TAMPERE UNIVERSITY OF TECHNOLOGY

MUBASHIR ALI

IMPLEMENTING CARRIER RECOVERY FOR LTE 20 MHZ ON TRANSPORT TRIGGERED ARCHITECTURE

MASTER OF SCIENCE THESIS

EXAMINERS: PROF. JARI NURMI
PROF. MIKKO VALKAMA, MSC OMER ANJUM

Examiner and topic approved by Faculty Council of Computing and Electrical Engineering, December 2011.

ABSTRACT

Degree Programme in Electrical Engineering
Mubashir Ali: Implementing Carrier Recovery for LTE 20 MHz on Transport Triggered Architecture
Master of Science Thesis: 49 pages
March 2013
Major subject: Radio Frequency Electronics
Examiners: Professor Jari Nurmi, Professor Mikko Valkama, MSc Omer Anjum
Keywords: Long Term Evolution, Orthogonal Frequency Division Multiplexing, Carrier Frequnecy Offset, Transport Triggered Architecture, Software Defined Radio.

Synchronization is a critical function in digital communications. Its failure may cause catastrophic effects on the transmission system performance. It is very important that the receiver is synchronised with the transmitter because it is not possible to correct frequencies/phases without any control mechanisms. Synchronization is different in Third Generation Partnership Project (3GPP) Long Term Evolution (LTE) for uplink and downlink because of the choice of multiple access scheme. Multiple access scheme for LTE downlink is Orthogonal Frequency Division Multiple Access (OFDMA) and Single Carrier-Frequency Division Multiple Access (SC-FDMA) for the uplink. OFDMA is susceptible to Carrier Frequency Offset (CFO). In case of a typical LTE system with a carrier frequency of 2.1 GHz, a frequency drift of 10ppm (10×10^{-6}) of the local oscillator can cause an offset of 21 kHz. LTE system employs a fixed subcarrier spacing of 15 kHz. This offset caused by the local oscillator corresponds to 1.40 subcarrier spacings. The receiver extracts the information from the received signal to synchronise and compensate for any carrier frequency/phase offset. Increasing demand for data driven applications has put stress on communication systems to provide high data rates and increased bandwidth. This demand has ever been increasing and requires new standards to evolve and efficient hardware. It has been difficult to develop hardware at the pace new communication standards are developing. It also increases the cost of deployment of a technology for a brief period of time without covering the huge capital invested in the network. In order to meet the pace of evolving standards and covering the huge network costs, industry needs Software-Defined Radio (SDR). SDR is a radio communication technology that is based on software defined wireless communication protocols instead of hardwired implementations. System components that are usually implemented in hardware are implemented by means of software on a computer or embedded system.

LTE carrier recovery algorithm for LTE downlink with 20 MHz system bandwidth has been implemented in this thesis. The architecture chosen for implementation is Transport Triggered Architecture (TTA) with the goal to achieve real time constraints

along with a certain flexibility and power consumption needed for an SDR platform. The target programming language is C with TTA specific extensions instead of hand optimized assembly with the aim to reduce the whole design time and still achieve the required optimizations and throughput. This design cycle time is also one of the important aspects for product development in the industry.

PREFACE

This M.Sc. thesis was completed in Department of Computer Systems of Tampere University of Technology (TUT) as a part of the International Masters Degree Program in RF Electronics.

I would like to express my sincere gratitude to my thesis supervisors, MSc Omer Anjum and Prof. Jari Nurmi, for giving me a chance to develop as a C programmer and for their patience during this period. I am extremely grateful to them for introducning me the new concepts and research topics in computer architecture, software definded radio and for their patience in answering my questions. Special thanks to Department of Computer Systems of TUT for giving me the opportunity and space for learning and research during my thesis work. I would also like to thank my colleagues and friends for the inspired and motivated work atmosphere they have provided.

I am also extremely grateful to my parents for their continuous support especially my father and my elder brother Mr. Haider Ali for providing me moral and financial support and courage during the whole period. Finally, I would like to remember my relatives who passed away while I was away from home.

Tampere, March 22, 2013

Mubashir Ali

CONTENTS

LIST OF ACRONYMS

1G, 2G, 3G, 4G	First Generation, 2nd Generation, 3rd Generation, 4th Generation
3GPP	3rd Generation Partnership Project
ADF	Architecture Definition File
AMPS	Advanced Mobile Phone Service
CDMA	Code Division Multiple Access
CFO	Carrier Frequency Offset
CORDIC	Coordinate Rotational Digital Computer
CP	Cyclic Prefix
DA	Data-Aided
DD	Data Directed
DL	Downlink
DRE	Dead Result Elimination
DSE	Design Space Exploration
EDGE	Enhanced Data rates for GSM Evolution
FDD	Frequency Division Duplex
FDMA	Frequency Division Multiple Access
FFO	Fractional Frequency Offset
FFSK	Fast Frequency Shift Keying
FFT	Fast Fourier Transform
FPGA	Field Programmable Gate Array
FU	Functional Unit
GPRS	General Packet Radio Service
GSM	Global Systems for Mobile Communications
GUI	Graphical User Interface
HDB	Hardware Database
HLL	High Level Language
IC	Interconnection

ICI	Inter-Carrier Interference
IDF	Implementation Definition File
IFO	Integer Frequency Offset
IMT	International Mobile Telecommunications
IS	Interim Standard
ISI	Inter-Symbol Interference
ISS	Instruction Set Simulator
ITU	International Telecommunication Union
LO	Local Oscillator
LSU	Load-Store Unit
LTE	Long Term Evolution
MIMO	Multiple Input Multiple Output
ML	Maximum Likelihood
MPSoC	Multi-Processor System on Chip
MU-MIMO	Multi-User MIMO
NDA	Non-Data Aided
NMT	Nordic Mobile Telephony
OFDMA	Orthogonal Frequency Division Multiple Access
OTA	Operation-Triggered Architecture
PAPR	Peak to Average Power Ratio
PCI	Physical-Layer Cell Identity
PDC	Personal Digital Cellular
PHS	Personal Handy Phone System
PPM	Parts Per Million
P-SCH	Primary Synchronization Signal
QAM	Quadrature Amplitude Modulation
QoS	Quality of Service
QPSK	Quadrature Phase Shift Keying
RB	Resource Block
RE	Resource Element
RF	Radio Frequency/Register File
RFO	Residual Frequency Offset
RISC	Reduced Instruction Set Computer
RS	Reference Symbol

RTL	Register Transfer Level
SC-FDMA	Single-Carrier Frequency-Division Multiple Access
SDR	Software Defined Radio
SFU	Special Functional Unit
SIMD	Single Instruction, Multiple Data
SODA	Signal Processing on Demand Architecture
S-SCH	Secondary Synchronization Signal
SU-MIMO	Single-User MIMO
TACS	Total Access Communication System
TCE	TTA-based Co-design Environment
TDMA	Time Division Multiple Access
TPEF	TTA Program Exchange Format
TTA	Transport Triggered Architecture
WLAN	Wireless Local Area Network

LIST OF SYMBOLS AND DEFINITIONS

$a_{k,l}^{(p)}$	Value of resource element (k,l) [for antenna port p]
h	Channel impulse response
I	Inter-carrier interference caused by neighboring subcarriers
(k,l)	Resource element with frequency-domain index k and time-domain index l
K_p	Joint set of OFDM symbol number and subcarrier indices for reference symbols
K_{SCH}	Subcarrier indices that contain the synchronization signals
l	OFDM symbol index in one subframe
N	FFT size
$N_{\mathrm{ID}}^{\mathrm{cell}}$	Physical layer cell identity
$N_{\mathrm{RB}}^{\mathrm{DL}}$	Downlink bandwidth configuration, expressed in multiples of $N_{\mathrm{sc}}^{\mathrm{RB}}$
$N_{\mathrm{RB}}^{\mathrm{max,DL}}$	Largest downlink bandwidth configuration, expressed in multiples of $N_{\mathrm{sc}}^{\mathrm{RB}}$
$N_{\mathrm{sc}}^{\mathrm{RB}}$	Resource block size in the frequency domain
N_g	CP length
N_s	Number of OFDM symbols in one slot
$\mathrm{N}_{\mathrm{RB}}^{\mathrm{DL}}$	Maximum number of downlink Resource Blocks
$N_{\mathrm{symb}}^{\mathrm{DL}}$	Number of OFDM symbols in a downlink slot
n	Time index within one OFDM symbol
R	Received reference symbols
R_{PSCH}	Received primary synchronization signal
R_{SSCH}	Received secondary synchronization signal
r	Received time-domain signal
T_{u}	Subcarrier modulation-symbol time

v	Additive Gaussian noise
X	Transmitted reference symbols
X_{PSCH}	Transmitted Primary synchronization signal
X_{SSCH}	Transmitted secondary synchronization signal
x	Transmitted OFDM signal
Δf	Subcarrier spacing
$\Delta\omega$	Constant carrier frequnecy error
ε_{CFO}	Frequency mismatch/offset between transmitter and receiver
ε_{FFO}	Fractional frequency mismatch/offset
ε_{IFO}	Integer frequency mismatch/offset
ε_{RFO}	Residual frequency mismatch/offset
φ	Constant carrier phase error
γ	Degradation on the desired subcarrier caused by the CFO

1. INTRODUCTION

Wireless technology has evolved very rapidly during the last couple of decades. New wireless communication methods, services and efficient hardware have been adopted throughout the world. The mobile radio communication industry has grown by orders of magnitude and this growth has been further accelerated by advancement in digital and radio frequency (RF) circuit fabrication, new large scale circuit integration, and other miniaturization technologies which make portable radio equipment smaller, cheaper, and more reliable. Digital switching techniques have facilitated the large scale deployment of affordable, easy to use radio communication networks. These trends will continue at even greater pace during the coming years. [1]

1.1. Evolution of Wireless Technology

The evolution of wireless communication networks is spread across four generations. These are first generation (1G) mobile networks, second generation (2G) mobile networks, third generation (3G) mobile networks, and the latest commercially available fourth generation (4G) mobile networks respectively.

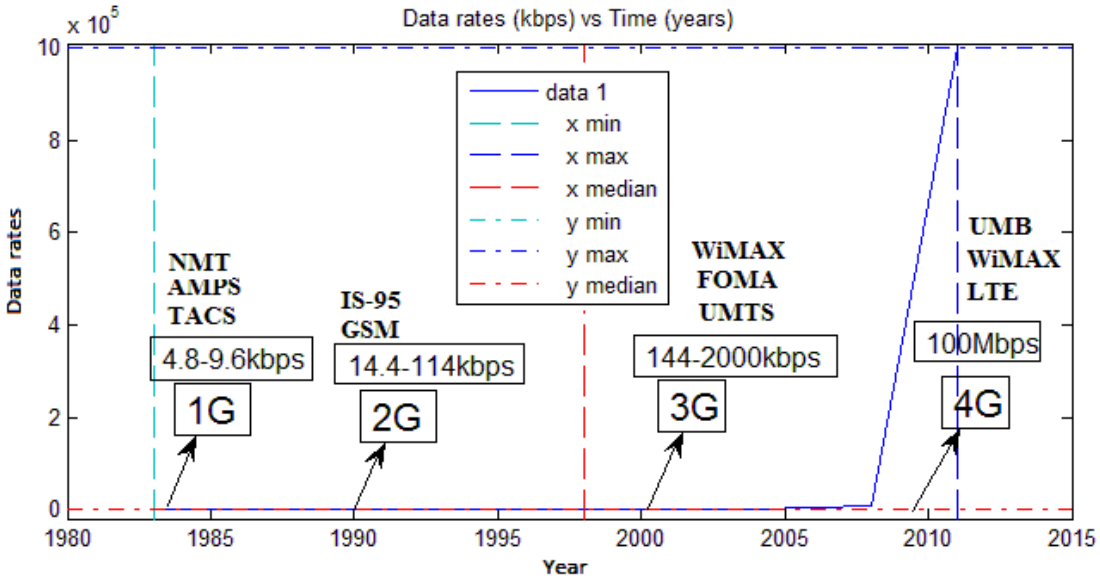

Figure 1.1. Wireless networks generations and technologies

Japan took the lead in the development of wireless technology, deploying the first cellular networks in Tokyo. Within a couple of years Nordic Mobile Telephony (NMT) started cellular operations in Europe. At the same time Advanced Mobile Phone Service (AMPS) started in the USA, while Total Access Communication System (TACS) started in the UK. These systems were called 'First Generation Mobile Systems', providing

speech services and were based on analogue transmission techniques. 1G network supported very low data rates. AMPS modem call data rate under good conditions is as high as 14.4 kbps and under poor conditions is as low as 4.8 kbps. NMT FFSK modem data rate was 12kbps.

In the early 1990s, digital transmission technology came into force, introducing the 'Second Generation Mobile System'. Key 2G systems in this generation included Global Systems for Mobile Communications (GSM), TDMA IS-136, CDMA IS-95, Personal Digital Cellular (PDC) and Personal Handy Phone System (PHS). IS 54 and IS 136 (where IS stands for Interim Standard) were the second generation mobile systems that constituted the D-AMPS. This was the digital advancement of the then existing AMPS in America. [2]

The third generation cellular networks were developed with the aim of offering high speed data and multimedia connectivity to subscribers. The International Telecommunication Union (ITU) under the initiative IMT-2000 defined 3G systems as being capable of supporting high speed data ranges of 144 kbps to greater than 2 Mbps. A few technologies are able to fulfill the International Mobile Telecommunications (IMT) standards, such as CDMA and UMTS. [2]

The three previous generations of mobile networks had their merits and demerits but none of them had the ability to completely replace the other. Even IMT-2000, a worldwide standard, was not able to break the bottleneck of high data rate and capacity, and this led to the formation of a new generation, referred to as 3GPP Long Term Evolution (LTE) or 4G. Orthogonal Frequency Division Multiple Access (OFDMA) and Frequency Division Multiple Access (FDMA) are used as the modulation access schemes for downlink and uplink respectively in LTE. It was thought that instead of developing new radio interfaces and new technology it would be better to integrate existing and newly developed wireless systems like the GPRS, EDGE, Bluetooth, WLAN and Hiper-LAN. [2]

Wireless networks evolution in terms of data rates, mobility and different technologies is depicted in the following figure. The figure is based on research carried out on existing and emerging wireless networks at IMEC Belgium.

