MIMO-OFDM LTE System based on a parallel IFFT/FFT on NoC-based FPGA

Kais Jallouli, Mohamed Mazouzi, Jean-Philippe Diguet, Alireza Monemi, Salem Hasnaoui

MIMO-OFDM LTE System based on a parallel IFFT/FFT on NoC-based FPGA

Kais Jallouli[1*], Mohamed Mazouzi[2], Jean-Philippe Diguet[3], Alireza Monemi[4] and Salem Hasnaoui[1]

[1*]Sys'Com (ENIT), University of Tunis El Manar (UTM), Tunis, Tunisia.
[2]CES-ENIS, University of Sfax, Sfax,Tunisia.
[4]Computer Sciences, Barcelona Supercomputing Center, 08034, Barcelona, Spain.
[3]IRL CNRS CROSSING, Adelaide, Australia.

*Corresponding author(s). E-mail(s): kais.jallouli@enit.utm.tn;
Contributing authors: mohammad.mazouzi@gmail.fr; jean-philippe.diguet@cnrs.fr;
alireza.monemi@bsc.es; salem.hasnaoui@enit.rnu.tn;

Abstract

The growing demand for wireless devices capable of performing complex communication processes has imposed an urgent need for high-speed communication systems and advanced network processors. This paper proposes a hardware workflow developed for the Long Term Evolution (LTE) communication system. It studies the Multiple-input, multiple-output orthogonal frequency-division multiplexing (MIMO-OFDM) LTE system. Specifically, this work focuses on the implementation of the OFDM block that dominates the execution time in high-speed communication systems. To achieve this goal, we have proposed an NoC-based low-latency OFDM LTE multicore system that leverages Inverse Fast Fourier Transform (IFFT) parallel computation on a variable number of processing cores. The proposed multicore system is implemented on an FPGA platform using the ProNoC tool, an automated rapid prototyping platform. Our obtained results show that LTE OFDM execution time is drastically reduced by increasing the number of processing cores. Nevertheless, the NoC's parameters, such as routing algorithm and topology, have a negligible influence on the overall execution time. The implementation results show up to 24% and 76% execution time reduction for a system having 2 and 16 processing cores compared to conventional LTE OFDM implemented in a single-core, respectively. We have found that a 4×4 Mesh NoC with XY deterministic routing connected to 16 processing tiles computing IFFT task is the most efficient configuration for computing LTE OFDM. This configuration is 4.12 times faster than a conventional system running on a single-core processor.

Keywords: Network-On-Chip (NoC), Multiple-Input Multiple-Output (MIMO), Orthogonal Frequency Division Multiplexing (OFDM), Inverse Fast Fourier Transform (IFFT)

1 Introduction

Digital communication and technological advances are based on several essential pillars, such as big data, real-time, and Internet of Things (IoT). Various studies announced an explosion in the volume of connected objects, especially with 5G spreading [1]. Wireless communication networks are one of the main important cores of the success of IoT. In this context, developing a communication network adapted to these objects becomes a vital necessity [2].

In IoT, we therefore need wireless communication networks that provide low latency, higher bandwidth, and energy consumption as low as possible. Orthogonal Frequency Division Multiplexing (OFDM) has proven itself as an effective multicarrier digital communication technique in the field of wireless communication, such as LTE and 5G [3].

The implementation of the OFDM system is based on Inverse Fast Fourier Transform (IFFT) and Fast Fourier Transform (FFT) for the modulation and the demodulation, respectively [4]. In this paper, we adopt the Long Term Evolution (LTE), that is one of the communication standards based on the OFDM system, especially for downlink transmission. The LTE OFDM system parameters such as the size of IFFT, the size of the cyclic prefix, and the amount of data to be transferred vary with the change of the channel bandwidth [5]. These different parameters affect the processing time and operations counts [6]. The crucial IFFT and FFT modules are the most computationally intensive blocks compared to other blocks in LTE system [7]. Due to the evolution of wireless communication systems and the demand for better implementation of IoT, the need for shorter execution time, lower latency, higher bandwidth, lower energy consumption, and having more complex hardware is constantly increasing. It is necessary to design an OFDM system that meets high-speed requirements while providing higher bandwidth. As a result, reducing the computation time of the IFFT algorithm [8] is a key issue for fast configurations of LTE, 5G, and beyond. Based on a distributed programming model, multiprocessor architectures are naturally more efficient than a single processor to process a set of independent or loosely-coupled tasks, since they allow parallel management of these tasks. With such an approach, we can take advantage of the IFFT parallelization. This work makes two contributions: first, it parallelizes IFFT using a Network-on-Chip (NoC)-based platform implemented on an FPGA device, and second, it conducts a guided design space exploration (DSE) of the NoC architecture to determine the best execution-time/resource-utilization trade-off.

The rest of this paper is organized as follows: Section 2 summarizes the related work on MIMO-OFDM LTE implementation. Section 3 describes the MIMO-OFDM LTE blocks and more precisely, the OFDM block. Section 4 presents the hardware development setup using the ProNoC prototype platform. Section 5 explains our proposed parallelization approach for the OFDM system. Section 6 presents the obtained experimental results. This section evaluates different architectural designs' performance. Finally, Section 7 concludes the paper and provides the future work.

2 Related works

Discrete Fourier Transform (DFT) is the most mathematical process used in Digital Signal Processing (DSP). It plays an essential role in several applications like communication systems based on OFDM [9]. Discrete Fourier Transform $X(k)$ of a function $x(n)$ is given by the equation (1).

$$X(K) = \sum_{n=0}^{N-1} x(n) W_N^{Kn} \tag{1}$$

where N is the DFT length and $x(n)$ and $X(k)$ are the input sequence of complex numbers and the output sequence of complex numbers, respectively. $W_N^{Kn} = e^{-j2\pi kn}$ are the transformation coefficients called Twiddle Factor.

The calculation of DFT requires a very large number of complex operations. The time complexity of DFT is $O(N^2)$. It becomes very expensive for large N. A faster method of computing DFT is the Fast Fourier Transform (FFT) algorithm developed by Cooley and Tukey [10]. The direct calculation of the DFT requires $N \log_2 N$ operations rather than N^2, where N is the size of FFT[9]. The time complexity of FFT is still very high and poses a huge and serious problem for real-time applications such as MIMO-OFDM LTE. As a result, reducing FFT computation time is a major concern [11]. However, IFFT/FFT can be parallelized as long as an increase in communications can be handled. Hence, an efficient implementation of IFFT/FFT blocks is required.

A solution consists of the parallel execution of IFFT/FFT modules on P processors of a multicore system. Moreover, NoC is an emerging technology that can satisfy the communication demands of such high-performance communication systems [12, 13]. On the other hand, the NoC should be configured according to the target application requirements. To select the most optimized architecture for a target application, the DSE is necessary. The most optimized NoC configuration is determined by considering one or more criteria such as execution time, power consumption, communication bandwidth, and hardware resource utilization.

A NoC-based FPGA prototype platform can facilitate the performance evaluation of a MIMO-OFDM LTE application. It also helps to identify the best NoC architecture. FPGAs are reconfigurable and cost-effective devices widely used for implementing high-performance digital communication systems [14, 15]. However, less effort is put into leveraging NoC in such architectures in today's systems. In this study, we aim to identify the most optimized NoC architecture that responds to the constraints of MIMO-OFDM LTE specifications.

