping block where the data symbols will be converted to data bits by employing the hard decision method. Symbols demapping block can be executed by processor in SW without requiring any HW accelerator.

7.2 Experimental Results, Comparison and Discussion

Table 7.1 depicts the required CC for performing each block of an OFDM receiver using FPGA HW accelerator or ARM processor SW. Furthermore, the achieved clock cycles are also compared against the HARP platform in Table 7.2. Due to having different clock speeds for running two different platforms, the value of speed-up for each accelerator is computed based on the ratio of the old execution time (the product of CC and the clock period) to the new execution time for a system. The maximum operating frequency of SDSoC is equal to 667.0 MHz while HARP is running at 200.0 MHz. According to the calculated speed-up of the SDSoC HW accelerators against HARP, the obtained speed-up from the implementation of TS, FOE and SD on the ZC706 evaluation board are 1.21X, 8.92X and 4.03X, respectively. In the case of FFT and CE, since they are algorithms of parallel nature, HARP shows better performance due to using template-based CGRAs, specialized for parallel tasks. Although the FPGA is also a worthy candidate platform for accomplishing parallel tasks, however, by the use of template-based CGRAs, the application developers could have near-optimal solutions since the interconnections among the PEs can be done manually. On the other hand, most of the CGRAs suffer from a fixed set of PEs and interconnections, which makes them dependent on various Design Space Exploration (DSE) techniques in order to improve them in terms of cost and performance [97]. Most of the DSE techniques are concerned with the interconnection between PEs and the configuration of their internal structure. Employing DSE techniques by the designer ensures the design of near-optimal applications in terms of cost and performance. Furthermore, the level of granularity is different among the FPGA (an example of fine-grained) and the CGRAs. It should also be noticed that the structure of FPGAs are based on LUTs not PEs, which resulted to accept any bit size. In order to execute an operation, coarse-grained devices require less PEs and consequently, lower resource utilization in comparison with the fine-grained devices. However, the flexibility of fine-grained devices is better than coarse-grained devices for executing new operations. Furthermore, coarse-grained devices have symmetry (arrays of PEs) in their structure. It results to high level of throughput and data parallelism. On the other hand, they are expensive and have high transient power dissipation because of occupying an area of a few million gates on a single chip. In this case-study, TS, FFT and CE blocks of an OFDM receiver are parallel in nature algorithms. Although it would be difficult to have a direct comparison between FPGA or CGRA in terms of the performance of the above-mentioned parallel algorithms, however, the parallel algorithms may have better performance on FPGA or

CGRA based on the amount of computations required by each of them. In the case of SD block as a serial in nature algorithm, it is performed by the processor software in both cases due to having faster execution than HW. As it can be observed from Table 7.2, SDSoC HW accelerators have better performance in terms of CC for parallel in nature algorithms than SDSoC SW. For the whole OFDM receiver, SDSoC HW showed speed-up of 1.75X against SDSoC SW. Furthermore, SDSoC HW showed also speed-up of 3.94X in comparison with the HARP due to having higher operating frequency and different NoC used for transferring the data.

Table 7.1: CC required for processing in two platforms at different stages. Moreover, * and ** stand for the algorithms that have been implemented by SW instead of HW and data transfer from HW to SW, respectively. [P.III]

	HARP (CC) at 200 MHz			SDSoC (CC) at 667 MHz				
Acc	Data transfer	HW	Execution Time (μs)	SW	Execution Time (μs)	Data Transfer	HW	Execution Time (μs)
TS	9,619	3,465	17.33	14,082	21.11	6,642	9,549	14.32
FOE	4,664	12,708	63.54	14,366	21.53	10,280	4,747	7.12
FFT	2,677	328	1.64	9,282	13.91	3,160	2,503	3.75
CE	1,435	250	1.25	5,276	7.91	2,016**	2,164	4.74
SD	-	2,253*	16.27	3,724	5.58	-	-	-
OFDM	-	31,521	157.6	46,730	70.1	-	26,677	40

Table 7.2: Performance comparison between SDSoC HW against HARP and SDSoC SW. * stand for the algorithms that have been implemented by SW instead of HW. [P.III]

Acc	Gain SDSoC HW vs HARP	Gain SDSoC HW vs SW
TS	1.21X	1.69X
FOE	8.92X	3.16X
FFT	0.44X	3.71X
CE	0.38X	4.86X
SD	4.03X*	-
OFDM	3.94X	1.75X

Node-by-node breakdown of resource utilization of the designed hardware accelerator on Zynq®-7000 all programmable SoC ZC706 evaluation kit is depicted in Table 7.3. Furthermore, the rightmost columns of the table present the comparison of resource utilization between HARP and ZC706 evaluation board. Regarding the SD block, it is implemented by processor software and thus, there is not any employed 48-bit DSP in ZC706 evaluation board or ALMs and 18-bit DSP in HARP for that. In order to have fare comparison between HARP and ZC706 evaluation board, it has to be considered that each ALM in Stratix-V FPGA device consists of two LUTs. As it is described in Chapter 5, the number of employed 18-bit DSP resources in HARP depends on the number of 32-bit multipliers instantiated. However, against 18-bit DSP resources in HARP, which are only used for multiplication, 48-bit DSP in ZC706 evaluation board is an arithmetic logic unit and consists of an add/subtract unit and a multiplier connected to a final add/subtract/accumulate engine and can be used only by accelerators (hardware functions). Accordingly, a direct comparison between the amount of utilized DSP resources by two different technologies is difficult and the results are only indicative.

The platform's total power dissipation is estimated by using Xilinx Power Estimator (XPE). According to the obtained results for an OFDM receiver on the ZC706 evaluation board, the tool showed 3.171 W for total on-chip power at an ambient temperature of 25°C. The total power dissipation is composed of static, Processor System (PS) +

Table 7.3: Synthesis results of the proposed accelerators on ZC706 evaluation board and also resource utilization summary of HARP (Stratix-V (5SGXEA4H1F35C1) FPGA device) against ZC706. Acc stands for Accelerator. [P.III]

Acc	Resource Utilization					
	ZC706				HARP	
	BRAM	FF	LUT	48-bit DSP	ALMs	18-bit DSP
TS	0	1,019	563	34	22,616	40
FOE	0	41,506	47,838	459	8,171	32
CE	24	29.664	5,143	153	22,809	56
FFT	4	26,900	24,894	103	25,908	66
SD	0	1,875	9,101	0	0	0
Total	28	100,964	117,539	749	98,729	230
%	(5.14)	(23.09)	(53.77)	(83.22)	(62)	(90)

dynamic and I/O power with the values of 0.243 W, 2.18 W and 0.748 W, respectively. In comparison with HARP platform with the total power dissipation of 3.9 W, Zynq®-7000 all programmable SoC ZC706 evaluation board showed a gain of 1.22X. However, since HARP and Xilinx Zynq SoC are running at two different clock frequencies on the different FPGA platforms, the comparison is not a completely fair and the result is only indicative. The author expects from HARP to achieve a higher operating frequency on the ZC706 board. Even if HARP runs at the same clock frequency as the OFDM receiver design implemented on ZC706, there would be a slight gain in speed over the HARP.

8 Design and Implementation of Multi-Purpose DCT/DST-Specific Accelerator on HARP

In this chapter, the efficient design and implementation of 4/8/16/32-point DCT and 4-point DST by using template-based CGRAs on HARP is presented. Designed template-based CGRAs for performing various sizes of DCT and DST are crafted according to the algebraic equations at different dimensions by employing most of the PEs at each context. The final architecture as a multi-purpose DCT/DST-specific accelerator is composed of five template-based CGRAs arranged over a NoC structure along with three RISC cores. Subsequent to integrating the CGRA nodes over HARP, the performance of each DCT/DST-specific accelerator separately along with the performance of the entire platform are recorded in terms of several high-level performance metrics. In the following sections, a concise explanation of 4/8/16/32-point DCT and 4-point DST algorithms is presented. As it is already described in Chapter 2, the HEVC standard supports 2D DCT/DST with the above-mentioned sizes. However, in order to prevent significant complexity issues of 2D transforms, 1D implementations are considered instead of 2D in which N-point 1D transforms should be applied to every row and column of the residual block separately. Furthermore, these transforms are designed to be finite precision approximations of DCT/DST without considering the transform size. The reason behind using finite precision approximations is to prevent encoder/decoder mismatch caused by different implementations, e.g., by different manufacturers. The process of finite precision approximations is composed of proper rounding and scaling operations for obtaining efficient compression performance, which ensures the interoperability of such transforms. More details about the investigations and experiments for the HEVC transforms are described in [98]. The signal flow-graph for 4/8/16/32-point DCT is depicted in Fig. 8.1 where the outputs are reordered in four different sections for four sizes of DCT [99]. As it can be seen from the figure, by increasing the number of inputs, which is always the power of two, a new set of cosine butterflies and a new set of odd transform coefficients are added. The coefficients of Fig. 8.1 are not normalized, which can be performed at design-time by multiplying by the factor of $\sqrt{2/N}$. The general equations for a 1D DCT and 1D DST can be expressed as following where $n \in (0, 1, ..., N-1)$ and k is the number of row or column.

$$1D\ \textbf{DCT}: \begin{cases} Y_k = \sqrt{\frac{2}{N}}\alpha_k \sum_{n=0}^{N-1} X_k \textbf{cos}(\frac{\pi(2n+1)k}{2N}) \\ k = 0, 1, ..., N-1 \ \textit{where}\ \alpha_k = \begin{cases} \frac{1}{\sqrt{2}}\ K = 0 \\ 1\ \textit{otherwise} \end{cases} \end{cases} \tag{8.1}$$

$$1D\ \mathbf{DST}: \begin{cases} Y_k = \sqrt{\frac{2}{N}}\beta_k \sum_{n=0}^{N-1} X_k \mathbf{sin}(\frac{\pi(2n+1)k}{2N}) \\ k = 0, 1, ..., N-1\ where\ \beta_k = \begin{cases} \frac{1}{\sqrt{2}}\ K = 0 \\ 1\ otherwise \end{cases} \end{cases} \tag{8.2}$$

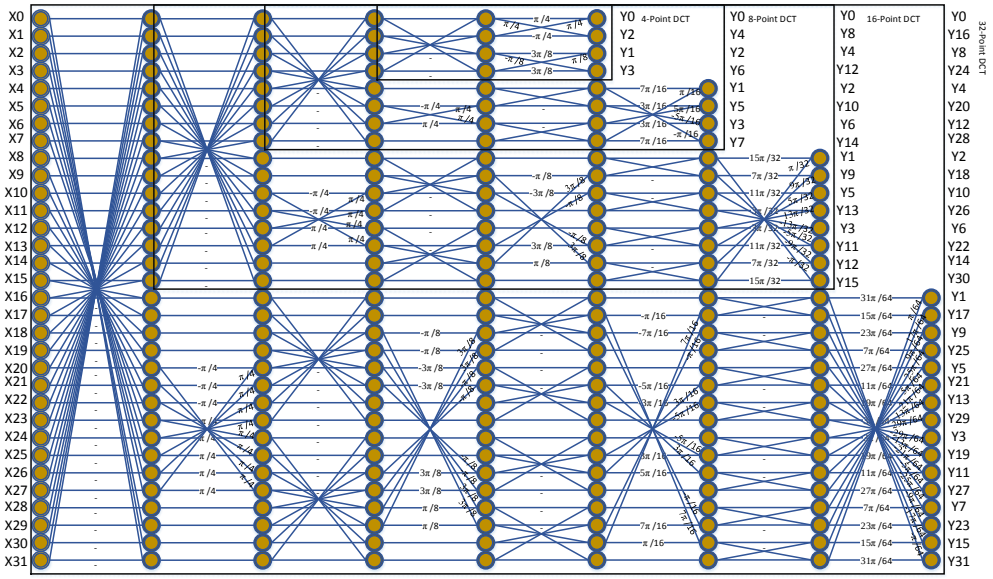

Figure 8.1: DCT Flow Graph for 4/8/16/32-point.

8.1 4-Point DCT

The 4-point DCT can be achieved from Eq. 8.3 and Eq. 8.4, which is equivalent to the expansion of the shown butterfly. In general the transform matrix coefficients for 1D 4-point DCT are $a = 0.5cos(\frac{\pi}{4})$, $b = 0.5cos(\frac{\pi}{8})$, $c = 0.5cos(\frac{3\pi}{8})$. However, the values of the matrix were found by [98] to be as follows; $a = 64, b = 83$ and $c = 36$, which are the most efficient transform matrix coefficients. In the following sections, the similar things are also applied for 8/16/32-point transform matrix coefficients by referring to them with these (a,b,c, ...) parameters while the details of the calculation of all the transform coefficients are not in the scope of this thesis.

$$DCT_{TCoeff}\ 4 \times 4 = \begin{bmatrix} a & a & a & a \\ b & c & -c & -b \\ a & -a & -a & a \\ c & -b & b & -c \end{bmatrix} \tag{8.3}$$

According to the above-mentioned transform matrix coefficients, the equations related to four outputs (Y_n) of 1D 4-point DCT can be written as following where X_n is the

input.

$$\begin{aligned} Y_0 &= a(X_0 + X_1 + X_2 + X_3) \\ Y_1 &= b(X_0 - X_3) + c(X_1 - X_2) \\ Y_2 &= a(X_0 + X_3 - (X_1 + X_2)) \\ Y_3 &= c(X_0 - X_3) - b(X_1 - X_2) \end{aligned} \tag{8.4}$$

The CREMA-generated accelerator for performing the above equations is shown in Fig. 8.2 where 6 additions, 6 subtractions and 6 multiplications are employed in order to accomplish 4-point 1D DCT in just one context.

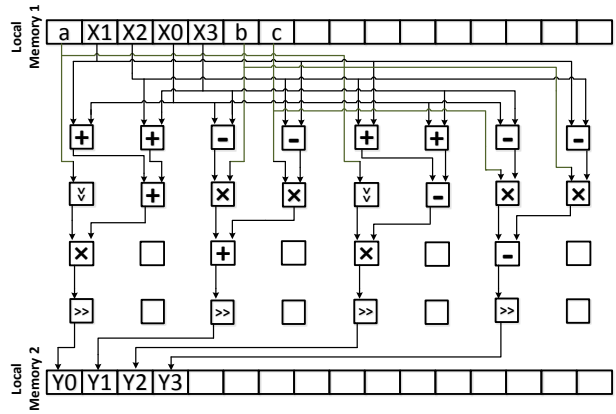

Figure 8.2: Context for the Calculation of 1D DCT 4-point. [P.VI]

8.2 8-Point DCT

In the case of 8-point DCT, the coefficient transform matrix can be decomposed into two matrices in which the computation of the even indexed coefficients are separated from the computation of the odd indexed coefficients due to the coefficient symmetry [99]. Although the values of the parameters according to Eq. 8.1 and the signal flow are $a = 0.5cos(\frac{\pi}{16})$, $b = 0.5cos(\frac{\pi}{8})$, $c = 0.5cos(\frac{3\pi}{16})$, $d = 0.5cos(\frac{\pi}{4})$, $e = 0.5cos(\frac{5\pi}{16})$, $f = 0.5cos(\frac{3\pi}{8})$ and $g = 0.5cos(\frac{73\pi}{16})$, however, they are recalculated by [98] as the most efficient transform matrix coefficients. Therefore, the values of parameters in this manuscript are as following; $a = 64, b = 83, c = 36, d = 89, e = 75, f = 50, g = 18$.

$$\begin{bmatrix} Y_0 \\ Y_2 \\ Y_4 \\ Y_6 \end{bmatrix} = \begin{bmatrix} d & d & d & d \\ b & f & -f & -b \\ d & -d & -d & d \\ f & -b & b & -f \end{bmatrix} \begin{bmatrix} X0 + X7 \\ X1 + X6 \\ X2 + X5 \\ X3 + X4 \end{bmatrix} \tag{8.5}$$

$$\begin{bmatrix} Y_1 \\ Y_3 \\ Y_5 \\ Y_7 \end{bmatrix} = \begin{bmatrix} a & c & e & g \\ c & -g & -a & -e \\ e & -a & -g & c \\ g & -e & c & -a \end{bmatrix} \begin{bmatrix} X0 - X7 \\ X1 - X6 \\ X2 - X5 \\ X3 - X4 \end{bmatrix} \tag{8.6}$$

According to the above-mentioned transform matrices coefficients, the equations related to eight outputs (Y_n) of 1D 8-point DCT can be expressed as Eq. 8.7 where they are also

simplified to be mapped on the template-based CGRA. The operations are reported in the same order as the butterfly develops Fig. 8.1, starting from the input coefficients to arrive at the output coefficients.

$$
\begin{aligned}
Y_0 &= d(\underbrace{(X_0 + X_3 + X_4 + X_7)}_{Z0}) + \underbrace{(X_1 + X_2 + X_5 + X_6)}_{Z4} \\
Y_1 &= \underbrace{a(X_0 - X_7)}_{Z10} + \underbrace{c(X_1 - X_6)}_{Z11} + \underbrace{e(X_2 - X_5)}_{Z12} + \underbrace{g(X_3 - X_4)}_{Z13} \\
Y_2 &= f(\underbrace{(X_1 + X_6 - (X_2 + X_5))}_{Z2}) + b(\underbrace{(X_0 + X_7 - (X_3 + X_4))}_{Z6}) \\
Y_3 &= \underbrace{c(X_0 - X_7)}_{Z30} - \underbrace{g(X_1 - X_6)}_{Z31} - \underbrace{a(X_2 - X_5)}_{Z32} - \underbrace{e(X_3 - X_4)}_{Z33} \\
Y_4 &= d(\underbrace{(X_0 + X_3 + X_4 + X_7)}_{Z0}) - \underbrace{(X_1 + X_2 + X_5 + X_6)}_{Z4} \\
Y_5 &= \underbrace{e(X_0 - X_7)}_{Z50} - \underbrace{a(X_1 - X_6)}_{Z51} + \underbrace{g(X_2 - X_5)}_{Z52} + \underbrace{c(X_3 - X_4)}_{Z53} \\
Y_6 &= f(\underbrace{(X_0 + X_7 - (X_3 + X_4))}_{Z6}) - b(\underbrace{(X_1 + X_6 - (X_2 + X_5))}_{Z2}) \\
Y_7 &= \underbrace{g(X_0 - X_7)}_{Z70} - \underbrace{e(X_1 - X_6)}_{Z71} + \underbrace{c(X_2 - X_5)}_{Z72} - \underbrace{a(X_3 - X_4)}_{Z73}
\end{aligned}
\tag{8.7}
$$

The AVATAR-generated accelerator for performing the above equations is shown in Fig. 8.3 where 16 additions, 7 subtractions and 10 multiplications are employed in order to accomplish 8-point 1D DCT in two contexts. The left side of the contexts are allocated to calculate the even indexed outputs while the right side is responsible for computing the odd indexed outputs.

8.3 16/32-Point DCT

As it can be observed from the second context of Fig. 8.4, the outputs Y0, Y4, Y8 and Y12 can be produced while switching the context, the new set of data can be loaded into the local memory. In order to produce the outputs Y2, Y6, Y10 and Y14, the third context and a part of the fourth context specified by dashed border line have to be employed. The remaining outputs of 16-point DCT can also be produced by using the fourth context twice (second time just the dashed border part).

As it can be observed from Fig. 8.5, Fig. 8.6 and Fig. 8.7, seven contexts are required in total in order to accomplish 32-point DCT. The second context is the same as the second context of 16-point DCT while the outputs Y0, Y8, Y16 and Y24 will be produced. The third context is designed to prepare the data for the fourth context. The fourth context is designed as a multi-purpose context, which is partitioned into four different segments, each can be employed for various purposes. The first segment can produce the outputs Y4, Y12, Y20 and Y28 from the received data of third context. The second segment is designed to receive the data from the fifth context, process the data, forward them to the third segment and finally, produce the outputs Y2, Y6, Y10, Y14, Y18, Y22, Y26 and Y30. The fourth segment will also receive data from the seventh context and produce the last set of 32-point DCT outputs.

