FACULTY OF INFORMATION TECHNOLOGY AND ELECTRICAL ENGINEERING
DEGREE PROGRAMME IN ELECTRONICS AND COMMUNICATIONS ENGINEERING

MASTER'S THESIS

EFFICIENT IMPLEMENTATION OF CHANNEL ESTIMATION ALGORITHM FOR BEAMFORMING

Author	Arttu Afflekt
Supervisor	Prof. Olli Silvén
Second Examiner	D.Sc. (Tech.) Tuomo Hänninen
Technical Advisor	D.Sc. (Tech.) Jussi Salmi

May 2020

Afflekt A. (2020) Efficient Implementation of Channel Estimation Algorithm for Beamforming. University of Oulu, Faculty of Information Technology and Electrical Engineering, Degree Programme in Electronics and Communications Engineering. Master's Thesis, 66 p.

ABSTRACT

The future 5G mobile network technology is expected to offer significantly better performance than its predecessors. Improved data rates in conjunction with low latency is believed to enable technological revolutions such as self-driving cars. To achieve faster data rates, MIMO systems can be utilized. These systems enable the use of spatial filtering technique known as beamforming. Beamforming that is based on the pre-acquired channel matrix is computationally very demanding causing challenges in achieving low latency. By acquiring the channel matrix as efficiently as possible, we can facilitate this challenge.

In this thesis we examined the implementation of channel estimation algorithm for beamforming with a digital signal processor specialized in vector computation. We present implementations for different antenna configurations based on three different approaches. The results show that the best performance is achieved by applying the algorithm according to the limitations given by the system and the processor architecture. Although the exploitation of the parallel architecture was proved to be challenging, the implementation of the algorithm would have benefitted from the greater amount of parallelism. The current parallel resources will be a challenge especially in the future as the size of antenna configurations is expected to grow.

Key words: telecommunications, 5G, MIMO, parallel architecture.

Afflekt A. (2020) Keilanmuodostuksen tarvitseman kanavaestimointialgoritmin tehokas toteutus. Oulun yliopisto, tieto- ja sähkötekniikan tiedekunta, elektroniikan ja tietoliikennetekniikan tutkinto-ohjelma. Diplomityö, 66 p.

TIIVISTELMÄ

Tulevan viidennen sukupolven mobiiliverkkoteknologian odotetaan tarjoavan merkittävästi edeltäjäänsä parempaa suorituskykyä. Tämän suorituskyvyn tarjoamat suuret datanopeudet yhdistettynä pieneen latenssiin uskotaan mahdollistavan esimerkiksi itsestään ajavat autot. Suurempien datanopeuksien saavuttamiseksi voidaan hyödyntää monitiekanavassa käytettävää MIMO-systeemiä, joka mahdollistaa keilanmuodostuksena tunnetun spatiaalisen suodatusmenetelmän käytön. Etukäteen hankittuun kanavatilatietoon perustuva keilanmuodostus on laskennallisesti erittäin kallista. Tämä aiheuttaa haasteita verkon pienen latenssivaatimuksen saavuttamisessa.

Tässä työssä tutkittiin keilanmuodostukselle tarkoitetun kanavaestimointialgoritmin tehokasta toteutusta hyödyntäen vektorilaskentaan erikoistunutta prosessoriarkkitehtuuria. Työssä esitellään kolmea eri lähestymistapaa hyödyntävät toteutukset eri kokoisille antennikonfiguraatioille. Tuloksista nähdään, että paras suorituskyky saavutetaan sovittamalla algoritmi järjestelmän ja arkkitehtuurin asettamien rajoitusten mukaisesti. Vaikka rinnakkaisarkkitehtuurin hyödyntäminen asetti omat haasteensa, olisi algoritmin toteutus hyötynyt suuremmasta rinnakkaisuuden määrästä. Nykyinen rinnakkaisuuden määrä tulee olemaan haaste erityisesti tulevaisuudessa, sillä antennikonfiguraatioiden koon odotetaan kasvavan.

Avainsanat: tietoliikennetekniikka, 5G, MIMO, rinnakkaisarkkitehtuuri.

TABLE OF CONTENTS

FOREWORD

First, I want to thank my supervisor Professor Olli Silvén for his continuous guidance and support through the writing process. I felt privileged to receive such detailed guidance. I also want to thank Nokia and my Technical Advisor Jussi Salmi for the great opportunity to work as a thesis worker as a part of his group. Your feedback has encouraged me and without your expertise, defining the final scope of this thesis all by myself would have been much more challenging. I would like to express my gratitude to the second examiner Tuomo Hänninen. I want to thank my senior colleagues. Their comments and insight have made this journey much more enjoyable.

I would like to extend particular thanks to my mother. I would like to thank you for providing me the opportunity to study so far. Without your valuable contribution I would have never reached this goal. You have always been there for me, even in difficult times.

My final and warmest thanks belong to my girlfriend Reetta for always believing in me. Your support and encouragement have been immeasurably valuable. The importance of the immense love you have given is beyond words. Thank you.

Helsinki, May 25, 2020

Arttu Afflekt

LIST OF ABBREVIATIONS AND SYMBOLS

ASIC	application specific integrated circuit
ASIP	application specific instruction set processor
AWGN	additive white gaussian noise
BPSK	binary phase shift keying
BS	base station
CAZAC	constant amplitude zero auto correlation
CPI	cycles per instruction
CPU	central processing unit
CGRA	coarse grained reconfigurable array
DFT	discrete Fourier transform
DLP	data level parallelism
DMA	direct memory access
DM-RS	demodulation reference signal
DOP	degree of parallelism
DPC	dirty paper coding
DSP	digital signal processor
EDGE	enhanced data rates for GSM evolution
FIR	finite impulse response
FPGA	field programmable gate array
GSM	global system for mobile communications
IDE	integrated development environment
IEEE	institute of electrical and electronics engineering
ILP	instruction level parallelism
ISI	inter symbol interference
LOS	line of sight
LTE	long term evolution
LTE-A	long term evolution advanced
MAC	multiply and accumulate
MIMO	multiple in multiple out
MISO	multiple in single out
MMSE	minimum mean square error
MU-MIMO	multi-user multiple in multiple out
SU-MIMO	single-user multiple in multiple out
NR	new radio
NRE	non-recurring engineering
OFDM	orthogonal frequency division multiplexing
PRB	physical resource block
PRACH	physical random access channel
PUCCH	physical uplink control channel
PUSCH	physical uplink shared channel
RX	receiving
SIMO	single in multiple out
SISO	single in single out
SIMD	single instruction multiple data
SISD	single instruction single data
SNR	signal-to-noise ratio

SRS	sounding reference signal
SVD	singular value decomposition
TDD	time division duplex
TTI	transmission time interval
TTA	transport triggered architecture
TX	transmitting
UE	user equipment
VLIW	very long instruction word
WCDMA	wideband code division multiple access
QAM	quadrature amplitude modulation
ZF	zero forcing

A_t	area of the antenna aperture
$\alpha_n(t)$	amplitude of the nth multipath component
B	bandwidth
B_c	coherence bandwidth
c	speed of light
C	channel capacity
$c(t)$	channel impulse response
$\mathbb{C}$	complex plane
δ	Dirac delta function
f	frequency
G_r	receive antenna gain
G_t	transmit antenna gain
γ_Σ	output SNR
$\mathbf{H}$	channel matrix
$\widehat{\mathbf{H}}_{LS}$	channel LS estimate
$\widehat{\mathbf{H}}_{LMMSE}$	channel LMMSE estimate
$\mathbf{I}$	Identity matrix
λ	wavelength
λ_i	ith eigenvalue
m	number of executed instructions
m_u	cyclic shift parameter
n	number of stages
M_r	number of receive antennas
M_t	number of transmit antennas
N_{tr}	length of the training sequence
$\mathbf{n}$	noise vector
N	noise
P	signal power
$\phi_n(t)$	Doppler phase shift of the nth multipath component
P_i	transmit power of ith stream
P_r	receive power
P_t	transmit power
R	distance between receiver and transmitter
$\mathbf{R}_H$	channel correlation matrix
$r(t)$	received signal
σ_i	ith singular value of Σ

σ^2	noise variance
$\mathbf{\Sigma}$	diagonal matrix from SVD
t	time to execute one stage
T_D	time delay
$T(n)$	throughput
$\tau_n(t)$	delay of the nth multipath component
$\mathbf{u}$	beamforming vector
$\mathbf{U}$	unitary matrix from SVD
$u(t)$	equivalent low-pass representation
$\mathbf{v}$	beamforming vector
$\mathbf{V}$	unitary matrix from SVD
$\mathbf{W}_Z$	zero forcing precoding matrix
$\mathbf{x}$	transmitted signal vector
$\mathbf{X}$	transmitted signal matrix
x_{LS}	general least squares solution
X_k	Zadoff-Chu base sequence
$X_{n.k}$	transmitted symbol at kth subcarrier of nth symbol in frequency domain
$\mathbf{y}$	received signal vector
$\mathbf{Y}$	received signal matrix
$Y_{n,k}$	received symbol at kth subcarrier of nth symbol in frequency domain

1 INTRODUCTION

The evolution of mobile network generations has been steady from the introduction of 1G in the early 1980s until today. A new mobile network generation has been released every ten years and the fifth-generation mobile network should be deployed during this decade.