Several IFFT/FFT dedicated architectures are found in the literature that are based on parallel and pipeline processing [16, 17]. The pipeline architecture is one of the most popular and most suitable solutions for signal processing applications and high-speed real-time communication systems such as MIMO-OFDM LTE, where high-throughput is a key constraint [18]. On the other hand, the parallel architecture is an interesting option when latency is the most important constraint [16]. These works, however, do not address a solution for distributing IFFT execution across multiple cores.

The main objective of this study is to implement an LTE OFDM system where the parallel execution of the IFFT algorithm is the challenge. This implementation choice considers execution time minimization while optimizing resource utilization. The main contribution of this paper is the DSE of the MIMO-OFDM LTE implementation based on a parallel IFFT algorithm on a prototype NoC platform. Considering the application speed-up under resource constraints, the second contribution is identifying a promising candidate architecture.

3 MIMO-OFDM LTE description

In this section, we provide a brief description of the MIMO-OFDM LTE blocks [19, 20] as shown in Figure 1. Note that, this work focuses on the implementation of the OFDM block, as it represents 50% of the processing time on a single-core system [21].

3.1 Overview of LTE OFDM system

Figure 2 shows a simplified block diagram of the LTE OFDM transceiver system that consists of two parts: the transmitter and the receiver. On the transmitter's side, the binary source bits are modulated to generate complex numbers using digital modulators. Afterward, a serial to parallel converter (S/P) allows the conversion of the combined data into parallel data. Moreover, these symbols are distributed on N orthogonal subcarriers that are often performed with an IFFT. Finally, the cyclic prefix (CP) is inserted along with a parallel to serial (P/S) conversion. On the receiver's side, the opposite operations are performed. Once the signal is at the receiver, the CP, is removed. Then, the operation of the FFT is carried out. Lastly, the S/P allows the conversion of data from serial to parallel form. Finally, the data are converted back to its original form.

Based on the LTE standard specification, the size of IFFT/FFT varies from 128 to 2048 for all indicated channel bandwidths. Table 1 outlines the LTE OFDM parameters [21].

4 Development and simulation setup

Design space exploration aims to find the best solution for a given cost function. Accordingly, performance evaluation techniques are essential to select the best architecture design for a target application. Various NoC parameters such as the mapping algorithm, network topology, switching techniques, and routing algorithm affect the overall system performance.

4.1 ProNoC prototyping platform

Manually writing a complete register transfer level (RTL) code for a complex system, particularly a heterogeneous multi-processor system-on-chip (MPSoC), is a time-consuming and error-prone process. To overcome this problem, we use the ProNoC tool [22] that facilitates the development and validation of a NoC-based system on FPGA. This prototyping tool generates the complete RTL code of the specified system. The generated NoC is optimized for FPGA implementation and supports several features such as virtual channels, different routing algorithms, and different topologies. ProNoC offers a high-performance interconnect for our application domain. Moreover, ProNoC is equipped with a graphical user interface (GUI), facilitating the development and validation of MPSoC on FPGA devices. The ProNoC design flow is (shown in Figure 3) composed of four main steps, which are interface generator, intellectual property (IP) generator, processing tile (PT) generator, and finally, the NoC-based MPSoC generator. Each step is equivalent to one layer in MPSoC design. Due

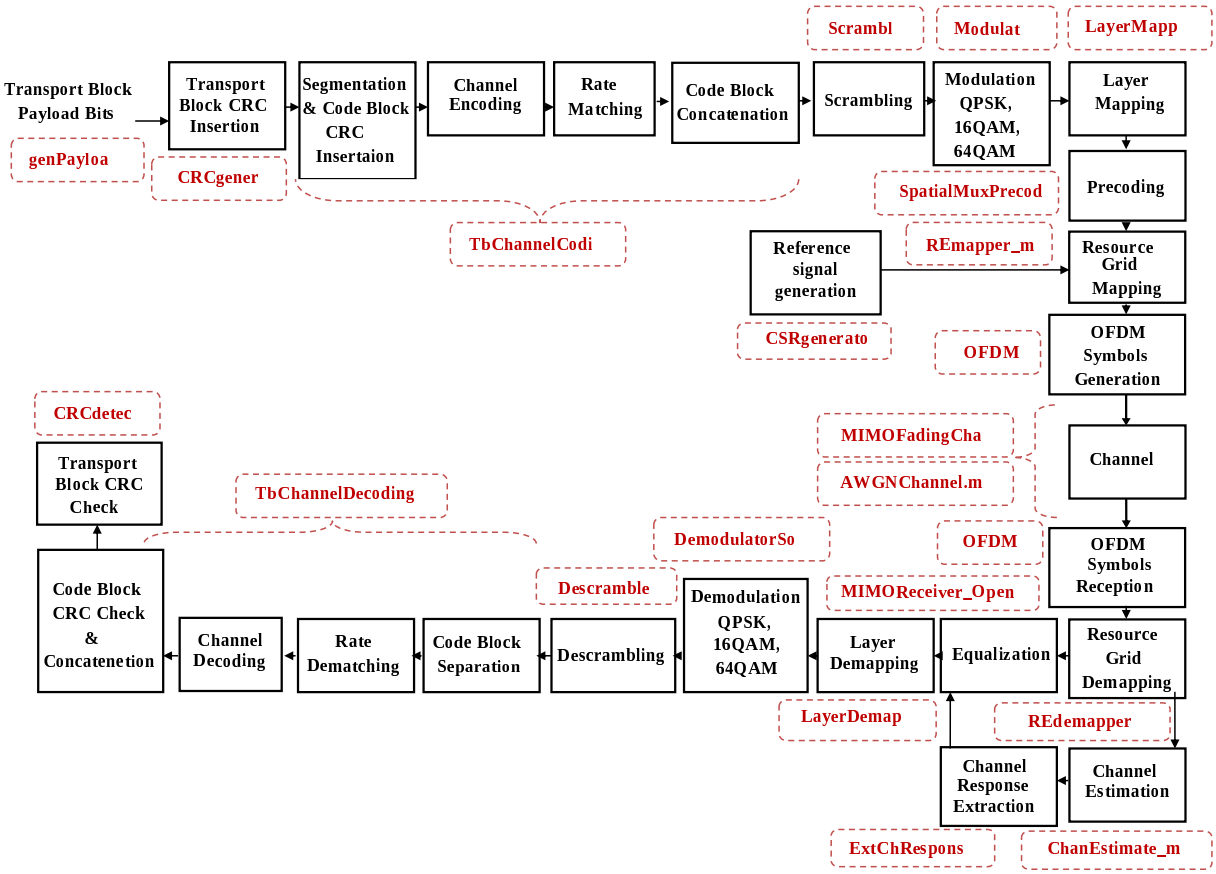


Fig. 1 MIMO-OFDM LTE blocks.

Table 1 OFDM LTE parameters

Channel bandwidth (MHz)	1.4	3	5	10	15	20
Subframe duration	1ms					
Subcarrier spacing	15kHz					
IFFT size(N)	128	256	512	1024	1536	2048
Number of data(N_d)	72	180	300	600	900	1200
Number of nulls (N_n)	56	76	212	424	636	848
OFDM symbol per slot	7/6 (Normal/extended)					

to the definition of an interface, the components can be easily connected to each other. An interface consists of a combination of several ports providing specific functionality. The IP generator makes a library for each IP, which can be a processor, bus, timer, memory, or interrupt controller.