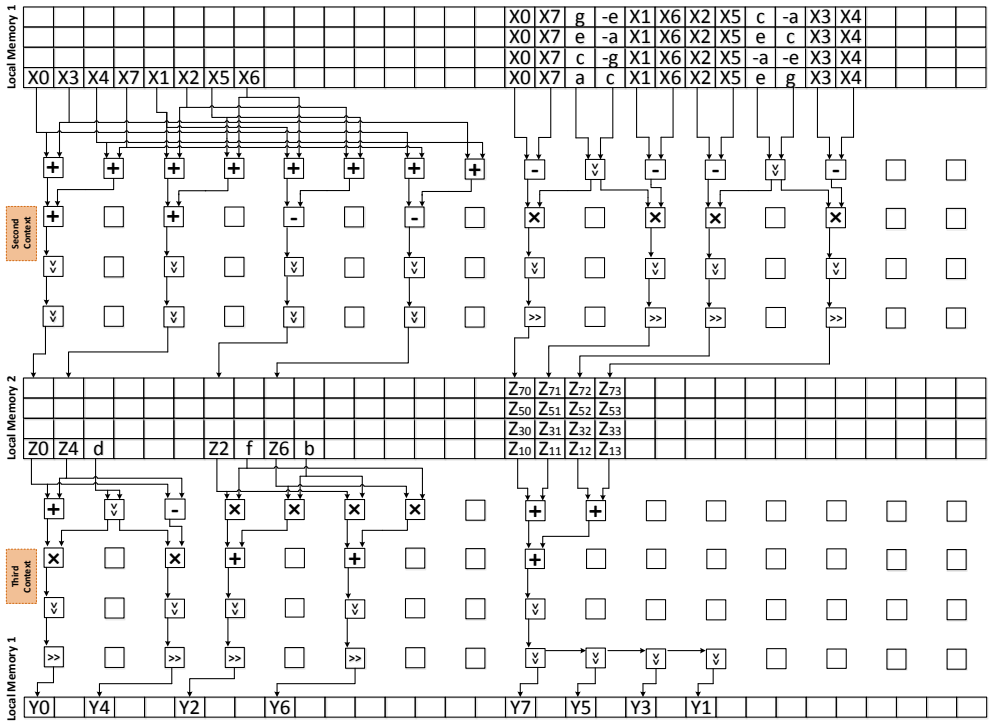

Figure 8.3: Second & Third Contexts for the Calculation of 1D DCT 8-point. [P.VI]

8.4 4-Point DST

Similar to 4-point DCT, the most efficient transform coefficient matrix for 4-point DST has also been found as written in Eq. 8.8 in which the values of parameters a, b, c and d are equal to 29, 55, 74 and 84, respectively.

$$DST_{TCoeff}\ 4 \times 4 = \begin{bmatrix} a & b & c & d \\ c & c & 0 & -c \\ d & -a & -c & b \\ b & -d & c & -a \end{bmatrix} \tag{8.8}$$

Then Eq. 8.9 can be expressed based on the above matrix. The CREMA-generated accelerator for performing 4-point 1D DST is shown in Fig. 8.2 where 5 additions, 1 subtraction and 8 multiplications are employed in a context.

$$\begin{aligned} Y_0 &= a \times X_0 + b \times X_1 + c \times X_2 + d \times X_3 \\ Y_1 &= c \times X_0 + c \times X_1 - c \times X_3 \\ Y_2 &= d \times X_0 - a \times X_1 - c \times X_2 + b \times X_3 \\ Y_3 &= b \times X_0 - d \times X_1 + c \times X_2 - a \times X_3 \end{aligned} \tag{8.9}$$

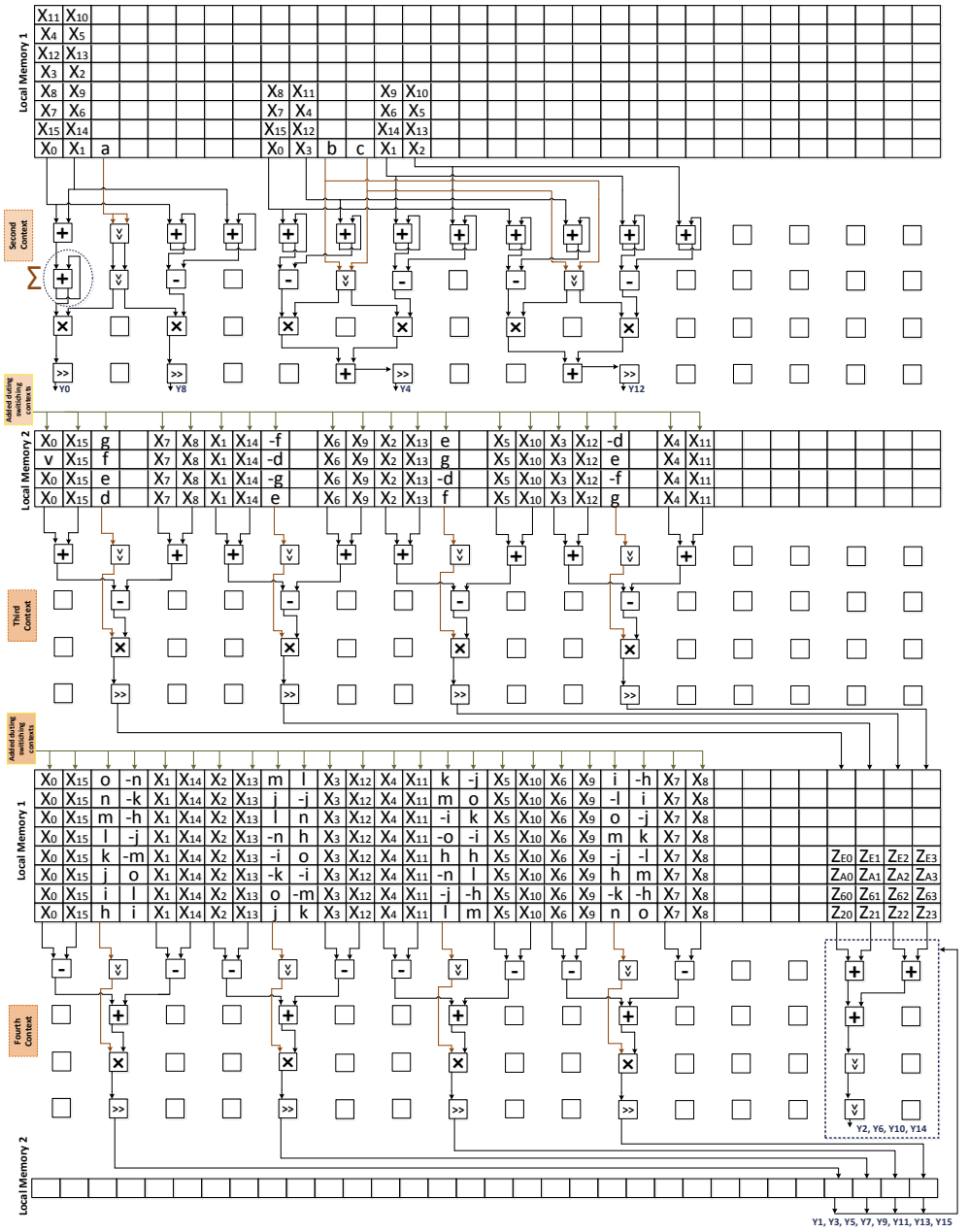

Figure 8.4: Second, Third & Fourth Contexts for the Calculation of 1D DCT 16-point.

8.5 Implementation of Multi-Purpose DCT/DST-Specific Accelerator on HARP

Subsequent to crafting DCT/DST-specific accelerators by using the template-based CGRAs at different dimensions in the most optimum way due to employing most of the PEs at each context, they are integrated over HARP as the slave nodes. As it is

shown in the overall processing architecture for DCT/DST, Fig. 8.9, N3 RISC core is responsible for transferring the configuration stream and data to be processed to No and N6 CGRAs, N4 RISC is responsible for N1 CGRA and N5 RISC is responsible for N2 and N8 CGRAs. This level of exchanging data between the master nodes and the slave nodes simultaneously demonstrates the functionality and technical capabilities of HARP. Each of these RISC cores has its own counter for general-purpose measurements while the entire platform works on a single clock source. The CC required for transferring the data from the data memory of RISC processor to the data memory of the CGRA node, from the data memory of the CGRA node to the local memory and also for processing the assigned algorithm are depicted in Table 8.1.

Table 8.1: Clock cycles required for data transfer and processing at different stages. D. Mem, Trans and Exe. stand for Data memory, Transfer and Execution, respectively. [P.VI]

Node-to -Node	Data Memory to Data Memory	Data Memory to CGRA's Local Memory	Transfer Total	Execution Total
N3-No	165	348	513	54
N3-N6	188	407	595	56
N4-N1	247	882	1129	67
N5-N2	304	4591	4895	179
N5-N8	616	8697	9313	354

Once the configuration streams and the data are loaded in parallel by the DMA device during the system start-up time, the slave nodes can start their execution simultaneously according to the algorithm data-flow. As it can be observed from the table, 513 CC and 595 CC are required for total data transfer from the data memory of N3 RISC core to No and N6 CGRA nodes for implementing 4-point DCT and 4-point DST in 54 CC and 56 CC, respectively. The implementation of 8-point DCT by N1 CGRA node is monitored by N4 RISC core in which the total data transfer and the process of the algorithm can be performed in 1129 CC and 64 CC, respectively. The execution of 16-point DCT and 32-point DCT by N2 and N8 CGRA nodes, respectively is controlled by N5 RISC core. They are the most time-consuming tasks of the platform due to their large matrix sizes and require 4895 CC and 9313 CC for data transfer and also 179 CC and 354 CC to be executed, respectively. Subsequent to the completion of each task, the results from the local memory of CGRA node should be transferred by the DMA device to the data memory of its host RISC processor for further processing. Although each CGRA node can also be designed to be used for processing of various sizes of DCT instead of just one size, which requires to be reconfigured at run-time, the critical path will become longer at the cost of operating frequency reduction. The percentage of the total execution time frame for the useful computation of No, N1, N2, N6 and N8 CGRA nodes excluding the data transfer total time is equal to 7.61%, 7.89%, 9.44%, 25.21% and 49.86%, respectively.

8.6 Measurements, Estimations, Evaluation and Comparisons

The modified platform for performing 4/8/16/32-point DCT and 4-point DST was synthesized on Stratix-V (5SGXMB9R3H4C2) FPGA device for prototyping the concept. The operating conditions are selected for both 0°C (low) and 85°C (high) junction temperatures and 23 mm heat sink with 200 LFPM airflow as a preset cooling solution. The achieved maximum operating frequencies after placement and routing are equal to 172.66 MHz 0°C and 161.75 MHz 85°C for slow timing model (900 mV). In the case of

fast timing model (900 mV), maximum operating frequencies were equal to 245.96 MHz and 232.97 MHz at 0°C and 85°C, respectively. The overall platform works on a single clock source and is simulated at 200.0 MHz operating frequency.

8.6.1 Resource Utilization

Node-by-node breakdown of resource utilization for the entire prototyped platform on FGPA device is depicted in Table 8.2. The resources are categorized in terms of the number of ALMs, Registers, Memory Bits and DSP elements. Around 30% of the ALMs, 8.6% of the registers and 42.7% of the memory bits are consumed by the overall design. It can be found out that the resource utilization of AVATAR-generated accelerators (N1 & N2 & N8) are approximately 4X more resource-consuming than CREMA-generated accelerators (No & N6) due to employing a higher number of PEs. Additionally, 51% of 18-bit DSP resources are utilized by the 32-bit multipliers instantiated in the PEs.

Table 8.2: Node-by-node Breakdown of Resource Utilization. [P.VI]

Node	ALMs	Registers	Memory Bits	(32-bit Multipliers) DSPs
No	5,792	4,121	2,364,672	(6) 12
N1	19,469	7,580	2,632,832	(10) 20
N2	20,659	7,887	2,632,992	(14) 28
N3	5,575	5,687	3,145,728	(6) 12
N4	5,630	5,662	4,194,304	(6) 12
N5	5,584	5,617	3,145,728	(6) 12
N6	6,088	3,454	2,364,672	(8) 16
N8	25,149	10,391	2,633,472	(34) 68
NoC	2,605	4,212	-	-
Total	95,551 30%	54,611 8.6%	23,114,400 42.7%	(90) 180 51%

8.6.2 Energy and Power Estimations

The platform's total power dissipation is estimated based on post P&R information using PowerPlay Power Analyzer Tool of Quartus II 15.0 at the operating frequency of 200.0 MHz and at the room temperature of 25°C. The total power dissipation is estimated by simulating the gate-level netlist of the entire platform and then generating the VCD file during the execution of DCT/DST by using ModelSim software while it yielded 'HIGH' level of confidence. The estimated dynamic, static and I/O thermal power dissipation are equal to 2492.9 mW, 1624.55 mW and 48.39 mW respectively, which makes a total of 4102.8 mW. Node-by-node breakdown of dynamic power dissipation and energy consumption of the system is depicted in Table 8.3. The AVATAR-generated accelerators (N1 & N2 & N8) require almost 1.8X-2.5X dynamic power consumption in comparison with the CREMA-generated accelerators (No & N6), which shows that the dynamic power dissipation increases as the size of PE arrays in the template-based CGRAs increases. The energy consumption for each node is computed as a product of dynamic power dissipation and execution time.

8.6.3 Evaluation and Comparisons

As it can be seen from Fig. 8.9, the current instance of HARP, which is modified for processing an OFDM receiver, consists of 256 PEs in total. Furthermore, the entire

Table 8.3: Dynamic power and energy estimation of each CGRA node and the NoC. GPP and IL stand for General Purpose Processing and Integration Logic, respectively. [P.VI]

Node	Accelerator Type	Dynamic Power (mW)	Active Time (μs)	Dynamic Energy (μJ)
N0	4-point DCT	182.34	0.27	0.05
N1	8-point DCT	363.5	0.34	0.12
N2	16-point DCT	407.92	0.9	0.37
N3	GPP	115.14	5.54	0.64
N4	Synchronization,	114.08	5.65	0.65
N5	Control	115.15	71.04	8.18
N6	4-point DST	193.71	0.28	0.05
N8	32-point DCT	456.85	1.77	0.81
NoC	-	10.03	-	-
IL	-	534.19	-	-
Total	-	2492.9	-	10.87

platform is running at 200.0 MHz operating frequency. By considering the total power dissipation of 4.1 W, the current instance of HARP delivers a performance of 51.2 GOPS and 0.012 GOPS/mW, which proves again the *architectural constant* for the HARP template on Altera Stratix-V chip in 28 nm.

In comparison with the other state-of-the-art platforms for executing the various sizes of DCT and DST dedicated for HEVC, our proposed architecture supports variable block sizes (4/8/16/32-point DCT and 4-point DST). However, in many other architectures, just one or few block sizes are supported, which shows the flexibility of HARP for the application developer. Considering FULL HD 1080p encoding at 30 fps, it requires the minimum throughput of 1920 $\times$ 1080 $\times$ 30 / (8 $\times$ 8) = 972,000 blocks/second in order to accomplish 8 $\times$ 8 DCT. The HARP template is able to perform 1D 8-point DCT and 2D 8-point DCT in 67 CC and 219 CC, respectively. Accordingly, in order to accomplish a complete 8 $\times$ 8 DCT block, HARP template demands 972,000 $\times$ 219 = 212.9 million cycles/second, which is equivalent to a clock frequency of 212.9 MHz. Considering the achieved 232.97 MHz maximum operating frequency at 85°C for fast timing model, HARP template is able to support 1080p format at 30 fps. By increasing the video quality to Full HD 1080p format at 60 fps and 4K UHD 2160p format at 30 fps, the clock frequency of 425.74 MHz and 851.47 MHz are required by HARP template, which can not be met on FPGA device. However, by integrating the architecture on ASIC, the required operating frequencies can be achieved. Based on the conducted experiment results from Chapter 6, the maximum operating frequency of 500.0 MHz can be obtained by synthesizing HARP on ASIC, which is enough to sustain 1080p format at 60 fps.

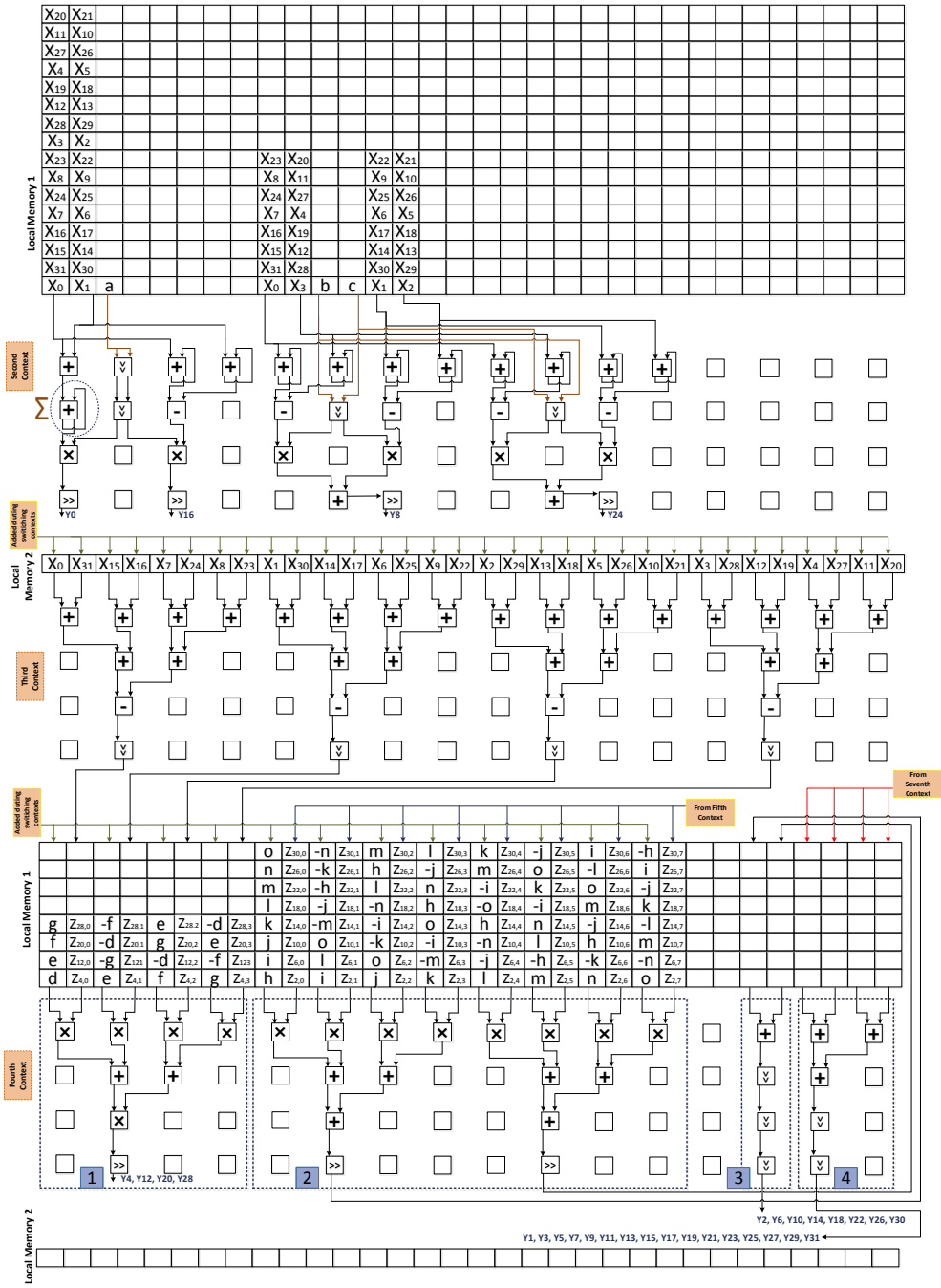

Figure 8.5: Second, Third & Fourth Contexts for the Calculation of 1D DCT 32-point.

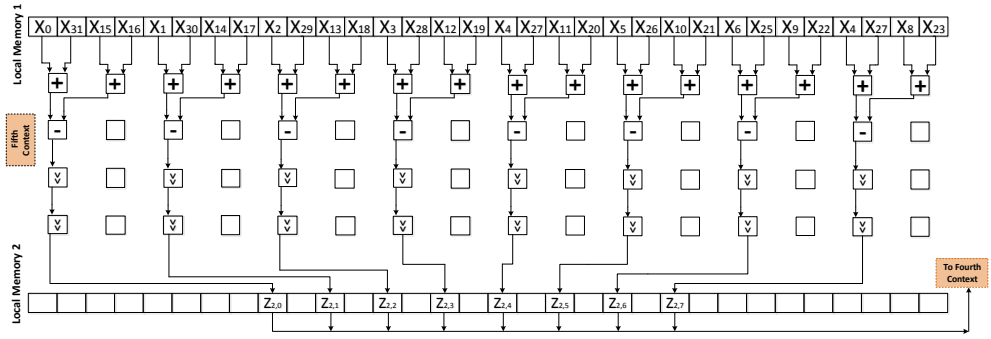

Figure 8.6: Fifth Context for the Calculation of 1D DCT 32-point.

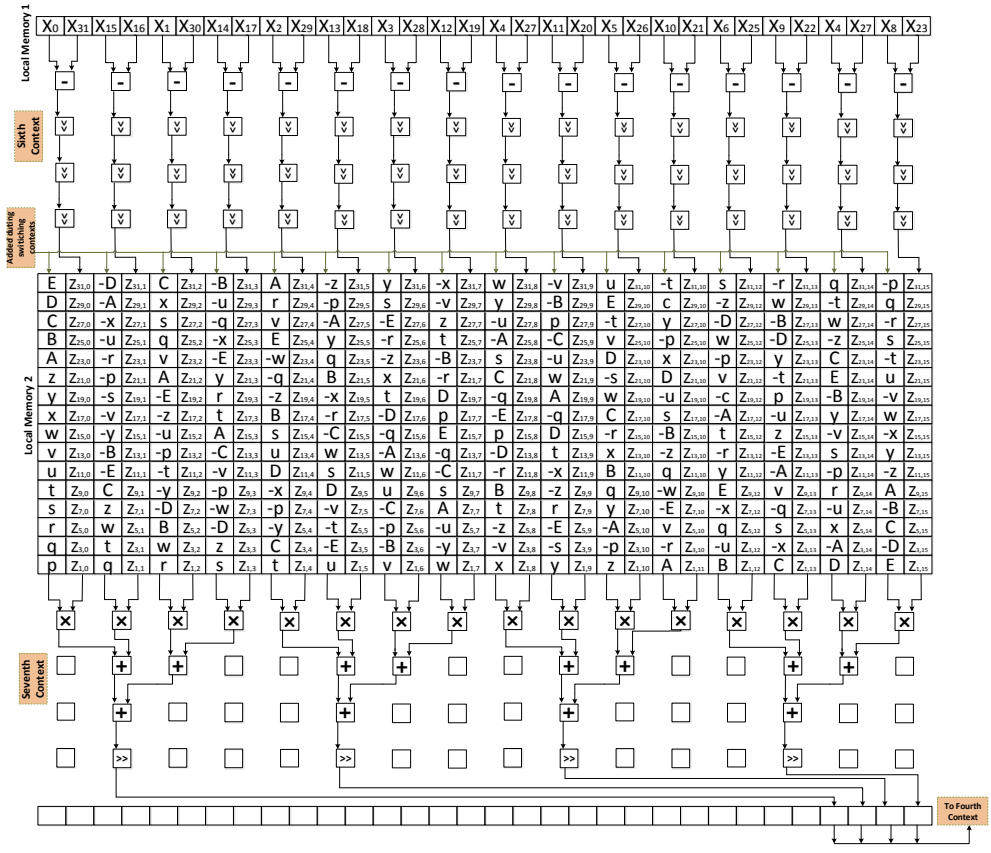

Figure 8.7: Sixth & Seventh Contexts for the Calculation of 1D DCT 32-point.