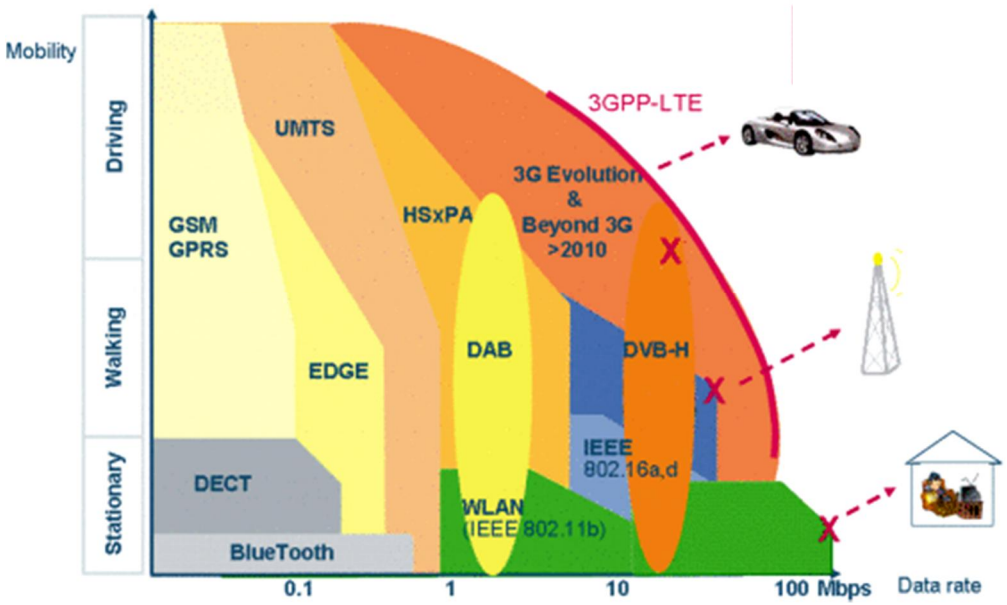

Figure 1.2. Wireless networks supported data rates and mobility [(C) Copyright IMEC]

1.2. OFDM & OFDMA

OFDM is a special case of Frequency Division Multiplexing (FDM). OFDM is both a modulation and multiplexing technique. The concept of OFDM has existed since around 1966. [5]

A single carrier system modulates information onto one carrier using frequency, phase, or amplitude adjustment of the carrier. For digital signals, the information is in the form of bits, or collection of bits called symbols, that are modulated onto the carrier. As higher bandwidths (data rates) are used, the duration of one bit or symbol of information becomes smaller. [32]

OFDM system breaks the available bandwidth into many narrower sub-carriers and transmits the data in parallel streams. OFDM symbols are much longer than symbols on single carrier systems of equivalent data rate. [32]

Each sub-carrier is modulated using varying levels of QAM modulation, e.g., QPSK, QAM, 64QAM or possibly higher orders depending on signal quality. Therefore, each OFDM symbol is a linear combination of the instantaneous signals on each of the sub-carriers in the channel.

There are two important aspects of OFDM. First, each OFDM symbol is preceded by a cyclic prefix (CP). This CP is effective in eliminating Inter-Symbol Interference (ISI). Second, the sub-carriers are very tightly spaced to make efficient use of band-

width, yet there is virtually no interference among adjacent sub-carriers. These two features are in fact closely related.

Figure 1.3. OFDM symbol with CP

OFDM symbol consists of two major components: CP and a Fast Fourier Transform (FFT) period. The duration of CP is determined by the highest anticipated degree of delay spread for the targeted application. When transmitted symbols arrive at the receiver by two paths of differing length, they are spread in time. However it is possible to have distortion from preceding symbol within the CP. With a CP of sufficient duration, preceding symbols do not spill over into the FFT period.

Once the signal is received and digitized, the receiver removes CP. The result is a rectangular pulse within each subcarrier, which has constant amplitude over the FFT period. [6]

The OFDM pulse shaping for simple rectangular pulse is shown in below:

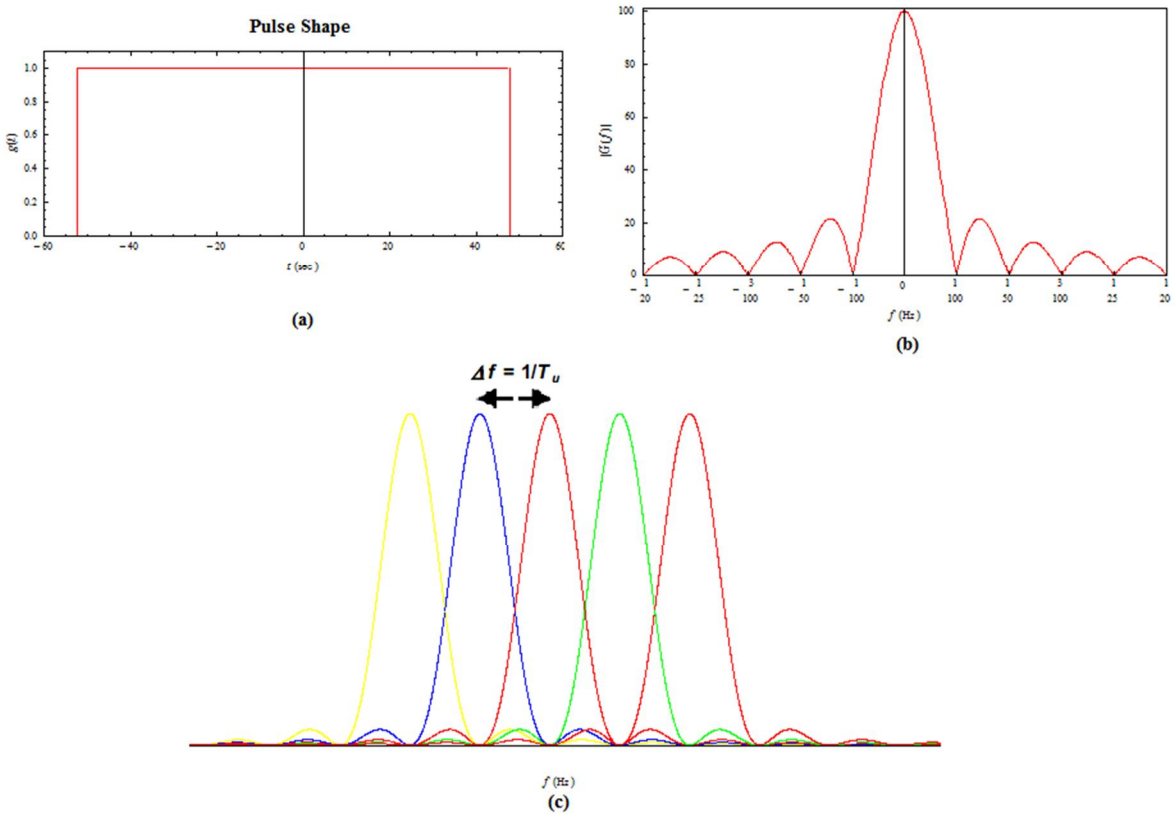

Figure 1.4. Pulse shaping and sub-carrier spacing

Figure 1.4 (c) shows tight frequency-domain packing of the subcarriers with a sub-carrier spacing $\Delta f = 1/T_u$, where T_u is the per-subcarrier modulation-symbol time (see Figure 1.4). In other words the subcarrier spacing is thus equal to the per-subcarrier modulation rate $1/T_u$, which gives orthogonal sub-carriers.

OFDMA is the multiple access scheme for LTE downlink. There is a difference between OFDM and OFDMA in the scheduling of the users. In an OFDM system, different users scheduled in time domain are allocated the full bandwidth. In OFDMA users are scheduled in both frequency and time domain in such a way that the available bandwidth is shared by the users. A very important advantage of OFDMA system is its scheduling property. This property allows scheduling decisions in the frequency domain. For instance, some sub-carriers can be modulated simultaneously with 64-QAM while others may be modulated with some other digital modulation technique like QPSK in order to take frequency dependencies into account in the radio link quality.

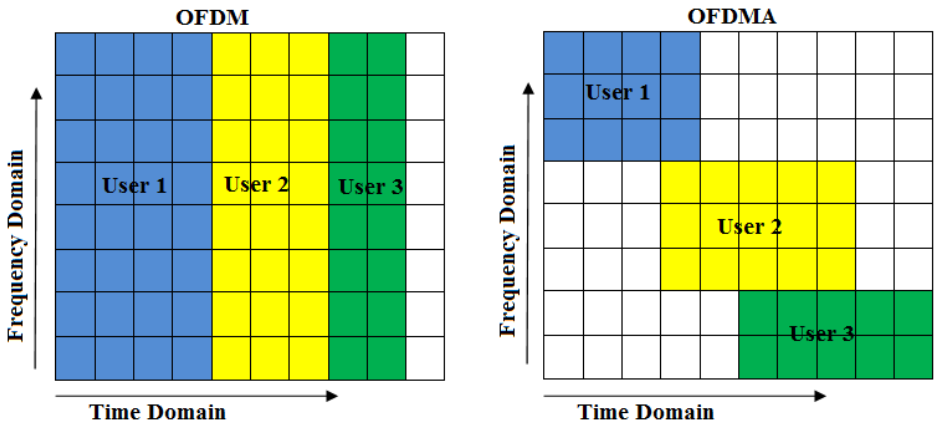

Figure 1.5. OFDM and OFDMA (Adapted from [36])

1.3. LTE (4G) Multiple Access Schemes

Actual work towards 3GPP Long Term Evolution (LTE) started in 2004 with the definition of the targets. After initial consolidation of proposals, the candidate scheme for the downlink was OFDMA, while the candidate scheme for the uplink was SC-FDMA. The choice of multiple-access schemes was made in December 2005, with OFDMA being selected for the downlink, and SC-FDMA for the uplink. Both of these schemes open up the frequency domain as a new dimension of flexibility in the system.

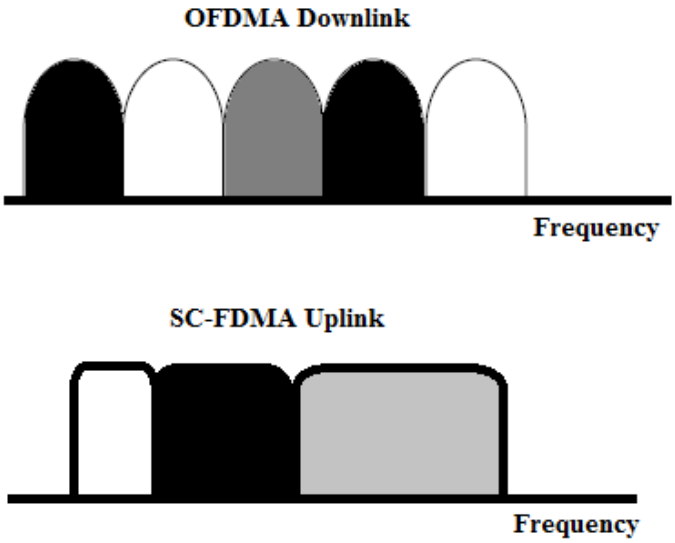

Figure 1.6. LTE multiple access technologies

Problem: Increasing demand for data driven applications has put stress on communication systems to provide high data rates and increased bandwidth. This demand has ever been increasing and requires new standards to evolve and efficient hardware. It has been difficult to develop hardware at the pace new communication standards are developing. It also increases the cost of deployment of a technology for a brief period of time without covering the huge capital invested in the network.

Proposed Solution: In order to meet the pace of evolving standards and covering the huge network costs industry needs SDR (Software-Defined Radio). [11] SDR is a radio communication technology that is based on software defined wireless communication protocols instead of hardwired implementations. System components that are usually implemented in hardware are implemented by means of software on a computer or embedded system. In other words, frequency band, air interface protocol and functionality can be upgraded with software download or configuring hardware instead of complete hardware replacement.

1.4. Related Work

SDR architectures have been proposed by CoreSonic [8], Sandbridge [9] and SODA (Signal Processing on Demand Architecture) in [10]. Multi-threaded processor architecture for SDR by Sandbridge is shown in the Figure 1.7. The design includes a unique combination of modern techniques such as a SIMD Vector/DSP unit, a parallel reduction unit, and a RISC-based integer unit. Each processor core provides support for concurrent execution for up to eight threads of execution. The platform allows the freedom to choose a particular set of operating modes (such as GSM, GPRS, WCDMA, etc.) and the desired local area connectivity (Bluetooth, Wireless LAN, etc). [9]

Some of the key focuses in this architecture are support for high-level programming language like C and compiler opimization for DSP. The need for compiler design in parallel with the DSP architecture design is particularly emphasized in their design cycle for the whole system. The proposed compiler analyzes the C code and extracts the DSP operations itself. [11]

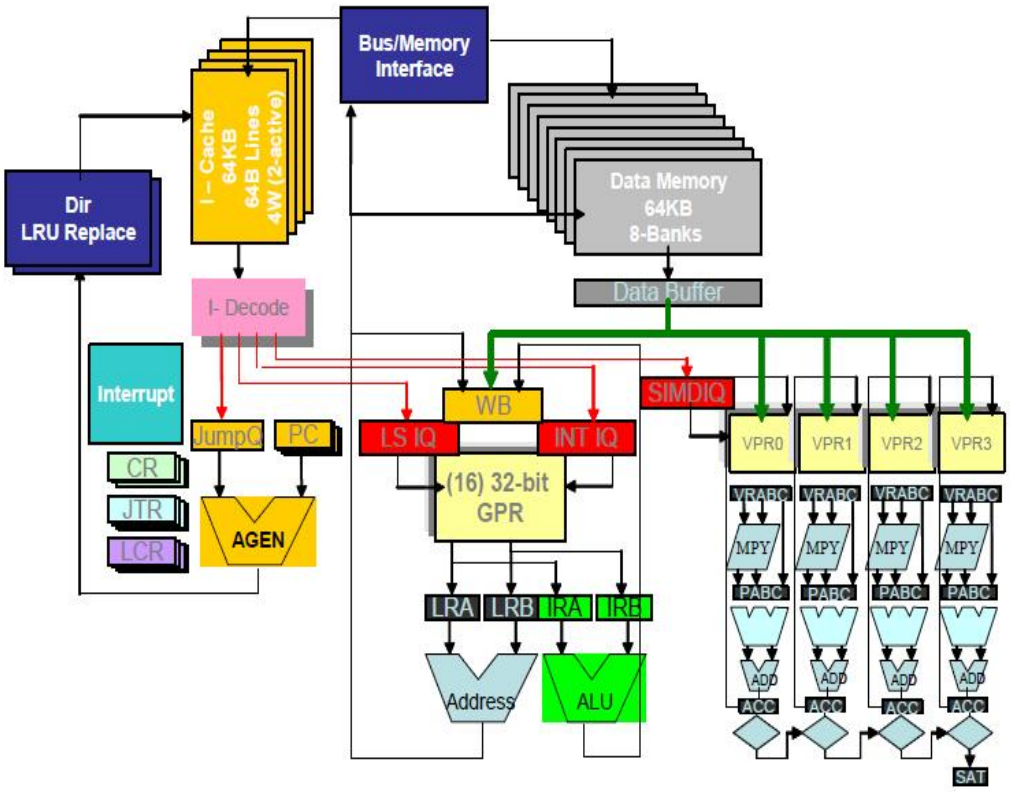

Figure 1.7. Multi-threaded processor SDR architecture [9]

A tradeoff exists between operating frequency and power consumption. The available state of the art SDR architectures have a drawback of either higher operating operating frequency or higher power consumtion. Compiler design optimization is another contributing factor as well. Considering these facts TTA (transport triggered architecture) is considered to be a possible solution.

LTE carrier recovery algorithm for LTE with 20 MHz system bandwidth has been implemented in this thesis. The architecture chosen for implementation is TTA with the goal to achieve real time constraints along with a certain flexibility and power consumption needed for an SDR platform. The target programming language is C with TTA specific extensions instead of hand optimized assembly with the aim to reduce the whole design time and still achieving the required optimizations and throughput. This design cycle time is also one of the important aspects for product development in the industry.

1.5. Organization of the Thesis

The thesis is organized as follows:

Chapter 1: discusses the evolution of the wireless technology, the technologies available and the challenges faced by future wireless technology.

Chapter 2: is about the LTE physical layer. LTE frame structure and the physical layer downlink signals used for carrier recovery.

Chapter 3: discusses the carrier recovery offset according to the mathematical model presented in [16].

Chapter 4: is about TTA framework, TTA based Co-Design Environment (TCE) toolset for designing and optimizing a TTA processor.

Chapter 5: describes the TTA design process and optimizations for a Carrier Recovery TTA and the Special Functional Units (SFUs) designed for the Carrier Recovery TTA. Results are discussed and comparison has been made between Carrier Recovery TTA and other architectures.