The increase in data rates as measured in bits per second has been exponential within successive generations. The growth of the data rates during the evolution of mobile networks is approximated in Figure 1 [1]. In the early days of the 2G network it was possible to achieve a data transfer rate approximately 15kbps using GSM technology. The introduction of EDGE technology made it possible to reach data rates up to 600kbps. Therefore, data rates experienced a forty-fold increase during a single mobile network generation.

3G network continued this trend by increasing the data rates up to 2Mbps with a new technology called WCDMA. Having the latest enhancements for the 3G network, it was possible to reach data rates around 40Mbps. Once again, tremendous increase in data rates was experienced within one generation. This trend continued with the 4G network whose latest enhancements increased data rates to gigabits per second [2]. The upcoming 5G network is not going to end this trend [1].

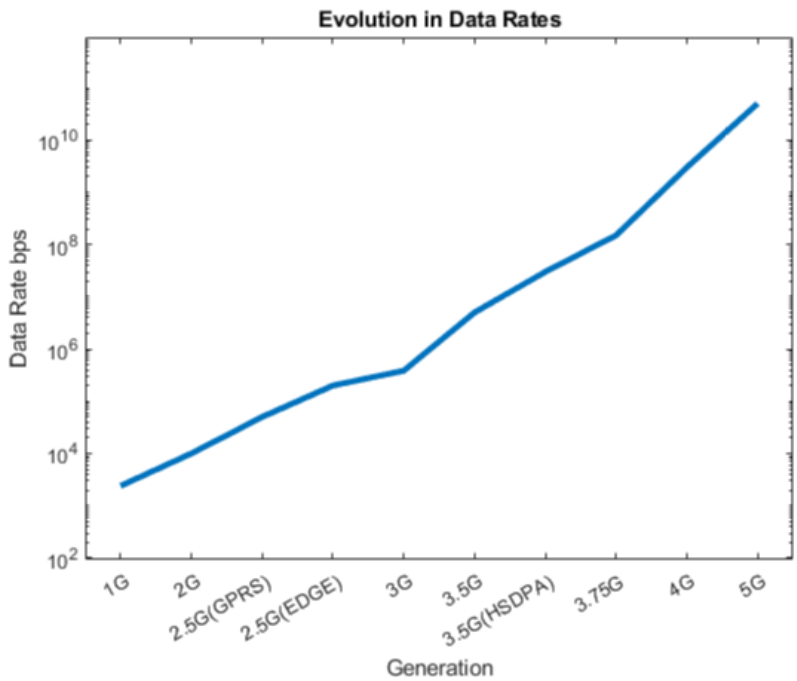

Figure 1. Approximate growth of data rates in logarithmic scale.

This massive growth in data rates has not been achieved just by increasing the transmission bandwidths. The latest mobile network generations (LTE and 5G), utilize advanced signal processing techniques to increase the spectral efficiency in conjunction with multiple input multiple output (MIMO) technology.

In theory, these techniques can increase data rates. However, utilizing these methods requires a great deal of heavy computation. Combining computationally demanding techniques with very low latency requirement is challenging. In order to meet the latency requirements, these

methods must be implemented with an architecture specialized in signal processing. Furthermore, one needs to implement these tasks of significance as efficiently as possible in order to achieve the expected latency [3].

One of these methods is known as beamforming. Beamforming is based on a channel matrix that contains the channel state information for each antenna pair. Channel state information can be obtained by estimating the channel based on the known pilot symbols. These estimates are used later in the calculation of the beamforming coefficients. Since this process has to be performed beforehand, it has a great impact on the total system load caused by the beamforming.

Channel estimation can be performed in countless ways. By favouring a channel estimation scheme with low complexity and adequate accuracy, excess signal processing workload could be reduced. Current antenna configurations consist of numerous antenna elements which makes the channel estimation demanding. The size of the antenna configuration directly determines the size of the channel matrix. Because each element of the channel matrix must be estimated, large antenna configurations poses challenges to channel estimation. Although processing of very large matrices is cumbersome it can be facilitated by using a parallel architecture. To ensure good performance, one should evaluate the feasibility of used parallel architecture on a regular basis due to rapidly growing number of antenna elements. In this thesis, we review couple of beamforming techniques and discuss the topic of channel estimation in wireless communications. The discussion emphasis is in efficient implementation of the channel estimation for beamforming.

This thesis is organized as follows. In Chapter 2 an overview of 5G is provided by introducing the preliminary requirements and usage scenarios specified for 5G. In addition, a brief introduction to multipath propagation is provided. At the beginning of Chapter 3, the multipath channel is linked to the multiantenna systems by introducing conventional methods to increase the channel capacity via spatial diversity schemes. This is followed by discussion of beamforming techniques and spatial multiplexing. In Chapter 4, we discuss channel estimation in single and multi-antenna systems. The architecture exploration is provided in Chapter 5. In Chapter 6, we go through the implementations and present the results. Then, in Chapter 7, we discuss our findings and speculate on the future of the subject. Finally, in Chapter 8 we provide a brief summary of this thesis.

2 MOBILE COMMUNICATION EVOLUTION

In this chapter the requirements and expectations towards the upcoming fifth mobile network generation are introduced. According to the specifications set by 3GPP, 5G will continue the trend of remarkable data rate growth seen between the previous successive generations [4]. In addition to significantly higher data rates, the network is expected to operate at ultra-low latency, even in challenging conditions such as when providing a solid and reliable experience for high mobility users. Initially, these visions and usage scenarios are reviewed. Since 5G is targeted to operate in much higher frequencies than the previous mobile network generations, we take a glance at the feasibility of the millimetre waves.

It is shown that the higher frequencies targeted for 5G offer significantly more bandwidth than ever before. Higher frequencies allow greater number of users to enter the network than previously. In crowded areas, current frequency resources are so heavily utilized that they cannot provide necessary capacity anymore so new frequency resources are required. Finally, to provide some preliminary knowledge for the future chapters, we take a look at the properties of the multipath channel.

2.1 Visions and Requirements

Continuous increase in the usage of the mobile networks has led to the development of a new mobile network generation. This mobile network generation is the fifth of its kind, hence the name 5G. 5G is also known as new radio, commonly abbreviated as NR. The standardization work of 5G is carried out by an organization called 3GPP. 3GPP is a collaboration of seven independent telecommunication standardization organizations [5]. One could view the role of 3GPP in 5G development as an author who maintains and develops the global rules and standards required for the future network. In the 3GPP specification release 15 the requirements for the 5G usage scenarios are the following [6][4]:

- **Enhanced Mobile Broadband (eMBB)** The requirement for eMBB is to provide better data rates for larger user density and mobility than 4G/LTE. Specification for the user densities and the data rates in both uplink and downlink are provided for different types of scenarios. One type use case is broadband access in a crowd. A rock concert could be a real-life example of this scenario. A specification is also provided for high speed vehicles and even for connectivity in airplanes.
- **Critical Communications (CC) and the Ultra Reliable and Low Latency Communications (URLLC)** This scenario corresponds to the situation where very low latency with very high service availability is required. From the radio access point of view the total latency can be reduced by improving the data processing in the radio and the baseband parts. Real scenarios could be for instance a process automation or a remote control such as self-driving cars.
- **Massive Internet of Things (mIoT)** Scenarios require support for very high density of devices.
- **Flexible Network Operation** Allows network to be configured in a more flexible way. This covers for example a network slicing, which gives the operator an ability to customize the network appropriate to the situation.