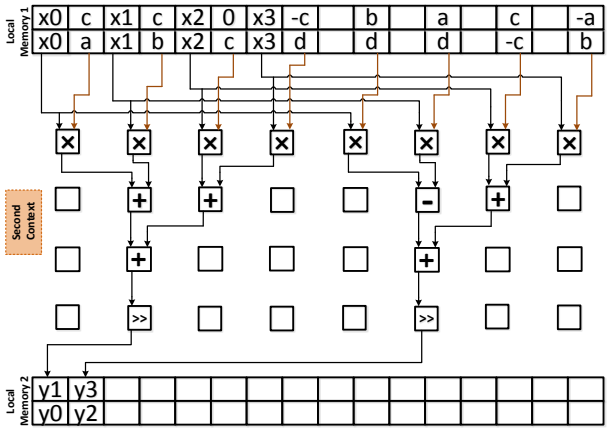

Figure 8.8: Context for the Calculation of 1D 4-point DST. [P.VI]

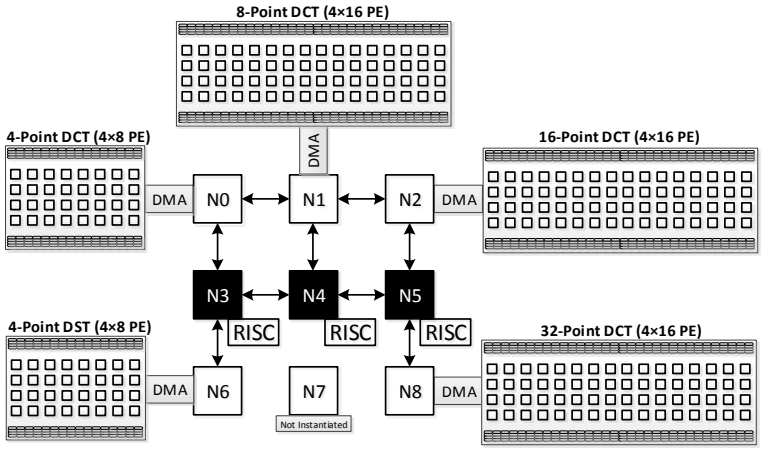

Figure 8.9: Abridged general view of Multi-purpose DCT/DST-specific accelerator on HARP platform. [P.VI]

9 Conclusion

In this thesis, the author tested, evaluated and verified the functionality and design features of Heterogeneous Accelerator-Rich Platform (HARP). The HARP template was designed as a heterogeneous multicore architecture in order to maximize the number of computational resources on a platform by integrating several template-based Coarse-Grained Reconfigurable Arrays (CGRAs) and Reduced Instruction-Set Computing (RISC) cores in a processor/coprocessor model while the backbone of communication is Network-on-Chip (NoC). The architecture of HARP is composed of nine nodes arranged in a mesh topology of three rows and three columns. The central node is always integrated with the RISC core, which acts as a supervisor node in order to monitor the functionality of the other slave nodes. Moreover, the supervisor node is responsible to transfer the configuration streams and data to be processed to the slave nodes and establish custom synchronization among all the cores in order to maintain the distributed control for data and configuration transfers as well as for parallel execution. The other nodes can be either integrated with template-based CGRA for parallel processing or even with the RISC core for general-purpose processing.

9.1 Main Results

As part of this design and test regime, the author designed a particular HARP instance by instantiating three RISC cores in the mid-row and a few template-based CGRAs around them. As the first test-case, Orthogonal Frequency Division Multiplexing (OFDM) receiver blocks have been selected as a competent candidate to be mapped on the HARP template. The OFDM receiver is a fine mixture of parallel and serial algorithms, which are demanded to be processed by the template-based CGRAs and RISC cores, respectively. The template-based CGRAs can be tailored to the computational requirements of the target application while the dynamic power dissipation can be mitigated. This combination provided a thorough exploration of the architectural features and identification of potential architectural fallacies and pitfalls. HARP was successfully evaluated by performing an OFDM receiver completely with the acceptable level of accuracy. Although the minimum requirement throughput defined by IEEE 802.11a/g for 16-QAM modulation has not been met due to having rather low clock frequency in the employed FPGA device, the required throughput set by the standard can be met easily by using a commercial off-the-shelf low cost device with at least 3.0X higher clock frequency. The utilization rate of the PEs employed for performing an OFDM receiver was measured to be 0.61%, which makes the HARP tamplate an excellent candidate for alleviating Dark Silicon issues.

The second test-case was design and implementation of various sizes of Discrete Cosine Transform (DCT) and Discrete Sine Transform (DST) dedicated for High Efficiency Video

Coding (HEVC) standard as the latest standard family on video compression. HARP was also successful in order to prove its technical capabilities as well as the functionality by acting as the the first complete multi-purpose DCT/DST-specific accelerator design for HEVC standard. It was able to execute 4/8/16/32-point 2D DCT and 4-point 2D DST by applying an N-point 1D transforms to each row and column of the residual block. The proposed architecture was capable of fully sustaining a format of Full HD 1080P at 30 fps on FPGA.

In both case-studies, the computationally intensive parallel tasks are designed and implemented by crafting template-based CGRAs while the serial in nature tasks are delegated to RISC cores as General Purpose Processors (GPPs). The designed CGRAs for performing an OFDM receiver as well as DCT/DST algorithms are crafted according to the algebraic equations in the most optimum way since most of the PEs are employed in each context. Optimal mapping is important at this level and should be noticed by designers since it will result in improvement of the performance, area utilization, power dissipation and execution time. In both case-studies, the entire HARP template including the RISC cores and designed application-specific accelerators on that has been prototyped on a 28 nm Altera Stratix-V Field Programmable Gate Array (FPGA) device at an operating frequency of 200.0 MHz and at the room temperature of 25°C. It was also evaluated for different high-level performance metrics in addition to the required number of Clock Cycles (CC) as the overall execution time such as resource utilization, maximum achieved operating frequency, power dissipation and energy consumption.

In this thesis, the author also presented a novel HW/SW codesign of an OFDM receiver using a High-Level Synthesis (HLS) tool. The employed HLS tool was from Xilinx called Software Defined System-on-Chip (SDSoC). The Zynq SoC is composed of an ARM processor besides a FPGA in order to improve the power efficiency as well as the system performance. In this case, the OFDM receiver blocks have been written in C/C++ code and realized on the FPGA and the ARM processor. The implementation of OFDM receiver blocks as the mixture of serial and parallel tasks on ZC706 board resulted in speed-up 3.49X compared to its implementation on the HARP template. However, in the case of parallel in nature tasks, CGRAs demonstrated more efficient implementation in comparison with high-level-synthesis for FPGAs since the designers could have near-optimal solutions for their target application.

As the preliminary result of this manuscript, the author concluded an architectural constant for the HARP template with the value of 0.012 Giga Operations Per Second per milliWatts (GOPS/mW). This value was conducted by considering the operating frequency, the total number of instantiated Processing Elements (PEs) at each HARP instance and total power dissipation. The architectural constant achieved from both case-studies proved that it does not change by scaling the computational resources of a platform. It ensured application-independent figure of merit and demonstrated the modularity and regularity of the HARP template as one of the key factors in order to have near optimal solution. According to the architectural constant of the HARP template, the cross-technology comparisons have been done with the other state-of-the-art platforms. Against the scaled performance levels of P2012, ADRES and MORPHEUS in terms of GOPS/mW, HARP showed a performance gain of 2.4X, 15X and 3X, respectively.

Another proceeding in order to have near optimal solution is mitigating the power dissipation by maintaining self-aware Dynamic Voltage and Frequency Scaling (DVFS). In this manuscript, the author also presented power mitigation of the HARP template

to be used as an OFDM receiver by applying Dynamic Frequency Scaling (DFS) on Stratix-V FPGA device and DVFS on 28nm Ultra-Thin Body and Buried oxide (UTBB) Fully-Depleted Silicon-On-Insulator (FD-SOI) ASIC technology. In this context, Feedback Control System (FCS) has been employed in RISC software in order to monitor the performance of each CGRA node continuously and tune the clock frequency of the CGRA nodes based on the selected worst-case execution time candidate. Every three RISC cores in the current HARP instance were responsible to store the counted clock cycles of their associated CGRA nodes. Subsequent to recognizing the worst-case execution time CGRA node, FCS degraded the performance of the other CGRA nodes to the equalization region in an user-defined and application-specific range of frequencies during several iterations. There is also possibility for the designed FCS to update itself automatically at run-time and define new performance equalization region in the case where the CGRAs are reconfigured for performing new tasks. In the case of FPGA implementation, since just the frequency can be scaled, the FCS technique with DFS has been applied, which resulted in mitigation of the dynamic power dissipation and total power dissipation by 20.2% and 16.8%, respectively. By moving to ASIC technology and scaling both frequency and voltage simultaneously, up to 82.98% dynamic power reduction was achieved. According to the conducted experiment results, self-aware computing models proved their ability to reduce the instantaneous power dissipation and accordingly the heat dissipation, which will result in mitigation the Dark Silicon issue.

As a summary of the main results of the experiments conducted in this thesis by the author, the HARP template was evaluated successfully to act as a power- and energy-aware general-purpose transceiver for IoT purposes. The main societal impact will be to make IoT energy consumption levels feasible for establishing smart homes and cities. Furthermore, HARP proved itself as a potentially competent hardware to be employed by 5G radio standards. The HARP template with the applied self-aware computing model showed its promising results to be instantiated in the dark area of the chip, which will result in Dark Silicon improvement. According to the provided important insights into the architecture in this thesis, the author recommended near optimal solution with some more proceedings in addition to the mapping of the applications on the CGRAs such as self-aware DVFS, modularity and regularity in the architecture and scalability in computational resources.

9.2 Future Developments

Many issues in the design of heterogeneous multicore architectures to act as a general-purpose power- and energy-aware transceiver platform for diverse needs of IoT remain to be resolved. Furthermore, the research in the area of Dark Silicon is still a hot topic in both industry and academia and tempts several research groups in the scientific community. The design and evaluation of HARP has already shown promising results, which call for further research to explore more opportunities to harness the Dark Silicon.

As the future work, it would be a great opportunity to develop, test and evaluate an extra-large scale power- and energy-aware heterogeneous multicore template-based architecture by increasing the size of the HARP to $2\times$ (3×3), which will be composed of 18 connected nodes as a 2D matrix of three rows and six columns. Such an extra-large scale heterogeneous multicore architecture can be used for harvesting Dark Silicon issue. The upgraded HARP can also act as a energy-efficient general-purpose transceiver

platform for future IoT scenario by supporting a large number of connected devices communicating at various standards. It can provide high throughput while consuming as little energy as possible. For the proposed architecture, the power dissipation should be minimized by adopting self-, power- and energy-aware computational models in the context of workload balancing and performance equalization. Furthermore, an appropriate compiler in order to port sequential code on top of the heterogeneous multicore architectures efficiently is still missing.

The HARP template is highly potential to be a suitable candidate for future IoT and 5G radio standards. It would be interesting to implement massively-parallel computationally-intensive signal processing algorithms for physical layer of IEEE 802.11n/ac transceiver with massive Multiple Input Multiple Output (MIMO) technology to achieve higher data rate networking and adaptability to available bandwidth at the hardware and software level. One more step towards making a general energy efficient transceiver architecture for IoT purposes would be the physical layer implementation of IEEE 802.15.4 standard for low-rate wireless personal area networks. It is the basis standard for different specifications released or under development, such as ZigBee and Radio Frequency Identification (RFID), which will be employed by IoT.

Although CGRAs give speed-up for computationally intensive tasks, most of them have a fixed set of PEs and interconnections, which are not optimum for various applications in terms of the cost and performance. In order to combat this issue, several Design Space Exploration (DSE) have been proposed while most of them are concerned with the interconnection between PEs and the configuration of their internal structure. The problem of such configuration techniques is the high hardware cost due to the need of substantial functional resources for initial PE design in order to have a reasonable performance. Therefore, the problem of interconnection between PEs for a particular application-specific accelerator and analyzing different DSE techniques in order to mitigate the hardware cost are still fully open. As a future task, we could focus on the exploration of novel DSE techniques for demonstrated template-based CGRAs in this manuscript.

By 2040, embedded systems will demand more electricity than the world energy resources can generate. IoT will soon connect 20 to 50 billion devices through wireless networks to the cloud. Despite all the advances in semiconductor technology and power- and energy-aware system design, the overall energy consumption of computing and communications systems is rapidly growing. However, the user expectations on features and battery life of on-line devices are increasing continuously, creating another incentive for seeking good trade-offs between performance and energy consumption. Moreover, the devices and communications in the periphery of 5G networks need to be optimized for lower energy consumption. Recently, approximate computing has emerged for making trade-offs between the accuracy of the results on one hand and the use of resources, power dissipation, energy consumption and time on the other hand. Approximate computing has huge opportunities for saving energy and time at the cost of some loss of accuracy or reliability of the results. Transprecision computing is based on adjustable word lengths according to the precision requirements in different parts of application algorithms, instead of using the worst-case precision throughout the computations. This can be achieved by using specifically designed multi-precision processors or by continually reconfiguring hardware according to the precision needed. The preliminary idea of the future work would be the exploration of using demonstrated template-based CGRAs for energy-efficient (transprecision and approximate) computing

on an open framework under development. It can be done for saving resources, time, power and energy while reaching acceptable levels of accuracy or precision. Therefore, the final target of the research is to benefit people and the smart living environments by bringing a new trade-off to the energy and performance of computing systems with the help of approximate computing and transprecision computing.

Bibliography

[1] S. Nouri, W. Hussain, and J. Nurmi, "Evaluation of a heterogeneous multicore architecture by design and test of an ofdm receiver," *in IEEE Transactions on Parallel and Distributed Systems*, vol. 28, no. 11, pp. 3171–3187, 2017.

[2] Q. Zhao and B. M. Maguire, "A survey of dynamic spectrum access," *IEEE Signal Processing magazine*, vol. 24, no. 3, pp. 79–89, 2007.

[3] J. Mitola and G. Maguire, "Cognitive radio: making software radios more personal," *IEEE Personal Communications*, vol. 6, no. 4, pp. 13–18, 1999.

[4] Y. Lin, H. Lee, M. Woh, Y. Harel, S. Mahlke, T. Mudge, C. Chakrabarti, and K. Flautner, "Soda: A high-performance dsp architecture for software-defined radio," *IEEE Micro*, vol. 27, no. 1, pp. 114–123, 2007.

[5] G. K. Rauwerda, P. M. Heysters, and G. J. M. Smit, "Towards software defined radios using coarse-grained reconfigurable hardware," *IEEE Transactions on Very Large Scale Integration (VLSI) Systems*, vol. 16, no. 1, pp. 3–13, 2008.

[6] G. Venkatesh, J. Sampson, N. Goulding, S. Gracia, V. Bryksin, J. L. Martinez, S. Swanson, and M. B. Taylor, "Conservation cores: reducing the energy of mature computations," *ASPLOS 10*, pp. 205–218, 2010.

[7] A. Pedram, S. Richardson, S. Galal, S. Kvatinsky, and M. Horowitz, "Dark memory and accelerator-rich system optimization in the dark silicon era," *in IEEE Design & Test*, no. 99.

[8] M. B. Taylor, J. Kim, J. Miller, D. Wentzlaff, F. Ghodrat, B. Greenwald, H. Hoffman, P. Johnson, J.-W. Lee, W. Lee, A. Ma, A. Saraf, M. Seneski, N. Shnidman, V. Strumpen, M. Frank, S. Amarasinghe, and A. Agarwal, "The raw microprocessor: a computational fabric for software circuits and general-purpose programs," *IEEE Micro*, vol. 22, no. 2, pp. 25–35, 2002.

[9] M. Taylor, "Is dark silicon useful?: harnessing the four horsemen of the coming dark silicon apocalypse," *In proceedings of the 49th Annual Design Automation Conference (DAC '12), NY, USA: ACM*, p. 1131–1136, 2012.

[10] A. Kulkarni, D. Stroobandt, A.Werner, F. Fricke, and M. Hübner, "Pixie: A heterogeneous virtual coarse-grained reconfigurable array for high performance image processing applications," *3rd International Workshop on Overlay Architectures for FPGAs (OLAF2017)*, pp. 1–6, 2017.

[11] C. B. C. et al., "Extra: Towards an efficient open platform for reconfigurable high performance computing," *2015 IEEE 18th International Conference on Computational Science and Engineering, Porto*, pp. 339–342, 2015.

[12] D. S. et al., "Extra: Towards the exploitation of exascale technology for reconfigurable architectures," *2016 11th International Symposium on Reconfigurable Communication-centric Systems-on-Chip (ReCoSoC), Tallinn*, pp. 1–7, 2016.

[13] J. Kylliäinen, T. Ahonen, and J. Nurmi, "General-purpose embedded processor cores - the coffee risc example," *In Processor Design:System-on-Chip Computing for ASICs and FPGAs, J. Nurmi, Ed. Kluwer Academic Publishers / Springer Publishers*, pp. 83–100, 2007.

[14] B. Mei, S. Vernalde, D. Verkest, H. D. Man, , and R. Lauwereins, "Adres: An architecture with tightly coupled vliw processor and coarse-grained reconfigurable matrix," *Field-Programmable Logic and Applications*, vol. 2778, pp. 61–70, 2003.

[15] F. Bouwens, M. Berekovic, A. Kanstein, G. Gaydadjiev, P. Diniz, E. Marques, K. Bertels, M. Fernandes, , and J. Cardoso, "Architectural exploration of the adres coarse-grained reconfigurable array," *Reconfigurable Computing: Architectures, Tools and Applications, Lecture Notes in Computer Science, Springer Berlin / Heidelberg*, vol. 4419, pp. 1– 13, 2007.

[16] B. Mei, F.-J. Veredas, and B. Masschelein, "Mapping an h.264/avc decoder onto the adres reconfigurable architecture," *International Conference on Field Programmable Logic and Applications*, pp. 622–625, 2005.

[17] H. Singh, M.-H. Lee, G. Lu, F. J. Kurdahi, N. Bagherzadeh, , and E. M. C. Filho, "Morphosys: An integrated recofigurable system for data-parallel and computation-intensive applications," *IEEE Trans. Computers*, vol. 49, no. 5, pp. 465–481, 2000.

[18] M.-H. Lee, H. Singh, L. Guangming, N. Bagherzadeh, F. J. Kurdahi, E. M. C. Filho, and C. A. Vladimir, "Design and implementation of the morphosys reconfigurable computing processor," *The Journal of VLSI Signal Processing-Systems for Signal, Image, and Video Technology, Kluwer Academic Publishers*, vol. 24, pp. 147–164, 2000.

[19] L. Guangming, H. Singh, L. Ming-Hau, N. Bagherzadeh, F. J. Kurdahi, E. M. C. Filho, and V. Alves, "The morphosys dynamically reconfigurable system-on-chip," *Proceedings of the First NASA/DoD Workshop on Evolvable Hardware*.

[20] H. Singh, L. Ming-Hau, L. Guangming, F. J. Kurdahi, N. Bagherzadeh, and E. M. C. Filho, "Morphosys: A reconfigurable architecture for multimedia applications," *Proc. XI Brazilian Symposium on Integrated Circuit Design*, pp. 134–139, 1998.

[21] V. Baumgarte, G. Ehlers, F. May, A. Nuckel, M. Vorbach, , and M. Weinhardt, "Pact-xpp a self-reconfigurable data processing architecture," *The Journal of Supercomputing*, vol. 26, no. 2, pp. 167–184, 2003.

[22] J. M. P. Cardoso and M. Weinhardt, "Xpp-vc: A c compiler with temporal partitioning for the pact-xpp architecture," *in Field-Programmable Logic and Applications: Reconfigurable Computing Is Going Mainstream, Editors: M. Glesner and P. Zipf and M. Renovell, Lecture Notes in Computer Science, Springer Berlin Heidelberg*, vol. 2438, pp. 864–874, 2002.

[23] H. H. et al., "Self-aware computing in the angstrom processor," *DAC Design Automation Conference 2012, San Francisco, CA*, pp. 259–264, 2012.

[24] W. Hussain, R. Airoldi, H. Hoffmann, T. Ahonen, and J. Nurmi, "Design of an accelerator-rich architecture by integrating multiple heterogeneous coarse grain reconfigurable arrays over a network-on-chip," *in Application-specific Systems, Architectures and Processors (ASAP)*, pp. 131–138, 2014.

[25] W. Hussain, R. Airoldi, H. Hoffmann, T. Ahonen, and J. Nurmi, "Harp2: An x-scale reconfigurable accelerator-rich platform for massively-parallel signal processing algorithms," *in the Springer's Journal of Signal Processing Systems*, 2015.

[26] N. S. Voros, M. Hübner, J. Becker, M. Kühnle, F. Thomaitiv, A. Grasset, P. Brelet, P. Bonnot, F. Campi, E. Schler, H. Sahlbach, S. Whitty, R. Ernst, E. Billich, C. Tischendorf, U. Heinkel, F. Ieromnimon, D. Kritharidis, A. Schneider, J. Knaeblein, and W. Putzke-Rming, "Morpheus: A heterogeneous dynamically reconfigurable platform for designing highly complex embedded systems," *ACM Trans. Embed. Comput. Syst.*, no. 70, p. 33, 2013.

[27] F. Thoma, M. Kuhnle, P. Bonnot, E. M. Panainte, K. Bertels, S. Goller, A. Schneider, S. Guyetant, E. Schuler, K. D. Muller-Glaser, , and J. Becker, "Morpheus: Heterogeneous reconfigurable computing," *International Conference on Field Programmable Logic and Applications, FPL*, pp. 409–414, 2007.