Chapter 6: summarizes and concludes the thesis.

2. LTE DOWNLINK PHYSICAL LAYER

The physical layer offers data transport services to higher layers in a system. LTE physical layer deals with signals in the OFDM format on the downlink (DL). The physical layer is expected to perform functions like modulation and demodulation of physical channels, frequency and time synchronization, multiple input multiple output (MIMO) antenna processing, transmit diversity (TX diversity) and RF processing.

As mentioned in the previous chapter multiple access schemes for the LTE physical layer are based on OFDMA with a cyclic CP in the downlink, and on SC-FDMA with a cyclic prefix in the uplink. Furthermore LTE frame structure and access schemes will be discussed in more detail in this chapter.

2.1. LTE Physical Layer Frame Structure

The Layer 1 is defined in a bandwidth agnostic way based on resource blocks, allowing the LTE Layer 1 to adapt to various spectrum allocations. Downlink and uplink transmissions are organized into radio frames with $T_{\mathrm{f}} = 307200 \times T_{\mathrm{s}} = 10$ ms duration. Two radio frame structures are supported:

- Type 1, applicable to FDD
- Type 2, applicable to TDD

10 subframes are available for downlink transmission and 10 subframes are available for uplink transmissions in each 10 ms interval for the FDD mode. Uplink and downlink transmissions are separated in the frequency domain. In half-duplex FDD operation, the user terminal cannot transmit and receive at the same time while there are no such restrictions in full-duplex FDD.

Type 1 frame structure with FDD mode has been focused in this work. The smallest time-frequency unit for downlink transmission is called a resource element (RE), shown in Figure 2.1 in the resource grid identified by the index pair (*k*,*l*) in a slot where $k = 0,..., N_{\mathrm{RB}}^{\mathrm{DL}} N_{\mathrm{sc}}^{\mathrm{RB}} - 1$ and $l = 0,..., N_{\mathrm{symb}}^{\mathrm{DL}} - 1$ are the indices in the frequency and time domains, respectively. $N_{\mathrm{RB}}^{\mathrm{DL}}$ is downlink bandwidth configuration, expressed in multi-

ples of N_{sc}^{RB}, N_{sc}^{RB} is resource block size in frequency domain and N_{symb}^{DL} is number of downlink OFDM symbols in a time slot.

A group of contiguous sub-carriers and symbols form a resource block (RB). Data is allocated to each user in terms of RB. For a frame structure using normal CP, a RB spans 12 consecutive sub-carrier spacings of 15 kHz, and 7 consecutive symbols over slot duration of 0.5 ms. In other words, resource blocks are actually two dimensional (time-frequency) units with a size of 12 sub-carriers times 0.5ms slots.

The following table shows the number of resource blocks for LTE system operating at different bandwidths in the frequency domain;

Table 2-1 System Bandwidth and number of resource blocks

System Bandwidth [MHz]	1.4	3	5	10	15	20
Number of RBs	6	15	25	50	75	100
Signal Bandwidth [MHz]	1.08	2.7	4.5	9	13.5	18

The following table shows the number of OFDM symbols in a slot in time domain;

Table 2-2 OFDM symbols in a slot

Cyclic Prefix	Sub-carrier Spacing	Number of OFDM symbols in a time slot
Normal	15 kHz	7
Extended	15 kHz	6
	7.5 kHz	3

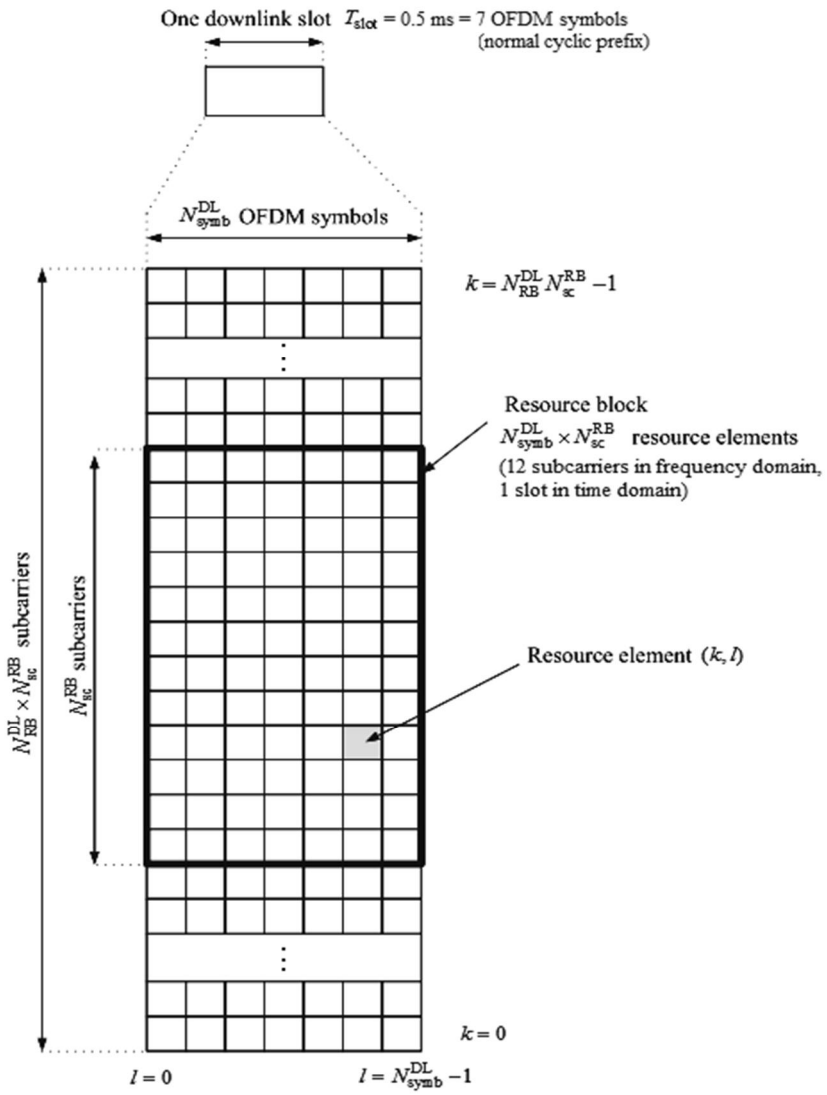

Figure 2.1. Downink resource grid. Relationship between a slot, OFDM symbols and Resource Blocks. [33]

The transmitted signal in each slot is described by one or several resource grids of $N_{RB}^{DL} N_{sc}^{RB}$ subcarriers and N_{symb}^{DL} OFDM symbols. The number of OFDM symbols in a slot depends on the cyclic prefix length and subcarrier spacing.

2.2. Downlink Frame Structure

Frame structure type 1 is applicable to both full duplex and half duplex FDD. Each radio frame is $T_f = 307200 \cdot T_s = 10$ ms long and consists of 20 slots of length $T_{slot} = 15360 \cdot T_s = 0.5$ ms, numbered from 0 to 19. A subframe is defined as two consecutive slots where subframe i consists of slots $2i$ and $2i+1$.

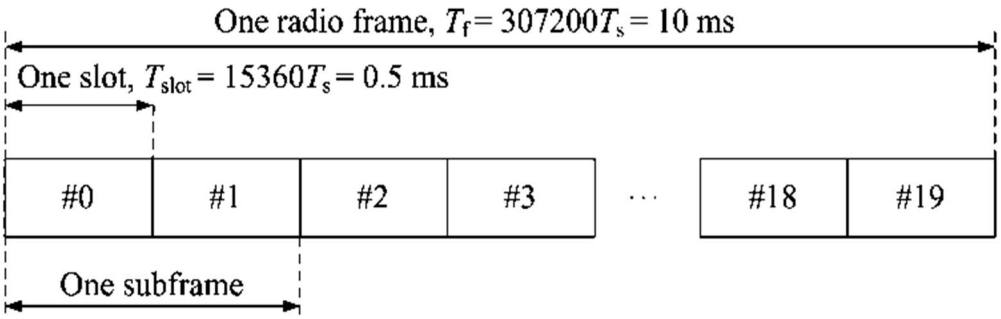

Figure 2.2. FDD Frame. Timing and symbol allocations shown for FDD with normal cyclic prefix (CP). [33]

2.3. LTE Downlink Physical Layer Signals

The LTE frame carries physical channels and signals. Channels carry information received from higher layers. Signals originate at the physical layer. The framing structure is common to uplink and downlink, but the physical signals and physical channels are different. [12]

The signals used in the carrier recovery algorithm are **reference**, **primary** and **secondary** synchronization signals.

2.3.1. Reference Signals

Reference signals are mapped to the resource elements in the frequency domain as shown in Fig. 2.3. Whenever there is one antenna port transmitting a reference signal on one resource element, all other antenna ports transmit a ‘zero’ symbol at this position. Thus, interference of the reference symbols transmitted from different antennas is avoided and channel estimation is simplified. Reference symbols are used in estimating the Residual Frequency Offset (RFO). The position of reference signal in OFDM systems is different based on the channel conditions and required Quality of Service (QoS). These may be located in block type, comb type, rectangular grid, parallelogram-shaped grid and hexagonal grid as depicted in the Figure. 2.3. The channel is assumed to be slow fading and reference symbols are in hexagonal configuration as shown in the Figure. 2.3 (e). There is only one antenna port in our case and all other ports are ignored. Reference symbols are transmitted in symbols (0 and 4) of each slot in case of normal CP and the first and the fourth in case of extended CP. The location of reference symbols varies on the subcarriers. Hexagonal configuration depicted in Figure. 2.3 (e) has been used in this work. Reference signals are transmitted every sixth symbol in hexagonal configuration. The hexagonal symbol arrangement reduces the reference symbol density and thus improves the spectral efficiency. Reference signals are transmitted on one or several of antenna ports 0 to 3.

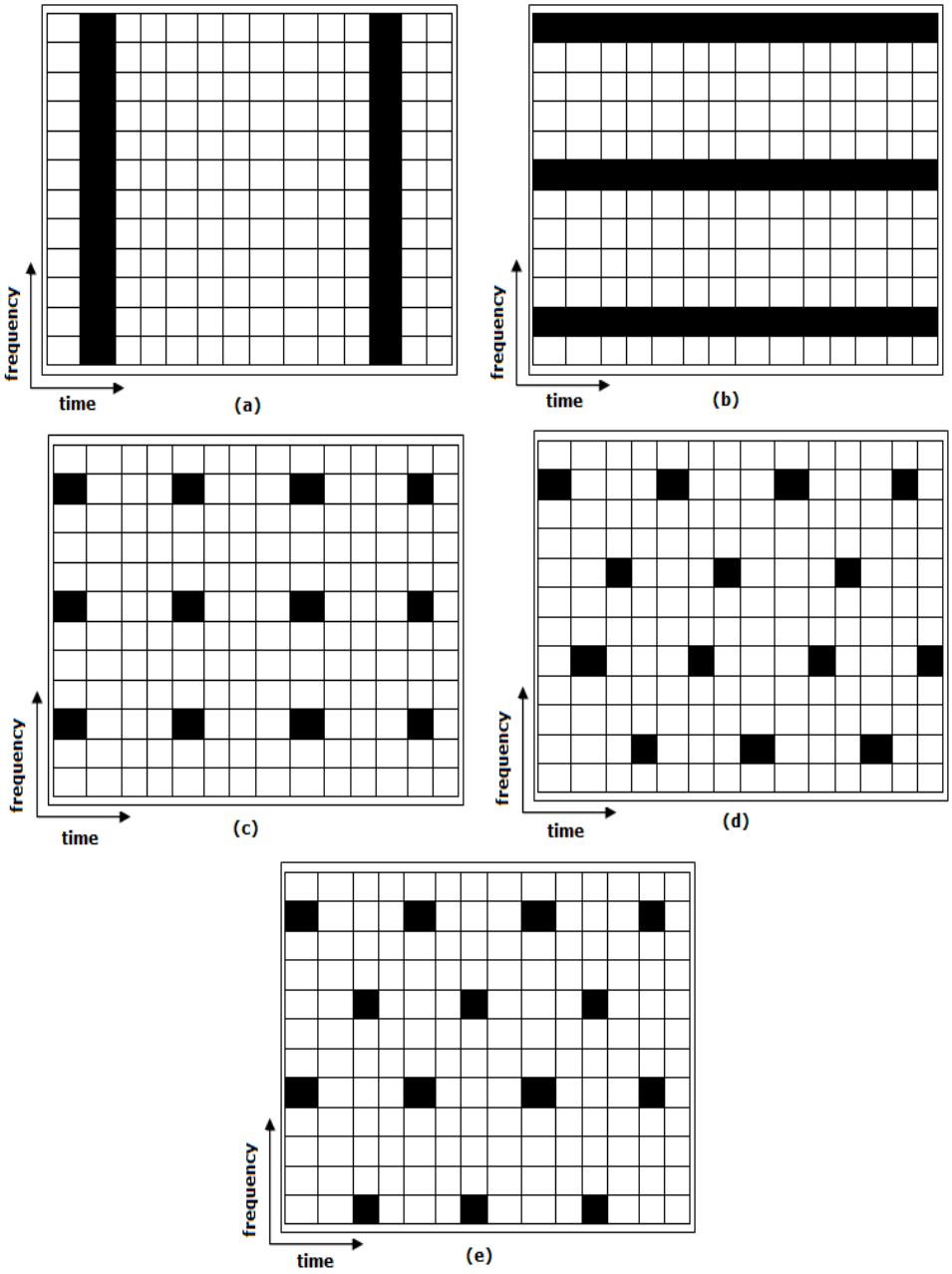

Figure 2.3. Illustration of several Reference signals pattern: (a) block type, (b) comb type, (c) rectangular grid, (d) parallelogram-shaped grid, (e) hexagonal grid (Adapted from [35])

2.3.2. Reference Signal Sequence Generation

The reference-signal sequence $r_{l,n_s}(m)$ is defined by,

$$r_{l,n_s}(m) = \frac{1}{\sqrt{2}}\left(1-2\cdot c(2m)\right) + j\frac{1}{\sqrt{2}}\left(1-2\cdot c(2m+1)\right), \quad m = 0,1,\ldots,2N_{\mathrm{RB}}^{\max,\mathrm{DL}} - 1 \tag{2.1}$$

where n_s is the slot number within a radio frame, $N_{RB}^{max,DL}$ is the largest downlink bandwidth and l is the OFDM symbol number within the slot.