A PT consists of a combination of IPs that are connected together using a Wishbone Bus [23]. The generated tile can be used as a single-core system subsequently in the final step. Moreover, a complete MPSoC in RTL can be generated using ProNoC by connecting PTs via a parameterizable NoC.

4.2 Design methodology and hardware workflow

4.2.1 Single-core system generation

We have generated a single-core system RTL code to evaluate a conventional LTE OFDM system running on a single processor using the ProNoC tool. Figure 4 illustrates the functional block diagram of this single-core system, which consists of a mor1kx processor [24], single-port memory, a wishbone bus, a timer, and a virtual JTAG UART.

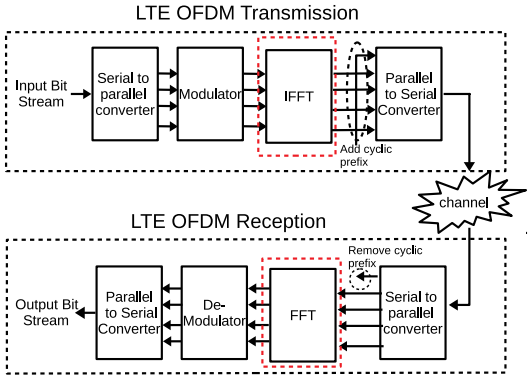


Fig. 2 LTE OFDM transceiver.

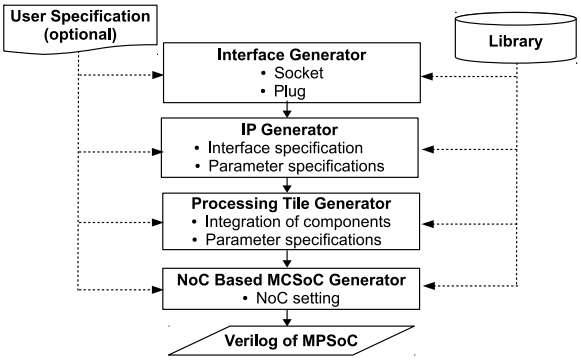


Fig. 3 ProNoC design flow [22]

Then we developed the conventional LTE OFDM application code purely in C programming language. The code is compiled using the mor1kx GNU toolchain [25]. Finally, the whole system is simulated using the Verilator simulator [26]. The RTL code is synthesized using the Quartus Prime 18.1 software tool to generate the SOF programming file. The design is then emulated using an Altera Stratix V GX FPGA. The LTE OFDM system output is directed to JTAG UART by our application. These results are verified by comparing them with the output obtained from the Matlab "OFDMTx.m" function [27].

4.2.2 NoC generation

The design methodology adopted for the generation of a network-on-chip architecture implemented in an FPGA device for the multicore LTE OFDM system is composed of eight steps, which are (1) the PT generation process, (2) NoC configurations, (3) NoC generation, (4) parallelization of the LTE OFDM system, (5) compilation of the parallel LTE OFDM system, (6) NoC design verification, (7) NoC design synthesis, and (8) NoC implementation.

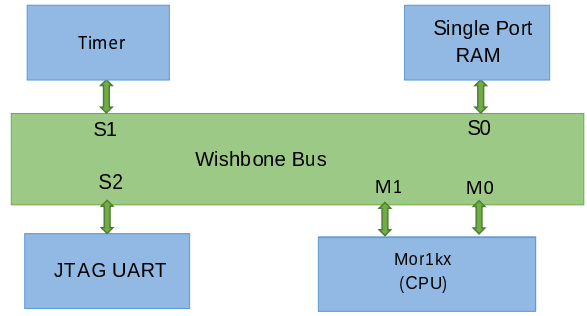


Fig. 4 Schematic of single-core architecture used to implement an LTE OFDM system in a single processor.

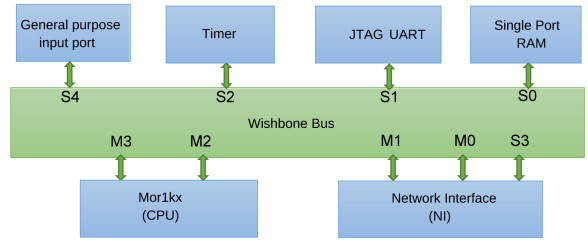


Fig. 5 Schematic of the PT used in the MPSoC design.

To generate a PT that can be connected to a NoC, we added an additional network interface IP to our previously single-core architecture. Figure 5 illustrates the PT adopted in the design of our MPSoC. In the next step, we specify the NoC configurations such as topology, router per row, router per column, VC number per port, buffer width per VC, payload width, routing algorithm, etc. Next, we generate the complete RTL code of the MPSoC with the help of the NoC-based MPSoC generator. It comprises the entire multicore hardware architecture, including the basic components of a NoC. We notice that the memory size of each processing tile should be configured differently to match the application memory requirements. The LTE OFDM system is parallelized by viewing our application as a combination of serial and parallel blocks, with the IFFT block computed using p processing tiles (detailed in section 5). The C code corresponding to each processing tile is developed individually based on the number of processing tiles computing the IFFT task. Then the application code of each PT is compiled using the mor1kx GNU toolchain. During the design verification, we simulated the proposed multicore LTE OFDM system using the ModelSim simulator. However, the NoC simulation using ModelSim is too slow, especially for a large NoC. For this reason, we simulate the whole system using the Verilator simulator.

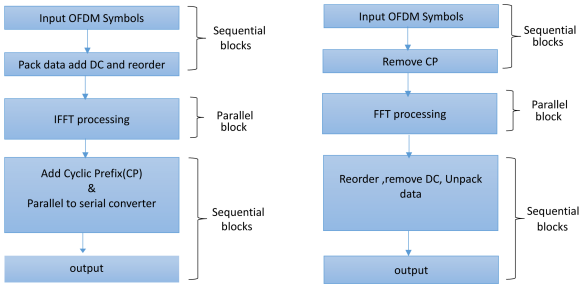


Fig. 6 OFDM LTE system block diagram.

During the design synthesis, the whole MPSoC is synthesized using the Intel Quartus prime 18.1. Finally, the complete MPSoC is prototyped in an Altera Stratix V GX FPGA using the USB blaster II. We start the Nios2-terminals of all processing tiles then run the program. To perform this, we need to check a start flag in our codes which is set by the user once all the Nios2-terminals are ready. We use a general purpose input (GPI) as a push-button that makes sure that all of the processing tiles start running the code simultaneously. For the performance evaluation, we are particularly interested in execution time and resource occupation of the target FPGA. We have changed the NoC configuration parameters such as the topology, the number of virtual channels per port, and routing algorithm to figure out the best architecture. In addition, we vary the number of processing tiles computing the IFFT task.