[28] D. Rossi, F. Campi, S. Spolzino, S. Pucillo, and R. Guerrieri, "A heterogeneous digital signal processor for dynamically reconfigurable computing," *in IEEE Journal of Solid-State Circuits*, vol. 45, no. 9, pp. 2481–2494, 2017.

[29] D. Melpignano, L. Benini, E. Flamand, B. Jego, T. Lepley, G. Haugou, F. Clermidy, and D. Dutoit, "Platform 2012, a many-core computing accelerator for embedded socs: Performance evaluation of visual analytics applications," *in Proc. 49th Annual Design Automation Conference (DAC '12). ACM , New York, NY, USA*, pp. 1137–1142, 2012.

[30] F. Conti, C. Pilkington, A. Marongiu, and L. Benini, "He-p2012: architectural heterogeneity exploration on a scalable many-core platform," *in Proc. of the 24th edition of the great lakes symposium on VLSI (GLS- VLSI '14), ACM, New York, NY, USA*, pp. 231–232, 2014.

[31] F. Garzia, W. Hussain, and J. Nurmi, "Crema, a coarse-grain re-configurable array with mapping adaptiveness," *in Proc. 19th International Conference on Field Programmable Logic and Applications (FPL 2009). Prague, Czech Republic: IEEE*, 2009.

[32] W. Hussain, F. Garzia, T. Ahonen, and J. Nurmi, "Designing fast fourier transform accelerators for orthogonal frequency-division multiplexing systems," *Journal of Signal Processing Systems, Springer*, vol. 69, pp. 161–171, 2012.

[33] M. L. Ferreira, J. C. Ferreira, and M. Hübner, "A parallel-pipelined ofdm baseband modulator with dynamic frequency scaling for 5g systems," *Applied Reconfigurable Computing. Architectures, Tools, and Applications, Springer International Publishing AG, part of Springer Nature 2018*, 2018.

[34] N. Voros, M. Hübner, D. Göhringer, and C. Antonopoulos, "Applied reconfigurable computing. architectures, tools, and applications," *14th International Symposium, ARC 2018, Santorini, Greece, May 2-4, 2018, Proceedings, Springer*, 2018.

[35] J. Heiskala and J. Terry, "Ofdm wireless lans: A theoretical and practical guide," *©2002 by Sams Publishing, SAMS, 201 West 103rd St., Indianapolis, Indiana, 46290 USA*.

[36] Altera, *Design Implementation and Optimization Quartus-II Handbook Version 13.1*. Altera Corporation, 2013.

[37] Altera, *Altera Product Catalog. 2015 Version 15.0,*. www.altera.com: Altera Corporation, 2014.

[38] P. D. Hennessy JL, "Computer architecture: A quantitative approach. 3rd edn. elseview morgan kaufmann, san francisco." 1990.

[39] C. Panis, "Vliw dsp processor for high-end mobile communication applications," *In Processor Design: System-on-Chip Computing for ASICs and FPGAs, J. Nurmi, Ed. Kluwer Academic Publishers / Springer Publishers*, pp. 83–100, 2007.

[40] Vassiliadis, N. Kavvadias, G. Theodoridis, and S. Nikolaidis, "A risc architecture extended by an efficient tightly coupled reconfigurable unit," *In Proc. ARC*, 2005.

[41] F. Garzia, T. Ahonen, and J. Nurmi, "A switched interconnection infra- structure to tightly-couple a risc processor core with a coarse grain reconfigurable array," *Research in Microelectronics and Electronics*, pp. 16–19, 2009.

[42] Xilinx, http://www.xilinx.com.

[43] Altera, http://www.altera.com.

[44] F. Lertora and M. Borgatti, "Handling different computational granularity by a reconfigurable icfeaturing embedded efpgas and a network-on-chip," *in The 13th Annual IEEE Symposium on Field-Programmable Custom Computing Machine (FCCM 2005)*, pp. 45–54, 2005.

[45] H. Wawrzinek, "Garp: A mips processor with reconfigurable coprocessor," *Proceedings of the Symposium on Field Programmable Custom Computing Machines*, 1997.

[46] T. J. Callahan, J. R. Hauser, and J. Wawrzynek, "The garp architecture and c compiler," *Computer*, vol. 33, pp. 62–69, 2000.

[47] N. S. Voros, A. Rosti, and M. Hubner, "Flexeos embedded fpga solution," *in Dynamic System Reconfiguration in Heterogeneous Platforms, Lecture Notes in Electrical Engineering, Springer Netherlands*, vol. 40, pp. 39–47.

[48] S. Vassiliadis, S. Wong, G. N. Gaydadjiev, K. Bertels, G. Kuzmanov, and E. M. Panainte, "The molen polymorphic processor," *IEEE Transactions on Computers*, vol. 40, pp. 1363–1375, 2004.

[49] E. M. Panainte, K. Bertels, and S. Vassiliadis, "The molen compiler for reconfigurable processors," *ACM Trans. Embed. Comput. Syst.*, vol. 6, 2007.

[50] A. Lodi, C. Mucci, M. Bocchi, A. Cappelli, M. D. Dominicis, and L. Ciccarelli, "A multi-context pipelined array for embedded systems," *International Conference on Field Programmable Logic and Applications*, p. 18, 2006.

[51] A. Lodi, M. Toma, F. Campi, A. Cappelli, R. Canegallo, and R. Guerrieri, "A vliw processor with reconfigurable instruction set for embedded applications," *IEEE Journal of Solid-State Circuits*, vol. 38, pp. 1876–1886, 2003.

[52] S. Pillement, O. Sentieys, and R. David, "Dart: a functional-level reconfigurable architecture for high energy efficiency," *EURASIP J. Embedded Syst.*, no. 5, p. 13, 2008.

[53] D. Raphael, C. Daniel, P. Sebastien, and S. Olivier, "A compilation framework for a dynamically reconfigurable architecture," *n Field-Programmable Logic and Applications: Reconfigurable Computing Is Going Mainstream, Editors: M. Glesner and P. Zipf and M. Renovell, Lecture Notes in Computer Science, Springer Berlin Heidelberg*, vol. 2438, pp. 1058–1067, 2002.

[54] S. Nouri, W. Hussain, and J. Nurmi, "Design and evaluation of correlation accelerator in ieee-802.11a/g receiver using a template-based coarse-grained reconfigurable array," *Nordic Circuits and Systems Conference (NORCAS): NORCHIP & International Symposium on System-on-Chip (SoC), Oslo, Norway*, pp. 1–6, 2015.

[55] W. Hussain, F. Garzia, and J. Nurmi, "Exploiting control management to accelerate radix-4 fft on a reconfigurable platform," *in Proc. International Symposium on System-on-Chip, Tampere, Finland: IEEE*, pp. 154–157, 2010.

[56] C. Brunelli, F. Garzia, and J. Nurmi, "A coarse-grain reconfigurable architecture for multimedia applications featuring subword computation capabilities," *n Journal of Real-Time Image Processing, Springer-Verlag*, pp. 21–32, 2008.

[57] hia Cheng Lo, S.-T. Tsai, and M.-D. Shieh, "A reconfigurable architecture for entropy decoding and idct in h.264," *VLSI Design, Automation and Test, 2009. VLSI-DAT '09. International Symposium on*, pp. 279–282, 2009.

[58] P. K. amd C. Bleakley, "Systolic algorithm mapping for coarse grained reconfigurable array architectures," *Reconfigurable Computing: Architectures, Tools and Applications, Lecture Notes in Computer Science, 2010, Springer Berlin Heidelberg,*, vol. 5992, pp. 351–357, 2010.

[59] R. Airoldi, F. Garzia, O. Anjum, and J. Nurmi, "Homogeneous mpsoc as baseband signal processing engine for ofdm systems," *International Symposium on System on Chip (SoC)*, pp. 26–30, 2010.

[60] "An iot endpoint system-on-chip for secure and energy-efficient near-sensor analytics, author =."

[61] W. Hussain, H. Hoffmann, T. Ahonen, and J. Nurmi, "Power mitigation by performance equalization in a heterogeneous reconfigurable multicore architecture," *in Journal of Signal Processing Systems, Springer*, pp. 287–297, 2017.

[62] H. Choi, Y. .Park, H.-H. Lee, , and C. Kim, "Adaptive dynamic frequency scaling for thermal-aware 3d multi-core processors," *In Computational Science and Its Applications-ICCSA 2012, Lecture Notes in Computer Science*, vol. 7336, pp. 602–612, 2012.

[63] X. Chen, Z. Xu, H. Kim, P. Gratz, J. Hu, M. Kishinevsky, U. Ogras, and R. Ayoub, "Dynamic voltage and frequency scaling for shared resources in multicore processor designs," *In Proceedings of the 50th Annual Design Automation Conference (DAC 13)*, no. 114, 2013.

[64] S. Jafri, M. Tajammul, A. Hemani, K. Paul, J. Plosila, and H. Tenhunen, "Energy-aware-task-parallelism for efficient dynamic voltage, and frequency scaling," *In CGRAs, 2013 International Conference on Embedded Computer Systems: Architectures, Modeling, and Simulation (SAMOSXIII)*, pp. 104–112, 2013.

[65] M.-H. Haghbayan, A.-M. R. A.Y., Weldezion, P. Liljeberg, J. Plosila, A. Jantsch, and H. Tenhunen, "Dark silicon aware power management for manycore systems under dynamic workloads," *In 2014 32nd IEEE International Conference on Computer Design (ICCD)*, pp. 509–512, 2014.

[66] Y. S. et al., "A self-aware processor soc using energy monitors integrated into power converters for self-adaptation," *Symposium on VLSI Circuits Digest of Technical Papers, Honolulu, HI*, pp. 1–2, 2014.

[67] R. Airoldi, F. Garzia, and J. Nurmi, "Improving reconfigurable hardware energy efficiency and robustness via dvfs-scaled homogeneous mp-soc," *IEEE International Symposium on Parallel and Distributed Processing Workshops and Phd Forum, Shanghai*, pp. 286–289, 2011.

[68] D. R. et al., "A self-aware architecture for pvt compensation and power nap in near threshold processors," *in IEEE Design & Test*, vol. 34, no. 6, pp. 46–53, 2017.

[69] G. J. Sullivan and J.-R. Ohm, "Recent developments in standardization of high efficiency video coding (hevc)," *in Proc. 33rd SPIE Appl. Dig. Image Process.*, vol. 7798, pp. 7798–7830, 2010.

[70] M. Budagavi, A. Fuldseth, G. Bjontegaard, V. Sze, , and M. Sadafale, "Core transform design in the high efficiency video coding (hevc) standard," *IEEE J. Select. Topics Signal Process.*, vol. 7, no. 6, pp. 1029–1041, 2013.

[71] P. Sjövall, V. Viitamäki, J. Vanne, T. D. Hämäläinen, and A. Kulmala, "High-level synthesis implementation of hevc 2-d dct/dst on fpga," *2017 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP), New Orleans, LA*, pp. 1547–1551, 2017.

[72] P. Sjövall, V. Viitamäki, A. Oinonen, J. Vanne, T. D. Hämäläinen, and A. Kulmala, "Kvazaar 4k hevc intra encoder on fpga accelerated airframe server," *IEEE International Workshop on Signal Processing Systems (SiPS), Lorient*, pp. 1–6, 2017.

[73] J. Nikara, R. Rosendahl, K. Punkka, and J. Takala, "Implementation of two-dimensional discrete cosine transform and its inverse," *12th European Signal Processing Conference, Vienna*, pp. 1537–1540, 2004.

[74] V. Viitamäki, P. Sjövall, J. Vanne, and T. D. Hämäläinen, "High-level synthesized 2-d idct/idst implementation for hevc codecs on fpga," *IEEE International Symposium on Circuits and Systems (ISCAS), Baltimore, MD*, pp. 1–4, 2017.

[75] A. C. Mert, E. Kalali, and I. Hamzaoglu, "An fpga implementation of future video coding 2d transform," *IEEE 7th International Conference on Consumer Electronics - Berlin (ICCE-Berlin), Berlin*, pp. 31–36, 2017.

[76] K. K. Singh and D. Pandey, "Implementation of dct and idct based image compression and decompression on fpga," *International Conference on Inventive Systems and Control (ICISC), Coimbatore*, pp. 1–4, 2017.

[77] M. Chen, Y. Zhang, and C. Lu, "Efficient architecture of variable size hevc 2d-dct for fpga platforms," *AEU - International Journal of Electronics and Communications*, vol. 73, pp. 1–8, 2017.

[78] P. Kitsos, N. S. Voros, T. Dagiuklas, and A. N. Skodras, "A high speed fpga implementation of the 2d dct for ultra high definition video coding," *18th International Conference on Digital Signal Processing (DSP), Fira*, pp. 1–5, 2013.

[79] R. Jeske, J. C. de Souza, G. Wrege, R. Conceicao, M. Grellert, J. Mattos, and L. Agostini, "Low cost and high throughput multiplierless design of a 16 point 1-d dct of the new hevc video coding standard," *VIII Southern Conference on Programmable Logic, Bento Goncalves*, pp. 1–6, 2012.

[80] E. D. Kusuma and T. S. Widodo, "Fpga implementation of pipelined 2d-dct and quantization architecture for jpeg image compression," *International Symposium on Information Technology, Kuala Lumpur*, pp. 1–6, 2010.

[81] R. E. Atani, S. Mirzakuchaki, and S. E. Atani, "Design and implementation of an image coprocessor," *International Conference on Image and Signal Processing*, pp. 498–507, 2008.

[82] C. Brunelli, F. Garzia, C. Giliberto, , and J. Nurmi, "A dedicated dma logic addressing a time multiplexed memory to reduce the effects of the system buss bottleneck," *in Proc. 18th International Conference on Field Programmable Logic and Applications, (FPL 2008), Heidelberg, Germany*, pp. 487–490, 2008.

[83] F. Garzia, C. Brunelli, and J. Nurmi, "A pipelined infrastructure for the distribution of the configuration bitstream in a coarse-grain reconfigurable array," *in Proceedings of the 4th International Workshop on Reconfigurable Communication-centric System-on-Chip (ReCoSoC'08). Univ Montpellier II*, pp. 188–191, 2008.

[84] T. Ahonen and J. Nurmi, "Hierarchically heterogeneous network-on-chip," *in Proc. Int. Conf. Comput. Tool*, p. 2580–2586, Sep. 2007.

[85] J. J. V. D. Beek, P. Borjesson, M.-L. Boucheret, D. Landstrom, J. Arenas, P. Odling, C. Ostberg, M. Wahlqvist, and S. Wilson, "A time and frequency synchronization scheme for multiuser ofdm," *IEEE Journal on Selected Areas in Communications*, vol. 17, no. 11, pp. 1900–1914, 1999.

[86] F. Garzia, C. Brunelli, L. Nieminen, R. Mastria, and J. Nurmi, "Implementation of a tracking channel of a gps receiver on a reconfigurable machine," *Proceedings of the International Conference of Computer as a Tool (EURI-CON '07)*, pp. 875–881, 2007.

[87] J. E. Volder, "The cordic trigonometric computing technique," *IRE Transactions on Electronic Computers*, pp. 330–334, 1959.

[88] J. W. Cooly and J. W. Tukey, "An algorithm for the machine calculation of complex fourier series," *Mathematics of Computation*, pp. 297–301, 1965.

[89] M.-O. Pun, M. Morelli, , and C.-C. J. Kuo, "Multi-carrier techniques for broadband wireless communications : a signal processing perspective," *copyright ©2007 by Imperial College Press*, 2007.

[90] V. S. Ryaben'kii and S. V. Tsynkov, "A theoretical introduction to numerical analysis," *CRC Press*, p. 243, 2006.

[91] http://www.mentor.com/.

[92] K. Ian and J. Rose, "Measuring the gap between fpgas and asics," *IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems*, vol. 26, no. 2, pp. 203–215, 2007.

[93] P. Bonnot, F. Lemonnier, G. Edelin, G. Gaillat, O. Ruch, and P. Gauget, "Definition and simd implementation of a multi-processing architecture approach on fpga," *In Proc. of Design, Automation and Test in Europe (DATE '08). ACM, New York, NY, USA*, pp. 610–615, 2008.

[94] https://www.synopsys.com/implementation-and-signoff/rtl-synthesis-test/dc-ultra.html.

[95] Xilinx, *Introduction to FPGA Design with Vivado High Level Synthesis*. http://www.xilinx.com: UG989 (v1.0), 2013.

[96] Xilinx, *SDSoC Environment Tutorial*. http://www.xilinx.com: UG1028 (v2016.3), 2016.

[97] Y. Kim, R. N. Mahapatra, and K. Choi, "Design space exploration for efficient resource utilization in coarse-grained reconfigurable architecture," *in IEEE Transactions on Very Large Scale Integration (VLSI) Systems*, vol. 18, no. 10, pp. 1470–1482, 2010.

[98] M. Budagavi, A. Fuldseth, G. Bjontegaard, V. Sze, and M. Sadafale, "Core transform design in the high efficiency video coding (hevc) standard," *IEEE Journal of Selected Topics in Signal Processing*, 2013.

[99] W. Chen, C. H. Smith, and S. C. Fralick, "A fast computational algorithm for the discrete cosine transform," *IEEE Transaction on Communications*, vol. 25, no. 9, 1977.

Publication I

Reprinted with permission from the Proceedings of the *International Conference on Design & Architectures for Signal & Image Processing (DASIP)*, Cracow, Poland, pp. 23-25., Sep. 2015, S. Nouri, W. Hussain, and J. Nurmi, "Implementation of IEEE-802.11a/g Receiver Blocks on a Coarse-Grained Reconfigurable Array". doi: 10.1109/DASIP.2015.7367254

Publication II

Reprinted with permission from the Proceedings of the *IEEE Nordic Circuits and Systems Conference (NORCAS): NORCHIP & International Symposium on System-on-Chip (SoC)*, Oslo, Norway, pp. 1-6., Oct. 2015, S. Nouri, W. Hussain, and J. Nurmi, "Design and evaluation of correlation accelerator in IEEE-802.11a/g receiver using a template-based Coarse-Grained Reconfigurable Array". doi: 10.1109/NORCHIP.2015.7364374

Publication III

Reprinted with permission from the Proceedings of the *the 8th International Symposium on Highly Efficient Accelerators and Reconfigurable Technologies (HEART2017)*, Bochum, Germany, ACM, New York, NY, USA, no 15, June 2017, S. Nouri, J. Rettkowski, D. Göhringer, and J. Nurmi, "HW/SW Co-design of an IEEE 802.11a/g Receiver on Xilinx Zynq SoC using High-Level Synthesis". doi.org/10.1145/3120895.3120908

HW/SW Co-design of an IEEE 802.11a/g Receiver on Xilinx Zynq SoC using High-Level Synthesis

Sajjad Nouri
Laboratory of Electronics and Communications Engineering, Tampere University of Technology
Tampere, Finland
sajjad.nouri@tut.fi

Jens Rettkowski
Application-Specific Multi-Core Architectures (MCA) Group, Ruhr-Universität Bochum
Bochum, Germany
jens.rettkowski@rub.de

Diana Göhringer
Chair for Adaptive Dynamic Systems,Technische Universität Dresden
Dresden, Germany
diana.goehringer@tu-dresden.de

Jari Nurmi
Laboratory of Electronics and Communications Engineering, Tampere University of Technology
Tampere, Finland
jari.nurmi@tut.fi

ABSTRACT

This paper presents an implementation of an Orthogonal Frequency-Division Multiplexing (OFDM) receiver using the high-level synthesis tool, from Xilinx called Software Defined System-on-Chip (SDSoC). The Zynq SoCs containing an ARM processor besides a Field Programmable Gate Array (FPGA) are introduced to improve the system performance and power efficiency. SDSoC provides an embedded C/C++ application programming interface for developing heterogeneous embedded systems. Thus, the OFDM receiver is written in C/C++ code to be realizable on the FPGA and the ARM processor. The OFDM receiver is composed of computationally intensive tasks which requires a HW/SW co-design to fulfill system requirements. In this work, the implementation of OFDM receiver blocks are evaluated on ZC706 board and compared against the Heterogeneous Accelerator-Rich Platform (HARP). Based on the achieved results, the complete execution of the IEEE 802.11a/g receiver shows an overall speed-up 3.49X compared to the HARP platform.

ACM Reference Format:
Sajjad Nouri, Jens Rettkowski, Diana Göhringer, and Jari Nurmi. 2017. HW/SW Co-design of an IEEE 802.11a/g Receiver on Xilinx Zynq SoC using High-Level Synthesis. In *Proceedings of The 8th International Symposium on Highly Efficient Accelerators and Reconfigurable Technologies Proceedings, Bochum, Germany, June 2017 (HEART'17),* 6 pages.
https://doi.org/10.1145/3120895.3120908

1 INTRODUCTION

The demand for executing various sets of operations by embedded systems as well as the increasing computational performance and rapid development in semiconductor technology have led to the use of Heterogeneous Multicore Architectures (HMAs) that use accelerators. Additionally, Internet-of-Things (IoT) and its future, Internet-of-Everything (IoE), increase significantly the need for heterogeneous platforms since the wide range of technologies belonging to IoT are used in a large spectrum of application domain with its own data models [1]. HMAs can also be a worth candidate for 5G baseband in terms of exploiting parallelism in order to support a large number of connected devices communicating at multiple different standards and different types of data transmission schemes. Another reason which has led to the growth and importance of HMAs is the Dark Silicon issue [2]. It states that on a single chip, all cores cannot be clocked at their maximum operating frequency because of the given thermal design power constraint. Therefore, a large fraction of chip has been forced to be powered-off (dark) or operated at a very low frequency (dim). One of the suggested solutions to combat with this issue is the use of application-specific accelerators instantiated in a HMA for performing computationally intensive kernels.