Pseudo-Random Sequence: c is the pseudo-random sequence. Pseudo-random sequences are defined by a length-31 Gold sequence. The output sequence $c(n)$ of length M_{PN}, where $n = 0,1,...,M_{PN}-1$, is defined by,

$$c(n) = \left(x_1(n+N_C) + x_2(n+N_C)\right) \bmod 2, \tag{2.2}$$

$$x_1(n+31) = \left(x_1(n+3) + x_1(n)\right) \bmod 2, \tag{2.3}$$

$$x_2(n+31) = \left(x_2(n+3) + x_2(n+2) + x_2(n+1) + x_2(n)\right) \bmod 2, \tag{2.4}$$

where $N_C = 1600$ and the first m-sequence is initialized with

$$x_1(0) = 1, \text{ and}$$
$$x_1(n) = 0, n = 1,2,...,30$$

The initialization of the second m-sequence is denoted by,

$$c_{ini} = \sum_{i=0}^{30} x_2(i) \cdot 2^i$$

The pseudo-random sequence generator is initialised with

$$c_{\text{init}} = 2^{10} \cdot \left(7 \cdot \left(n_s + 1\right) + l + 1\right) \cdot \left(2 \cdot N_{ID}^{cell} + 1\right) + 2 \cdot N_{ID}^{cell} + N_{CP} \tag{2.5}$$

at the start of each OFDM symbol where , N_{ID}^{cell} is the physical layer cell ID and

$$N_{CP} = \begin{cases} 1 & \text{for normal CP} \\ 0 & \text{for extended CP} \end{cases}$$

2.3.3. Reference Signal Mapping to Resource Elements

The reference signal sequence $r_{l,n_s}(m)$ is mapped to complex-valued modulation symbols $a_{k,l}^{(p)}$ used as reference symbols for antenna port p in slot n_s according to:

$$a_{k,l}^{(p)} = r_{l,n_s}(m'), \tag{2.6}$$

where

$$k = 6m + (v + v_{\text{shift}}) \bmod 6$$

$$l = \begin{cases} 0, N_{\text{symb}}^{\text{DL}} - 3 & \text{if } p \in \{0,1\} \\ 1 & \text{if } p \in \{2,3\} \end{cases}$$

$$m = 0,1,\ldots,2 \cdot N_{\text{RB}}^{\text{DL}} - 1$$

$$m' = m + N_{\text{RB}}^{\max,\text{DL}} - N_{\text{RB}}^{\text{DL}}$$

The variables v and v_{shift} define the position in the frequency domain for the different reference signals where v is given by:

$$v = \begin{cases} 0 & \text{if } p = 0 \text{ and } l = 0 \\ 3 & \text{if } p = 0 \text{ and } l \neq 0 \\ 3 & \text{if } p = 1 \text{ and } l = 0 \\ 0 & \text{if } p = 1 \text{ and } l \neq 0 \\ 3(n_{\text{s}} \bmod 2) & \text{if } p = 2 \\ 3 + 3(n_{\text{s}} \bmod 2) & \text{if } p = 3 \end{cases}$$

The cell-specific frequency shift is given by $v_{\text{shift}} = N_{\text{ID}}^{\text{cell}} \bmod 6$. Resource elements (k,l) used for transmission of cell-specific reference signals on any of the antenna ports in a slot shall not be used for any transmission on any other antenna port in the same slot and set to zero.

Figure 2.4 illustrates the resource elements used for reference signal transmission according to the above definition. The colored resource elements denote resource elements being used for reference signal transmission on antenna port.

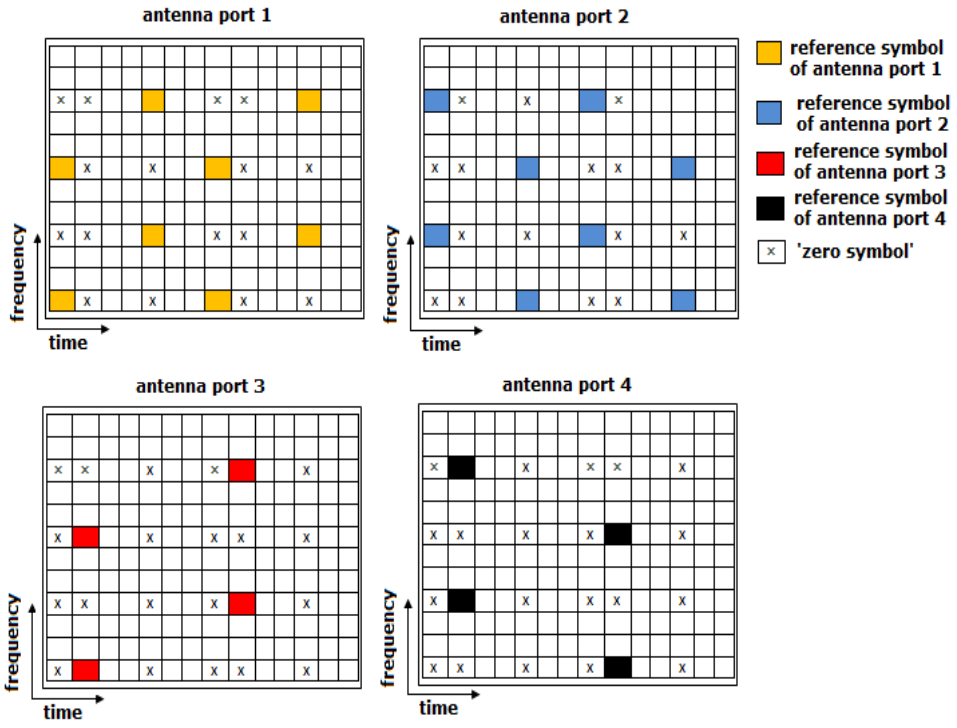

Figure 2.4. Reference signals in LTE downlink. (Adapted from [12])

2.3.4. Synchronization Signals

LTE standard has defined two synchronization signals: these are primary (P-SCH) and secondary (S-SCH). The two signals are identified with different colors in the Fig. 2.5. These signals are located on 62 subcarriers within 72 reserved subcarriers symmetrically arranged around DC-carrier in the first slot in the sixth and seventh OFDM symbols of the first and sixth subframes [15]. The synchronization signals are transmitted twice per 10 ms on predefined slots as depicted in Figure 2.5. Signal configuration is shown in the following figure:

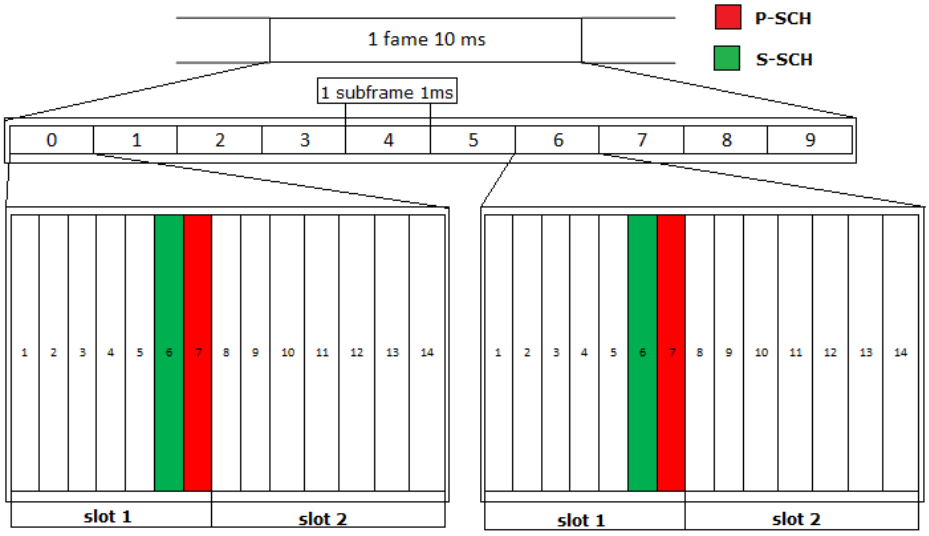

Figure 2.5. Primary and Secondary synchronization signals (Adapted from [12])

Physical-Layer Cell Identity (PCI): PCI is used in the sequence generation and resource element mapping of primary and secondary synchronization signals.

There are 504 unique physical-layer cell identities. In the LTE air interface, Physical Layer Cell Identity (PCI or CellID) is used for cell identification. The physical-layer cell identities are grouped into 168 unique physical-layer cell-identity groups, each group containing three unique identities. The grouping is such that each physical-layer cell identity is part of one and only one physical-layer cell-identity group. The PCI is expressed as:

$$N_{\mathrm{ID}}^{\mathrm{cell}} = 3N_{\mathrm{ID}}^{(1)} + N_{\mathrm{ID}}^{(2)}, \qquad (2.7)$$

A physical-layer cell identity '$N_{\mathrm{ID}}^{(1)}$', is uniquely defined by a number in the range of 0 to 167, representing the physical-layer cell-identity group, and a number $N_{\mathrm{ID}}^{(2)}$ in the range of 0 to 2, representing the physical-layer identity within the physical-layer cell-identity group.

2.3.5. Primary Synchronization Signal Sequence Generation and Resource Element Mapping

The primary synchronization signal carries the physical layer identity 0, 1, or 2. The sequence $d(n)$ used for the primary synchronization signal is generated from a frequency-domain Zadoff-Chu sequence according to the following expression:

$$d_u(n) = \begin{cases} e^{-j\frac{\pi un(n+1)}{63}} & n = 0,1,...,30 \\ e^{-j\frac{\pi u(n+1)(n+2)}{63}} & n = 31,32,...,61 \end{cases}, \qquad (2.8)$$

where the Zadoff-Chu root sequence index u is given by Table 2.3.

Table 2-3 Root indices for Primary Synchronization Signal

$N_{\mathrm{ID}}^{(2)}$	Root index u
0	25
1	29
2	34

It is assumed that the primary synchronization signal is not transmitted on the same antenna port as any of the downlink reference signals. Furthermore it is also assumed that any transmission instance of the primary synchronization signal is not transmitted on the same antenna port, or ports, used for any other transmission instance of the pri-

mary synchronization signal. The sequence $d(n)$ shall be mapped to the resource elements according to:

$$a_{k,l} = d(n), \tag{2.9}$$

$$\text{where} \quad n = 0,...,61 \quad \text{and} \quad k = n - 31 + \frac{N_{\text{RB}}^{\text{DL}} N_{\text{sc}}^{\text{RB}}}{2}$$

The primary synchronization signal is mapped to the last OFDM symbol in slots 0 and 10.

2.3.6. Secondary Synchronization Signal Sequence Generation and Resource Element Mapping

The secondary synchronization signal carries the physical layer cell identity group. The sequence $d(0),...\ d(61)$ is used for the second synchronization signal. It is an interleaved concatenation of two length-31 binary sequences. The concatenated sequence is scrambled with a scrambling sequence given by the primary synchronization signal.

The combination of two length-31 sequences defining the secondary synchronization signal differs between subframe 0 and subframe 5 according to:

$$d(2n) = \begin{cases} s_0^{(m_0)}(n)c_0(n) & \text{in subframe 0} \\ s_1^{(m_1)}(n)c_0(n) & \text{in subframe 5} \end{cases}, \tag{2.10}$$

$$d(2n+1) = \begin{cases} s_1^{(m_1)}(n)c_1(n)z_1^{(m_0)}(n) & \text{in subframe 0} \\ s_0^{(m_0)}(n)c_1(n)z_1^{(m_1)}(n) & \text{in subframe 5} \end{cases}, \tag{2.11}$$

where $0 \le n \le 30$. The indices m_0 and m_1 are derived from the physical-layer cell-identity group $N_{\text{ID}}^{(1)}$ expressed as:

$$m_0 = m' \bmod 31$$
$$m_1 = (m_0 + \lfloor m'/31 \rfloor + 1) \bmod 31$$
$$m' = N_{\text{ID}}^{(1)} + q(q+1)/2, \quad q = \left\lfloor \frac{N_{\text{ID}}^{(1)} + q'(q'+1)/2}{30} \right\rfloor, \quad q' = \lfloor N_{\text{ID}}^{(1)}/30 \rfloor$$

Same antenna port as for the primary synchronization signal is used for the secondary synchronization signal. The sequence in equation 2.9 is used for mapping secondary synchronization signals to resource elements but expression for *k* differs:

$$k = n - 31 + \frac{N_{RB}^{DL} N_{sc}^{RB}}{2}$$
$$l = N_{symb}^{DL} - 2 \qquad \text{in slots } 0 \text{ and } 10$$
$$n = -5,-4,...,-1,62,63,...66$$

The index values of *n* are reserved and not used for transmission of the secondary synchronization signal.

2.4. Downlink Multi-Antenna Transmission

Multiple input and multiple output (MIMO) antenna technology is a key feature for the LTE downlink. Different downlink MIMO modes are specified for LTE which can be adjusted and flexibly configured in accordance with channel conditions and data rate requirements. “It is noteworthy that full selection of MIMO options is available only for the downlink in LTE.” [34]

Spatial multiplexing is an important MIMO technique. It is also referred to as “true MIMO”. It is the technique to transmit different data streams simultaneously over the same set of radio resources. If spatially multiplexed data streams belong to one user, it is referred to as single-user MIMO (**SU-MIMO**). In case spatially multiplexed data streams belong to different users, it is multi-user MIMO (**MU-MIMO**). Both SU-MIMO and MU-MIMO are supported in LTE downlink. SU-MIMO directly impacts the data rate of the user and MU-MIMO helps to increase the capacity of the network by accommodating multiple users.

Multi-antenna transmission with up to 8 transmit antennas is supported. The maximum number of codeword is two irrespective to the number of antennas with fixed mapping between code words to layers. [33] A codeword is a block of information bits that can be encoded, scrambled, and modulated separately before it is transmitted in a subframe over the air interface. [34]

MIMO and the principle of spatial multiplexing are displayed in Figure 2.6.

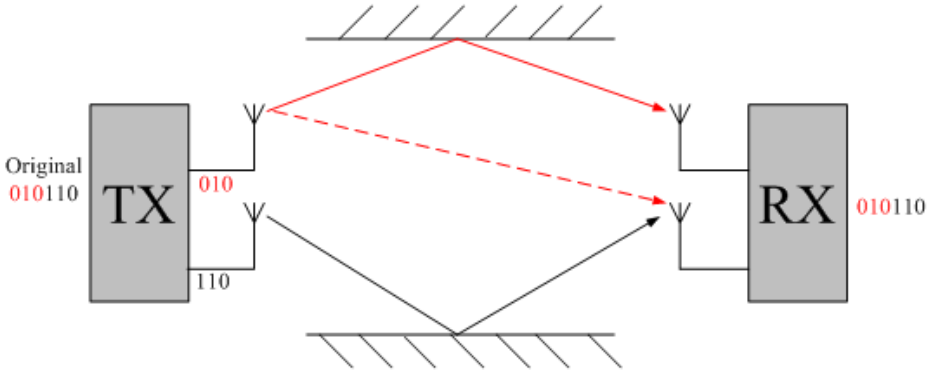

Figure 2.6. MIMO and Spatial Multiplexing (Adapted from [34])

Other important MIMO techniques are transmit diversity and beamforming.

In **Transmit diversity** antennas at the transmitter transmit the same data streams simultaneously or in other words replicas of the same signal are transmitted. The information content of each stream remains the same but each antenna uses a different coding. Transmit diversity does not increase the data rate but it increases the robustness of the transmission against fading effects by using the diversity effect.

Beamforming is the shaping of the antenna radiation pattern. The shapped antenna beam is used to focus the transmission energy in the direction of the receiver. It is also used to mitigate possible interferers. Beamforming is done with an antenna array that contains multiple antenna elements which can be individually modified in gain and phase.

There are different MIMO modes mentioned in [34], starting from Mode 1 upto Mode 7. “Each MIMO mode requires a different type and amount of control signaling to be conveyed to the terminal”. [34]

3. LTE CARRIER RECOVERY

Synchronization is a critical function in digital communications. Its failure may cause catastrophic effects on the transmission system performance. It is very important that the receiver is synchronised with the transmitter because it is not possible to correct frequencies/phases without any control mechanisms.

Synchronization is different in 3GPP LTE for uplink and downlink because of the choice of multiple access schemes. Multiple access scheme for LTE downlink is OFDMA and SC-FDMA for the uplink. OFDMA is susceptible to Carrier Frequency Offset (CFO). In case of a typical LTE system with a carrier frequency of 2.1 GHz, a frequency drift of 10ppm (10×10^{-6}) of the local oscillator can cause an offset of 21 kHz. LTE system employs a fixed subcarrier spacing of 15 kHz. This offset caused by the local oscillator corresponds to 1.40 subcarrier spacings.