International Telecommunication Unions sector for radiocommunication (ITU-R) summarizes similar type of usage scenarios according to their importance in their document "IMT Vision – Framework and overall objectives of the future development of IMT for 2020 and beyond". These features are partially illustrated in Figure 2.1 [7] with numerical specifications for the possible future 5G network. [7]

In Figure 2.1 IMT-2020 and IMT-advanced are the specifications set by ITU for 5G and 4G, respectively. Furthermore, this figure visualizes the requirements for eMBB in compact form. Features such as mobility, connection density, traffic capacity and data rates can be compared to the corresponding values provided by 3GPP in table 7.1-1 in [4]. Although the values introduced in Figure 2.1 are said to be only target values for the research, it should be noted that the numbers given by the 3GPP specification predict very promising enhancements for the data transmission rates and the network capacity. Both features are promised also for high velocity users, e.g. train passengers.

As shown in the listing of 3GPP 5G-scenarios, as well as in Figure 2.1, the latency requirement is heavily emphasized. Reliable low latency network is naturally a mandatory requirement for self-driving cars, as one example. Low latency requirement addresses challenges towards the system design, which will be pointed out in the subsequent chapters.

In the following chapters it is noticed that the higher data rates are not achieved just by increasing the bandwidth. It is shown that a great deal of signal processing is required in order to achieve the expected specifications and to tackle the challenges set by the mobile environment. In order to meet the specifications, new technology must be developed and utilized.

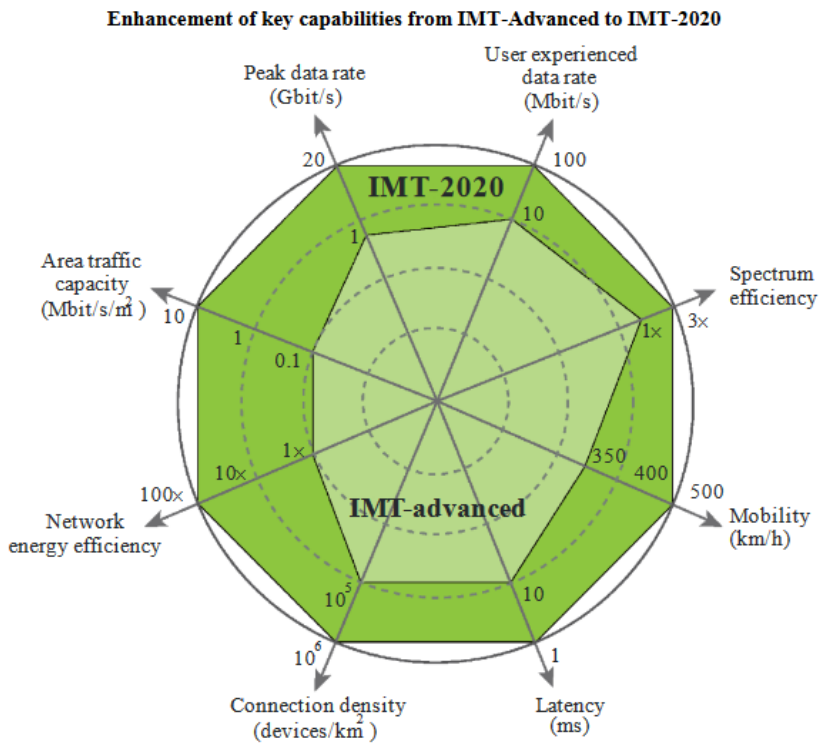

Figure 2.1. Comparison of fourth and fifth mobile network generations. © [2015] ITU

Several papers and publications provide summaries of the key enabling technologies for 5G [8][1][9]. Since new frequencies are considered essential, their feasibility for wireless communication is discussed in the next section.

2.2 Towards New Frequencies

Fundamentally, wireless communication is implemented using electromagnetic waves. These waves are divided into different frequency ranges which together form the electromagnetic spectrum. The spectrum of electromagnetic radiation is shown in Figure 2.2 [10]. Parts from long wave radio towards higher frequencies are traditionally used in wireless communications as shown in Figure 2.2.

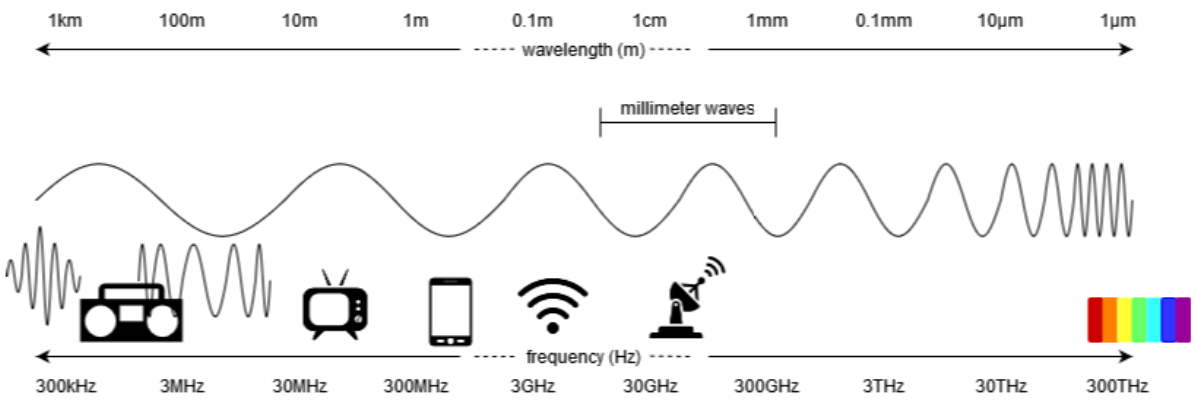

Figure 2.2. Spectrum of electromagnetic radiation.

Useable parts of the spectrum are allocated for the different wireless technologies. Today, the situation is that the lower parts of the frequency range are heavily occupied. The frequencies are from approximately hundreds of MHz up to a few GHz. For example, these frequencies are used by older mobile network generations. When all the frequency resources are utilized, in order to allocate bandwidth for new users, number of frequency resources must be increased. Since lower frequencies are already heavily occupied, new frequencies are found from the higher frequencies.

The new high frequency range envisioned for the use of 5G communications is often referred as millimetre waves. These millimetre waves correspond to the frequency range from 3GHz (some context 6GHz+) up to 300GHz [8]. Using the millimetre wave frequency range offers several evident enhancements for the network. High frequency carrier waves enable allocation of larger bandwidths for the same number of users compared to their lower frequency counter-parts [1].

Above-mentioned property is also easily recognized by observing the famous theorem for channel capacity provided by C.E. Shannon. The theorem for additive white Gaussian (AWGN) channel where channel output corresponds to the input corrupted by noise is given by:

$$C = B \log_2 \left(1 + \frac{P}{N}\right) \tag{2.1}$$

In Equation (2.1), C is the channel capacity in bits/s, B is the bandwidth in Hz, P and N are the signal power and the noise, respectively [11]. Given theorem states that the channel capacity increases linearly with respect to the transmission bandwidth, when signal-to-noise ratio (SNR) is assumed constant. Although we will see slightly different type of expressions for the channel capacity in the subsequent chapters, it should be noted that the same key findings considering

bandwidth and SNR still apply. Actually, the emphasis on the subsequent chapters is to enhance the capacity without increasing the bandwidth.

It is predicted in [8] that in terms of frequency these new wave lengths could increase the capacity by 10GHz or more. All the previous network generations operated only within couple of GHz. Compared to the previous generations, frequency resources envisioned for 5G are tremendous. With this approximation, 5G alone would consume many times more frequency resources than all the previous generations together. However, the question arises are these frequencies feasible for wireless communication in cellular networks?

Free space propagation in line-of-sight (LOS) is often modelled using the Friis transmission equation. Generally adopted idea that the higher frequencies tend to have a lesser free space propagation follows from too narrow inspection of the previously mentioned equation. Friis formula can be written as [12]

$$P_r = P_t + G_t + G_r + 20\log\left(\frac{c}{4\pi R f}\right)\ [dBm]\,, \tag{2.2}$$

where P_t and G_t are the transmit power and the transmit antenna gain, P_r and G_r are the equivalent receive antenna quantities. f, R and c are the frequency, distance between the receiver and the transmitter and the speed of light, respectively.