In this context, Field-Programmable-Gate-Arrays (FPGAs) can be employed for integrating multiple processing elements (PEs) on a single chip to build heterogeneous Multi-Processor Systems-on-Chip (MPSoCs), to be consistent with Section II. Since programming of FPGAs as well as designing application-specific accelerators by using hardware description language might be expensive and complicated, High-Level Synthesis tools such as VivadoHLS [3] and SDSoC (Software-Defined SoC) [4] have been introduced. SDSoC provides an embedded C/C++ application programming interface for developing heterogeneous embedded systems on the Zynq MPSoC/SoC platform. It includes a C/C++ compiler which can compile each thread of an application by a processor in software/hardware.

The main contribution of this paper is a novel HW/SW codesign of an Orthogonal Frequency-Division Multiplexing (OFDM) receiver using high-level-synthesis. The OFDM receiver contains algorithms of parallel and serial nature, i.e., Fast Fourier Transform (FFT), Correlation, Convolution and Complex Matrix-Vector Multiplication (MVM) which requires hardware/software co-design. This mixture of parallel and serial tasks are computationally intensive and time consuming. The computationally intensive and parallel tasks are implemented into hardware by compiling the functions written in C/C++ code. The functions are optimized by pragmas to be efficiently implemented in the FPGA. Furthermore, in this

HEART'17, June 2017, Bochum, Germany

ACM ISBN 978-1-4503-5316-8/17/06…$15.00
https://doi.org/10.1145/3120895.3120908

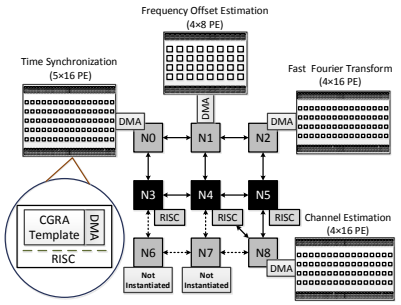

Figure 1: HARP Template, the black colored N3, N4 and N5 are the supervisor nodes containing the RISC processor, the rest of the nodes are slave and can be either RISC or CGRA as a coprocessor; General view of IEEE 802.11a/g receiver on HARP platform.

work, the system is compared to the Heterogeneous Accelerator-Rich Platform (HARP) [5] in terms of performance. The system presented in this paper shows a speedup of 3.49x compared to the HARP system.

This paper is organized as follows. In Section II, related work about OFDM receivers implemented in HMA is presented. Section III explains the algorithms employed at different blocks of the OFDM receiver. Section IV presents the implementation of the OFDM receiver baseband processing on the ZC706 evaluation board. In Section V, the experimental results are discussed and compared to the HARP system. In the last section, the conclusion is given.

2 RELATED WORK

Until now, several state-of-the-art heterogeneous multicore platforms have been developed in order to accelerate many specific algorithms and investigate Dark Silicon issue. In the following, some of the most promising heterogeneous platforms are described briefly. Platform 2012 (P2012) is a homogeneous MPSoC consisting of four clusters connected by a Network-on-Chip (NoC) architecture [6]. Each cluster contains 16 general-purpose Reduced Instruction Set Computing (RISC) processors which are locally synchronous but globally asynchronous. In addition to the homogeneous version, there is also a heterogeneous extended version of P2012, called He-P2012 [7]. MORPHEUS ([8], [9]) is a heterogeneous dynamically reconfigurable platform comprising fine-grained, middle-grained and coarse-grained reconfigurable devices communicating over a 64-bit NoC. FlexEOS as an embedded FPGA is employed for fine-grained algorithms and constructed from multi-function logic cells. Besides FlexEOS, MORPHEUS has DREAM platform as a middle-grained reconfigurable Digital Signal Processor (DSP) core in order to exploit instruction level parallelism. DREAM consists of a 32-bit RISC core processor and PiCoGA-III reconfigurable datapath. The coarse-grained one is XPP-III which is composed of a dataflow array and Very Long Instruction Word (VLIW) processor as a heterogeneous platform optimized for streaming applications. Software oriented approach was also provided for ease of development and implementation. HARP is designed as a template-based architecture written in parametrized VHDL and programmable in

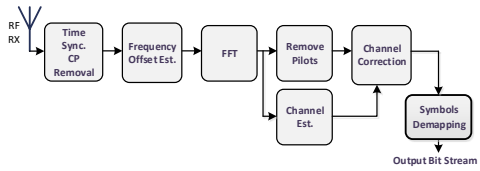

Figure 2: A simplified block diagram of the OFDM receiver.

C language for maximizing the number of computational resources [10]. HARP is composed of nine nodes connected to each other over a NoC in a 3×3 mesh topology, shown in Figure 1. The CGRAs are template-based and scalable. Each CGRA is equipped with R rows × C columns of reconfigurable Processing Elements (rPEs) where the values of 8 or 16 can be assigned to C while the value of R is based on the application requirements and algorithms. The functionality of rPE as well as interconnection among them can be implemented at design time. HARP is designed in a way that designers can instantiate each node either with a CGRA of a specific dimension or a RISC core or even leave it as a data routing resource. The central node is usually considered as a supervisor node which is responsible for transferring data between its own data memory and data memories of the slave nodes. Slave nodes contain a template-based CGRA or RISC core, a data memory and a Direct Memory Access (DMA) device [11]. An example of application mapping on HARP is also depicted in Figure 1 where an OFDM receiver is designed by crafting template-based CGRAs and then arranged over a NoC structure. As it is reported, HARP delivers a performance of 48 Giga Operations Per Second (GOPS) and 0.012 GOPS/mW for Altera Stratix-V chip in 28 nm. Compared to other state-of-the-art platforms, HARP shows a performance gain of 2.4X against the scaled performance of P2012 on a 28 nm FPGA which is estimated to be 0.005 GOPS/mW. Moreover, HARP's performance shows a gain of 3X in comparison to the scaled performance of MORPHEUS which has the value of 0.02 GOPS/mW at 90 nm CMOS.

3 OFDM RECEIVER ALGORITHMS

OFDM, as a special case of multicarrier modulation, is introduced and developed to combat with Inter-Channel Interference (ICI), Inter-Symbol Interference (ISI), frequency selective fading and also multipath channel fading in wireless communications. In an OFDM system, a transmitted data stream is divided into several parallel streams over different parallel subcarriers to achieve higher data rates [12]. The OFDM receiver, shown in Figure 2, is composed of several blocks to perform the inverse operation of the transmitter. In the following sections, each block of the receiver as well as the employed algorithms will be explained.

3.1 Time Synchronization

Time Synchronization is the first block of the OFDM receiver. It finds the exact moment of individual OFDM symbols and the starting point of the FFT window. One approach is a Cyclic Prefix (CP) correlation based method. Each of the received data symbols (y_n) should be correlated with its delayed version and thereafter the maximum values have to be found [13]. The following two steps perform a Time Synchronization:

(1) Correlation between y_n and conjugation of y_{n-D}. The length of delay (D) is equal to the length of CP (16).

$$c_n = y_n y^*_{n-D} \tag{1}$$

(2) Calculating the square modulus of produced outputs from the last step and finding the maximum value which is also the right position of the FFT window.

$$z_n = \sum_{i=0}^{L-1} c_{i+n} \tag{2}$$

3.2 Frequency Offset Estimation

The next block of the OFDM receiver is Frequency Offset Estimation where the amount of Carrier Frequency Offset (CFO) caused by device impairments should be estimated [12]. In other words, the received baseband signal after downconversion will be centered at f_Δ instead of zero. This can cause ISI and inaccuracy of demapping the data symbols. To estimate the CFO, special training symbols added in the transmitter can be employed. Each received packet contains short and long training symbols in addition to the data symbols which are predefined samples known for the receiver. At the receiver side, the down-conversion of the signal r_n can be expressed as

$$r_n = x_n e^{j2\pi f_\Delta n T_s} \tag{3}$$

where x_n is the transmitted signal. By applying the delay-and-correlate method on the output of Equation (3) and its delayed version, CFO can be estimated as depicted in Equation (4) and (5). The amount of delay, D, is equal to 16 which is calculated by multiplying the period of short training symbols and frequency spacing (0.8 μs $\times$ 20.0 MHz).

$$z = \sum_{n=0}^{L-1} r_n r^*_{n+D} \tag{4}$$

$$\hat{f_\Delta} = -\frac{1}{2\pi D T_s} \angle z \tag{5}$$

In Equation (5), T_s is the sampling period and $\angle$ takes the angle of z. Subsequent to finding the CFO, frequency offset correction has to be performed as the last step of this block by multiplying the estimated CFO and the received signal based on the following equation where $r_n{}'$ is the corrected signal, n is the sample index and N is the number of samples in a symbol.

$$r_n{}' = r_n \times e^{-j2\pi f_\Delta \frac{n}{N}} \tag{6}$$

3.3 Fast Fourier Transform

FFT is the next block of the OFDM receiver which is required for converting the corrected received signal from time domain to frequency domain. FFT is the special case of the Discrete Fourier Transform (DFT) which can be performed efficiently by using radix-2^m structures where $m \in Z^+$ [16]. By increasing the value of m, the number of execution stages of the FFT will be decreased. However, the arithmetic resources required by the FFT structural unit as well as its complexity will be increased.

3.4 Channel Estimation

Before the OFDM data symbols arrive at the receiver, they may get distorted during the transmission. Therefore, channel estimation is required for estimating the channel frequency response to resolve correctly the symbols. Frequency domain channel estimation can be performed using pilot subcarriers and interpolation. According to the IEEE 802.11a/g specifications, there are four regular pilot subcarriers which include constant known data. At the receiver side, the received noisy data symbols, Y_n, can be written as Equation (7) where n, H_n and N_n stand for the number of subcarriers, channel response and additive noise, respectively.

$$Y_n = X_n H_n + N_n \tag{7}$$

The process of channel estimation is composed of two steps: estimation of H_n and correction of Y_n as close as possible to X_n [14]. The channel response can be computed by multiplying the inverse of the diagonal matrix formed from transmitted pilots, M^{-1}, and the received noise-impaired pilots, P_{Rx} shown in Equation 8.

$$\tilde{H_k} = M^{-1} P_{Rx} \tag{8}$$

Here, k and $\tilde{H_k}$ are representing the number of pilots and the channel response of the received pilots, respectively. As the next step, the channel response of the remaining subcarriers requires to be computed by using Linear Interpolation which is a method for approximating the value at each position among two adjacent known values. It can be mathematically expressed as the following:

$$\hat{H_n} = \sum_{i=1}^{N_p-1} \sum_{j=1}^{N_s} \tilde{H_k}(i) + ((\tilde{H_k}(i+1) - \tilde{H_k}(i)) \times \frac{j-1}{N_s}) \tag{9}$$

where $\tilde{H_n}$, N_p, N_s and μ stand for the extension of channel frequency response of the pilots for the remaining subcarriers, number of pilots and samples and the step size, respectively. Subsequent to channel estimation, channel equalization is required to be implemented as an effort to correct the received noisy data symbols. Therefore, the received data symbols are needed to be divided by the estimated channel response according to the following equation.

$$\hat{Y_n} = \frac{Y_n}{\hat{H_n}} \tag{10}$$

3.5 Symbols Demapping

Subsequent to performing all required synchronization operations, the last step is the demodulator. This is required to determine the transmitted data bits for each received data symbol. In general, there are two decision methods for this purpose: hard and soft. In this research work, the first method is used for performing symbols demapping for 16-QAM (Quadrature Amplitude Modulation) constellation points. At the transmitter side of the OFDM, data bits are modulated with one of the schemes such as 16-QAM which comprises four bits per symbol. Moreover, since QAM changes both the amplitude and phase of the carrier, transmitted data symbols are complex and composed of Quadrature and In-phase carriers. Thus, at the receiver side, the complex plane should be divided into decision boundaries which include the set of points. Each point includes Quadrature and In-phase parts which are equivalent to real and imaginary parts of the received data symbols, respectively. As the complex plane is divided into In-phase and Quadrature areas, there are four zones for each leftmost and rightmost two bits. Therefore, the received data symbols can be mapped to data bits according to the decision boundaries and the closest constellation points to them. In other words, a data symbol in each of 16 decision areas will be mapped to the bits assigned to that constellation point [12].

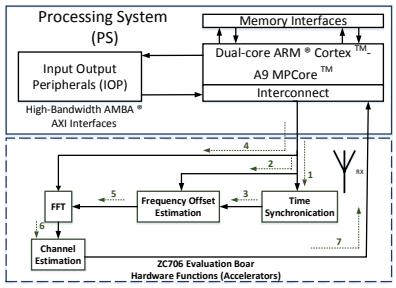

Figure 3: General view of IEEE 802.11a/g receiver on ZC706 evaluation board.

4 IMPLEMENTATION OF BASEBAND PROCESSING ON THE ZC706 EVALUATION BOARD

In this section, the instantiation and implementation of the OFDM receiver baseband processing on the Xilinx Zynq SoC using the high-level synthesis tool, SDSoC development environment, as an application for a Linux host is presented. SDSoC provides an embedded C/C++ application programming interface for implementing heterogeneous embedded systems. Each block of the OFDM receiver is written in C++ code based on the aforementioned algorithms. The overall platform for the OFDM receiver can be customized with application-specific hardware accelerators which are connected to the platform through a data motion network. The entire architecture of the OFDM receiver on the ZC706 evaluation board is depicted in Figure 3. It has to be mentioned that the platform for the ZC706 development board is the default provided by the SDSoC tool. The order of executing the hardware accelerators as well as exchanging the results and data symbols among the different blocks is specified with the green colored numbers and dashed arrows. Moreover, the hardware/software connectivity and the timeline for programmed hardware functions to be called during the process of the OFDM receiver are shown in Figure 4. As the first step, the data symbols are loaded from the ARM processor to the FPGA hardware by a DMA (dashed arrow number 1). SDSoC provides different DMA configurations. In this case, for array arguments, `axi_dma_sg` (scatter-gather DMA) data mover is required. In general, the data motion network in the SDSoC environment is composed of the following components: hardware interface for the accelerators, data movers between the Processing System (PS) and the FPGA exchanging data through AXI ports from the ARM processor. As it can be seen from Figure 4, the CPU establishes each hardware function and the data transfer for each function call. Once all calls and transfers are completed and the inputs of hardware functions become available, accelerators can start their actual computation based on the algorithm data flow controlled by the software. It has to be mentioned that setting up Data Mover (DM) for each new array can be performed in parallel with the data transfer of the previous one. As an example, setting up DM for the first input of TS which is an array with the size of 80 will take 1005 Clock Cycles (CC). Afterward, this array can be transmitted to the hardware function in 1317 CC while at the same time, setup DM for the second array can be established. In other words, the total process of transferring data from the CPU to the FPGA hardware functions will take 6642 CC, expressed in Equation (11).

$$6642\ CC = 5334\ CC + (5334\ CC - 4026\ CC) \tag{11}$$

Subsequent to performing the previous stages, the correlation algorithm as well as seeking the index of the time offset can be implemented in 9549 CC. The simplified algorithm written for performing the Time Synchronization block in SDSoC development environment based on Equation (1) and (2) is depicted in Algorithm 1. As it can be observed from the highlighted parts, loop pipelining and loop unrolling can be employed. The `pragma` *pipeline* improves the performance of the hardware function by implementing the loop in a concurrent manner. Loop pipelining transforms the sequential execution of operations into parallel operations. Subsequent to applying pipelining, the cycles to execute all operations are reduced. The efficiency of this `pragma` is limited mainly by data dependency. The `pragma` *loop unrolling* improves also the performance by exploiting the parallelism between loop iterations. Unrolling creates copies of the loop body which are executed concurrently. The programmer can determine how many copies are created by inserting a factor. The available hardware resources and data dependency give an upper limit of how many copies can be created. In this case, HLS unroll factor is chosen to be 4 as the most effective factor by using trial and error. Furthermore, since the design was too large, it was not synthesizable for the larger unroll factors.

Algorithm 1 CP correlation based method for TS

```
1:  Initialize Location to 0, and Max to −1
2:  for i:=1 to 80 step 1 do
3:      #pragma HLS PIPELINE enable_flush rewind off
4:      for j:=1 to 80 step 1 do
5:          #pragma HLS PIPELINE II=1
6:          #pragma HLS unroll factor=4
7:          c ← y(j)y*_D(j + i) + c
8:      z = real(c) ∗ real(c) + imag(c) ∗ imag(c)
9:      if (z > Max) then
10:         Max = SM
11:         TimeOffsetIndex = i
```

As the next step, the short training symbols as well as the index of time offset have to be transferred from the ARM processor to the hardware accelerator for performing frequency offset estimation and correction in 10280 CC (dashed arrows number 2 & 3). The computation of Frequency Offset Estimation block can be executed in 4747 CC. In parallel with the computation of Frequency Offset Estimation, twiddle factors can be loaded from CPU to the FPGA hardware functions which are required for implementing the FFT block. In the next stage, FFT is performed by the use of the radix-4 algorithm on the results of the Frequency Offset Estimation block which are exchanged directly among the accelerators in 6487 CC. Once the FFT is performed completely, the data is transferred to the next block (dashed arrow 6) in order to implement the Channel Estimation which is the last hardware function. Then the results of the Channel Estimation block should be transferred back to the PS in order to convert the data symbols to data bits. As it

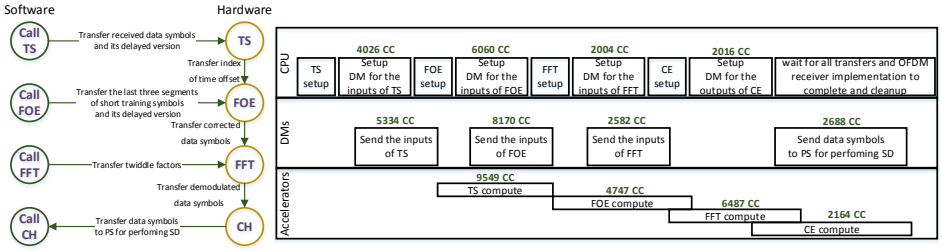

Figure 4: HW/SW connectivity with direct connection and timeline for hardware function calls. DM stands for Data Mover.

is mentioned above, the hard decision method is employed for Symbols Demapping which is just based on the comparison among the received data symbols and the constellation points located in the same decision boundaries. This task can be performed completely in software without requiring any accelerator.

5 EXPERIMENTAL RESULTS, COMPARISON AND DISCUSSION

The number of clock cycles required for implementing each block in hardware on the FPGA and in software on the ARM processor in comparison with the HARP platform are shown in Table 1. The speed-up for each accelerator is calculated based on the execution time (since the platforms compared are running at different clock speeds) which is the product of the total number of clock cycles and the clock period. Accordingly, the overall speed-up is the ratio of the old execution time to the new execution time for a system. The clock frequency of HARP and SDSoC are equal to 200 and 667 MHz, respectively. The calculated speed-up of the SDSoC HW accelerators compared to HARP is given in Table 1. As it can be observed, the implementation of Time Synchronization, Frequency Offset Estimation and Symbols Demapping on the ZC706 evaluation board shows overall speed-ups with the values of 1.21X, 8.92X and 4.03X, respectively. However, in the case of the FFT and channel estimation, HARP has better performance which shows the ability of CGRAs for executing parallel tasks. Although the FPGA is also a worthy platform in executing parallel tasks, however by the use of CGRA, the designers can implement near-optimal solutions for their target applications since the placement and routing can be performed manually. One of the reported problems of CGRAs is a fixed set of PEs and interconnections which are not optimal for various applications in terms of cost and performance. In order to tackle this issue, several Design Space Exploration (DSE) techniques have been proposed which are concerned with the interconnection between PEs [15]. Therefore, the users of CGRAs can design near-optimal applications in terms of cost and performance by employing the DSE techniques. Another difference between CGRA and FPGA is related to the level of granularity of the employed reconfigurable devices. Fine-grained devices can be characterized by PEs with granularity of 2 or 4 bits. However, FPGAs can potentially accept any bit-size while their concept is based on LUTs not PEs. Compared to coarse-grained devices, fine-grained devices require more PEs and also a combination of PEs to perform a single operation which results to large resource demanding. On the other hand, fine-grained devices might be better in terms of flexibility for performing new tasks. In CGRAs, each cell performs 16-bit or 32-bit operations. CGRAs provide high level of data parallelism and throughput because of the symmetry in their structure. However, they occupy an area of a few million gates, which makes them expensive and can have a potentially high transient power dissipation. The algorithms related to TS, FFT and CE are parallel in nature. Based on the amount of computations required by each of them as well as their algorithms, they may have better performance on FPGA or CGRA. Regarding the Symbols Demapping block, it was implemented by SW in both platforms instead of HW due to its faster implementation. Moreover, the estimated speed-up gained by hardware accelerators compared to the software-only measured clock cycles can be observed from the rightmost column of the Table 1. As it was predicted, considerable speed-up can be achieved by using the hardware accelerators. For the whole OFDM receiver, ZC706 evaluation board could give speed-up of 3.94X compared to the HARP platform due to the different clock frequency and NoC used for exchanging the data. Table 2 presents the resource utilization summaries of each of the designed accelerator on Zynq®-7000 all programmable SoC ZC706 evaluation kit. Table 2 also presents the comparison of the resource utilization of HARP and ZC706 evaluation board. It has to be mentioned that a single ALM in Stratix-V FPGA device is composed of two LUTs. The total number of 18-bit DSP resources utilized in the case of HARP is 230 (90%) which depends on the number of 32-bit multipliers instantiated. Each 32-bit multiplier instantiated in a PE requires two 18-bit DSP elements to be synthesized on the FPGA. In the case of the ZC706 evaluation board, a 48-bit DSP block is an arithmetic logic unit used by hardware functions and composed of an add/subtract unit and a multiplier connected to a final add/subtract/accumulate engine. It means that against HARP platform where DSP was just used for multiplication, the employed DSP blocks in the ZC706 evaluation board can act as an arithmetic logic unit. Regarding the power consumption, it is estimated by using Xilinx Power Estimator (XPE) tool for the whole OFDM receiver on the ZC706 evaluation board. It shows 3.171 W for total on-chip power at an ambient temperature of 25°C which includes 0.243, 2.18 and 0.748 for static, Processor System (PS) + dynamic and I/O power, respectively. In this context, Zynq®−7000 all programmable SoC ZC706 evaluation board shows an overall improvement in terms of total power dissipation compared to the HARP platform with the value of 3.9 W (1.22X). Regarding the comparison among HARP platform and Xilinx Zynq

Table 1: Clock cycles required for processing in two platforms at different stages. Moreover, * and † stand for those algorithms which have been executed by SW instead of HW and data transfer from HW to SW, respectively.