The receiver extracts the information from the received signal to synchronise and compensate for any carrier phase offset. In Figure 3.1 it can be seen that before the OFDM symbols are demodulated the carrier frequency offset must be compensated.

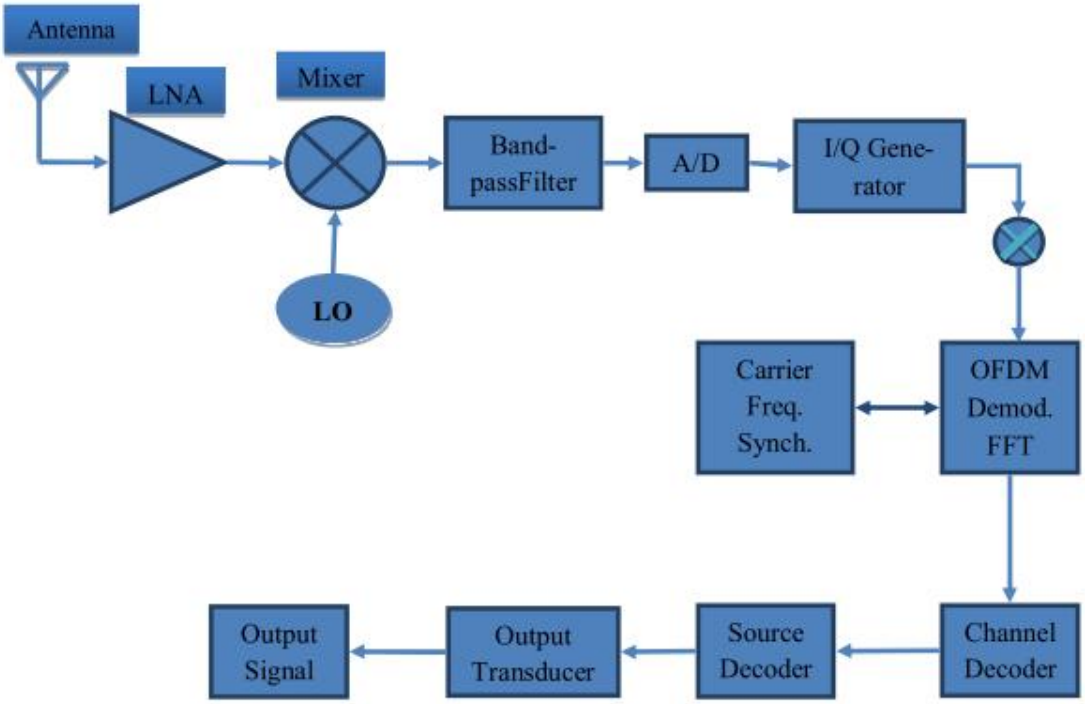

Figure 3.1 OFDM Receiver

3.1. Carrier Recovery

Information is sent over a carrier by doing some changes in its fundamental characteristics like phase, frequency and amplitude, in a digital communication system. These fundamental characteristics modifications must be detected by the receiver in order to recover the data correctly.

Carrier recovery is an important aspect in synchronizing digital communication systems because errors can occur in the carrier frequency/phase due to many reasons. Error may be caused by the transmitter local oscillator during up-conversion and modulation, receiver local oscillator used for down-conversion or demodulation, up-conversion or down-conversion in repeaters, and Doppler shift in mobile channels. The traditional solution is to adaptively adjust the local oscillator to match the frequency and phase of the received signal. Carrier recovery can be performed in a number of ways in different systems:

- Sufficiently accurate frequency adjustment is enough for coherent and non-coherent systems.
- Coherent detection is required for accurate phase recovery
- Quadrature carriers require accurate phases for complex symbols or alphabets.

It is assumed that the receiver is able to generate a reference carrier whose phase and frequency are identical to those of the carrier's at the transmitter. When the receiver exploits the knowledge of the carrier's phase to detect the signal, the process is called coherent detection. Carrier signal is generated by the local oscillator (LO). Generally the LO at the receiver is not synchronous with the LO at the transmitter. To match the receiver's local oscillator carrier frequency and phase to that of the transmitter's, carrier recovery is done and strictly required in coherent detection system such as OFDM systems. Generally carrier recovery is performed in two steps: first step is coarse frequency adjustement; second step is fine frequency adjustment and phase recovery.

Carrier synchronization error can affect the carrier phase and carrier frequency. There might be rotation in the constellation resulting in constant carrier phase error φ.

$$\bar{A}_k = e^{j\varphi}.A_k \tag{3.1}$$

There might be a time-varying rotation of the constellation resulting in constant carrier frequency error Δω.

$$\bar{A}_k = e^{jkT\Delta\omega}.A_k \tag{3.2}$$

In the past analog and hybrid methods utilizing both analog and digital solutions had been used for carrier recovery. Two salient features of these solutions are:

- Directly adjusting the demodulator local oscillator
- Phase-locked loops and related circuitory

There is an increase in utilization of digital synchronization techniques during the last decade. Digital synchronization principles can be classified into three categories:

Data-Aided (DA): By transmitting a separate sequence known to the receiver (reference symbols, pilot symbols, preambles/midambles)

Data-Directed (DD): By utilizing information extracted from the detected symbol.

Non-Data-Aided (NDA): This method does not depend on the detected symbol.

It is evident that DA exhibits the best synchronization performance, but there is some overhead in terms of some part of the bandwidth lost for pilot signals or training sequences. There are different carrier recovery methods. A well-known carrier recovery is the **Maximum Likelihood (ML)** method. This method is applicable to all the above mentioned DA, DD and NDA configurations. ML theory has been used in this work.

3.2. LTE Downlink Carrier Recovery Algorithm

General OFDM systems can be synchronized by splitting the carrier frequency offset (CFO) into Fractional Frequency Offset (FFO), Integer Frequency Offset (IFO), and Residual Frequency Offset. These offsets can be estimated individually [16;17;18]. The entire implementation of this work is based on standardized synchronization signals and reference symbols of LTE as explained in the previous chapter, following the approach presented in [16].

As discussed earlier in the previous chapter the LTE standard defines two kinds of signals utilized for synchronization at the receiver. First one is the dedicated synchronization signals and the second is the cell-specific reference signals. Their exact details used in the implementation are discussed in the following subsection:

3.2.1. Carrier Frequency Offset Estimation Method

The three stage CFO compensation scheme for 3GPP LTE is explained here. It is assumed that the channel is slow fading, so that the variation of the channel impulse response within one subframe of duration 1 ms is negligible.

The following equation defines the system model in time domain:

$$r = \{x * h + v\} \cdot e^{i2\pi\epsilon_{CFO}(n+l(N+N_g))/N}, \tag{3.3}$$

'r' the received time-domain signal, 'x' is the transmitted OFDM signal, 'h' is the channel impulse response, 'v' is the additive Gaussian noise, l is the OFDM symbol index in one subframe, 'N' denotes FFT size, 'n' is the time index within one OFDM symbol, 'N_g' the CP length and ε_{CFO} is frequency mismatch between transmitter and receiver.

When discrete Fourier transform is applied to the eq. 3.3, we have the corresponding frequency response.

$$R = \gamma . XH + I + V, \tag{3.4}$$

where,

$$\gamma = \frac{\sin(\pi\varepsilon')}{N sin(\pi\varepsilon'/N)} . e^{i\pi\varepsilon'(N-1)/N} . e^{i2\pi\epsilon_{CFO}l(N+N_g)/N}, \tag{3.5}$$

$$I = \sum_{p\neq k'} XH \cdot \frac{\sin(\pi(p+\varepsilon_{CFO}-k))}{N \sin(\pi(p+\varepsilon_{CFO}-k)/N)} . e^{i\pi(p+\varepsilon_{CFO}-k)(N-1)/N} . e^{i2\pi\epsilon_{CFO}l(N+N_g)/N}, \tag{3.6}$$

The factor 'γ' is the degradation on the desired subcarrier caused by the CFO. 'I' is the inter-carrier interference caused from neighboring subcarriers. Expressions for 'γ' and 'I' are derived in [16]. The total CFO is split into FFO, IFO and RFO:

$$\varepsilon_{CFO} = \varepsilon_{FFO} + \varepsilon_{IFO} + \varepsilon_{RFO}, \tag{3.7}$$

FFO is the major source of inter-carrier interference (ICI); it removes the orthogonality between the subcarriers. Therefore, it must be compensated before the FFT operation in the OFDM receiver.
The expression for the FFO mismatch has been derived in [16] for the LTE subframe;

$$\varepsilon_{FFO} = -\frac{1}{2\pi} arg\left\{\sum_{l=1}^{N_f}\sum_{n=1}^{N_g} r_{l,n} r_{l,n+N}^{*}\right\}, \tag{3.8}$$

where, 'r' is the received symbols, 'N_f' is the number of received OFDM symbols in one subframe, and 'N_g' is the CP length. The estimation range for this stage is -0.5f_s to 0.5f_s, where 'f_s' is the subcarrier spacing. CP is used to estimate the FFO within the estimation range according to [16]. FFO estimation is independent of the presence of IFO

and RFO. FFO can be further reduced by using more antennas at the receiver, by increasing the CP length or by taking more OFDM symbols into account.

IFO is the carrier offset by an integer multiple of subcarrier spacing. IFO is estimated after the fractional part of the CFO has been corrected. IFO has two effects on the received signal. First one is the FFT index shift i.e. frequency translation of a subcarrier by an integer multiple of subcarrier spacing if LO at transmitter and receiver are at different frequencies. The second one is the phase rotation of the symbols if the two oscillators are not in phase with each other. These two effects can be compensated by using a ML estimator proposed in [19]. The standardized synchronization signals are used for the frequency and phase correction. The IFO estimator based on the idea presented in [19] is represented by the following equation:

$$\varepsilon_{IFO} = arg\max_i\{Re\left[e^{\frac{j2\pi iN_g}{N}}.\sum_{k\in K_{SCH}}(R^*_{SSCH}R_{PSCH})(X^*_{SSCH}X_{PSCH})^*\right]\}, \quad (3.9)$$

where 'R_{SSCH}', 'R_{PSCH}', 'X_{SSCH}' and 'X_{PSCH}' are the received and transmitted secondary and primary synchronization signals respectively. The set 'K_{SCH}' represents the subcarrier indices that contain the synchronization signals. 'i' ranges from -31 to 31 and corresponds to the set of integer offsets that can be estimated. In LTE synchronization signals exist in every fifth subframe only, therefore estimation of 'ε_{IFO}' can only done in these subframes.

RFO, extent of RFO depends on the magnitude of the estimation error of FFO. Its impact on the current OFDM symbols may not be clearly visible, but the estimation error results in an increase in the phase shift for the OFDM symbols to be received subsequently. It is necessary to improve the estimation on the subframe basis by utilizing the reference symbols in the frequency domain to correct the RFO after the FFT.
It is assumed that 'ε_{FFO}' and 'ε_{IFO}' have been corrected to an extent that the magnitude of the interference is negligible. The resulting RFO estimator expressed in [16] is:

$$\varepsilon_{RFO} = -\frac{1}{2\pi}\frac{N}{N_s(N+N_g)}.\arg\{\sum_{(k,l)\in K_p}\left(R_{l,k}R^*_{l+N_s,k}\right)(X_{l,k}X^*_{l+N_s,k})^*\}, \quad (3.10)$$

where 'R' and 'X' are the received and transmitted reference symbols respectively, 'N_s' is the number of OFDM symbols in one slot and 'K_p' is the joint set of OFDM symbol number and subcarrier indices for reference symbols.

3.3. LTE Uplink Carrier Recovery

OFDMA properties are less favourable for the uplink. This is mainly due to weaker peak-to-average power ratio (PAPR) properties of OFDMA signal, resulting in worse uplink coverage. [34]

Synchronization is different in the uplink because of a slightly different multiple access scheme in the uplink, SC-FDMA. Both OFDMA and SC-FDMA are multiple access versions of OFDM. SC-FDMA signals have better PAPR properties in comparison to an OFDMA signal. This is one of the main reasons for selecting SC-FDMA as LTE uplink multiple access scheme. Two types of uplink reference signals are supported in the uplink:

- Demodulation reference signal
- Sounding reference signal

"The same set of base sequences is used for demodulation and sounding reference signals in the uplink" [34], but different sequences were used for the reference and synchronization signals in the downlink.

Another aspect of uplink carrier recovery is the uplink MIMO scheme. Uplink MIMO schemes for LTE differ from downlink MIMO schemes in order to limit terminal complexity. Many user terminals may transmit simultaneously on the same resource block, also known as Spatial Division Multiple Access (SDMA). Only one transmit antenna is required on the terminal end in this scheme. The terminals sharing the same resource block must apply mutually orthogonal pilot patterns, i.e. demodulation reference signals.

MU-MIMO can be used in the uplink. [34]

4. TCE FRAMEWORK

Move Processor and Transport Triggered Architecture are relevant terms, used interchangeably referring to the same architecture. The Innovative and exciting concept was introduced by Prof. Henk Corporaal at the Delft University of Technology, The Netherlands in 1990's. [20]

The concept is based on recent research performed within the MOVE project at the Delft University of Technology. Unlike most traditional architectures, TTAs function through programmed operations that are triggered by internal data transports. As a result the new architecture alleviates bottle-necks, allows for new code generation optimizations, and exploits hardware more efficiently. TTAs can be used as a template for automatic generation of application specific processors and are well-suited for embedded system design. [21]

4.1. Transport Triggered Architecture (TTA)

TTA concept is relatively new in the field of computer architecture. The main difference between TTAs and conventional Operation-triggered architectures (OTAs) is based on the way they are programmed. In case of OTAs operations are specified and then programmed, and data transports are handled and scheduled by the hardware. In case of TTAs the compiler is responsible for translating the operations into data operations. [23]

A TTA processor consists of functional units (FUs), special functional units (SFUs), register files (RFs), interconnection (IC) network, control logic, and separate instruction and data memories as shown in Fig. 4.1. The IC is responsible for transferring data between the FUs and the RFs. It is composed of data transport buses and sockets, which are connecting the FUs and the RFs to each others. A common approach to enhance the performance of TTA processors is to increase the number of functional units which operate concurrently.