5 Proposed parallelization approach

In this section, we present our proposed approach for an LTE OFDM system as a combination of serial and parallel tasks as illustrated in Figure 6. Focusing on the NoC paradigm, the LTE OFDM system can be decomposed into three blocks. Two sequential blocks that perform the sequential tasks each run on a single processing tile. These sequential tasks are packing the input data, inserting the null component DC, reordering the input data, and adding the cyclic prefix. A parallel block performs the execution of the IFFT algorithm on p processing tiles. The overall system performance of the LTE OFDM system can be improved by exploring a variety of architectural designs and increasing the number of processing tiles computing the IFFT task. This process is continued until the high computational requirement of the real-time MIMO-OFDM LTE application is satisfied.

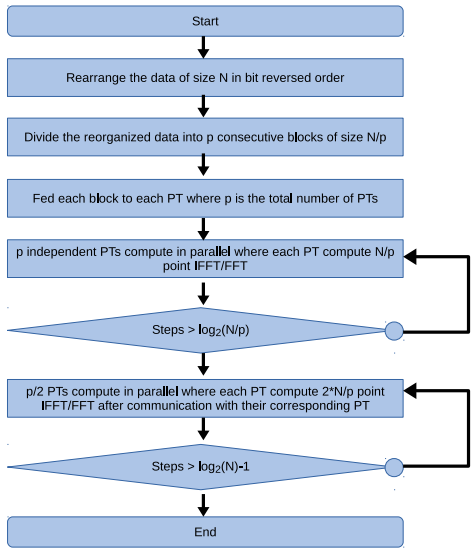


Fig. 7 Flowchart of our parallel FFT/IFFT.

5.1 Parallel IFFT/FFT algorithm

Our parallel IFFT/FFT algorithm can be decomposed into three stages, as shown in Figure 7 [28]. During the first stage, the input data is reorganized in bit-reversed order. The N reorganized complex numbers are then divided into p consecutive blocks, with each block containing $fracNp$ complex numbers. These complex numbers are sent to their corresponding processing tile as illustrated in Algorithm 1. Where N is the size of the IFFT and p is the number of processing tiles.

During the second stage, which corresponds to the first $\log_2(\frac{N}{P})$ steps of the IFFT, p processing tiles are executed in parallel, each performing an IFFT of size $\frac{N}{P}$. In this step, no communication between tiles is required, as Algorithm 2 illustrates.

In the final stage corresponding to the rest of $\log_2 p$ steps of the IFFT algorithm, $\frac{p}{2}$ processing tiles, each performs an IFFT of size $2\frac{N}{p}$ in parallel after receiving its data from its respective processing tile as shown in Algorithm 3. In this stage, the communication between tiles is required because the index distance between IFFT Butterfly diagram is larger than $\frac{N}{P}$. The proposed IFFT algorithm reduces the operation counts and minimizes the latency of the LTE OFDM system. The time complexity of this algorithm is reduced to $O(N\log_2\frac{N}{p})$ compared to the time complexity of serial FFT/IFFT $O(N\log_2 N)$. Figures 8, 9, 10, 11 illustrate the mapping exploration

examples of 2048-point IFFT/FFT onto 4 and 16 processing tiles.

Algorithm 1 Data Preprocessing for each processing element (PE)

Input: $a=(a_0,a_1,a_2,a_3,...,a_{N-1})$, N=size_of(IFFT), p=number of processors
Output: (PE[0].$c[\frac{N}{p}]$,..., PE[$p-1$].$c[\frac{N}{p}]$)
b=bit_reverse(a)
for i=0 to p-1 **do**
  for k=0 to $\frac{N}{p}$ **do**
    PE[i].$c[k]=b[i\times\frac{N}{p}+k]$
  end for
end for

Algorithm 2 IFFT of size $\frac{N}{p}$ on each PE

Input: $c=(c_0,c_1,c_2,c_3,...,c_{\frac{N}{p}-1})$, N=size_of(IFFT), p=number of processors
Output: $x=(x_0,x_1,x_2,x_3,...,x_{\frac{N}{p}-1})$
$x=c$
for i=0 to $\log_2(\frac{N}{P})$-1 **do**
  $l=2^i$, $q=\frac{N}{2l}$, $w=e^{\frac{i2\pi}{N}}$, $z=w^q$
  for $k=0$ to $\frac{N}{p}$ **do**
    $Q=i\times\frac{N}{p}+k$
    if $Q \bmod l = Q \bmod 2l$ **then**
      $m=Q \bmod l$
      $x[k]=x[k]+x[k+l]\times z^m$
      $x[k]=x[k]-x[k+l]\times z^m$
    end if
  end for
end for

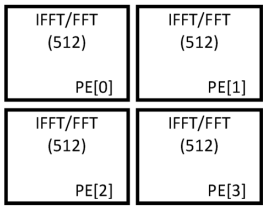
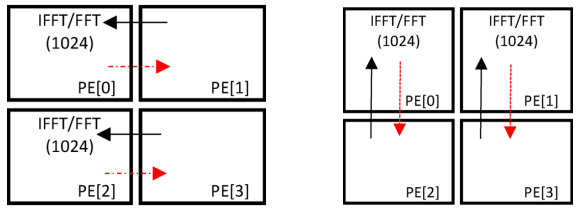


Fig. 8 Mapping exploration of steps 0 to 8 of 2048-point IFFT/FFT to 4 processing tiles where no communication between PEs is required.

Algorithm 3 IFFT of size $2\frac{N}{p}$ on $\frac{p}{2}$ PE where communication between PE is required

Input: $x=(x_0,x_1,x_2,x_3,...,x_{\frac{N}{p}-1})$, N=size_of(IFFT), p=number of processors
Output: $y=(y_0,y_1,y_2,y_3,...,y_{\frac{N}{p}-1})$
$y=x$
$j=\log_2(p)+1$
for $e=0$ to $\log_2(p)-1$ **do**
  $t=2^e$, $q=\frac{N}{2l}$, $w=e^{\frac{i2\pi}{N}}$, $z=w^q$,
  $j=j$-1, $l=2^{(e+\log_2\frac{N}{p})}$, $v=2^j$
  for $i=0$ to $p-1$ **do**
    if $i \bmod t = i \bmod 2t$ **then**
      Receive data from PE[$i+\frac{p}{v}$] and store them into $x[\frac{N}{v}]-x[\frac{N}{v}+\frac{N}{p}-1]$
      for $k=0$ to $\frac{N}{p}-1$ **do**
        $m=i\times\frac{N}{p}-1+k \bmod l$
        $y[k]=y[k]+y[k+l]\times z^m$
        $y[k]=y[k]-y[k+l]\times z^m$
      end for
      Send transformed data in $y[\frac{N}{v}]-y[\frac{N}{v}+\frac{N}{p}-1]$ to the PE[$i+\frac{p}{v}$]
    else
      Send data of this PE to PE[$i-\frac{p}{v}$]
      After transformation receive data from PE[$i-\frac{p}{v}$], store them into y.
    end if
  end for
end for



(a) step 9 (b) step 10

Fig. 9 Mapping exploration of step 9 and step 10 of 2048-point IFFT/FFT to 4 processing tiles where communication between PEs is required.