By observing the rightmost term, it certainly seems that the high frequencies seem to have a greater attenuation compared to the lower ones. In fact, the received power would attenuate according to the formula, if the antenna gains G_t and G_r are assumed to be constant as well as less than or equal to unity [12]. Actually, this property holds for the omnidirectional isotropic antennas whose G terms equal to one. However, when we are dealing with the directional transmit antenna, the gain of the receiving antenna is given by [13]

$$G_t = \frac{4\pi A_t}{\lambda^2}\,, \tag{2.3}$$

where A_t is the area of the antenna aperture and the wavelength is given by λ. Since the wavelength gets shorter as the frequency increases, it seems that the antenna gain is frequency dependent and it increases with the frequency.

The behaviour of both 3GHz and 30GHz signals in LOS was investigated in [12]. The propagation loss for the both frequencies is given as a function of a distance in Figure 2.3 [12]. From the figure we see that the both frequencies behave similarly, and the signals have attenuated by approximately the same amount [12]. According to these results, path loss is not a problem for short range LOS communications.

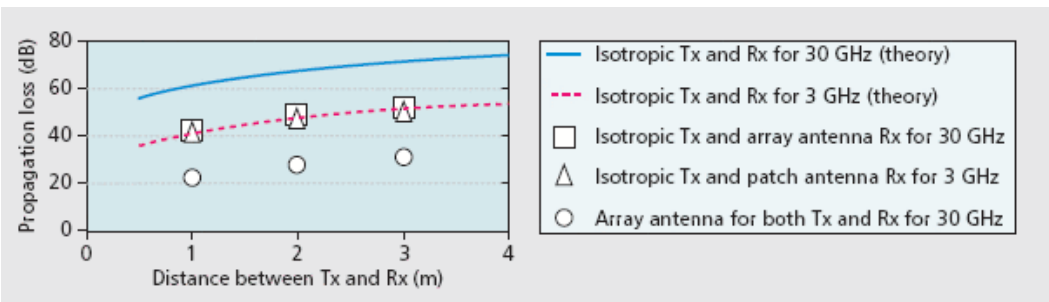

Figure 2.3. Signal attenuation in LOS scenario. © [2014] IEEE

However, it is pointed out in [9] that the millimetre waves handle physical obstacles poorly. If the signal is not under a line of sight, its attenuation doubles per decade. In addition to this, blocking loss of 15-40 dB is also introduced. Similar results are also highlighted in [14] and it is said that these high attenuations from the building materials can completely surpass the signal and prevent its access from outdoor radios to indoor devices. The signal's poor ability to penetrate certain construction materials can be ignored if separate networks are built indoors and outdoors. According to [14], heavy rain significantly attenuates the millimetre waves. The signal can experience a massive attenuation of 1-10dB/km depending on the rate of the rain. This challenge is tackled in [15] by pointing out that the current cell sizes are so small that this does not become a problem. This clearly suggests that rain is a negligible nuisance to dense networks.

Based on this quick review, the new frequencies envisioned for 5G seem to be potential but not problematic. To overcome the challenges related to the millimetre wave propagation many articles have suggested to take advantage of the short wavelengths and to use beamforming with (massive) MIMO antenna elements.

2.3 Multipath Propagation and OFDM

When a signal is transmitted across the wireless channel, it experiences several challenges that makes reliable communication harder. When the signal propagates through the free space, its power tends to attenuate as a function of distance. Adding obstacles such as buildings will introduce the signal to shadowing, which also attenuates the signal. These effects are known as large-scale propagation effects. There are also small-scale effects in the multipath channel, which are discussed in this section. [16]

In multipath environment the received signal consists of one line of sight component and several multipath components, as shown in Figure 2.4. To put it simply, if a single pulse is transmitted, several delayed and distorted copies of this pulse are received.

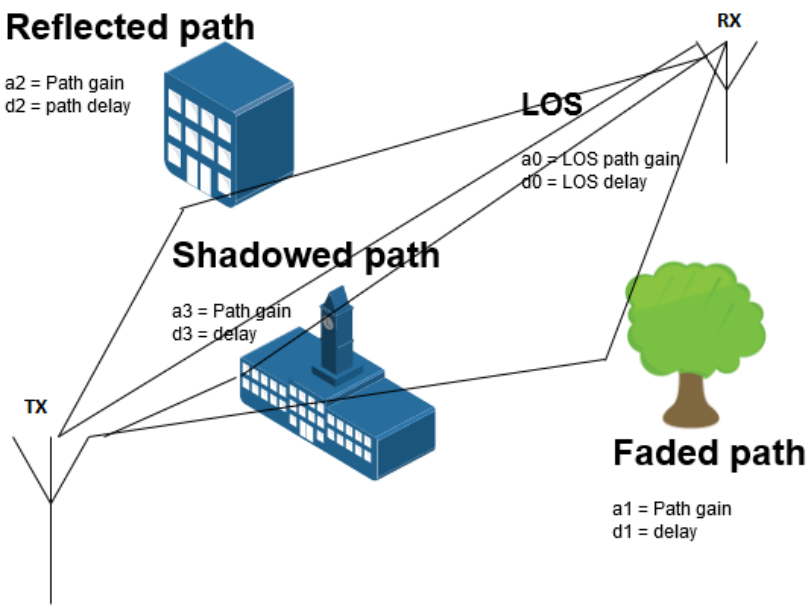

Figure 2.4. Multipath environment.

If the transmitted signal $s(t)$ is presented as $s(t) = Re\{u(t)e^{j2\pi f_c t}\}$, where $u(t)$, f_c and t are the equivalent low-pass representation for the transmitted signal, the frequency of the carrier wave and time, respectively. The received signal can be expressed as

$$r(t) = Re\left\{\sum_{n=0}^{N(t)} \alpha_n(t)\, u\big(t - \tau_n(t)\big) e^{j\left(2\pi f_c\left(t-\tau_n(t)\right)+\phi_{D_n}\right)}\right\}, \tag{2.4}$$

where $N(t)$ gives the number of the multipath components at time t, $n = 0$ corresponds to the LOS component and $\tau_n(t)$, ϕ_{D_n} and $\alpha_n(t)$ are the delay, Doppler phase shift, and the amplitude of the nth multipath component, respectively [16]. Notice that the transmitted signal is not the only time-dependent variable in Equation (2.4). The variables that represent the properties of the multipath channel are time dependent as well. As shown in Figure 2.5 [16], the other party moving, the physical structure of the channel changes, and the channel is described with a new set of parameters.

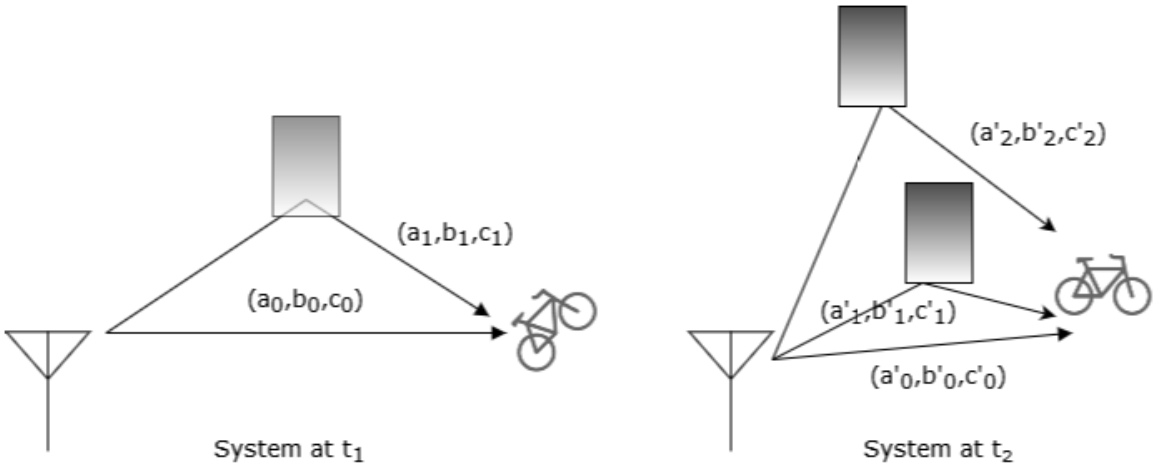

Figure 2.5. Time-varying nature of the multipath channel.