Platform	HARP (CC) at 200 MHz			SDSoC (CC) at 667 MHz					Gain	Gain
Accelerator	Data transfer	HW	Execution Time (μs)	SW	Execution Time (μs)	Data transfer	HW	Execution Time (μs)	(SDSoC HW vs HARP)	(SDSoC HW vs SW)
TS	9,619	3,465	17.33	14,082	21.11	6,642	9,549	14.32	1.21X	1.69X
FOE	4,664	12,708	63.54	14,366	21.53	10,280	4,747	7.12	8.92X	3.16X
FFT	2,677	328	1.64	9,282	13.91	3,160	2,503	3.75	0.44X	3.71X
CE	1,435	250	1.25	5,276	7.91	2,016†	2,164	4.74	0.38X	4.86X
SD	-	2,253*	16.27	3,724	5.58	-	-	-	4.03X*	-
OFDM Receiver	-	31,521	157.6	46,730	70.1	-	26,677	40	3.94X	1.75X

Table 2: Synthesis results of the proposed accelerators on ZC706 evaluation board and also resource utilization summary of HARP (Stratix-V (5SGXEA4H1F35C1) FPGA device).

Accelerator	Resource Utilization					
	ZC706				HARP [5]	
	BRAM	FF	LUT	48-bit DSP	ALMs	18-bit DSP
TS	0	1,019	563	34	22,616	40
FOE	0	41,506	47,838	459	8,171	32
CE	24	29.664	5,143	153	22,809	56
FFT	4	26,900	24,894	103	25,908	66
SD	0	1,875	9,101	0	0	0
Total	28	100,964	117,539	749	98,729	230
%	(5.14)	(23.09)	(53.77)	(83.22)	(62)	(90)

SoC, as the clock frequencies are different, largely based also on the different FPGA platforms used, the comparison is not a completely fair. Probably HARP would achieve a higher clock frequency on the ZC706 board, but we do not know as we do not have such an implementation of HARP reported. Even if HARP would run at the same clock frequency as implemented design on ZC706, we would achieve a slight speed-up over HARP. Therefore, a direct comparison is difficult and the results are only indicative.

6 CONCLUSION

In this work, a novel HW/SW codesign of IEEE 802.11a/g baseband processing using a high-level synthesis tool is presented. The HW/SW codesign is mapped to an ARM processor and an FPGA. The IEEE 802.11a/g receiver has computationally intensive parallel and serial tasks that have a high potential to be accelerated by the FPGA using a high-level synthesis tool. Therefore, important insights into the explored platform can be provided from the implementation results as well as a comparison with another state-of-the-art heterogeneous platform (HARP). It can be concluded that the implementation of inherently parallel algorithms such as FFT is more efficient by employing Coarse-Grained Reconfigurable Architectures (CGRAs) instead of using high-level-synthesis for FPGAs. Moreover, the users of CGRAs can implement near-optimal solutions for their target applications by employing various design space exploration techniques. On the other hand, for the mixture of serial and parallel tasks, Xilinx Zynq SoC platform is more powerful since all parts of an algorithm can be mapped into hardware/software. In contrast to the HARP platform and also software implementation of the receiver on Zynq, the presented approach shows an overall speed-up of 3.49X and 1.75X, respectively.

ACKNOWLEDGMENTS

This research work is jointly conducted by the Laboratory of Electronics and Communications Engineering, TUT, Tampere, Finland and the Application-specific Multi-Core Architectures Research Group, RUB, Bochum, Germany. This work was partially funded by the Academy of Finland under contract # 258506 (DEFT), TUT Graduate School, Tuula and Yrjö Neuvo fund, HPY Research Foundation, Tekniikan Edistämissäätiö (TES) Foundation and HiPEAC.

REFERENCES

[1] D. Pizzolli et al., "Cloud4IoT: A Heterogeneous, Distributed and Autonomic Cloud Platform for the IoT," 2016 IEEE International Conference on Cloud Computing Technology and Science (CloudCom), Luxembourg City, 2016, pp. 476-479. doi: 10.1109/CloudCom.2016.0082

[2] M.B. Taylor, "Is dark silicon useful?: harnessing the four horsemen of the coming dark silicon apocalypse.", In proceedings of the 49th Annual Design Automation Conference (DAC âĂŹ12) (pp. 1131-1136). NY, USA: ACM.

[3] "Introduction to FPGA Design with Vivado High Level Synthesis", UG989 (v1.0), July 2, 2013. Available at: http://www.xilinx.com

[4] "SDSoC Environment Tutorial", UG1028 (v2016.3) November 30, 2016. Available at: http://www.xilinx.com

[5] S. Nouri, W. Hussain and J. Nurmi, "Evaluation of a Heterogeneous Multicore Architecture by Design and Test of an OFDM Receiver", IEEE Transactions on Parallel and Distributed Systems, 2016, submitted and under review.

[6] D. Melpignano and et. al., "Platform 2012, a Many-Core Computing Accelerator for Embedded SoCs: Performance Evaluation of Visual Analytics Applications", in Proc. 49th Annual Design Automation Conference (DAC '12). ACM, New York, NY, USA, 1137-1142.

[7] F. Conti and et. al., "He-P2012: architectural heterogeneity exploration on a scalable many-core platform", in Proc. of the 24th edition of the great lakes symposium on VLSI (GLS- VLSI '14), pp. 231-232, ACM, New York, NY, USA.

[8] N. S. Voros and et. al., "MORPHEUS: A Heterogeneous Dynamically Reconfigurable Platform for Designing Highly Complex Embedded Systems", ACM Trans. Embed. Comput. Syst. 12, 3, Article 70, 33 pages, April 2013.

[9] F. Thoma and et. al., "MORPHEUS: Heterogeneous Reconfigurable Computing", International Conference on Field Programmable Logic and Applications, FPL 2007, pp. 409-414, 27-29 Aug. 2007.

[10] W. Hussain and et. al., "HARP2: An X-Scale Reconfigurable Accelerator-Rich Platform for Massively-Parallel Signal Processing Algorithms", in Journal of Signal Processing Systems, Springer, vol. 85, issue 3, pp. 341-353, 2016.

[11] C. Brunelli and et. al., "A Dedicated DMA Logic Addressing a Time Multiplexed Memory to Reduce the Effects of the System Bus Bottleneck", in Proc. 18th International Conference on Field Programmable Logic and Applications, (FPL 2008), Heidelberg, Germany, pp. 487-490, 8-10 September 2008.

[12] J. Heiskala and J. Terry, "OFDM Wireless LANs: A Theoretical and Practical Guide", Copyright ©2002 by Sams Publishing, SAMS, 201 West 103rd St., Indianapolis, Indiana, 46290 USA.

[13] J.-J. van de Beek and et. al., "A time and frequency synchronization scheme for multiuser OFDM", IEEE Journal on Selected Areas in Communications, vol.17, no.11, pp.1900-1914, Nov. 1999.

[14] Man-On Pun and et. al., "Multi-carrier techniques for broadband wireless communications : a signal processing perspective", copyright ©2007 by Imperial College Press, December 2007.

[15] Y. Kim and et. al., "Design Space Exploration for Efficient Resource Utilization in Coarse-Grained Reconfigurable Architecture," in IEEE Transactions on Very Large Scale Integration (VLSI) Systems, vol. 18, no. 10, pp. 1471-1482, Oct. 2010. doi: 10.1109/TVLSI.2009.2025280

[16] J. W. Cooley and J. W. Tukey, "An algorithm for the machine calculation of complex Fourier series", Math. Comp., vol. 19, pp, 297-301, April 1965.

Publication IV

Reprinted with permission from *IEEE Transactions on Parallel and Distributed Systems*, vol. 28, no. 11, pp. 3171-3187, Nov. 2017, S. Nouri, W. Hussain, and J. Nurmi, "Evaluation of a Heterogeneous Multicore Architecture by Design and Test of an OFDM Receiver". doi: 10.1109/TPDS.2017.2706691

Publication V

Reprinted with permission from the Publisher of the *Elsevier Journal of Microprocessors and Microsystems (MICPRO)*, Sep. 2018, S. Nouri, D. Rossi and J. Nurmi, "Power Mitigation of a Heterogeneous Multicore Architecture on FPGA/ASIC by DFS/DVFS Techniques". doi.org/10.1016/j.micpro.2018.09.010

Power Mitigation of a Heterogeneous Multicore Architecture on FPGA/ASIC by DFS/DVFS Techniques

Sajjad Nouri[a,*], Davide Rossi[b], Jari Nurmi[a]

[a] *Laboratory of Electronics and Communications Engineering, Tampere University of Technology, Tampere, Finland*
[b] *Department of Electrical, Electronic and Information Engineering "Guglielmo Marconi" (DEI), University of Bologna, Bologna, Italy*

Abstract

This article presents an integrated self-aware computing model in a Heterogeneous Multicore Architecture (HMA) to mitigate the power dissipation of an Orthogonal Frequency-Division Multiplexing (OFDM) receiver. The proposed platform consists of template-based Coarse-Grained Reconfigurable Array (CGRA) devices connected through a Network-on-Chip (NoC) around a few Reduced Instruction-Set Computing (RISC) cores. The self-aware computing model exploits Feedback Control System (FCS) which constantly monitors the execution-time of each core and dynamically scales the operating frequency of each node of the NoC depending on the worst execution-time. Therefore, the performance of the overall system is equalized towards a desired level besides mitigating the power dissipation. Measurement results obtained from Field-Programmable Gate Array (FPGA) synthesis show up to 20.2% dynamic power dissipation and 16.8% total power dissipation savings. Since FCS technique can be employed for scaling the frequency and the voltage and on the other hand, voltage supply cannot be scaled on the FPGA-based prototyped platform, the implementation is also estimated in 28nm Ultra-Thin Body and Buried oxide (UTBB) Fully-Depleted Silicon-On-Insulator (FD-SOI) Application-Specific In-

*Corresponding author
Email addresses: sajjad.nouri@tut.fi (Sajjad Nouri), davide.rossi@unibo.it (Davide Rossi), jari.nurmi@tut.fi (Jari Nurmi)

tegrated Circuit (ASIC) technology to scale voltage in addition to frequency and get more benefits in terms of dynamic power dissipation reduction. Subsequent to synthesizing the whole platform on ASIC and scaling the voltage and frequency simultaneously as a Dynamic Voltage and Frequency Scaling (DVFS) method, significant dynamic power dissipation savings by 5.97X against Dynamic Frequency Scaling (DFS) method were obtained.

Keywords: Reconfigurable, CGRA, Network-on-Chip, Heterogeneous, Accelerator, Multicore, FFT, Time Synchronization, Channel Estimation, Frequency Offset Estimation, Receiver, OFDM, FCS, DVFS, Power Mitigation, FPGA, ASIC.

1. Introduction

In 1965, Moore's Law claimed that the number of transistors that can be integrated on a single chip doubles every one to two years [1]. After that in 1974, Dennard showed that the power dissipation density can be kept constant by scaling the CMOS devices while voltage and current should be proportional to the linear dimensions of a transistor [2]. However, this trend is not continuing anymore since Dennard ignored the baseline of power per transistor established by leakage current in modern technologies, resulting to enormous heat dissipation. Then RAW microprocessor architecture as a multicore has been introduced in order to solve the problem of heat dissipation and power density by exploiting several processors operating in parallel at a lower frequency instead of a single core operating at very high frequency [3]. Although introducing RAW microprocessor was a big step for dealing with the mentioned issue, the experimental work explained in [4] showed that only 7% of a 300 mm^2 chip can be operated at full frequency under a power budget of 80W which put an end to multicore scaling. It also stated that on a single chip, all cores cannot be clocked at their maximum operating frequency for a given Thermal Design Power (TDP) constraint which forced a large fraction of chip to be to be switched-off (dark) or to be operated at very low frequency (dim). Then, four different approaches

as *Four Horsemen* (Shrink, Dim, Specialized and Deus Ex Machina [5]) have been suggested in order to deal with the issue of "utilization wall" that causes the Dark Silicon problem [4]. Among them, *Specialized Horseman* proposed domain-specific accelerators and in particular Coarse-Grained Reconfigurable Arrays (CGRAs) to be instantiated to avoid the dark and dim parts of the chip [5]. CGRAs are operated at a very low frequency and can be reconfigured at run-time for multiple applications and yield tremendous performance improvements exploiting computational units working in parallel [10]. Thus, the dark part of the chip can be employed for performing massively-parallel workloads of critical-priority applications [5]. Consequently, the execution time of those kernels would be reduced significantly in addition to operating at a very low frequency.

Despite all the benefits can be obtained from CGRAs, they have potentially high power dissipation. Power consumption can be mitigated by employing various techniques such as Dynamic Voltage and Frequency Scaling (DVFS) and clock gating [8]. There are several case-studies in the literature, presented in the next section, which show the effects of DVFS method for mitigating the power and maximizing the performance. The thermal problems can be solved by employing DVFS which keeps the temperature under a given threshold [6]. According to the level of granularity, DVFS schemes can be varied from fine-grained to coarse-grained which are suitable for modifying the frequency of each resource separately (better energy efficiency) or scaling the performance of the whole platform (maximum performance), respectively [7]. With the rising complexity of extreme-scale computer systems and increasing the need of maximum performance besides minimum energy consumption, SElf-awarE Computing (SEEC) models have been proposed in order to create self-adaptive computing systems with the ability of changeable behavior according to the performance requirements [8]. In this context, Feedback Control System (FCS) is an advanced technique to reduce the dynamic power dissipation by constantly monitoring the execution time and recognizing the worst-case one and then scaling the frequency and the voltage to meet the desired performance level. These systems

can also exploit the inertia of heat dissipation to boost performance of some resources which can be useful for execute some urgent task, but only for short times only to avoid overheating [26].

This experimental work presents the power mitigation of the Heterogeneous Accelerator-Rich Platform (HARP) [9] platform to be used as an Orthogonal Frequency Division Multiplexing (OFDM) receiver by applying Dynamic Frequency Scaling (DFS) method on Field-Programmable Gate Array (FPGA) and also on Application-Specific Integrated Circuit (ASIC) technology. The designed platform, modified to be used as an OFDM receiver, is composed of seven nodes, three Reduced Instruction-Set Computing (RISC) processors and four specialized template-based CGRAs in order to perform a fine mixture of parallel and serial tasks provided by OFDM receiver. It should be mentioned that although one RISC core is also sufficient for performing this task, however due to the data dependency between the nodes to perform an OFDM receiver completely as well as increasing the level of parallelism, three RISC cores are employed. The RISC processors are responsible for continuously monitoring the execution time of each computing-node, calculating the worst-case execution time and then upgrading or downgrading the clock frequencies and supply voltage of the slave nodes to minimize the overall power dissipation. Furthermore, the power overhead of these controlling cores is also taking into account when the energy improvement of FCS is computed. For this purpose, the FCS technique is implemented in the processor software of three RISC cores to perform dynamic frequency scaling by identifying the worst-case execution time and then tune the operating voltage and frequency of the computing-nodes to meet the performance thresholds. In order to evaluate the benefits of voltage scaling on top of the FCS technique, we have also synthesized the HARP system in 28nm Ultra-Thin Body and Buried oxide (UTBB) Fully-Depleted Silicon-On-Insulator (FD-SOI) ASIC technology as an extension version of [31] and estimated the power consumption of the nodes at different supply voltages, starting from the nominal voltage supply of the technology (1V) down to the deep near threshold regime (0.5V). Therefore, power saving can be guaranteed as a vital issue

for telecommunication systems while dark and dim part of the chip can remain operable because of overall heat dissipation reduction. The proposed approach reduces the dynamic power consumption on FGPA by 20.24% and by 97.61% on ASIC when exploiting DVFS.

The rest of the paper is organized as follows. Section II presents a brief survey of already existing related work about DVFS method in multicore platforms. Section III describes shortly the architecture of HARP platform with its overall functionality, the architecture of integrated template-based CGRAs, the internal structure of the Network-on-Chip (NoC) nodes and the execution flow. In Section IV, applying the FCS technique on OFDM receiver test case for mitigating the power and equalizing the performance of the overall multicore platform on FPGA and ASIC is provided. Section V presents the achieved results in terms of power saving as well as measurements and estimations of different performance metrics subsequent to applying the FCS technique with frequency and voltage scaling. Finally, in Section VI, conclusions are discussed.

2. Related Work

HARP is designed in order to maximize the number of computational resources for accelerating many specific computationally intensive algorithms besides investigating the issues related to Dark Silicon. In addition to HARP, other state-of-the-art homogeneous and heterogeneous computing platforms have been introduced so far such as Platform 2012 ([12], [13]) and MORPHEUS ([14], [15]) which is one of the most promising heterogeneous platforms. MORPHEUS has been designed by Rossi et. al. [16] as a heterogeneous digital signal processor for dynamically reconfigurable computing based on a 64-bit NoC. The platform has been developed by combining a Fine-grained embedded FPGA, a Mid-grained configurable processor and a CGRA for exploiting DFS to mitigate the power consumption. The supervisor node is an ARM 926EJ-S RISC processor which monitors communication, synchronization and reconfiguration mechanisms. This platform has been fabricated in 90-nm technology with the

size of 110 mm^2 delivering 120 GOPS on a video surveillance motion detection application with the power dissipation of 1.45 W and peak power consumption of 2.5 W. In another case-study [17], Fulmine, a 65 nm System-on-Chip (SoC) based on a tightly coupled multicore cluster supported with specialized blocks for computationally intensive tasks has been developed to be used as the emerging class of smart secure near-sensor data analytics for Internet-of-Things (IoT) end-nodes without voltage scaling. The proposed multicore platform achieved the power consumption of 20 mW on average at 0.8V, up to 25 MIPS/mW in software, 315 GOPS/W, low energy, potentially high speed and low-effort data exchange among processing engines. As a general comparison among the state-of-the-art homogeneous and heterogeneous computing platforms, Fulmine is heavily specialized for IoT application and does not support general purpose, MORPHEUS does not support template-based CGRAs hence might be worst in terms of performance, while HARP provides a better trade-off in terms of general purpose ability, performance and scalability since the template-based CGRA can be specialized for different application domains. Moreover, in the case of HARP, computational resources can be allocated at design time such that a core does not over or under perform relative to the overall execution time frame.

Many authors have tried to mitigate the power dissipation of different homogeneous/heterogeneous multicore platforms by using DVFS techniques and SEEC models. In [10], the authors have applied software-defined FCS on a heterogeneous platform and equalized its performance which resulted in mitigation of the overall dynamic power dissipation by 20.7%. In another case study [18], the thermal problems of 3D multicore processors, caused by high power density because of the stacking of multiple layers vertically, are investigated by employing an adaptive DFS technique which resulted in reduction of the peak temperature by up to 10.35°C. In [19], the energy level of shared resources such as NoC and Last-Level Caches (LLCs) in multicore processor designs is reduced by 56% by applying a DVFS method. In [20], energy aware task parallelism has been presented for CGRAs which relies on resource allocation graphs, autonomous

parallelism, voltage and frequency selection algorithms. They achieved considerable reduction in energy, power and cofiguration memory requirements of up to 36%, 28% and 36%, respectively in comparison with three different state-of-the-art DVFS algorithms. In another case-study [21], proportional integral derivative controller technique as a DVFS technique was proposed for manycore systems to keep the operating temperature within the thermal design power bounds and therefore, mitigate the power consumption based on the certain power limit. They proved the effectiveness of their method by enhancing the system throughput up to 43%. In [22], authors achieved 8.4X energy saving in a self-aware processor with the ability of self-adaption by monitoring the energy consumption. In [23], authors concluded that self-aware systems can play an important role for the IoT systems by adding the following features: enabling complex high-volume IoT architectures, performing configuration and adaption at run-time, enabling safety-critical application by adding planning and modeling to the IoT system's infrastructure and etc. In another case-study [24], the authors employed DVFS technique to a homogeneous multi-processor architecture and achieved the improvement of the overall system performance by a factor of 3 compared to clock gating and also considerable energy saving. In a case-study performed by Rossi et.al. [25], a software controllable self-aware architecture exploiting Body Biasing (BB) has been implemented in 28-nm ultra thin body and box fully depleted silicon on insulator technology for compensation of parameters, operational voltage and temperature and for implementation of low-power modes in near-threshold processors. BB is an advanced technique in which a voltage will be applied to the body contact of CMOS transistors and accordingly, the effective transistor threshold voltage can be shifted and also the leakage power consumption can be reduced. In this study, the wide range forward BB (FBB) and reverse BB (RBB) were employed for reducing the design time margins and introducing a low-power mode and state-retentive sleep mode. According to the conducted experiment results in [25], design margins reduced enormously while the energy efficiency of the processor improved by 32% with a hardware cost of less than 1% and a runtime cost for software control of less

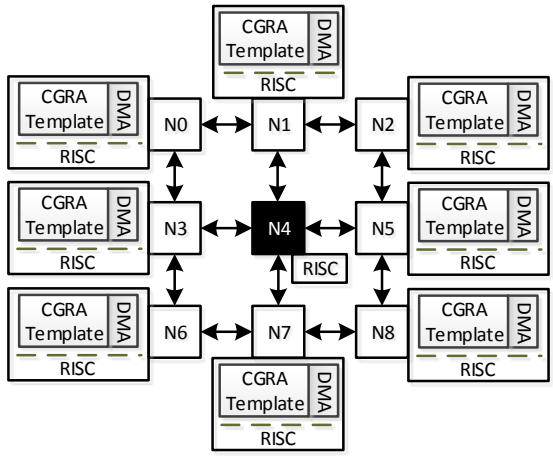

Figure 1: An overview of the HARP architecture applying a processor/coprocessor model [30].

than 0.01%. Furthermore, 24x area reduction for the compensation loop and 21.2x better efficiency were achieved in [25] compared to their previous design.