5.2 Multicore OFDM LTE system

While analyzing the LTE OFDM transmitter, we partitioned it into three blocks. Then, we map each block into its corresponding tiles. The target platform is composed of $p+2$ processing elements or tiles (PE) denoted as PE[0] to PE[$p+1$]. PE[p] is the source tile containing an LTE downlink subframe made

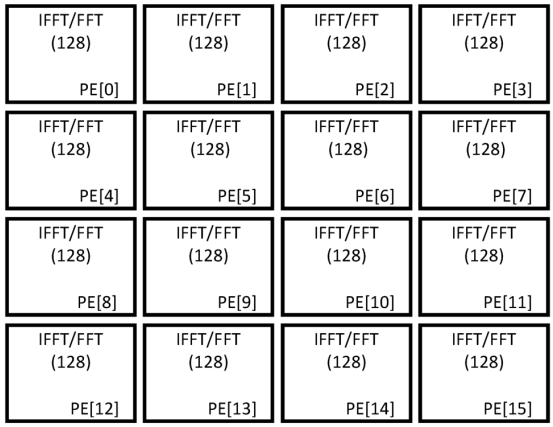


Fig. 10 Mapping exploration of the 0th to 6th of 2048-point IFFT/FFT onto 16 processing tiles where no communication between PEs is required.

up of a matrix of complex numbers with the size of (Nd, numSymb). Nd is the number of data to be transferred and numSymb is the number of OFDM symbols in an LTE subframe when using a normal cyclic prefix. PE[p] performs the first sequential block that includes packing the input data, inserting the null component DC, and data reordering. In addition, a matrix of complex numbers with the size of (N, numSymb) is generated by this tile. PE[p] reads the generated matrix column by column i.e. OFDM symbol by OFDM symbol then reorganizes each column in bit-reverse order and divides the reorganized N complex numbers into p blocks of $\frac{N}{p}$ complex numbers. Finally it sends each data block to its corresponding PE (PE[0] to PE[$p-1$]). Similarly, PE[p] treats the rest of the OFDM symbols until the end of subframe. This time, all PEs wait until they receive all of their corresponding data (14 subsequences of size $\frac{N}{p}$). The p PEs perform the IFFT task computation according to our parallel IFFT algorithm presented in Section 5.1. PE[p] waits until it receives the results of the computation of the p tiles. Last but not least, it reconstructs an OFDM symbol per OFDM symbol and adds prefix cyclic to each OFDM symbol until it finishes all the OFDM symbols.

6 Results

In this work, we explore various architectural designs implemented in an FPGA device for the LTE OFDM system using the ProNoC prototype platform. We vary the number of processing tiles computing the IFFT task and some NoC parameters such as topology and routing algorithms. We have considered all channel bandwidths supported by LTE except the 15 MHz because its IFFT size is 1536, and our proposed OFDM is based on parallel IFFT in which its size should be a power of two. We report a comparison of performance evaluation of the LTE OFDM system having a different number of processing tiles, including a single processing tile result as a reference. Table 2 enumerates the specific parameters of the NoC used to implement our multicore LTE OFDM system. For the different configuration designs, we use Mesh and Torus topologies. We use different routing algorithms for each topology, i.e. deterministic, partially, and fully adaptive.

Table 2 NoC parameters used to implement multicore LTE OFDM

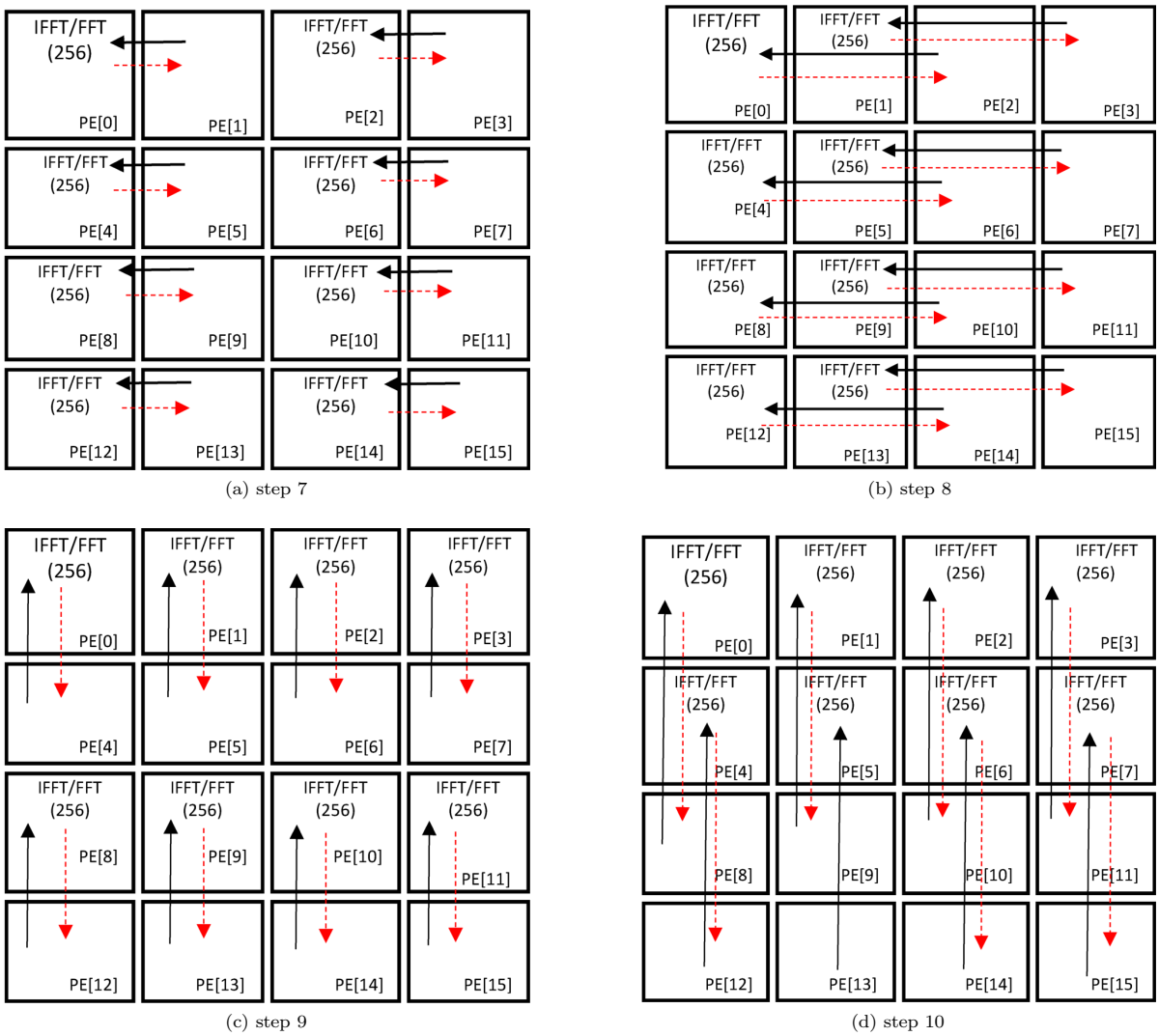
Parameters	Value
Number of tiles computing IFFT task	2,4,8,16
Topology	Mesh, Torus
Payload width	32
Number of virtual channel per port	4
Buffer flits per VC	4
Packet switching technique	Wormhole
Routing algorithm	XY, West-First, Fully adpative

6.1 Mesh network-on-chip

We begin our evaluation performance by using the parameters listed in Table 2 in which we use Mesh topology and XY deterministic routing algorithm. This work analyzes the execution time and the FPGA resource utilization for different LTE OFDM systems. We simulated different designs in which the IFFT task is computed by 1, 2, 4, 8 and 16 processing tiles.