Rearranging Equation (2.4), the received signal $r(t)$ can be written as a convolution with the channel impulse response $c(\tau, t)$ [16]

$$r(t) = Re\left\{\left(\int_{-\infty}^{\infty} c(\tau, t) u(t-\tau)\right) e^{j2\pi f_c t}\right\}, \tag{2.5}$$

where $c(\tau, t)$ is given by [16]

$$c(\tau, t) = \sum_{n=0}^{N(t)} \alpha_n(t) e^{-j\phi_n(t)} \delta\big(t - \tau_n(t)\big), \tag{2.6}$$

where $\phi_n(t)$ holds all the frequency and phase components. The Dirac delta function is denoted by δ. [16] Channel impulse response-based modelling has also been used in the methods discussed in the following chapters. Another important property of the multipath transmission is

the distinction between narrowband and wideband signals. A well-known distinction between the wideband and the narrowband signals is done using a measure called coherence bandwidth. The coherence bandwidth is determined being approximately the inverse of the delay spread, i.e. the inverse of the time difference between the first and last multipath component $B_c \approx \frac{1}{T_D}$ [16].

Narrowband signals tend to have a bandwidth significantly smaller than the coherence bandwidth. In this situation the fading of the signal is highly correlated, and the frequency response is ideally flat, thus it is called flat fading. Incidentally, when the signal bandwidth exceeds the coherence bandwidth it is considered as a wideband signal which experiences frequency selective fading. Wideband signals are also more vulnerable to a phenomenon called inter symbol interference (ISI). ISI is a type of interference where the consecutive symbols slightly overlap in time distorting the transmitted symbols [16]. Several techniques introduced in the later chapters will assume a narrowband model.

One way to reduce ISI and achieve narrowband signals is a modulation technique called orthogonal frequency division multiplexing (OFDM). OFDM is a modulation technique which is used in both LTE and 5G systems and is one of the key technologies used in the modern wireless systems [17]. Instead of producing a one high data rate stream with large bandwidth, OFDM divides the frequency in several orthogonal subcarriers that overlap in the frequency domain as illustrated in Figure 2.6 [18]. Although subcarriers are allowed to overlap, it can be seen that when one subcarrier reaches maximum, the rest are zero. As a result, we have many slowly modulated parallel narrowband channels that can be used to transmit the same information as the high data rate stream using one carrier wave [19]. In addition, since several MIMO techniques assume narrowband transmission, OFDM creates a robust basis for sophisticated MIMO communication systems.

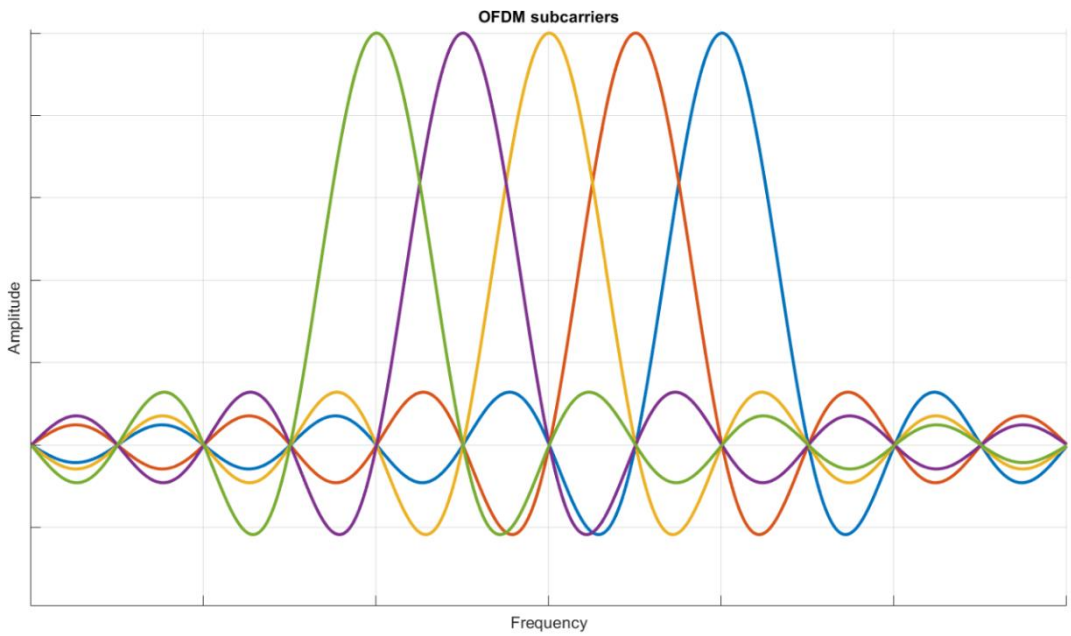

Figure 2.6. Orthogonal frequency division multiplexing.

3 MULTIANTENNA SYSTEMS

In Chapter 1 we went through the evolution of mobile networks and witnessed a tremendous increase in data rates. In the previous chapter we learned that it is possible to enhance the data rates by increasing the transmission bandwidths. However, the increased data rates have not been accomplished just by continuously increasing the bandwidth. It is also possible to boost up the transmission rate by increasing the spectral efficiency. One way to enhance the data rates in terms of the spectral efficiency is to use multiple antennas for the transmission or the reception of the signal. Multiple antennas in conjunction with diversity or beamforming techniques can provide auspicious enhancements for the channel capacity and the data rates.

In this chapter, we discuss multiantenna systems. To begin with, we describe how the conventional transmit diversity methods affect the channel capacity. These methods take advantage of multiple antennas in either transmitting or receiving ends, hence they can be classified as single in multiple out (SIMO) or multiple in single out (MISO) systems. When multiple antennas are deployed in the both ends of the wireless channel, the system is naturally called a multiple in multiple out (MIMO) system. When MIMO system is used in a heavy multipath environment it is possible to enhance the channel capacity significantly more than with traditional methods.

Different type of antenna configurations seen in wireless communications are illustrated in Figure 3.1. MIMO techniques introduced in this chapter are referred as beamforming techniques due to the physical form of the radiation pattern that originates from the antenna elements [20]. Beamforming techniques are advanced precoding methods. They are used to increase the channel capacity by intelligently exploiting the diversity and multiplexing gains provided by MIMO. In addition, multiantenna systems can be utilized to generate a highly directional transmission using phased array technique. This provides a highly directional gain, which helps to reduce the inter-user interference and delay spread which helps fighting ISI [16].

In this thesis a channel estimation algorithm for beamforming was implemented. This chapter emphasizes the importance of the channel estimation in multiantenna systems. The majority of the schemes introduced in this chapter are based on the channel coefficients making the channel estimation mandatory. We see that beamforming techniques are computationally demanding. Because adapting heavy computation to low latency is challenging, this further emphasizes the importance of the efficient implementation of channel estimation.

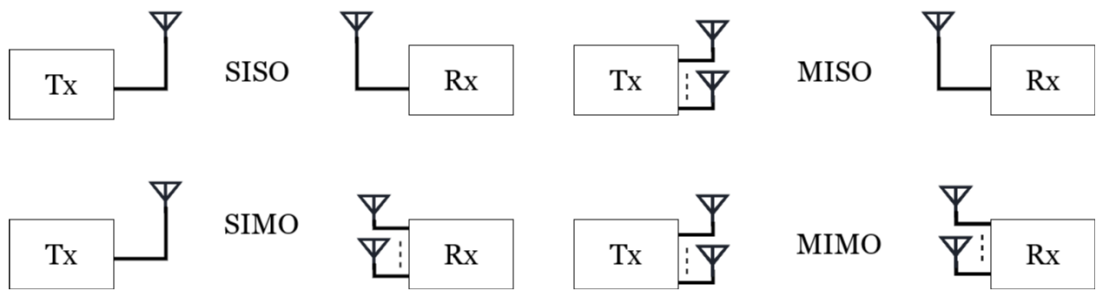

Figure 3.1. Different type of antenna systems.

3.1 Space Diversity

Diversity techniques can be used to mitigate the effect of fading. As the signal attenuates, signal-to-noise ratio (SNR) declines, and this affects to the performance of the channel. By using multiple antennas for the transmission or the reception, several signal paths are formed between the TX and the RX-antenna pairs. Therefore, the same information can be sent through several independent paths improving the reliability of the transmission. [21] This type of diversity is called space diversity, antenna diversity or spatial diversity.