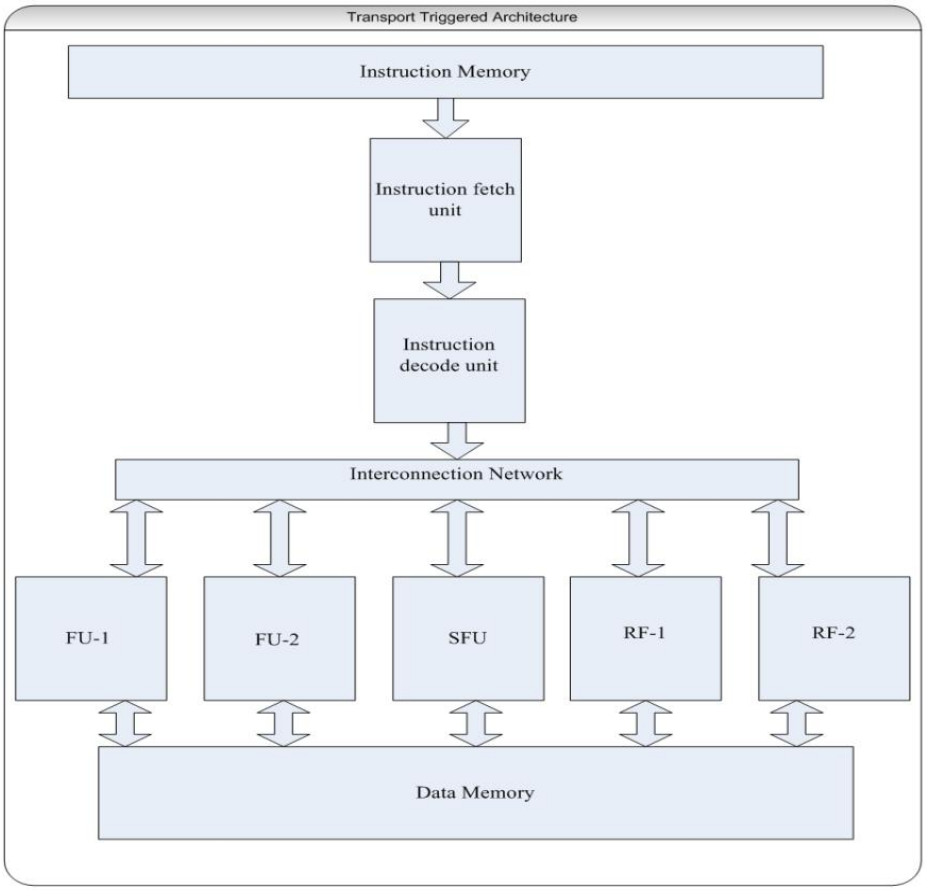

Figure 4.1. Organiztion of TTA

A TTA is programmed by specifying the required data transports. The number of data transports or moves depends on the number of transport buses. Each move specifies a transport from a FU output to one of the FU inputs or the corresponding general purpose register. The FU interconnection network is not limited in any sense as long as point to point transports can be specified. The network is not required to be fully connected, although full connectivity would ease the code generation process. It is the task of the compiler to optimize the required number of transports, given certain connectivity, such that the number of execution cycles is minimized. Limiting bus connections is an important tool in reducing bus loads, yielding faster transport times. [23]

4.2. Hardware Aspects of TTA

TTAs have several characteristics which make them very interesting from a hardware design point of view:

Modularity: TTAs are constructed using a limited number of building blocks. They are built by proper connection of FUs and bus-connections. FUs are completely independent of each other and of the interconnection network. FUs can therefore be designed separately, and pipelined independently. The modularity of TTAs allows the

hardware design process to be automated. Different TTAs can easily be configured by assembling different combinations of these blocks.

Flexibility and Scalability: TTAs have large flexibility because the interconnection network is separate from the FUs and both can be designed independently. TTAs have virtually no constraints on how to design and pipeline the network, and different FUs as long as both obey the interface specification. FUs can implement any functionality even if it requires more than two operands and multiple results.

Processor Cycle Time: The processor can be optimized for operation throughput, instead of latency if needed. This requires extensive pipelining those FUs which constrain the achievable cycle time. The result is cycle time which is limited by the time needed for inter FU data transports only. The implementation can be tuned to optimize this time by using advanced bus implementation techniques.

Hardware Efficiency: There are several aspects contributing to an economic usage of hardware resource: TTAs basically support only one operation format. The compiler is in control of every data transport. Many of these transports can be optimized away. Many values produced during the course of a program do not need to be allocated in general purpose registers; the register file traffic rate reduces. FU logic can be split into independent parts, used for different functionality without a large impact on the interconnection network and register file. These parts may be used concurrently after splitting which results in a larger efficiency. [23]

4.3. TTA Programming Model

TTA has only one instruction, the *move* instruction which implements operand transports. The need to support only a single instruction combined with the static operation scheduling makes the control logic of the TTA very simple.

The operand transports of TTA are visible to the programmer. In a traditional operation based assembly, an addition *r3 = r1 + r2* could be executed as:
add r3, r1, r2

In TTA assembly, addition is executed by defining the operand transports:

r1 - > add.o1
r2 -> add.t
add.r -> r3

Or, if there are multiple transport busses available, the input operands could be transferred simultaneously:

```
r1 -> add.o1, r2 -> add.t
add.r -> r3
```

It is notable that the operation latency is also visible to the programmer. In the above example the latency of the addition operation is 1 clock edge. The operation result can be read from the output on the next instruction after the triggering move.

One of the advantages of the operand transport paradigm is software bypassing [24] and Dead Result Elimination (DRE). For example the following code in operation based assembly:

```
add r3, r1, r2
shift r6, r3, r4
store r3, r6
```

could be executed on a TTA with two transport buses as:

```
r1 -> add.o1, r2 -> add.t
add.r -> shift.o1, r4 -> shift.t
add.r -> store.o1, shift.r -> store.t
```

As the example demonstrates, the addition result can be bypassed to the shift and store operations. Likewise, the shift result is bypassed to store. Furthermore, as the results of addition and shift can be bypassed to the next instructions, dead result elimination removes the unnecessary result writes to the register file. In the example, these optimizations decreased register file utilizations by removing two register writes and reads.

TTA also makes it easy to schedule code for custom MIMO operations. The greatest advantage in case of MIMO operations is that all the input and output operands do not have to be transported in a single instruction cycle. The input port registers of a function unit hold their values until they are overwritten and the operation results stay in the output port registers as long as the next operation is triggered and new results are written to the outputs. This makes it possible to divide the operand transports to multiple instruction cycles which puts less pressure on the register file. For example, if there was an operation with eight inputs and all the operands were stored in a register file, there would have to be eight read ports in a register file in order to provide all the operands in a single instruction cycle. However, in the case of TTA, the input operands can be transported in multiple instruction cycles depending on the available register file read ports. For example, if there are two read ports, the input operands can be transported in four instruction cycles. [25]

4.4. TCE (TTA-based Co-design Environment)

TCE is an open source toolset for designing application specific processors based on transport triggered architecture. TCE provides a complete co-design flow from C programs down to synthesizable VHDL and parallel program binaries. Processor customization points include the register files, function units, supported operations, and the interconnection network. TCE has been developed in Tampere University of Technology since early 2003. [26]

The most essential tools in TCE are the processor design tool *ProDe*, a retargetable high level language compiler *tcecc*, the retargetable *Instruction-Set Simulators (ISS) ttasim*, *proxim* (graphical user interface version) and the processor generator *ProGe*. These tools allow high level language (HLL) to RTL design flow. The retargetability of the tools means that they automatically adapt to the processor architecture at the runtime.

TCE allows the designer to customize TTA processors, for example the designer can change the number of function units in the architecture, modify which functions are included in a function unit and even create new operations. The register files can be customized as well. The designer can change the register width, the register file size, and the register file port count. The number of register files is also customizable. The interconnection network can also be tailored. The number of transport buses and connections between FUs, RFs and transport buses are also customizable. [25]

The TCE design flow is depicted in the following state flow diagram:

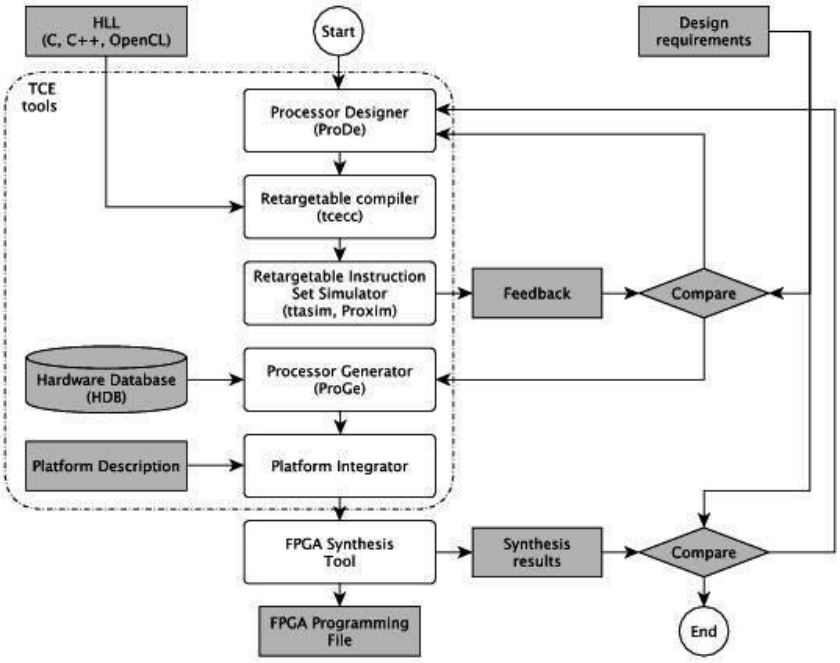

Figure 4.2. TCE design flow [25]

4.5. TTA Design with TCE

The goal of TCE design flow depicted in Fig. 4.2 is to produce a TTA processor which is able to execute specific application while complying with the restrictions set by the design requirements. The design flow inputs are HLL (high level language) source code of the desired application and the design requirements. These requirements can define the amount of FPGA resources the implementation can use, the target execution time or performance, the minimum clock frequency or the maximum allowed energy consumption. At the beginning of the design flow, the designer uses *ProDe* to create starting point architecture. *ProDe* stores the processor architecture description in XML-format to an architecture definition file (ADF). Or alternatively a pre-existing ADF can be used as the starting point.

The next step is to compile the source code for the starting point architecture with the *tcecc* compiler which oututs a *TTA Program Exchange Format* (TPEF) binary file. The retargetable instruction set simulator *ttasim* gets the ADF and TPEF files as imputs and produces the simulation results, such as execution cycle count, processor resource utilization data and optionally execution traces which can be utilized to extract profiling data. The simulation results are the feedback from the flow. The designer analyses the feedback and modifies the architecture accordingly with *ProDe*. Then a new iteration is started and the new feedback shows how the modifications affected the results. The iteration process is also known as manual processor *Design Space Exploaration* (DSE). TCE also uses an experimental explorer tool which can be used to automate the processor design space exploration. In the automated DSE, the designer sets goals for explora-

tion and then the explorer modifies the processor architecture until the given goals are reached. [27]

Processor design space exploration is continued until the design requirements are met or the designer determines that the results are adequate. It should be noticed that the maximum clock frequency of the processor at this point is unknown. Thus, the actual run time of the application is also unknown because the simulation results only show the instruction cycle count. If the design requirements set a target clock frequency, the execution time and the FPGA resource usage, the processor must be implemented and synthesized. The RTL implementation of the processor is generated with the *ProGe* tool. Before generating the RTL, processor architecture resources such as the FUs and RFs must be mapped to their actual RTL implementations.The FU and RF implementation are stored in hardware databases (*HDB*), and the resource mapping is done by simply defining which HDB entry is used for each architecture resource. The mapping information is written to an *Implementation Definition File* (*IDF*). When the mapping is done, *ProGe* uses the ADF and IDF to create the processor RTL implementation by generating the IC network and connecting the FUs and RFs together.

Only the processor core is implemented at this point. The core must be interfaced with the target platform in order to execute the applications. The processor can be synthesized with a third party synthesis tool after the integration process. The synthesis produces an FPGA device programming file which is used to configure the design onto the FPGA for execution. The synthesis results are valuable feedback for the design flow. The results determine the actual resource usage and maximum clock frequency of the processor. The designer compares these results with the design requirements to determine whether the iterative design process can be finished.

TCE also allows the designer to exploit custom hardware operations which can be used to accelerate the application. TCE allows the custom operations to be tested and evaluated without having the RTL implementation of the custom operation, due to the separation of the processor architecture and implementation. The custom operation design flow is illustrated in Fig. 4.3. The flow begins by searching for a custom operation candidate. Application profiling can be helpful for this purpose. When a candidate is found, the designer creates a *custom operation compiler definition* by using the Operation Set Editor tool *OSEd*. The compiler definition simply describes the name of the operation and the number of input and output operands. In order to simulate the operation, a *simulation model* is needed. This model implements the operation behavior using C/C++. Usually the software function in the accelerated program can be exploited in defining the simulation model. For example, if the custom operation replaces a function, the function code can be utilized as the simulation model.

Using new custom operation requires modifications to the processor architecture and the HLL source code. The designer must add a function unit containing the custom operation to the architecture. At this point, the custom operation latency must be defined. Sometimes it can be difficult to determine the operation latency before the operation hardware is implemented. The designer can either take a suitable guess or change the latency between iterations to find out which latencies would be feasible.

For the software to be able to use the custom operation, the designer needs to modify the source code to utilize the new operation. This is done by calling the operation via TCE-specific *operation macros or intrinsics*. In case the custom operation implements a function from the original code, these function calls are replaced with calls to the operation macros. When the modifications are ready, the application can be compiled and simulated. Feedback from the simulation will reveal how the custom operation affected the execution cycle count. If the results do not satisfy, another custom operation can be tested by starting a new iteration.

If the custom operation speedup was adequate and the designer chooses to include the custom operation to the architecture, the custom operation must be implemented. This is the only step in the TCE design flow where the designer is required to write RTL code. TTA Unit Tester verifies that the RTL implementation of the custom operation is equal to its simulation model. When custom operation implementation is ready, the custom FU is added to a hardware database with the *hdbeditor* tool. This allows the FU to be reused in later designs. [25]

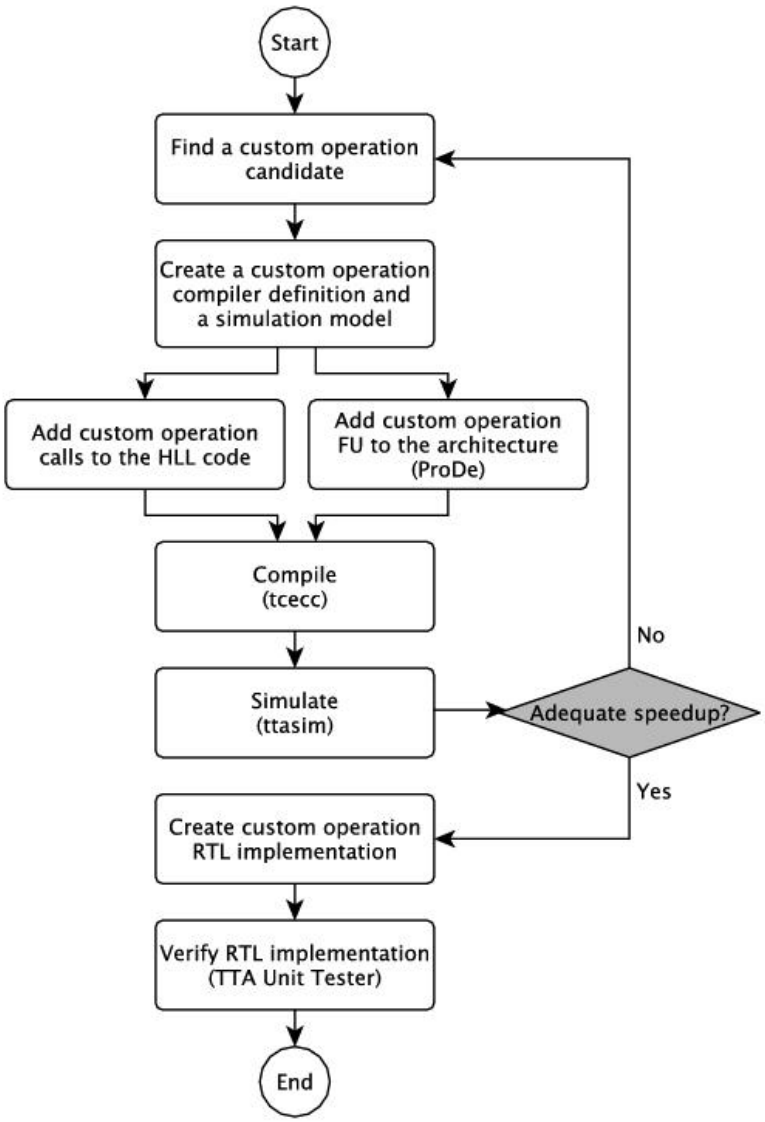

Figure 4.3. TCE custom operation design flow. [25]

5. IMPLEMENTATION

The implementation details of the work are discussed in this chapter.