In this paper, the proposed FCS technique with the high-level of complexity due to applying on three RISC cores simultaneously, reduces the overall power consumption (as a vital issue for telecommunication systems and IoT purposes) significantly on FGPA and ASIC when exploiting DFS/DVFS methods.

3. The Heterogeneous Multicore Platform

The HARP template, as is depicted in Figure 1, consists of nine nodes arranged in a topology of three rows and three columns over a NoC. The central node which is responsible for General-Purpose Processing (GPP) and overall platform supervision is always integrated with a RISC core (called COFFEE [27]) while the rest of the nodes can be employed to be a RISC core or instances of the template-based CGRAs as coprocessors for performing computationally intensive tasks. According to the application requirements, nodes can exchange data between each other at run-time in the case of data dependency or even

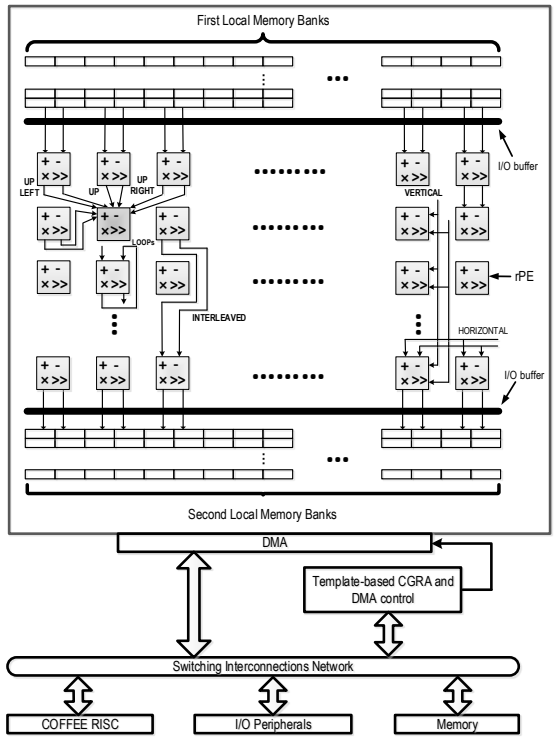

Figure 2: The architecture of scalable template-based CGRA [30].

work independently and simultaneously.

The template-based CGRAs integrated in the HARP template (called CREMA, AVATAR [28]), shown in Figure 2, vary in terms of their sizes based on the application requirements while their architectural features are almost the same. Template-based CGRAs are equipped with the arrays of Processing Elements (PEs) while the number of rows and columns of PEs can be scaled-up/down based on the proposed applications and their algebraic expressions in order to be performed efficiently. The functionality of each PE and also interconnection between PEs in a point-to-point fashion with multiple routing possibilities can

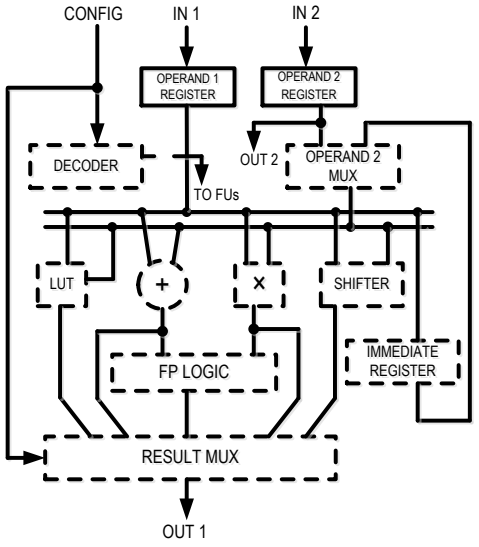

Figure 3: PE Core Template.

be specified at design time by the designer. The internal structure of each PE is depicted in Figure 3. Every PE receives two input operands and executes a 32-bit integer or a floating-point operation (IEEE-754 format). A PE is composed of a Look-Up Table (LUT), adder, multiplier, shifter, immediate register and floating-point logic. The blocks which are shown with the dashed borders are selectivity instantiated according to the processing requirements of an algorithm's algebraic expression at design-time while the two input operand registers are always instantiated. At runtime, the functionality and routing are controlled by reconfiguration. The data can be interleaved between data local memories and the PEs by using the I/O buffers which are made of sixteen 16 or 32 $\times$ 1 multiplexers and 16 or 32 32-bit registers for CREMA and AVATAR, respectively. Each PE has interconnections with neighboring PEs in a point-to-point fashion with the following routing possibilities, i.e., local, interleaved and global. In order to perform an application, a number of configuration contexts, designed

at the system design-time and enabled at run-time, may required. A context is the pattern of interconnections among all PEs and the set of operations to be performed by each PE at any clock cycle. According to the execution flow of an application, the contexts can be switched at run-time by deploying the configuration words, consisting of an address and operation field to determine the task of each PE and its destination address, respectively. They are stored during the system startup time at configuration memory of PEs by the Direct Memory Access (DMA) device [29]. The overall control flow of the designed template-based CGRAs, performed by COFFEE RISC core, is programmable in C language. Once an application-specific accelerator is designed, it can be integrated with one of the existing network nodes.

As it can be observed from Figure 1, nodes are connected to each other in a point-to-point fashion. The slave nodes integrated with the template-based CGRA can exchange data between their local memories directly or with the help of RISC cores. Each node of the NoC has one master and two slave interfaces. The master one, integrated with the RISC core, can be employed for writing to the network and transferring data within a node while the slave interfaces are used for controlling the clock frequency and supply voltage selection, and integrating the data memory. The central node is integrated with the RISC core as a supervisor node for transferring data in the form of packets among its own data memory and data memories of the slave nodes. The target devices for transferring the data packet gets selected by using the switches integrated in each node according to the information in the routing field of the transported packet which can point to the data memory of the respective node or the DMA slave. Moreover, in the case of data dependency between the nodes, the supervisor node is responsible for establishing synchronization for data transfer between two different nodes by using an allocated shared memory space. The software/hardware co-design flow for template-based CGRAs integrated over HARP can be listed as follows:

- Defining the functionalities of the PEs and routing among them by using

a Graphical User Interface (GUI) tool;

- Generating the configuration files for mapping and run-time reconfiguration;
- Loading the configuration data in the template-based CGRAs at the system start-up time by using DMA device which can be used for switching the contexts and performing reconfiguration;
- Loading the data to be processed into one of the local memories of the template-based CGRAs by using the DMA device monitored by the host RISC core;
- Enabling a context for configuring the functionalities of the PEs and the routing between them;
- Processing the data over the PE array;
- Switching the context in order to reconfigure the template-based CGRAs for performing the new task;
- As required, loading the new set of data and iterating the above-mentioned steps;
- Transferring back the results from the local memory of a CGRA node to the data memory of its host RISC core for further processing;
- Iterating the above phases until the algorithm completes its execution.

More information about the execution flow of performing an particular application by the use of template-based CGRAs as well as the application mapping on the HARP platform and the internal structure of the NoC can be found in [30].

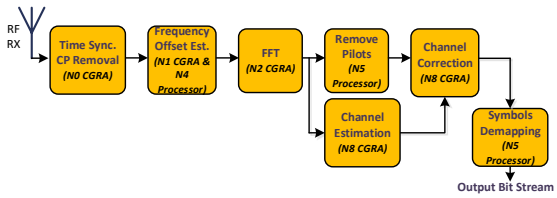

Figure 4: A simplified block diagram of an OFDM receiver.

4. Equalization of the OFDM Receiver Performance by Frequency and Voltage Scaling

Prior to discussing about the equalization of the OFDM receiver performance by using DFS and DVFS methods, let us have a brief explanation about the design and implementation of an OFDM receiver blocks on HARP as well as the reasons behind selecting OFDM application as a test-case.

4.1. Design and Implementation of an OFDM Receiver Blocks on HARP

OFDM is an important data transmission scheme in Software Defined Radio (SDR) technology because of providing high data rates and it is also a candidate to be employed for 5G wireless systems. OFDM receiver blocks as a critical part due to retrieving the data after the noisy channel, are composed of most computationally intensive and time-consuming tasks such as Fast Fourier Transform (FFT), Correlation, Convolution and Complex Matrix-Vector Multiplication (MVM) which require parallel processing, shown in Figure 4. Furthermore, there are some tasks such as Frequency Offset Estimation which require the GPP of some serial in nature algorithms such as Taylor series and CORDIC algorithms. Such a fine mixture of parallel and serial algorithms makes the OFDM receiver as a valuable test-case in order to evaluate almost all the design features and technical capabilities of the HARP template as well as to identify potential architectural fallacies and pitfalls.

As it is depicted in Figure 4, the parallel algorithms are implemented by crafting template-based CGRAs while the general-purpose tasks such as Fre-

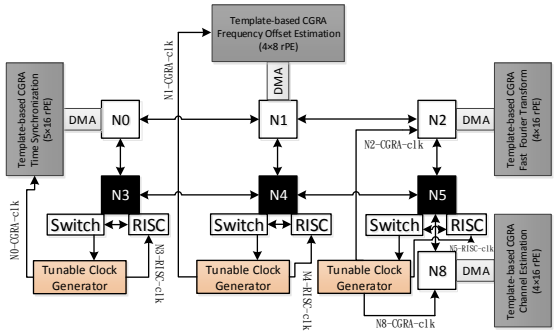

Figure 5: A simplified overview of the HARP architecture with three RISCs and four template-based CGRAs in a processor/coprocessor model [31].

quency Offset Estimation are executed by using both template-based CGRA and RISC processor. Subsequent to mapping the kernels on the designed CGRA accelerators, they can be integrated over HARP in such a way that both master and slave nodes can exchange data with each other (Figure 5). According to Figure 4, the order of exchanging the data in Figure 5 for performing the OFDM receiver blocks should be initiated by N0 CGRA node (belonging to Time Synchronization block). After passing through N1 and N2 which are associated with Frequency Offset Estimation and FFT, respectively, ending up with N8 CGRA node (at Channel Estimation block). The detail of design and implementation of each block of an OFDM receiver which led to scale CGRAs to different dimensions can be found in [30].

4.2. Performance Equalization by DFS/DVFS Methods

The overall architecture of HARP, modified for performing an OFDM receiver functionality is depicted in Figure 5. Each of three RISC cores acts as an controller and observer for monitoring the performance of their associated CGRA nodes and transferring the configuration stream and data to be processed in a way that node N3 RISC is responsible for N0 CGRA while N4 RISC is responsible for N1 CGRA and N5 RISC is responsible for N2 and N8 CGRAs.

Since there are multiple CGRA nodes with different dimensions in the platform those are supposed to exchange data with each other, running a core faster than the other is not desirable. Subsequent to transferring the configuration stream by three RISC cores in parallel, the data will be loaded into the local memories of N0 in order to implement Time Synchronization block. Then the number of clock cycles required for the execution of the Time Synchronization block counted by using a special counter of the RISC processor. The counted number of clock cycles for complete execution of Time Synchronization block will be transfered to the N4 RISC core and stored at reserved locations in its data memory for seeking the worst-case execution time. Total clock cycles required for performing each block of OFDM receiver contains the following stages: data memory to data memory, data memory to local memory of the CGRA and vice versa and also the CGRA execution time. Since each computing-node has to wait until the other one completes its execution because of the data dependency between the receiver blocks, all three RISC cores establish synchronization between each other by writing 'read' and 'write' flags in their shared memory location.

The CGRA accelerator blocks of the platform form a software-defined macro-pipeline. By completion the process of Time Synchronization block, an acknowledgment will be sent by DMA's master from N3 to N4 which gives the permission to N4 for starting the process of Frequency Offset Estimation block. Other CGRAs will also perform their tasks with the same procedure and at the end of the first iteration, the data related to the counted number of clock cycles of four worker-nodes will be retrieved in order to recognize the stored most time consuming CGRA (worst-case). The total number of clock cycles related to the nodes N0, N1, N2 and N8 are equal to 9985, 18039, 1931 and 1685 CC, respectively [30]. Based on the achieved results, node N5 CGRA, which contains the Frequency Offset Estimation block, is the most time consuming one and can be selected as a worst-case candidate by master node N4 RISC core. Therefore, N4 as a supervisor node will notify the other two RISC cores about the selected worst-case candidate. From the second round of iteration, nodes N3,

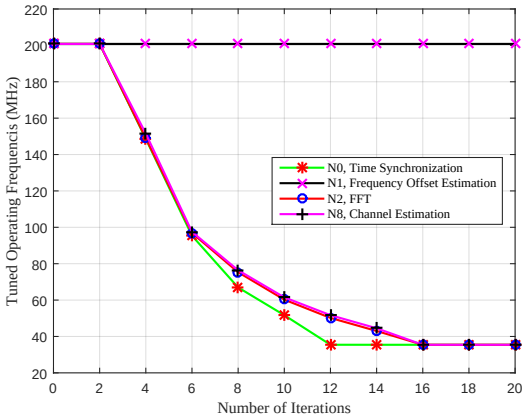

Figure 6: Tuning the operating frequencies in the range of ≈35.0-200.0 MHz on FPGA prototype [31].

N4 and N5 RISC cores will tune the operating frequency of the CGRA cores belonging to them in order to approach the defined equalization region. The FCS technique has been implemented completely in RISC software to perform dynamic frequency scaling by reducing each core operating frequency to approach the performance goal which can be identified by comparing the counted clock cycles of the CGRAs and selecting the worst-case execution-time. The clock frequencies of the CGRAs can be updated through a module emulating a DVFS Power Management Unit (PMU). This PMU includes a 32-bit general-purpose register, defined as allocated 4-bit field (16 bits in total for four cores) for each one when in DVFS mode (on ASIC), the corresponding supply voltage supporting the clock frequency is also selected. When the frequency is updated the PMU clock gates the part of the systems where the frequency or voltage is changed. This guarantees that the subsystem is not operating during the transitions of frequency and voltage, and that operation is safely restored once the voltage/frequency are stable.

Each RISC core can tune the clock frequency of its associated CGRA by

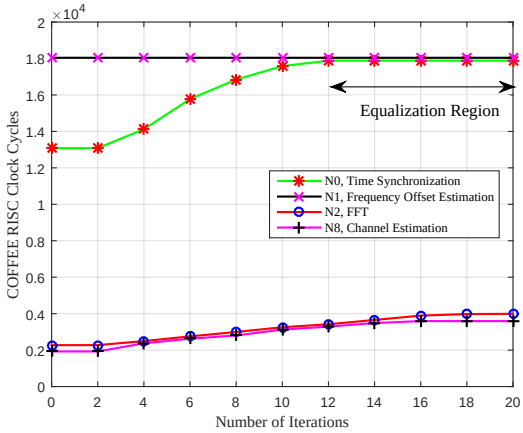

Figure 7: Performance Equalization of the CGRAs based on the Worst-Case Execution Time on FPGA prototype [31].

using this 4-bit field between 16 different frequencies in the range of 35.0-200.0 MHz on the FPGA prototype (shown in Fig. 6). Meanwhile, the operating frequency of the three RISC cores is maintained fixed at 100.0 MHz and is set as a reference of all measurements. At each iteration, FCS software adds or subtracts a 4-bit field by "0001" in order to update the current operating frequency of the CGRA. It has to be mentioned that during the iterations, the clock frequency of the RISC cores will remain the same. As it can be observed from Fig. 6, the clock frequency of all the CGRA cores operate at the maximum possible operating frequencies during the system start-up time, near to 200.0 MHz (achieved after placement and routing, to be explained in detail in the next section). During the first iteration, the execution time of the CGRA cores and accordingly, the worst-case candidate can be identified. From the second iteration onwards, FCS will automatically update the target and start to tune the operating frequency of other CGRA nodes for twenty iterations in order to approach the equalization region. The performance equalization of the CGRA cores is also depicted in Fig. 7. It can be seen that by reaching the

12^{th} iteration, N0 successfully approached the goal while running at 35.0 MHz operating frequency. However, for the other two computing-nodes (N2 and N8), because of the smaller workload, lower computation complexity algorithm and relatively shorter execution time, their performance could not degraded to the equalization region even with running at 35.0 MHz operating frequency during the last iterations.

Subsequent to synthesizing on FPGA, the same procedure is performed by synthesizing the modified HARP template on ASIC technology and dynamically scaling both frequency and the voltage within the range of 55.0-500.0 MHz and 0.5-1V, respectively. In this case, the dynamic power dissipation of the nodes is estimated in several operating conditions, first just DFS at 1V and then DVFS, explained in detail in the next section. Furthermore, since the comparison among power mitigation of the platform on FPGA and ASIC is not a completely fair, as the clock frequencies are different, we reduced the maximum operating frequency of ASIC from 500.0 MHz to 200.0 MHz which is the same as FPGA one.

5. Measurements and Estimations

The overall HARP platform with the applied FCS was first synthesized on a Stratix-V FPGA device (5SGXEA4H1F35C1) for prototyping the concept and then in 28nm UTBB FD-SOI ASIC technology for estimating the added benefits of including also voltage scaling.

5.1. FPGA Evaluation

We will first take a look at the FPGA prototype results here. Node-by-node breakdown of resource utilization on FPGA is provided in Table 1 in terms of the following metrics: Adaptive Logic Modules (ALMs), Registers, Memory Bits and DSP elements. It has to be mentioned that the number of employed 18-bit DSP resources depends on the number of multiplications performed by the template-based CGRAs where for each 32-bit multiplier, two 18-bit DSP elements are required. More details about tailoring the template-based CGRAs for

performing OFDM receiver algorithms as well as instantiating 32-bit multipliers by using PEs on CGRAs based on their algebraic expressions are explained in [30]. As it can be observed from Table 1, the logic utilization increase due to applying FCS technique is just around 1% compared to [30], which is almost negligible. Subsequent to synthesizing the overall platform on Stratix-V FPGA device by using Quartus II 15.0, two timing models were selected in order to measure the operating frequencies after placement and routing. The timing models used by the Quartus II software could cover worst-case voltage to the minimum and maximum supported Vdd operating conditions for Slow 900mV 85°C and Fast 900mV 0°C, respectively. By using these timing models, the timing of FPGA can be verified without the need to implement physical simulation. In this regard, the maximum achieved operating frequencies for slow timing model at an operation voltage of 900 mV are equal to 163.61 MHz and 188.29 MHz at temperatures of 85°C and 0°C, respectively. In the case of fast timing model (900 mV), the maximum operating frequencies are equal to 246.12 MHz at 0°C and 223.51 MHz at 85°C. First of all, the clock frequency of 170.0 MHz has been used for running the simulations using the ModelSim simulator and then it is increased up to 200.0 MHz as the average of the achieved operating frequencies without any timing error on the particular FPGA instance used.

The platform's power dissipation has been estimated based on post placement and routing (post P&R) information, not functional simulations, using PowerPlay Power Analyzer Tool of Quartus II 15.0 at an ambient temperature of 25°C. The power analyzer provided us a "HIGH" confidence metric in our power dissipation estimations acquired from simulating the gate-level netlist. The power estimations are performed in two cases: inactive state of FCS with the fixed operating frequency of 200.0 MHz and active state of FCS with the tunable operating frequency after every iteration started from 200.0 MHz. Node-by-node breakdown of dynamic power dissipation for the both cases is shown in Table 2. Nodes N0, N1, N2 and N8 are belong to Time Synchronization, Frequency Offset Estimation, FFT and Channel Estimation, respectively while

Table 1: Resource Utilization summary for Stratix-V FPGA device [31].

Node	**ALMs**	**Registers**	**Block Memory Bits**	**(32-bit Multipliers) DSPs**
N0	22,729	8,612	2,633,472	(20) 40
N1	8,255	7,855	2,364,672	(16) 32
N2	24,149	11,326	2,633,472	(28) 56
N3	5,590	5,648	3,145,728	(6) 12
N4	5,646	5,668	4,194,304	(6) 12
N5	5,601	5,700	3,145,728	(6) 12
N8	25,974	15,975	2,633,144	(33) 66
NoC	2,533	4,073	-	-
Total	100,477 63%	66,823 11%	20,750,520 53%	(115) 230 90%

nodes N3, N4 and N5 are belong to General Purpose Processing, Synchronization and Control. In the case of inactive state of FCS, the tool estimated 1243.84 mW, 2623.72 mW, 27.37 mW and 3894.93 mW as static, dynamic, I/O and total power dissipation, respectively. Moreover, it can be found out that by scaling up the size of CGRA (N1 compared to N0, N2 and N5, Fig 5), dynamic power dissipation will be increased in the range of 1.5X-2X. Once the FCS starts to tune the operating frequency of the cores based on the worst-case execution time, the dynamic power dissipation will also start to be reduced, up to 20.2% for the total dynamic power in this case study. In the case of FCS active, the estimated static, dynamic, I/O and total power dissipation for the overall platform showed the value of 1121.19 mW, 2092.6 mW, 27.37 mW and 3239.2 mW, respectively. Compared to the inactive state of FCS, the total power dissipation of the platform is reduced by 16.8%. Although it is proved that the FCS will decrease the instantaneous dynamic power dissipation and therefore the heat dissipation and the dark/dim part of the chip, the energy consumption on the FPGA proto-

Table 2: Dynamic power dissipation of each node and the NoC before/after applying FCS on FPGA prototype [31].