In order to exploit space diversity, attention must be paid to the separation of the antenna elements. When the antenna elements are separated by a sufficient spacing, it is expected that their signalling paths are statistically independent and do not experience deep fades simultaneously. Sufficient distance between the adjacent antenna elements depends on several things. However, since the antenna spacing is clearly more limited in the mobile devices due to their small size, low fading correlation can already be achieved with distance of $\frac{\lambda}{4}$, where λ is the wavelength of the signal. [22] High frequencies make the antenna spacing very practical. Since the wavelength shrinks as the frequency increases, multiple antenna elements can be integrated to a small area without losing the sufficient distancing. Diversity techniques can also be applied in terms of frequency or time diversity, where the signal replicas are transmitted using independent time or frequency resources [16]. However, these two techniques will require an expansion in bandwidth and are out of the scope of this thesis [16].

System with receiver diversity is a multi-antenna system with one transmit antenna and several receive antennas, which is essentially a SIMO system. The idea behind the receiver diversity is to bring robustness to the transmission by introducing several independent signalling paths between the RX-TX-antenna pairs. The performance gained from the receiver diversity depends on the technique which is used for combining the signals from the different antennas. Maximal-ratio combining (MRC) is a technique which combines all the received signals by co-phasing the individual components and weighting them with an appropriate scaling factor in order to reach the maximal SNR. Most combining techniques including MRC are linear combiners which is illustrated in Figure 3.2 [16]. [16]

In MRC, the signal in every branch is multiplied by a coefficient $\alpha_i = a_i e^{-j\theta_i}$, where θ_i is the phase of the signal on ith branch. θ_i cancels the phase of every signal branch enabling coherent addition. If we assume an identical noise power spectral density (PSD) in every branch the total or the output SNR of the combiner γ_Σ can be expressed as follows [16]

$$\gamma_\Sigma = \frac{r^2}{N_{tot}} = \frac{1}{N_0} \frac{(\sum_{i=l}^{M} a_i r_i\ )^2}{\sum_{i=l}^{M} a_i^2}, \tag{3.1}$$

where a_i, r_i and N_0 are the scaling coefficient, the received signal on the ith branch and the noise, respectively. Because MRC wanted to maximize SNR, coefficient a_i should be chosen accordingly. For example, it is possible to show using the Cauchy-Schwartz inequality that by weighing the signals by respective amplitudes will lead to the optimal SNR. Output SNR will correspond to the sum of the SNR of every branch. [16] When an equal average SNR is assumed on each individual branch with i.i.d. Rayleigh fading, the expected SNR achieved using the maximal-ratio combining is given by [16]

$$\bar{\gamma}_\Sigma = M\bar{\gamma}, \tag{3.2}$$

where $\bar{\gamma}$ denotes for the average branch SNR and M is the number of the RX-antennas. As seen from (3.2), using M receiving antennas, an M-fold increase in SNR was achieved on average. [16]

A much more straightforward combining method is called selection combining (SC). In selection combining the branch with the highest SNR would be chosen as the received signal. This approach eliminates the need for phase compensation leading to a reduced complexity. If we assume equivalent fading and noise with (3.2), the average SNR using the selection combining is given by: [16]

$$\bar{\gamma}_{\Sigma} = \bar{\gamma} \sum_{i=1}^{M} \frac{1}{i} \tag{3.3}$$

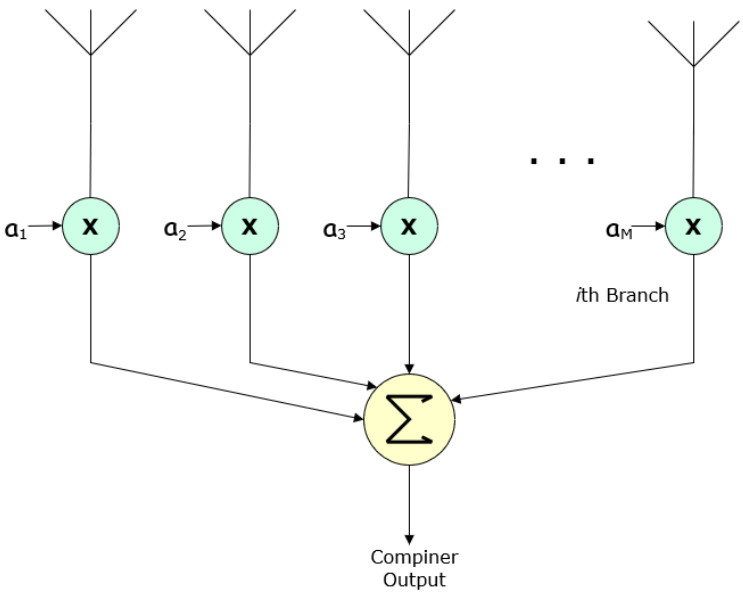

Figure 3.2. Illustration of the linear combiner.

The best efficiency from the selection combining is clearly achieved with two receiving antennas. Although MRC offered higher gain, SC offers a low complexity alternative making it also an interesting alternative for some purposes [16].

Space diversity can be exploited in the transmitting end as well. These setups can be classified as MISO type of antenna systems shown in Figure 3.1. If the channel is known to the transmitter in advance, i.e. the complex channel gain for the ith transmit antenna $r_i e^{j\theta_i}$ is known, the transmitted signal $s(t)$ is multiplied with a complex gain $\alpha_i = a_i e^{-j\theta_i}$ before the transmission. It is immediately seen that the phase term θ_i of the complex gain term α_i corresponds to the conjugate of the corresponding channel gain. If all paths are co-phased according to their channel gain, a constructive interference should happen at the receive position. This sort of prefiltering is applied to all transmit antennas. As all the signals are pre-distorted according to their channel gains, channel acts as a matched filter and the received signal is a coherent combination of the transmitted signals. This method can provide similar SNR gain as its receiver counterpart. [16]

If the channel is not known to the transmitter, the previous method is inapplicable. Just by equally dividing the transmit power along the antennas no gain would be obtained in this situation [16]. However, using the well-known space-time coding method known as the Alamouti scheme it is possible to obtain transmit diversity to some extent [16]. When the Alamouti scheme is used with two transmit antennas, SNR at the receiver is given by [16]

$$\gamma_i = \frac{(|h_1^2| + |h_2^2|)E_s}{2N_0}, \tag{3.4}$$

where h_1 and h_2 are the channel gains. In this section, we saw that by using multiple transmitter or receiver antennas, it was possible to achieve a better signal-to-noise ratio. Recalling Equation (2.1), by increasing the signal-to-noise ratio we also increase the channel capacity. Although, the increase is only logarithmic, the capacity still increases. This is especially important when the signal-to-noise ratio is very low. The increase in SNR is called array gain. Better SNR also increases the reliability of the transmission which in turn is called diversity gain [16]. By comparing the introduced transmit diversity schemes, we noticed that the schemes which utilized the channel information for filtering the transmitted signals achieved the best array gain. In addition, SNR at the receiver was maximized with the price of system complexity. We have already witnessed that the channel estimation plays an important role only by reviewing some of the conventional multiantenna techniques.

3.2 Spatial Multiplexing

When we are dealing with a multiantenna system that contains multiple transmitting and multiple receiving antennas, there exists a different type of technique to improve channel capacity. This technique is called spatial multiplexing. In order to use spatial multiplexing, we have to employ multiple antennas in both ends. This type of antenna configuration corresponds to the MIMO system shown in Figure 2.1. In principle, spatial multiplexing differs from the transmit diversity in that it sends different information via multiple spatial stream instead of sending the same symbol all over again. Each of these streams are capable of transmitting information independently, hence the name multiplexing.

The primary goal with the transmit and the receive diversity methods was to increase the channel capacity by fighting against fading. This led to an increased SNR, thus achieving a somewhat logarithmic increase in the channel capacity. Spatial multiplexing takes completely different approach in some sense. In a fading multipath environment, there is also the possibility of independent fading between different transmitting and receiving antennas. This makes communication via independent spatial stream possible [21]. For a system with n RX antennas and m TX antennas, it is shown that with a high SNR under i.i.d. Rayleigh-fading the channel capacity can be written as: [21]

$$C = \min\{n, m\} \log_2(SNR) \tag{3.5}$$

Although the capacity given by Equation (3.5) also requires certain other conditions to be fulfilled than high SNR, multiplicative increase in the channel capacity sounds promising.