5.1. Implementation in C

The algorithm is implemented in C language using data in fixed point format (least significant 12 bits for fractional part and the most significant 4 bits for the fixed part) and making the code as efficient as possible by getting rid of the unnecessary computations, such as divisions and multiplications. C extensions have been used for operations like FFT, Cordic and complex multiplications to speed up the process.

5.1.1. Fractional Carrier Frequency Offset

Fractional carrier frequency offset (FCFO) is implemented by extracting the indices of the complex CPs from the received OFDM symbol stream. The number of CP points varies with the system bandwidth. For a 10 MHz bandwidth there are 1024 CP points and for 20 MHz it is 2048 CP points.The received CP points are stored in two different arrays with a shift of 2K FFT points in case of 20 MHz system bandwidth. These two arrays are then multiplied and their sum is taken to find a maximum value. Fractional estimate is computed from this maximum value by taking it phase. This estimate is then used to correct the received symbol stream.

5.1.2. Integer Carrier Frequency Offset

Integer carrier frequency offset (ICFO) is independent of the system bandwidth. It is computed with the help of primary and secondary synchronization symbols. It is based on the correlation of the received primary and secondary synchronization symbols with the locally generated primary and secondary symbols at the receiver and then finding the peak value which results in a perfect ineteger value indicating that the receiver is able to locate the centre frequency of a carrier. The received symbol stream is then corrected based on this integer value.

5.1.3. Residual Carrier Frequency Offset

The residual offset is corrected with Residual Carrier Frequency Offset (RCFO). RCFO is computed with the help of reference symbols. The number of reference symbols depends on the system bandwidth. Residual carrier offset is computed by correlating the

reference symbols and summing the results of the correlations. Finally phase is calculated based on the summation of the correlations.

5.2. Channel Characteristics and Synchronization Parameters

As mentioned earlier received symbol stream is in fixed point format (least significant 12 bits for fractional part and the most significant 4 bits for the fixed part). A floating point standard compliant LTE simulation Matlab-model for Downlink Physical Layer has been used as a reference to verify the results and performance of the system. Results matched the floating point simulation. The simulation parameters shown in the Table 5-1 have been used and the results have been verified:

Table 5-1

Channel	AWGN Block Fading
Channel Type	Typical Urban (TU)
SNR	15 dB
Number of Faders	15
Number of Users	1
Bandwidth	20 MHz
Transmission Mode	Open Loop Spatial Multiplexing
Introduced CFO	3.1457
FCFO	0.1457
ICFO	3.0000
RCFO	0
Data Format	12 bit fixed point

5.3. TTA Processor for Carrier Recovery

The TTA processor used in the carrier recovery algorithm is shown in Fig. 5.2. The architecture is not fully connected but still the compiler has a lot of scheduling freedom because of many connections still present in the interconnect network. There is a significant amount of power consumption in this architecture that can be saved by removing those connections which are not utilized at all or the removal of which does not affect the performance of the application beyond a certain acceptable limit to meet the real time constraints. Unnecessary capacitance is removed by the removal of those interconnections from the bus and decreasing the size of the instruction word. The TCE provides a Connection Sweeping Tool which removes the unneceassry connections and gives different cycle count for different sets of TTA connections for the designed TTA. The most optimum TTA is then selected from this set. Fig. 5.1 shows different sets of con-

nections for the same TTA and the cycle count that is required for performing RCFO only. A considerable amount of power is saved by implementing these measures.

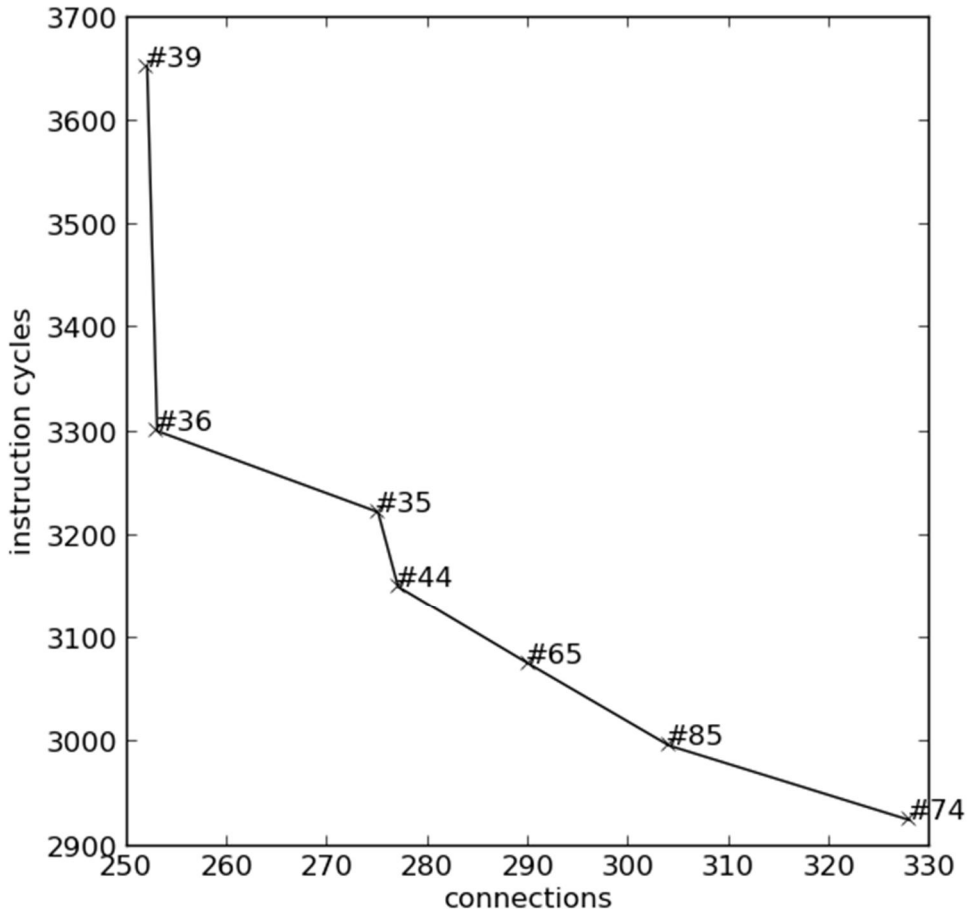

Figure 5.1. Bus connections and cycle count for performing RCFO

SFUs can also be added to the architecture for certain portions of the application program to improve the performance of the system. Addition of SFUs can help in reduction of the number of instructions in the instruction memory and thus reducing the number of fetches. SFUs can also help to accelerate the program as they realize the software portion of the code as a special hardware unit. The control mechanism of the TTA makes automated clock gating of the FUs possible, and this property is used to make SFUs inactive when not used. It further reduces the power consumption.

5.4. TTA with Custom Functional Units

The architecture for carrier recovery TTA is shown in Figure. 5.2. It consists of a total of nine functional units and three register files of 16x32 bit registers each, and interconnection network consisting of 10 buses. The instruction memory, dual-port data

memory and control unit are not displayed in Fig. 5.2. Special functional units include Complex multiplier, Cordic for calculating sine/cosine value and a functional unit for FFT which is in fact a specialized TTA for calculating 2K point FFT. In this context the architecture can be regarded as a multi-TTA design.

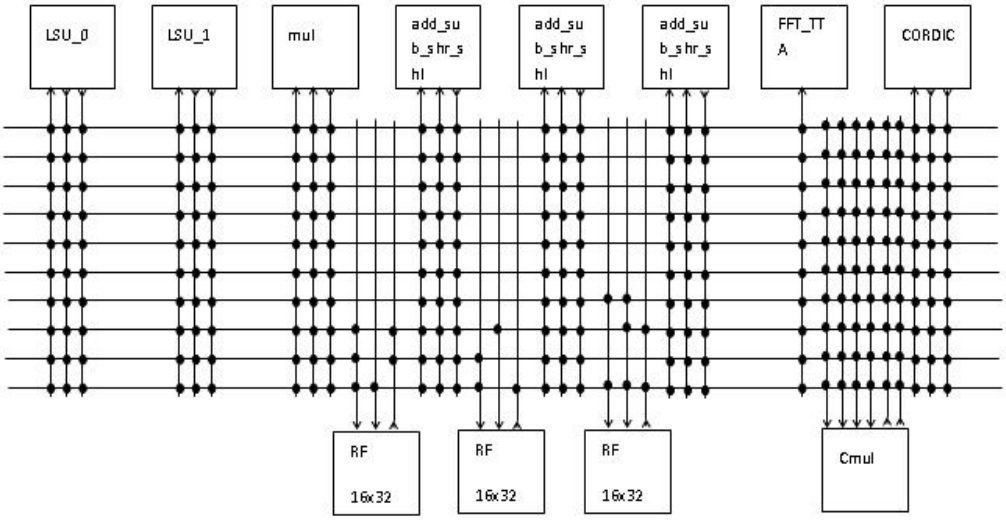

Figure 5.2. Starting point Fully Connected Carrier Recovery TTA

5.5. CORDIC Functional Unit

Sine and cosine values need to be computed in certain parts of the carrier offset compensation algorithm. CORDIC algorithm was chosen for this task. It is similar to shift-and-add algorithms. It is a simple and efficient algorithm for calculating the sines and cosines of a value using basic arithmetic operations like addition, subtraction, comparisons and shifts. A precomputed lookup table with as many entries as the desired accuracy in bits is required. The acronym CORDIC stands for Coordinate Rotational Digital Computer. The cordic algorithm is not accurate, but is simple to implement with limited hardware without multipliers and small storage space. [28]

Since it has been called many times in the algorithm a special function unit has been dedicated for this operation. The algorithm calculates the values in an iterative process, fifteen iterations to calculate an output. Since the CORDIC core is implemented using a pipelined approach, all the iterations are performed in parallel. The valid range of CORDIC algorithm is between $\pi/2$ to $-\pi/2$, so some preprocessing of the input is needed to bring the values in the valid range. To calculate the sine and cosine 'x' is some constant value, 'y' is zero and 'z' is fed with the angle. After the calculations are done the outputs must be placed in the correct quadrants by the post processor. There is input port for reading the operand (angle) and ports for the outputs (values of sine and cosine). The addition of special functional unit for CORDIC calculations has a huge im-

pact on the cycle count as without this functional unit the cycles taken to complete the task are 3518937 which is almost 11 times worse than the cycle count measured when the architecture has FU for CORDIC.

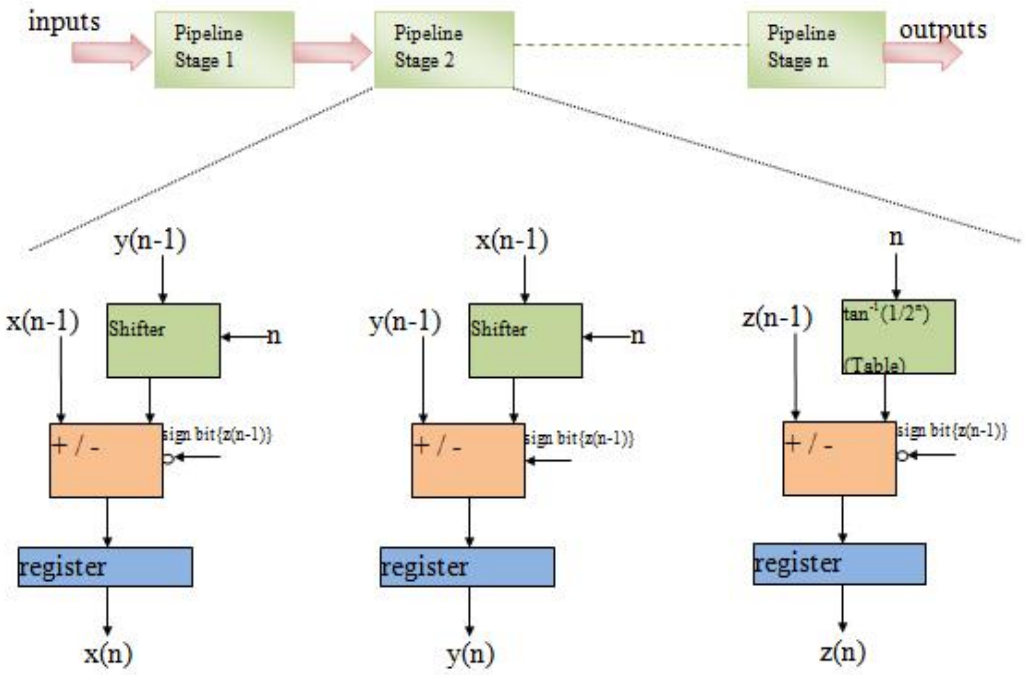

Figure 5.3. Cordic FU

5.6. Complex Multiplier Functional Unit

The algorithm operates on complex values and there are considerable amount of complex multiplications. It would be better to have a separate functional unit dedicated to complex multiplications. The operands of complex multiplier are 16 bits each. The latency of complex multiplier is two clock cycles. Its data path is shown in Figure. 5.4. In each cycle an output can be read because of the pipelined implementation but this complex multiplication operation depends on the scheduling done by the compiler. There are four input ports for reading the real and imaginary parts of the operands and two output ports for writing the real and imaginary parts of the result.

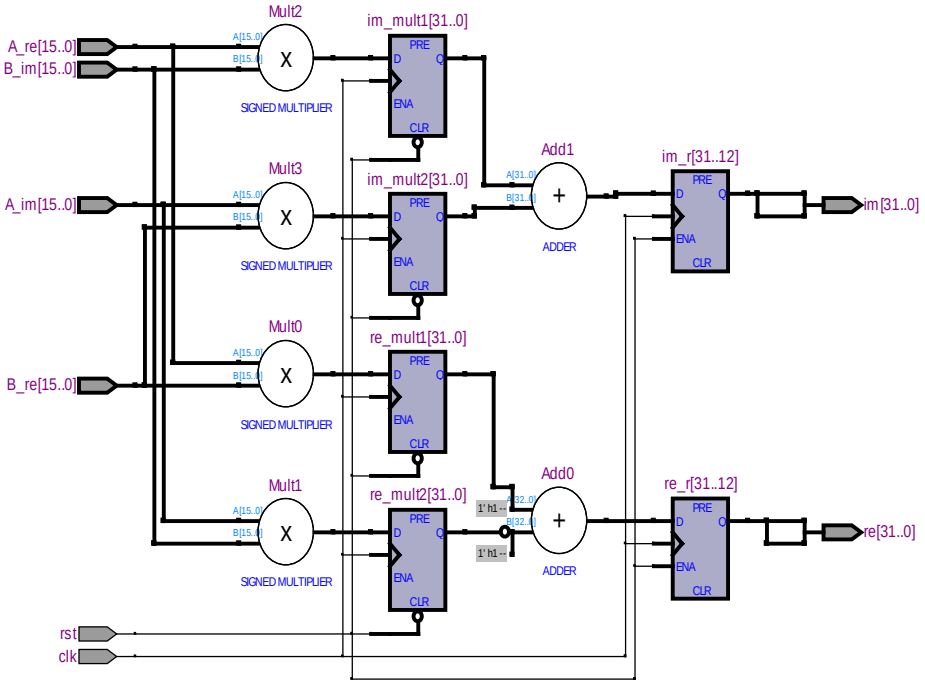

Figure 5.4. Complex Multiplier Architecture

5.7. FFT Functional Unit

The estimators for fractional and residual carrier frequency offsets operate on data in frequency domain so it is important to transform the time domain received data to frequency domain. For this reason 2K point FFT needs to be taken overall six times in the algorithm. FFT itself is computationally very intensive algorithm. To meet the real time constraints and low power requirements a TTA designed for FFT as shown in Figure. 5.5 has been used in the architecture as a special functional unit.