Node	**Dynamic Power FCS Inactive [30] [mW]**	**Dynamic Power FCS Active [mW]**	**Gain %**
N0	414.35	284.43	31.35
N1	272.23	272.83	≃0.0
N2	526.07	391.23	25.63
N3	114.47	104.13	9.03
N4	113.82	114.61	≃0.0
N5	114.52	105.07	8.25
N8	448.01	344.61	23.07
NoC	10.10	14.34	-
Integration Logic	609.07	461.35	24.25
Total	2623.72	2092.6	20.24

type for completing the operations will remain approximately the same, as the decrease of clock frequency will increase the active time proportionally.

5.2. ASIC Evaluation

As the next step, the overall platform is synthesized on 28nm FD-SOI ASIC technology in order to scale both frequency and voltage simultaneously and accordingly, to achieve more power dissipation and energy reduction. The different blocks of the system were synthesized with Synopsys Design Compiler 2014.09 on a 28nm UTBB FD-SOI RVT standard cell library. The power estimations were performed with Synopsys PrimeTime 2013.06 assuming 20% of switching activity. In order to estimate the power consumption of the blocks at different voltages, the libraries have been characterized down to 0.5V with 0.1V steps using Cadence Liberate. The design was synthesized at 1.0V (slow-slow corner, 125C), while the signoff was performed at different operating voltages

Table 3: Power consumption and area utilization of the nodes synthesized on ASIC at the operating frequency of 500.0 MHz (0.9V, ss, 125C) and typical conditions (tt, 25C, 1V) for estimating the power numbers.

Node	**Area [mm2]**	**Leakage Power @ 1V [mW]**	**Dynamic Power [mW] @ 1V, 500 MHz**
N0	1.78	0.0298	51
N1	0.9	0.0146	24.75
N2	1.81	0.0292	48.75
N3	0.79	0.0136	10.91
N4	0.79	0.0130	15
N5	0.79	0.0128	19.65
N8	1.82	0.0294	48.6
Total	8.68	0.1424	218.66

(from 1.0V to 0.5V with steps of 0.1V), in order to provide to the software the knowledge of the maximum operating frequency and power consumption of the system at the different voltage supplies evaluated in this work. Table 3 shows the post-syntesis results in which the maximum achieved operating frequency is 500.0 MHz at nominal voltage in the slow corner (ss, 125°C, 0.9V) and leakage power and dynamic power consumption are measured in typical operating conditions (tt, 25°C, 1V). Similar to FPGA, the area occupation and also the leakage and dynamic power consumption of the CGRA nodes will be almost doubled by increasing the size of template-based CGRAs from CREMA (N1) to AVATAR (N0, N2 and N8).

Subsequent to synthesizing the whole platform on ASIC, we started to scale down both frequency and voltage simultaneously in a way that at each phase, the dynamic power consumption has been measured. However, the measurement of updated leakage power is ignored due to its negligible value. Node-by-node breakdown of dynamic power measurements at each stage with scaled-down frequencies and voltages is depicted in Table 4 where gains are calculated against

Table 4: Impact of DVFS method on the dynamic power dissipation of the nodes as well as the entire platform on ASIC prototype at scaled down voltages and frequencies. DP stands for Dynamic Power.

Node	**DP [mW] @ 0.9V 480.0 MHz**	**DP [mW] @ 0.8V 366.0 MHz**	**DP [mW] @ 0.7V 238.0 MHz**	**DP [mW] @ 0.6V 119.0 MHz**	**DP [mW] @ 0.5V 55.0 MHz**
N0	33.58	19.15	9.65	3.5	1.17
N1	16.3	9.29	4.68	1.7	0.57
N2	32.1	18.3	9.22	3.34	1.12
N3	7.18	4.09	2.06	0.75	0.25
N4	9.88	5.63	2.84	1.03	0.35
N5	12.94	7.38	3.72	1.35	0.45
N8	32	18.25	9.2	3.34	1.12
Total	149.96	85.51	43.09	15.62	5.23
Gain %	31.42	60.89	80.29	92.86	97.61

dynamic power at 1V and 500.0 MHz. It can be observed that by scaling down the operating frequency and the voltage from 500.0 MHz and 1V to 480.0 MHz and 0.9V, respectively, the dynamic power dissipation is reduced by 31.42%. By moving forward in scaling down the operating frequency and the voltage down to 55 MHz and 0.5V, the dynamic power dissipation reduction showed 97.61% saving against to the first stage (500.0 MHz and 1V).

Then we kept the maximum operation frequency at the fixed value of 200.0 MHz, the same operating condition as FPGA, and started to estimate the dynamic power consumption in three following cases: inactive state of FCS, active state of FCS with DFS and also with DVFS. As it can be seen from Table 5, the dynamic power dissipation of N1, Frequency Offset Estimation block, as the worst-case candidate is constant in all three cases while the dynamic power dissipation of other nodes started to be reduced (in total by 64.29%) once the FCS starts to tune the operating frequency of the cores. Therefore, all the cores

Table 5: Dynamic power dissipation of each node before applying FCS, after applying FCS with DFS and also with DVFS on ASIC.

Node	DP[mW] FCS Inactive, 200 MHz	DP [mW] FCS Active with DFS @ 1V	DP [mW] FCS Active with DVFS
N0	20.4	5.61	1.7
N1	9.9	9.9	9.9
N2	19.5	5.36	1.12
N3	4.36	1.2	0.25
N4	6	1.65	0.35
N5	7.86	2.16	0.45
N8	19.44	5.35	1.12
Total	87.46	31.23	14.89

approached successfully to the equalization region, targeted automatically by FCS, while running at 55.0 MHz operating frequency. In order to get more benefits in terms of power mitigation, the supply voltages are also scaled down to 0.5V in parallel with the frequency scaling which resulted in mitigation of the total dynamic power dissipation by 82.98%.

5.3. Mixed Comparison between FPGA and ASIC technologies

In order to have a fair comparison between the OFDM receiver implementation on two different technologies, FPGA and ASIC, the estimated dynamic power dissipation is compared in the case of 200.0 MHz maximum operating frequency for running the whole platform at the condition of active state of FCS with DFS. The results, depicted in Table 6, show the significant power mitigation of ASIC implementation. The general comparison as a visual summary of total dynamic power dissipation of FPGA and ASIC is also shown in Figure 8 by considering the following cases: FPGA with inactive state of FCS, FPGA with active state of FCS, ASIC with inactive state of FCS, ASIC with active state of FCS and DFS and also ASIC with active state of FCS and DVFS. Apply-

Table 6: General comparison of the impact of DFS technique on FPGA and ASIC dynamic power dissipation at the same operating condition.

	Dynamic Power [mW] **FPGA**	**Dynamic Power [mW]** **ASIC**
Node	**FCS Active with DFS**	**FCS Active with DFS @ 1V**
N0	284.43	5.61
N1	272.83	9.9
N2	391.23	5.36
N3	104.13	1.2
N4	114.61	1.65
N5	105.07	2.16
N8	344.61	5.35
Total	2092.6	31.23

ing FCS on FPGA resulted in dynamic power dissipation reduction by 1.25X while on ASIC the amount of achieved power saving with DFS is 7X which can be further reduced by 5.97X with DVFS. This approach paves the way for self-aware systems for energy efficient OFDM receiver, mapped on HARP, by mitigating the signal transition activity over the entire platform with reference to the worst-case performing core.

6. Conclusions

This paper presents the power mitigation of a Heterogeneous Accelerator-Rich Platform (HARP) on Stratix-V Field-Programmable Gate Array (FPGA) device and 28nm Ultra-Thin Body and Buried oxide (UTBB) Fully-Depleted Silicon-On-Insulator (FD-SOI) Application-Specific Integrated Circuit (ASIC) technology by employing a Dynamic Frequency and Voltage Scaling (DVFS) technique in an Orthogonal Frequency-Division Multiplexing (OFDM) receiver test case. The platform consists of three Reduced Instruction Set Computing (RISC) cores and four template-base Coarse-Grained Reconfigurable Ar-

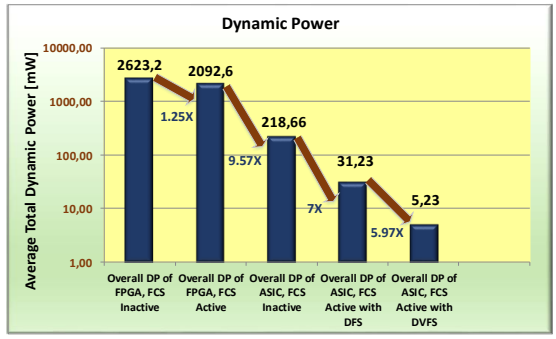

Figure 8: General Comparison of Total Dynamic Power of FPGA Inactive FCS, FPGA Active FCS, ASIC Inactive FCS, ASIC Active FCS with DFS and with DVFS. DP stands for Dynamic Power.

rays (CGRAs) nodes arranged over a Network-on-Chip (NoC). Template-based CGRAs are crafted with different sizes according to the algebraic expressions of computationally intensive tasks of OFDM receiver blocks. Due to the high-level of switching activity on the platform and high power dissipation of CGRAs, dynamically scaling down the operating frequency and the voltage for mitigating the power dissipation is vital. To face this issue, Feedback Control System (FCS) implemented in RISC software continuously monitors the performance of each CGRA and can be employed to tune the clock frequency of the CGRA nodes by selecting a worst-case execution time candidate. RISC processors are responsible to monitor the performance of their associated CGRA cores continuously and store the counted clock cycles in a reserved location of their data memory. FCS uses several iterations to degrade the performance of the CGRA nodes to the equalization region in an user-defined and application-specific range of frequencies. Furthermore, in some cases where the CGRAs are reconfigured at run-time for performing the new tasks, FCS can update itself automatically and define new performance equalization region in order to make the balance between the computing-nodes. In the case of FPGA implementation, the dynamic power dissipation and total power dissipation of the platform showed 20.2% and

16.8% reduction, respectively, subsequent to applying the FCS technique with Dynamic Frequency Scaling (DFS). Furthermore, by moving to ASIC technology, both frequency and voltage have been scaled simultaneously which resulted in significant dynamic power reduction while the nodes approached the equalization region. Achieved results from prototyping the heterogeneous multicore architecture on FPGA and in particular ASIC proved that SElf-awarE Computing (SEEC) models such as FCS by the use of DFS/DVFS techniques can be practically and realistically fruitful as one step forward to mitigate the Dark Silicon issue by reducing the instantaneous power dissipation and therefore the heat dissipation.

Acknowledgment

This research work is jointly conducted by the the Laboratory of Electronics and Communications Engineering, TUT, Tampere, Finland and Department of Electrical, Electronic and Information Engineering "Guglielmo Marconi" (DEI), University of Bologna, Bologna, Italy. This work was partially funded TUT Graduate School, Tuula and Yrjö Neuvo fund, HPY Research Foundation, Tekniikan Edistämissäätiö (TES) Foundation, DELTA doctoral training network, HiPEAC collaboration grant and Nokia Scholarship.

[1] G. E. Moore, "Cramming more components onto integrated circuits", Electronics, Volume 38, Number 8, April 19, 1965.

[2] A. Pedram, S. Richardson, S. Galal, S. Kvatinsky and Mark Horowitz, "Dark Memory and Accelerator-Rich System Optimization in the Dark Silicon Era," in IEEE Design & Test , vol. 34, no. 2, pp. 39-50, April 2017. doi: 10.1109/MDAT.2016.2573586.

[3] M. B. Taylor, J. Kim, J. Miller, D. Wentzlaff, F. Ghodrat, B. Greenwald, H. Hoffman, P. Johnson, Jae-Wook Lee, W. Lee, A. Ma, A. Saraf, M. Seneski, N. Shnidman, V. Strumpen, M. Frank, S. Amarasinghe, A. Agarwal, "The Raw

microprocessor: a computational fabric for software circuits and general-purpose programs," Micro, IEEE, vol. 22, no. 2, pp. 25,35, Mar/Apr. 2002.

[4] G. Venkatesh, J. Sampson, N. Goulding, S. Gracia, V. Bryksin, J. L. Martinez, S. Swanson and M. B. Taylor, "Conservation cores: reducing the energy of mature computations", ASPLOS 10, pp. 205-218, 2010.

[5] M.B. Taylor, Is dark silicon useful?: harnessing the four horsemen of the coming dark silicon apocalypse. In proceedings of the 49th Annual Design Automation Conference (DAC 12) (pp. 1131-1136). NY, USA: ACM.

[6] A.K. Coskun, J.L. Ayala, D. Atienza, T.S. Rosing, Y. Leblebici,"Dynamic Thermal Management in 3D Multicore Architectures", In: Proc. of Design, Automation & Test in Europe Conference & Exhibition, Nice, France, pp. 1410-1415,April 2005.

[7] W. Kim, M. Gupta, G.-Y. Weil and D. Brooks, "System level anaysis of fast, per-core DVFS using on-chip switching regulators." in Proc. International Symposium on High Performance Computer Architecture (HPCA), pp. 123-134, 2008.

[8] H. Hoffmann et al., "Self-aware computing in the Angstrom processor," DAC Design Automation Conference 2012, San Francisco, CA, pp. 259-264, 2012. doi: 10.1145/2228360.2228409

[9] W. Hussain, R. Airoldi, H. Hoffmann, T. Ahonen and J. Nurmi, "HARP^2: An X-Scale Reconfigurable Accelerator-Rich Platform for Massively-Parallel Signal Processing Algorithms", in Journal of Signal Processing Systems, Springer, 2015.

[10] W. Hussain, H. Hoffmann, T. Ahonen and J. Nurmi, "Power Mitigation by Performance Equalization in a Heterogeneous Reconfigurable Multicore Architecture", in Journal of Signal Processing Systems, Springer, pp. 287-297, June 2017. doi="10.1007/s11265-016-1142-5"

[11] W. Hussain, H. Hoffmann, T. Ahonen and J. Nurmi, "Constraint-Driven Frequency Scaling in a Coarse Grain Reconfigurable Array", in Proc. International Symposium on System-on-Chip 2014, Tampere, Finland.

[12] D. Melpignano, L. Benini, E. Flamand, B. Jego, T. Lepley, G. Haugou, F. Clermidy and D. Dutoit, "Platform 2012, a Many-Core Computing Accelerator for Embedded SoCs: Performance Evaluation of Visual Analytics Applications", in Proc. 49th Annual Design Automation Conference (DAC '12). ACM, New York, NY, USA, 1137-1142, 2012.

[13] F. Conti, C. Pilkington, A. Marongiu and L. Benini, "He-P2012: architectural heterogeneity exploration on a scalable many-core platform", in Proc. of the 24th edition of the great lakes symposium on VLSI (GLS- VLSI '14), pp. 231-232, ACM, New York, NY, USA, 2014

[14] N. S. Voros, M. Hübner, J. Becker, M. Kühnle, F. Thomaitiv, A. Grasset, P. Brelet, P. Bonnot, F. Campi, E. Schüler, H. Sahlbach, S. Whitty, R. Ernst, E. Billich, C. Tischendorf, U. Heinkel, F. Ieromnimon, D. Kritharidis, A. Schneider, J. Knaeblein, and W. Putzke-Röming, "MORPHEUS: A Heterogeneous Dynamically Reconfigurable Platform for Designing Highly Complex Embedded Systems", ACM Trans. Embed. Comput. Syst. 12, 3, Article 70, 33 pages, April 2013.

[15] F. Thoma, M. Kuhnle, P. Bonnot, E. M. Panainte, K. Bertels, S. Goller, A. Schneider, S. Guyetant, E. Schuler, K. D. Muller-Glaser, and J. Becker, "MORPHEUS: Heterogeneous Reconfigurable Computing", International Conference on Field Programmable Logic and Applications, FPL 2007, pp. 409-414, 27-29 Aug. 2007.

[16] D. Rossi, F. Campi, S. Spolzino, S. Pucillo and R. Guerrieri, "A Heterogeneous Digital Signal Processor for Dynamically Reconfigurable Computing," in IEEE Journal of Solid-State Circuits, vol. 45, no. 8, pp. 1615-1626, Aug. 2010. doi: 10.1109/JSSC.2010.2048149

[17] F. Conti et al., "An IoT Endpoint System-on-Chip for Secure and Energy-Efficient Near-Sensor Analytics," in IEEE Transactions on Circuits and Systems I: Regular Papers, vol. 64, no. 9, pp. 2481-2494, Sept. 2017. doi: 10.1109/TCSI.2017.2698019

[18] H.J. Choi, Y.J .Park, H.-H. Lee, and C.H. Kim, "Adaptive dynamic frequency scaling for thermal-aware 3d multi-core processors", In Computational Science and Its Applications-ICCSA 2012, Lecture Notes in Computer Science. ISBN: 978-3-642-31127-7, (Vol. 7336 pp. 602612), 2012.

[19] X. Chen, Z. Xu, H. Kim, P.V. Gratz, J. Hu, M. Kishinevsky, U. Ogras and R. Ayoub, "Dynamic voltage and frequency scaling for shared resources in multicore processor designs", In Proceedings of the 50th Annual Design Automation Conference (DAC 13). Article 114 , 7 pages, NY, USA: ACM, 2013. doi:10.1145/2463209.2488874

[20] S.M.A.H. Jafri, M.A. Tajammul, A. Hemani, K. Paul, J. Plosila and H. Tenhunen,"Energy-aware-task-parallelism for efficient dynamic voltage, and frequency scaling", In CGRAs, 2013 International Conference on Embedded Computer Systems: Architectures, Modeling, and Simulation (SAMOS XIII), pp. 104112, 2013. doi:10.1109/SAMOS.2013.6621112

[21] M.-H. Haghbayan, A.-M. Rahmani A.Y., Weldezion, P. Liljeberg, J. Plosila, A. Jantsch and H. Tenhunen,"Dark silicon aware power management for manycore systems under dynamic workloads", In 2014 32nd IEEE International Conference on Computer Design (ICCD), pp. 509512) , 2014.

[22] Y. Sinangil et al., "A self-aware processor SoC using energy monitors integrated into power converters for self-adaptation," Symposium on VLSI Circuits Digest of Technical Papers, Honolulu, HI, pp. 1-2, 2014. doi: 10.1109/VLSIC.2014.6858424

[23] M. Möstl, J. Schlatow, R. Ernst, H. Hoffmann, A. Merchant and A. Shraer, "Self-aware systems for the Internet-of-Things," 2016 Interna-

tional Conference on Hardware/Software Codesign and System Synthesis (CODES+ISSS), Pittsburgh, PA, pp. 1-9, 2016.

[24] R. Airoldi, F. Garzia and J. Nurmi, "Improving Reconfigurable Hardware Energy Efficiency and Robustness via DVFS-Scaled Homogeneous MPSoC," 2011 IEEE International Symposium on Parallel and Distributed Processing Workshops and Phd Forum, Shanghai, pp. 286-289, 2011. doi: 10.1109/IPDPS.2011.160

[25] D. Rossi et al., "A Self-Aware Architecture for PVT Compensation and Power Nap in Near Threshold Processors," in IEEE Design & Test, vol. 34, no. 6, pp. 46-53, Dec. 2017. doi: 10.1109/MDAT.2017.2750907

[26] A. Bartolini, R. Diversi, D. Cesarini and F. Beneventi, "Self-Aware Thermal Management for High Performance Computing Processors," in IEEE Design & Test. doi: 10.1109/MDAT.2017.2774774

[27] J. Kylliäinen, T. Ahonen, and J. Nurmi, "General-purpose embedded processor cores - the COFFEE RISC example", In Processor Design: System-on-Chip Computing for ASICs and FPGAs, J. Nurmi, Ed. Kluwer Academic Publishers / Springer Publishers, ch. 5, pp. 83-100, ISBN-10: 1402055293, ISBN-13: 978-1-4020-5529-4, June 2007.

[28] F. Garzia, W. Hussain and J. Nurmi, "CREMA, A Coarse-Grain Reconfigurable Array with Mapping Adaptiveness", in Proc. 19th International Conference on Field Programmable Logic and Applications (FPL 2009). Prague, Czech Republic: IEEE, September 2009.

[29] C. Brunelli, F. Garzia, C. Giliberto, and J. Nurmi, "A Dedicated DMA Logic Addressing a Time Multiplexed Memory to Reduce the Effects of the System Buss Bottleneck", in Proc. 18th International Conference on Field Programmable Logic and Applications, (FPL 2008), Heidelberg, Germany, pp. 487-490, 8-10 September 2008.

[30] S. Nouri, W. Hussain and J. Nurmi, "Evaluation of a Heterogeneous Multicore Architecture by Design and Test of an OFDM Receiver", IEEE Transactions on Parallel and Distributed Systems, p. 1722 May 2017. 10.1109/TPDS.2017.2706691

[31] S. Nouri and J. Nurmi, "Power mitigation of a heterogeneous multicore architecture by frequency scaling in an OFDM receiver test case", Nordic Circuits and Systems Conference (NORCAS), Linkoping, Sweden, 23-25 Oct. 2017. DOI: 10.1109/NORCHIP.2017.8124987

Publication VI

Reprinted with permission from the Publisher of the *IEEE Nordic Circuits and Systems Conference (NORCAS): NORCHIP & International Symposium on System-on-Chip (SoC)*, Tallinn, Estonia, Oct. 2018, S. Nouri, R. Ghaznavi-Youvalari, and J. Nurmi, "Design and Implementation of Multi-Purpose DCT/DST-Specific Accelerator on Heterogeneous Multicore Architecture".

Tampereen teknillinen yliopisto
PL 527
33101 Tampere

Tampere University of Technology
P.O.B. 527
FI-33101 Tampere, Finland

ISBN 978-952-15-4253-4
ISSN 1459-2045