6.1.1 Execution time versus number of processing tiles

We evaluate the execution time and the speed-up metric of our implemented multicore LTE OFDM system at various channel bandwidths (i.e., 1.4 MHz, 3 MHz, 5 MHz, 10 MHz, 20 MHz) using the ProNoC tool's design methodology, as shown in Figure 12. The horizontal axis represents the number of processing tiles computing the IFFT task in the LTE OFDM system. The left vertical axis depicts the execution time of the LTE OFDM system in terms of clock cycles, whereas the right vertical axis depicts the speed-up metric of the OFDM system. For all examined channel bandwidth, the execution time of



(a) step 7

(b) step 8

(c) step 9

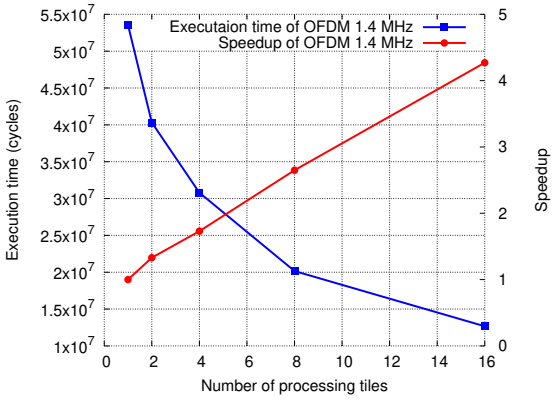
(d) step 10

Fig. 11 Mapping exploration of 7th to 10th steps of 2048-point IFFT/FFT to 16 processing tiles where communication between PEs is required.

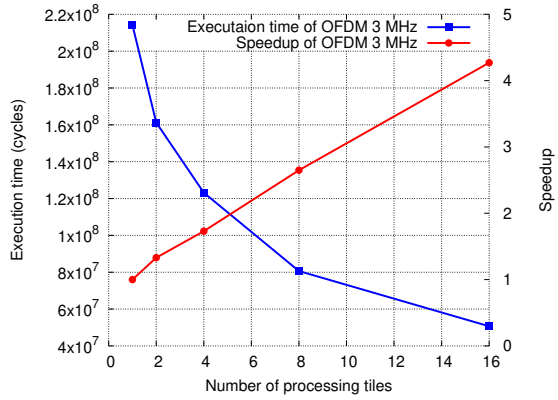
our multicore LTE OFDM system decreases significantly with the increase in the number of processing tiles computing IFFT task. For all configurations, the multicore LTE OFDM system performs better than the conventional LTE OFDM system, ranging from 24% to 76% improvement for 2 to 16 processing tiles computing IFFT task, respectively. The LTE OFDM system shows a shorter execution time with a larger number of processing tiles. Notice that the decrease in the execution time is not linear according to Amdahl Law [29]. The speed-up metrics of our multicore OFDM range from 1.33 to 4.27 times faster for 2 to 16 processing tiles computing IFFT task, respectively, compared to a single-core system.

6.1.2 Error rate verification

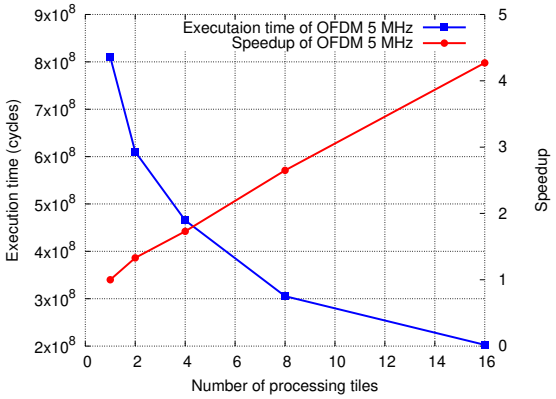
To verify our proposed parallelization approach and to test the bit error rate (BER), we checked the output results of different multicore systems having different numbers of cores with a Matlab OFDM simulation. To do so, we conducted tests with an OFDM LTE subframe without noise as the input sequence with a variable number of processing cores (i.e., 1, 2, 4, 8, and 16) and different channel bandwidths. The tests show zero BER in all cases. The number of tiles is limited to 16 as the FPGA board logic resources are mostly consumed in that configuration. However, we have tested a multicore



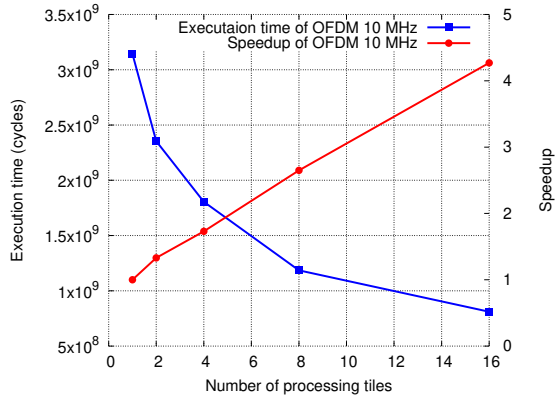
(a) 1.4 MHz channel bandwidth



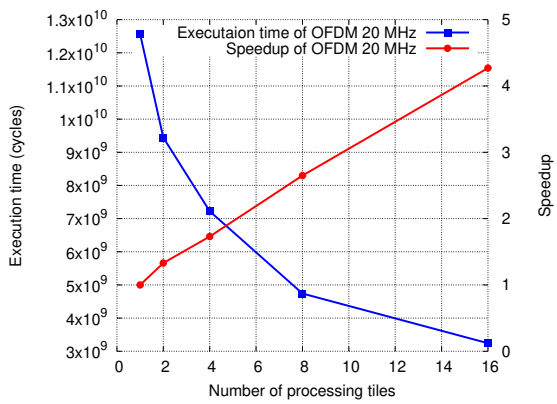
(b) 3 MHz channel bandwidth



(c) 5 MHz channel bandwidth



(d) 10 MHz channel bandwidth



(e) 10 MHz channel bandwidth

Fig. 12 Execution time of the proposed multicore LTE OFDM system in Mesh NoCs under XY deterministic routing algorithm versus number of processing tiles at different channel bandwidths.

system with up to 64 tiles using the RTL simulator, and the error rate is still zero. We conclude that our multicore implementation is not loose or faulty.

6.1.3 FPGA resource utilization versus number of processing tiles

Figure 13 summarizes the FPGA resource utilization required to implement different configurations of the

proposed multicore LTE OFDM system in Mesh topology under XY routing algorithm for a different number of processing tiles computing IFFT task at different channel bandwidths. As expected, the increase in the number of processing tiles leads to acceleration with a trade-off in higher logic utilization. However, generating a multicore LTE OFDM system beyond 16 processing tiles computing IFFT is impossible due to the limited embedded memory in the target FPGA board. An LTE OFDM system with 16 processing tiles computing IFFT task consumes 34%, 47%, and 74% of available logic resources, ALMs, and total block RAMs of the target FPGA board, respectively. As shown in Figure 13, we can also notice that there is an additional cost in memory usage for four processing tiles at 10 MHz compared to 1.4 MHz and 5MHz using the same number of processing tiles. In fact, for the OFDM system at 10 MHz with four tiles, each tile needs to have a memory size of 4 Mbits compared to 2 Mbits for the OFDM system at 5MHz.