The algorithm followed to calculate the FFT is Cooley and Tukey [29] because of its reduced computational complexity as compared to calculating DFT in a straightforward way using the eq. 5.1.

$$X_m = \sum_{n=0}^{N-1} x_n e^{-j2nm\pi/N} \quad , \qquad (5.1)$$

Cooley and Tukey algorithm decomposes the whole DFT into smaller DFTs and reduces the complexity from N^2 to *Nlog(N)*. *N* is the number of points of FFT and it is equal to R^r, where *R* is the radix of FFT algorithm with usually a value power of two. The most common used FFT is *radix-2* in which *N* must be a power of two. In this unit mixed radix 4/2 is used, to obtain performance boost from radix4 and still maintaining the support of FFT sizes of power of two, supported sizes are from 32 to 2048.

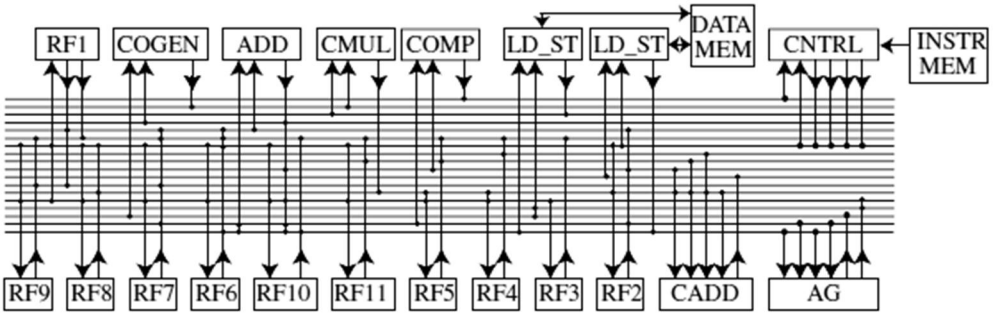

Figure 5.5. FFT TTA [37]

Since the FFT unit requires large memory bandwidth, i.e. two memory accesses required in every cycle the unit is operating, a local memory is allocated for the FFT unit. With aid of the local memory the special unit can be completely parallel to the rest of the system. In order to reduce overhead caused by data transport to and from the FFT FU, two memory arrays are used for the unit. By using two arrays result can be read and new data can be written to the unit, while the previous operation is still calculating. The unit is controlled by simple control flow; host processor will trigger the operation by defining which memory array the unit is ought to use and the host can read the results after the computations are complete, for which the latency is fixed.

5.8. Evaluation

It took **331733** cycles on a fully connected **TTA** for implementing Carrier recovery for LTE with 20 MHz Bandwidth. The architecture must be able to run around **335 MHz** which could be achieved on a **65 nm** ASIC processor in order to meet the real time constraint of 1 ms. Operating frequency of around **84 MHz** was achieved on **Stratix-II** FPGA for the fully connected design. Some of the connections were removed from some of the RFs in order to increase the frequency. This resulted in an increase in frequency to around **120 MHz** on Stratix-II without any significant change in the cycle count. But there are still many connections that can be removed which may result in further increase in the frequency. According to a rough estimate the real time constraint might be met if implementation is done on an ASIC.

As a consequence the architecture was then synthesized on **130 nm** and **1.5 V ASIC** technology at **200MHz** with a power consumption of **46.38mW**. The total energy consumption for task completion is **0.0968mJ**.

As mentioned earlier the system has data in complex format as input, 16 bits for the real part and 16 bits for the imaginary part, stored in separate memory locations. Both the real and imaginary parts can be stored on the same 32 bit memory location on upper and lower 16 bits. In this way data can be read/written at the same time, thus saving a lot of read and write operations.

Packing the real and imaginary part of data into the same memory location should not make much difference in terms of throughput and power consumption at the logic level. Because the same address goes to both real and imaginary part of data and data can be read or written by using two halves of 32 bit data bus. But this packed memory will fit into a slightly smaller die area due to less clearance around the memory module(s) and will require a single address decoder in comparison to two decoders in case real and imaginary parts are stored in separate memory locations.

The carrier recovery program was then run on a different architecture in order to evaluate the impact of SFUs. This architecture was fully connected containing four real multipliers, three register files of 16x32 bit registers each, interconnection network consisting of 10 buses, instruction memory, dual-port data memory and a control unit. The difference between this architecture and the one shown in Figure. 5.1 is that it **does not have special functional units** for CORDIC, FFT and Complex Multiplication. The effect on throughput of the system was very evident in this case as it took **4451425** cycles to complete the task which is almost **13.42** times slower than the architecture using the special functional units.

Carrier recovery program was then run on **COFFEE RISC** architecture [31] in order to compare its performance with a TTA without functional units. COFFEE RISC took **6377378** to complete the task. COFFEE needs a very high clock frequency to meet the real time constraints.

The same task was also performed on **TI's** famous fixed point VLIW **DSP TMS320C6416**. It took around **3928880** cycles to complete. A single TMS320C6416 core fabricated on **130nm** technology must consume at least **200mW** at **500MHz** and **1.2V**. This single DSP core is not enough to meet the real time constraints.

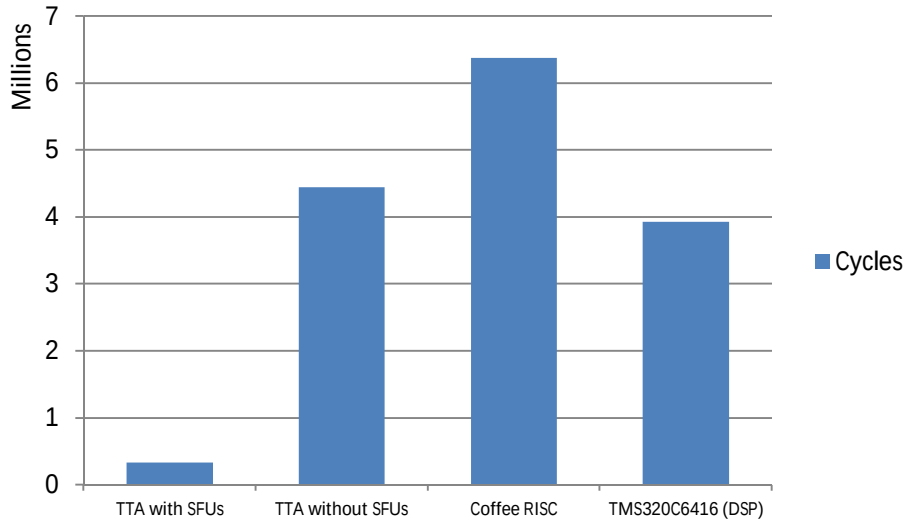

Figure 5.5. Cycle Count for different Architectures to complete the task

6. SUMMARY AND CONCLUSION

There is still room for further improvement as the code has been written in C but if it is written in assembly language program has full control over scheduling the instructions and hence improvements can be made in terms of cycle count and power consumption but writing assembly code needs much more effort and time. The other approach to increase the performance of architecture is to increase the available resources and exploit them using OpenCL but the support of Opencl in TCE is still under development. Further improvements can be made by using an MPSoC containing multiple TTAs and distributing the carrier recovery application code on the TTA cores inside the MPSoC, by compromising chip area and power consumption.

6.1. Summary

LTE carrier recovery for 20 MHz system for the downlink has been studied and implemented in this work. The carrier recovery algorithm has been coded in C language. A single TTA core has been synthesized for this purpose. TCE toolset designed at TUT has been used for this purpose. TCE provides a very good framework giving control to the designer in different aspects of design space exploration with its GUI and command prompt interface thus making it possible to develop TTAs for general or specific application needs. Specialized functional units (SFUs) were designed during the TTA design process for computationally intensive and recurring operations like CORDIC for calculating sine and cosine values, CMUL for complex multiplication, and FFT_TTA for FFT computation. Special function units have been discussed briefly and it has been shown that how these SFUs affect the overall performance of the system. A short comparison of TTA and other architectures e.g. RISC and DSP has been made which shows much better performance of TTA as compared to other architectures.

A **fully connected TTA** took **331733** cycles to Carrier recovery for LTE with 20 MHz Bandwidth on **65 nm** technology running around **335 MHz**. Operating frequency was **84 MHz** on **Stratix-II** for the fully connected design, **200 MHz** on **1.5 V ASIC 130 nm** technology. The Carrier Recovery algorithm execution took **4451425** cycles on architecture which **does not have special functional units** for **CORDIC** and **Complex Multiplication,** which is almost 13 times slower than a fully connected TTA. **COFFEE RISC** core took **6377378** to complete the task. **TI's** famous fixed point **DSP TMS320C6416** core at **500MHz** fabricated on **130nm** technology consuming **200mW** and **1.2V** took around **3928880** cycles to complete the task.

Architectures compared with TTA have to run either at a very high clock frequency and higher power. There must be some multi-core solution available which is able to run at comparatively lower frequency and consumes lower power, in order to meet the real time constraints.

REFERENCES

[1]. Theodore S. Rappaport, Wireless Communications: Principles and Practice.

[2]. Ajay R. Mishra, Advanced Cellular Network Planning and Optimiztion: 2G/2.5G/3G…Evolution to 4G.

[3]. Copyright IMEC.

[4]. Stefania Sesia, Issam Toufik, and Mathew Baker, LTE, The UMTS Long Term Evolution, FROM THEORY TO PRACTICE.

[5]. Ajay R. Mishra, Cellular Technology in Emerging Markets: 2G, 3G and beyond, Nokia Siemens Networks.

[6]. Jim Zyren, Dr. Wes McCoy, Overview of the 3GPP Long Term Evolution Physical Layer: White Paper.

[7]. Erik Dahlman, Stefan Parkvall, and John Sköld, 4G LTE/LTE-Advanced for Mobile Broadband, [8]. Dake Liu, Coresonic AB and Linköping University, Anders Nilsson and Eric Tell, Coresonic AB, Di Wu and Johan Eilert, Linköping University, Bridging Dream and Reality:Programmable Baseband Processors for Software-Defined Radio.

[9]. John Glossner, Tanuj Raja, Erdem Hoknek, and Mayan Moudgill, A Multi-threaded Processor Architecture for SDR, Sandbridge Technologies, Inc.

[10]. Yuan Lin, Hyunseok Lee, Mark Woh, Yoav Harel, Scott Mahlke, Trevor Mudge, Chaitali Chakrabarti, and Krisztián Flautner, SODA: A High-Performance DSP Architecture for Software-Defined Radio, IEEE Micro (Micro's Top Picks in Computer Architecture for 2006), Vol. 27, No. 1, Jan./Feb. 2007, pp. 114-123.

[11]. Omer Anjum, Tapani Ahonen, Fabio Garzia, Jari Nurmi, Claudio Brunelli and Heikki Berg, State of the art baseband DSP platforms for Software Defined Radio: A survey, 6 Jun, 2011.

[12]. LTE Resource Guide, Anritsu

[13]. John B. Anderson, Digital Transmission Engineering, Second Edition.

[14]. John G. Proakis, Digital Communications, Fourth Edition.

[15]. B.P. Lathi, Modern Digital and Analog Communication Systems, Third Edition.

[16]. Qi Wang, Christian Mehlf¨uhrer, and Markus Rupp, Carrier Frequency Synchronization in the Downlink of 3GPP LTE, The 21st Annual IEEE International Symposium on Personal, Indoor and Mobile Radio Communications (PIMRC'10)

[17]. C. R. N. Athaudage and K. Sathananthan, Cramer-Rao Lower Bound on Frequnecy Offset Estimation Error in OFDM Ssytems With Timing Error Feedback Compensation, 2005

[18]. Paul H. Moose, A Technique for Orthogonal Frequency Division Multiplexing Frequency Offset Correction, IEEE Transactions on Communication. Vol. 42, 10, October 1994

[19]. D. Toumpakaris, J. Lee, and H. Lou, Estimation of integer carrier frequency offset in OFDM systems based on the maximum likelihood principle, IEEE transactions on broadcasting, vol. 55, March 2009

[20]. Prof. Jari Nurmi, Springer, Processor Design, System on Chip Computing for ASICs and FPGAs, 2007.

[21]. Henk Corporaal, Microprocessor Architectures from VLIW to TTA, Delft University of Technology, The Netherlands, John Wiley, 1998.

[22]. Henk Corporaal, Jan Hoogerbrugge, Kluwer, Code generation for transport triggered architecture, 1995.

[23]. TTAs: Missing the ILP complexity wall, Henk Corporal, Delft University of Technology, The Netherlands.

[24]. V. Guzma, P. Jääskeläinen, P. Kellomäki, and J. Takala, Impact of Software By-passing on instruction level parallelism and register and register file traffic. In Proc. Int. Workshop Embedded Computer Syst.: Architectures, Modeling and Simulation, pages 23-32, 2008.

[25]. Otto Esko, Mar 2010, ASIP Integration and Verification Flow for FPGA, Master's Thesis, Tampere University of Technology. 15. 17. 19 p.

[26]. TTA-based Co-design Environment v 1.6 User Manual, Tampere University of Technology, Department of computer systems, Version 28, 18 May, 2012.

[27]. J. Mäntyneva, Automated Design Space Exploration of Transport Triggered Architectures. Master's thesis, Tampere University of Technology, Finland, Apr. 2007.

[28]. Jörg Arndt, Matters Computational, Ideas, Algorithms, Source Code, Nurnberg Germany, June 2010.

[29]. Cooley, James W., and John W. Tukey, "An algorithm for the machine calculation of complex Fourier series," Math. Comput. 19, 297–301 (1965).

[30]. Kylliäinen J, Nurmi J, Kuulusa M (2003) COFFEE – A Core for Free. In Proc. International Symposium on SoC, pp 17–22.

[31]. http://focus.ti.com/lit/an/spra811c/spra811c.pdf

[32]. Louis Litwin and Michael Pugel, The Principles of OFDM, January 2001.

[33]. Evolved Universal Terrestrial Radio Access (E-UTRA) and Evolved Universal Terrestrial Radio Access Network (E-UTRAN), Physical Channels and Modulation, Release 8, 3GPP TS 36.211, 10, 52, 53 p.

[34]. Christina Geβer, Long Term Evolution, A concise introduction to LTE and its measurement requirements, Rohde & Schwarz, Ist Edition 2011.

[35]. Tzi-Dar Chiueh and Pei-Yun Tsai, OFDM Baseband Receiver Design for Wireless Communications, John Wiley & Sons (Asia) Pte Ltd, Dec 4 2007.

[36]. Christina Gessner and Andreas Roessler, LTE technology and LTE test, a deskside chat, Rohde & Schwarz, April 2009.

[37]. Teemu Pitkänen, Risto Mäkinen, Jari Heikkinen, Tero Patanen, and Jarmo Takala, Low-Power, High Performance TTA Processor for 1024-Point Fast Fourier Transform, Volume LNCS 4599, pp. 233–240. Springer-Verlag, Berlin, Germany, 2007, 3 p.