More specifically, the number of antennas alone does not determine the increase in capacity. The key parameter is actually the number of spatial streams which is heavily dependent on the channel. A narrowband MIMO-system can be expressed as [16]

$$\begin{bmatrix} y_1 \\ \vdots \\ y_{M_r} \end{bmatrix} = \begin{bmatrix} h_{11} & \dots & h_{1M_t} \\ \vdots & \ddots & \vdots \\ h_{M_r1} & \dots & h_{M_rM_t} \end{bmatrix} \begin{bmatrix} x_1 \\ \vdots \\ x_{M_t} \end{bmatrix} + \begin{bmatrix} n_1 \\ \vdots \\ n_{M_r} \end{bmatrix}, \tag{3.6}$$

which can be simplified as $\mathbf{y} = \mathbf{Hx} + \mathbf{n}$. In the previous equation $\boldsymbol{y}$ represents the M_r-dimensional vector of received symbols, $\mathbf{H}$ is the channel matrix of size $M_r \times M_t$ wherein each element represents the complex channel gain of the antenna pair determined by the subscript. $\mathbf{x}$ is the M_t-dimensional vector of the transmitted symbols and $\mathbf{n}$ is the noise vector. From Equation (3.6) we see that the received symbol y_m in the mth receive antenna is a linear combination of all transmitted symbols multiplied with the corresponding channel coefficients. If we imagine that the channel matrix $\mathbf{H}$ was a diagonal matrix, then each received symbol would correspond to only one transmitted symbol. If this was the case, the symbols would be completely independent and resolvable. [16]

It is possible to present the channel matrix $\mathbf{H}$ as a product of three matrix by conducting the singular value decomposition (SVD) [16]

$$\mathbf{H} = \mathbf{U\Sigma V}^{\mathrm{H}}, \tag{3.7}$$

where the matrices $\mathbf{U}$ and $\mathbf{V}$ are unitary matrices of size $M_r \times M_r$ and $M_t \times M_t$, respectively, and $\mathbf{\Sigma}$ is a diagonal matrix of size $M_r \times M_t$. Furthermore, the columns of the matrix $\mathbf{U}$ are called the left singular vectors of the matrix $\mathbf{H}$ and the columns of the matrix $\mathbf{V}$ are called right singular vectors of the matrix $\mathbf{H}$. The values σ_i that are on the diagonal of the matrix $\mathbf{\Sigma}$ are called the singular values of the matrix $\mathbf{H}$. [16]

The singular values on the diagonal of the matrix $\mathbf{\Sigma}$ can be either zero or non-zero. The number of non-zero singular values can be determined from the rank of the channel matrix $\mathbf{H}$. When the channel retains a highly scattering environment, it is possible that $\mathbf{H}$ can achieve full rank. The other extreme would happen when the channel gains have very high correlation between each other. If the channel coefficients are highly correlated the rank of the channel matrix is one meaning that $\mathbf{H}$ has only one non-zero singular value. [16]

In fact, the number of non-zero diagonal elements, i.e. non-zero singular values, of the matrix $\mathbf{\Sigma}$ determine the maximum number of spatial streams. Since the rank of the channel matrix cannot exceed $\min(M_r, M_t)$, the number of spatial streams is also limited by the number of antennas. Therefore, the number of spatial streams is determined by the size of the antenna configuration and the nature of the multipath channel. By conducting SVD on the channel matrix we were able to represent it as a diagonal matrix multiplied with two other matrices. This suggests that if the transmission could be precoded so that it sees the channel matrix as diagonal, it could be possible to receive independent symbols.

3.3 Single-User Beamforming

We have seen that by coherently combining the signals transmitted or received using multiple antennas provided array and diversity gains that enhanced the performance of the channel. The remaining topics in this chapter deal with increasing channel capacity in MIMO systems. In this section a beamforming technique called eigen-beamforming is discussed. When the eigen-beamforming is used to transmit a single spatial stream, it is evident that it has a lot in common

with the previously introduced spatial diversity techniques. If multiple spatial streams are used, eigen-beamforming has shown to be the optimal precoding strategy for single-user MIMO [20]. Both eigen-beamforming techniques using single and multiple spatial streams are introduced in this section.

In the case of single spatial stream both ends perform precoding and decoding on the transmitted and the received symbols. The transmitted symbol is precoded using the vector $\mathbf{v}$ and the received symbol is decoded using vector $\mathbf{u}$, where $|\mathbf{u}| = |\mathbf{v}| = 1$. This results in a received signal: [20]

$$\mathrm{y} = \mathbf{u}^{\mathrm{H}}\mathbf{H}\mathbf{v}x + \mathbf{u}^{\mathrm{H}}\mathbf{n} \tag{3.9}$$

It is shown in [20] that the channel can be seen as an AWGN SISO channel when this technique is used. Therefore, the channel capacity for single spatial stream case is given by: [20]

$$C = \log_2\left(1 + \frac{P_t}{\sigma^2}|\mathbf{u}^{\mathrm{H}}\mathbf{H}\mathbf{v}|^2\right) \tag{3.10}$$

The vectors $\mathbf{u}$ and $\mathbf{v}$ should be chosen so that the signal-to-noise ratio is maximized. Furthermore, it is shown in [20] that the vectors $\mathbf{u}$ and $\mathbf{v}$ should be chosen as the first left and the first right singular vectors of the channel matrix $\mathbf{H}$. We saw in the context of spatial multiplexing that it was possible to identify the independent signalling paths using SVD. Eigen-beamforming-based transmission suggests that the path with the strongest gain should be used for the transmission. This technique can provide a σ_{max}^2-fold increase in SNR, where σ_{max} corresponds to the largest singular value of the channel matrix $\mathbf{H}$ [16]. When the channel matrix is zero mean spatially white, the array gain, i.e. the increase in SNR is in the range between $\max(M_t, \mathrm{M_r})$ and $M_t M_r$ [16]. In contrast to the maximal-ratio combining where multiple antennas were deployed only on one side, SNR increased approximately with the numbers of antennas in the multiple antenna side. Under the above assumptions, eigen-beamforming using a single spatial stream can further improve the signal-to-noise ratio. Since this approach provided a better SNR, we achieved diversity and array gains. If we use eigen-beamforming utilizing several spatial streams, it is also possible to obtain multiplexing gains which can lead to even better capacity.

If multiple spatial streams are used, it is shown that the transmit precoding coefficients should be chosen as the right singular vectors of the channel matrix $\mathbf{H}$ [20]. This seems to suggest that all possible spatial streams should be used. In addition, if the receiver uses the left singular vectors for decoding the transmission, the channel model eventually simplifies and is written as: [20]

$$\mathbf{y} = \mathbf{\Sigma}\mathbf{x} + \mathbf{U}^{\mathrm{H}}\mathbf{n} \tag{3.11}$$

When reminding that $\mathbf{\Sigma}$ is a diagonal matrix whose number of non-zero elements is given by the rank of $\mathbf{H}$, Equation (3.11) explicitly shows the independent nature of the spatial streams.

Channel capacity C for a system utilizing the eigen-beamforming with multiple spatial streams is given by [20]

$$C = \sum_{i=1}^{M} \log_2\left(1 + \frac{\lambda_i^2 P_i}{\sigma^2}\right), \tag{3.12}$$

where P_i is the transmit power of the ith stream and λ_i is the ith eigenvalue of $\mathbf{HH}^{\mathrm{H}}$. Earlier it was stated that the channel gain associated with the ith spatial stream was given by the ith singular value. Eigenvalues of $\mathbf{HH}^{\mathrm{H}}$ are related to the singular values of $\mathbf{H}$ as follows $\sigma_i = \sqrt{\lambda_i}$ and are adopted due to practical reasons. [20]

From Equation (3.12) it can be seen that the capacity increases with respect to the number of spatial streams. In theory, if the gain of all spatial streams were one and their SNR were identical, Equation (3.12) would clearly correspond to the capacity of AWGN SISO channel given by Shannon's theorem (2.1) multiplied by the number of streams. Under these rather surrealistic assumptions, a multiplicative increase in the channel capacity would have been achieved without increasing the bandwidth. In reality, the channel gain is neither unity nor identical between the individual antenna elements. Therefore, the channel capacity is optimised by adjusting the transmit powers according to water-filling principle [20].