Table 3 indicates the percentage of resources utilized for a 20 MHz LTE channel bandwidth. The number of ALMs is low compared to the total RAM block usage. The number of RAM blocks becomes high with an LTE OFDM system implemented on a NoC architecture comprising 16 processing tiles computing IFFT task. The ALM usage rises from 11% in an OFDM system with a speedup of 1.33 to 34% in an OFDM system with a speedup of 4.27.The RAM block usage increases from 46% to 73% for the same speedup.

Table 3 LTE OFDM 20 MHz: Resource Utilization and speedup on the Altera Stratix V GX

LTE channel bandwidth(MHz)	20			
Speedup	1.33	1.73	2.65	4.27
Logic Utilization ALMs	11%	11%	18%	34%
Total blocks memory	34%	52%	50%	47%
Total RAM blocks	46%	69%	70%	73%

6.2 Effect of routing algorithm in execution time and resource utilization

We explore the influence of the routing algorithm in our proposed architecture by analyzing three different routing algorithms, i.e., the XY deterministic routing algorithm, fully adaptive routing, and West-first partially adaptive routing. The simulation results show that the routing algorithm does not influence LTE OFDM system execution time. However, it affects the resource utilization of our multicore LTE OFDM system. Mesh NoCs with fully adaptive routing algorithms requires more additional resources (logic utilization and registers) than those with deterministic and partially adaptive routing algorithms.

6.2.1 Effect of network topology in execution time and resource utilization

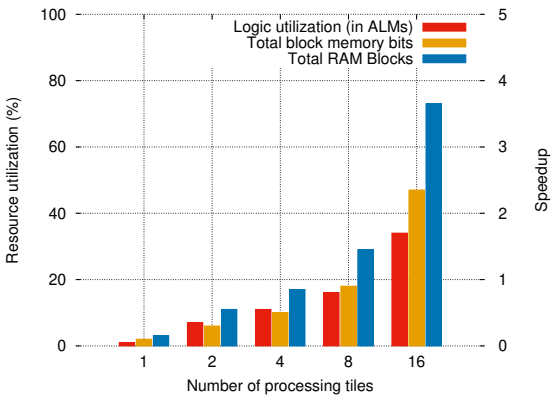
To observe the influence of the NoC topology on LTE OFDM performance, we configured our system with a Torus topology under different routing algorithms. We did not get any performance improvement in the execution time of our multicore LTE OFDM system compared to an OFDM system implemented under the Mesh topology. The Torus architecture leads to a slight increase in terms of resource utilization and power consumption due to the long connection wires compared to the Mesh topology. Table 4 presents the resource utilization comparison of the LTE OFDM system at a 20 MHz LTE channel bandwidth for 2 to 16 processing tiles computing IFFT task. As illustrated by the results, Mesh NoC under XY routing is more interesting. ALMs saving can range from 7.3% for 16 processing tiles to 11.2% for 2 processing tiles computing IFFT task. Total registers saving can range from 5.6% for 16 processing tiles to 8.4% for 2 processing tiles computing IFFT task.

Table 4 Comparison of resources utilization (Mesh and Torus) of a multicore OFDM LTE system the Altera Stratix V GX

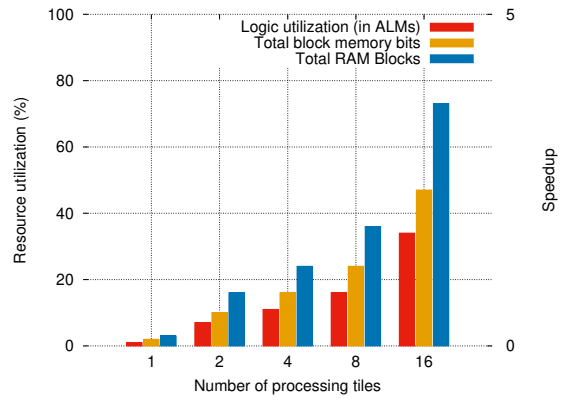
LTE channel bandwidth(MHz)	**20**			
Number of processing Tiles computing IFFT	2	4	8	16
TExec$_{Mesh}$/Texec$_{Torus}$	1			
ALMs Mesh saving compared to Torus	11.2 %	10.1 %	8.3 %	7.3 %
Registers Mesh saving compared to Torus	8.4 %	7.9 %	6.5 %	5.6 %

6.3 Synthesis and choice of architecture

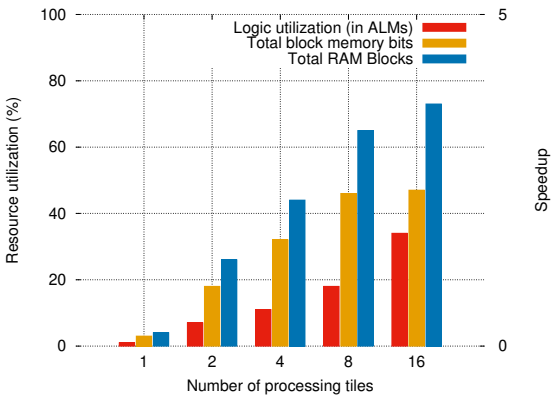
Our experiments show that changing the deterministic routing algorithm to adaptive or partially adaptive cannot improve the execution time performance, but it increases resource utilization. In addition, we show that the change of Mesh to Torus topology cannot improve the overall system performance of our multicore OFDM LTE system. Hence, we conclude



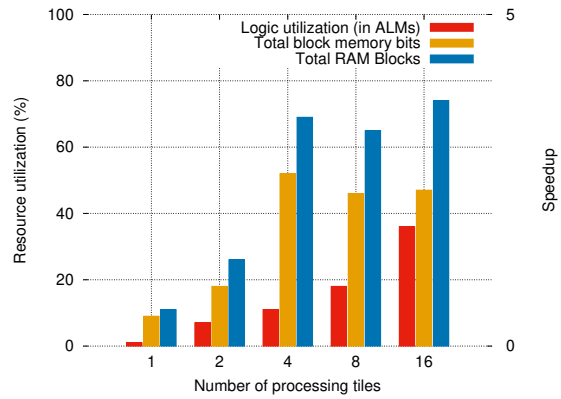
(a) 1.4 MHz channel bandwidth



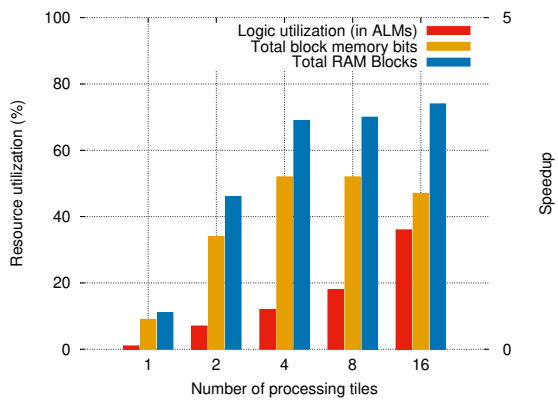
(b) 3 MHz channel bandwidth



(c) 5 MHz channel bandwidth



(d) 10 MHz channel bandwidth



(e) 20 MHz channel bandwidth

Fig. 13 Resource utilization comparison (Altera Stratix V GX FPGA) of LTE OFDM system implemented in Mesh NoCs under XY deterministic routing algorithm versus number of processing tiles at different channel bandwidths.

that the initial architecture that is a deterministic Mesh NoC having a maximum of 16 processing tiles is the best architecture in terms of execution time and resource utilization. Mesh architecture with a deterministic routing algorithm appears to be a good candidate for developing a communication system for IoT technology.

7 Conclusion

In this paper, we use a prototype platform to design and evaluate different architectures for the LTE OFDM system. It mainly relies on the parallelization of IFFT where it can be generalized to other future standards based on OFDM such 5G. We explore a variety of multicore architectures by varying the number of processing tiles computing IFFT task, changing some NoC parameters such as topology and routing algorithm. For real-time applications, execution time is a major concern. Our Altera Stratix V GX FPGA implementation results demonstrate that an LTE OFDM system at a 20 MHz channel bandwidth with 16 processing tiles computing IFFT task in a Mesh NoC under XY routing algorithm leads to a 76% reduction in execution time compared to a single core implementation.

References

[1] Hsieh HC, Chen JL, Benslimane A (2018) 5G virtualized multi-access edge computing platform for IoT applications. Journal of Network and Computer Applications 115:94–102

[2] Younus MU, ul Islam S, Ali I, Khan S, Khan MK (2019) A survey on software defined networking enabled smart buildings: Architecture, challenges and use cases. Journal of Network and Computer Applications 137:62–77

[3] Liu S, Huang S, Li M, Dong X, Qiu D, Yang R (2019) Novel low-complexity low-latency orthogonal frequency division multiplexing transmitter. Wuhan University Journal of Natural Sciences 24(1):71–78

[4] LaSorte N, Barnes WJ, Refai HH (2008) The history of orthogonal frequency division multiplexing. In: IEEE GLOBECOM 2008-2008 IEEE Global Telecommunications Conference, IEEE, pp 1–5

[5] Ferreira ML, Barahimi A, Ferreira JC (2016) Dynamically reconfigurable LTE-compliant OFDM modulator for downlink transmission. In: 2016 Conference on Design of Circuits and Integrated Systems (DCIS), IEEE, pp 1–6

[6] Nouri S, Hussain W, Nurmi J (2017) Evaluation of a heterogeneous multicore architecture by design and test of an OFDM receiver. IEEE Transactions on Parallel and Distributed Systems 28(11):3171–3187

[7] Huang J, Ruan F, Su M, Yang X, Yao S, Zhang J (2016) Analysis of orthogonal frequency division multiplexing (OFDM) technology in wireless communication process. In: 2016 10th IEEE International Conference on Anti-counterfeiting, Security, and Identification (ASID), IEEE, pp 122–125

[8] Dali M, Guessoum A, Gibson RM, Amira A, Ramzan N (2017) Efficient FPGA implementation of high-throughput mixed radix multipath delay commutator FFT processor for MIMO-OFDM. Advances in Electrical and Computer Engineering 17(1):27–38

[9] Kumar GG, Sahoo SK, Meher PK (2019) 50 years of FFT algorithms and applications. Circuits, Systems, and Signal Processing 38(12):5665–5698

[10] Cooley JW, Tukey JW (1965) An algorithm for the machine calculation of complex Fourier series. Mathematics of computation 19(90):297–301

[11] Lee JS, Sunwoo MH (2003) Design of new DSP instructions and their hardware architecture for high-speed FFT. Journal of VLSI signal processing systems for signal, image and video technology 33(3):247–254

[12] Gaur MS, Laxmi V, Zwolinski M, Kumar M, Gupta N, et al (2015) Network-on-chip: Current issues and challenges. In: 2015 19th International Symposium on VLSI Design and Test, IEEE, pp 1–3

[13] Bhaskar AV, Venkatesh T (2021) Performance analysis of network-on-chip in many-core processors. Journal of Parallel and Distributed Computing 147:196–208

[14] Hao M, Karsthof L, Rust J, Demel J, Bockelmann C, Dekorsy A, Al Houry A, Mackenthun F, et al (2018) FPGA-based baseband solution for high performance industrial wireless communication. In: 2018 IEEE 23rd International Conference on Digital Signal Processing (DSP), IEEE, pp 1–5

[15] Bruno JS, Almenar V, Valls J (2019) FPGA implementation of a 10 GS/s variable-length FFT for OFDM-based optical communication systems. Microprocessors and Microsystems 64:195–204

[16] Shirbhate R, Panse T, Ralekar C (2015) Design of parallel FFT architecture using Cooley Tukey algorithm. In: 2015 International Conference on Communications and Signal Processing (ICCSP), IEEE, pp 0574–0578

[17] Li L, Wyrwicz AM (2018) Parallel 2D FFT implementation on FPGA suitable for real-time mr image processing. Review of Scientific Instruments 89(9):093,706

[18] Mahdavi M, Edfors O, Öwall V, Liu L (2019) A low latency FFT/IFFT architecture for massive MIMO systems utilizing OFDM guard bands. IEEE Transactions on Circuits and Systems I: Regular Papers 66(7):2763–2774

[19] Mousavi H, Amiri IS, Mostafavi M, Choon C (2019) LTE physical layer: Performance analysis and evaluation. Applied computing and informatics 15(1):34–44

[20] Zarrinkoub H (2014) Understanding LTE with MATLAB: from mathematical modeling to simulation and prototyping. John Wiley & Sons

[21] Makni M, Baklouti M, Niar S, Biglari-Abhari M, Abid M (2015) Heterogeneous multi-core architecture for a 4G communication in high-speed railway. In: 2015 10th International Design & Test Symposium (IDT), IEEE, pp 26–31

[22] Monemi A, Tang JW, Palesi M, Marsono MN (2017) ProNoC: A low latency network-on-chip based many-core system-on-chip prototyping platform. Microprocessors and Microsystems 54:60–74

[23] Sharma M, Kumar D (2012) Wishbone bus architecture-a survey and comparison. arXiv preprint arXiv:12051860

[24] mor1kx (Online; accessed Nov. 2021) an OpenRISC processor IP core. http://github.com/openrisc/mor1kx

[25] Lampret D, Chen CM, Mlinar M, Rydberg J, Ziv-Av M, Ziomkowski C, McGary G, Gardner B, Mathur R, Bolado M (2003) Openrisc 1000 architecture manual. Description of assembler mnemonics and other for OR1200

[26] Snyder W, Wasson P, Galbi D (Online; accessed Nov. 2021) Verilator-convert Verilog code to C++/SystemC. http://www.veripool.org/wiki/verilator

[27] MATLAB (Online; accessed Nov. 2021) HDL OFDM receiver. https://www.mathworks.com/help/wireless-hdl/ug/hdlofdmreceiver.html

[28] Jallouli K, Mazouzi M, Ahmed AB, Monemi A, Hasnaoui S (2018) Multicore MIMO-OFDM LTE optimizing. In: 2018 International Conference on Internet of Things, Embedded Systems and Communications (IINTEC), IEEE, pp 166–170

[29] Amdahl GM (1967) Validity of the single processor approach to achieving large scale computing capabilities. In: Proceedings of the April 18-20, 1967, spring joint computer conference, pp 483–485