Overall it is clear that these methods offer significant increase in the channel capacity. However, the capacity is improved with the price of increased computation. Introduced beamforming techniques consists of computationally demanding operations such as matrix multiplications and decompositions. In order to meet the latency requirements, these features must be implemented as efficiently as possible. In this thesis we aim to ease the workload caused by the beamforming by investigating the properties of the mandatory channel estimation for beamforming and implementing it efficiently using a suitable architecture. This topic is discussed in the following chapters.

The computational efforts of beamforming can be further reduced by using limited feedback precoding, which consists of a codebook provided by the 3GPP specifications. This codebook is composed in order to reduce the computational load. The idea behind the codebook-based beamforming is to look up the parameters from the codebook instead of calculating them directly. [23]

3.4 Multi-User Beamforming

Beamforming can also be applied in multi-user scenarios. The key difference between the multi-user and the single user beamforming is that all spatial streams are not addressed to the same user. Therefore, these undesirable signals must be considered in the precoding strategy [20].

In [24] it is shown that if the channel is subject to some additional interference and the distribution of the interference is known, using a proper precoding method it is possible to adapt the transmission to this interference. In addition, it has been shown that using this method the channel capacity would correspond to the capacity without the interference. This method is known as the dirty paper coding (DPC) and it has been suggested to be the theoretically optimal multi-user precoding strategy. However, it has been pointed out that this method is very hard to implement in reality which makes it impractical [20].

One popular multi-user beamforming technique is a linear precoding scheme called zero-forcing (ZF) beamforming. ZF has attracted attention due to its more feasibility. ZF is a computationally lighter option than DPC which does not lose too much relevant information. It is shown by Weasel *et.al* that ZF type of precoding is the optimal coding strategy under a total power constraint in MU-MIMO situation. Zero forcing precoding matrix $\mathbf{W}_{\mathrm{Z}}$ under a total power constraint is given by [25]

$$\mathbf{W}_{\mathrm{Z}} = \mathbf{H}^{\mathrm{H}}(\mathbf{HH}^{\mathrm{H}})^{-1}, \tag{3.13}$$

where, $\mathbf{H} \in \mathbb{C}^{NxM}$ is the channel matrix. M corresponds to the number of transmit antennas, N is the number of users and $\mathbf{H}^{\mathrm{H}}$ denotes the conjugate transpose of the channel matrix $\mathbf{H}$. ZF solution is said to transform the multiuser channel into the several independent subchannels which is done by finding the inverse channel of the given channel matrix $\mathbf{H}$. It is evident that the zero forcing method also relies heavily on the channel information. Multi-user beamforming is discussed in more detail in [20].

Beamforming techniques introduced in this chapter offered a method to improve the channel capacity via complex spatial signal processing tasks. As discussed earlier, this sets challenges towards the system design due to the strict latency requirements of the 5G network. This emphasizes the importance of the efficient implementation of these techniques. Naturally, it is very important to optimize the chosen beamforming technique itself like it was done with the eigen-beamforming. By using the eigenvalues of the channel covariance matrix instead of the singular values of the channel matrix additional computation is avoided and better efficiency is achieved [20].

3.5 Summary

We found that the beamforming techniques presented in this chapter improved the channel capacity. However, these methods are computationally demanding and completely dependent on the channel matrix. By acquiring the channel coefficients that form the channel matrix efficiently, we can facilitate the attainment of the desired latency requirements on our part. This subject is addressed in the subsequent chapters.

4 CHANNEL ESTIMATION

We have seen that the multiantenna systems, especially MIMO systems can provide enhanced channel capacity via advanced beamforming schemes. However, these schemes were dependent on the channel matrix. Furthermore, since the multipath channel shifts the phase and scales the amplitude of the signal, in order to utilize high order modulations such as quadrature amplitude modulation (QAM), coherent detection of the signal also requires the channel parameters [26]. If the channel parameters were unknown, we would be limited to using differential modulation methods. This would significantly reduce the number of bits per symbol and decrease the signal-to-noise ratio [27]. In summary, channel state information also plays an important role in other parts of the system than beamforming. The parameters describing the channel are obtained from a process called channel estimation.

The channel between the UE and BS can change rapidly. Channel estimation should be performed within a short period of time in order to prevent the channel from becoming substantially different during the estimation. Because the estimation must be performed quickly and frequently, its efficient implementation is crucial. Recalling we are implementing a channel estimation algorithm for beamforming. The importance of the efficiency is emphasized by the fact that the estimation is part of an already computationally demanding entity. In this chapter we discuss channel estimation by getting to know its basics and presenting solutions to reduce complexity.

4.1 Overview

In [28], channel estimation in OFDM systems is categorized into four different approaches: iterative channel estimation, parametric model-based estimation, channel frequency response based and estimation for MIMO-systems whereas frequency response-based estimation is referred as the traditional approach. In this chapter emphasis is on the latter two.

Channel frequency response-based channel estimation in OFDM is commonly further divided into two different categories. These categories are decision-directed estimation and pilot-assisted estimation. The graphical distinction between these schemes is provided in Figure 4.1 [28]. In the decision-directed approach the whole bandwidth is loaded with training symbols and they are utilized for the detection of subsequent data symbol. In pilot-assisted method, the pilot signals are inserted in either comb-type or block-type and the frequency responses on different data symbols can be obtained using different interpolation techniques [28].

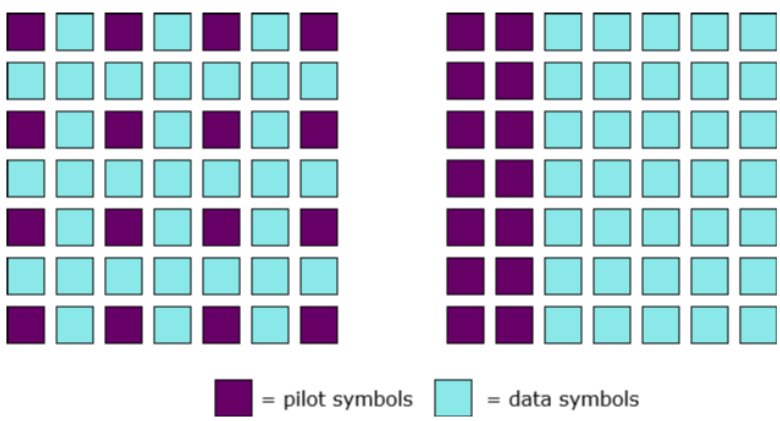

Figure 4.1. Comb-type PA and DD.

Commonly, channel estimation methods assume a system model based on the time-varying channel impulse response introduced in the second chapter. When the general assumptions of the OFDM system are considered, the received signal at the kth subcarrier of the nth symbol $Y_{n,k}$ is obtained from the channel frequency response as follows

$$Y_{n,k} = H_{n,k}X_{n,k} + W_{n,k}, \tag{4.1}$$

where $X_{n,k}$ denote the transmitted symbol e.g. pilot symbol and $W_{n,k}$ stands for the additive Gaussian noise with zero mean. $H_{n,k}$ is the corresponding channel frequency response. [28]

4.2 Channel Estimation in Single Antenna Systems

Since the received signal is corrupted by the noise and distorted by the time variant multipath channel, the acquisition of channel parameters is done by estimating them. A frequently encountered estimation approach is known as the least squares (LS) method. In LS, the goal is to find the parameters that minimizes the squared error of $|\mathbf{A}\bar{\mathrm{x}} - \mathbf{b}|^2$, where $\mathbf{A}$ is $N \times M$ matrix, $\bar{\mathrm{x}}$ and $\mathbf{b}$ are vectors of size $M \times 1$ and $N \times M$, respectively. Least squares method is used to find an approximate solution to the systems that are overdetermined. This means that there are N equations, M unknowns and $N > M$. The solution is found to be a vector $\bar{x}_{LS}$ for which $\mathbf{A}\bar{x}_{LS}$ is the orthogonal projection of the vector $\boldsymbol{b}$. The general solution for the linear least squares is given by [23]:

$$x_{LS} = (\mathbf{A}^{\mathrm{H}}\mathbf{A})^{-1}\mathbf{A}^{\mathrm{H}}\mathbf{b} \tag{4.2}$$

The least squares approach is used especially when no statistical information regarding the channel is available, or it is ignored due to some other reason. When Equation (4.1) is presented in a vector form, the task to minimize the squared distance between $\mathbf{Y}$ and $\mathbf{XH}$ for the nth symbol is given by

$$\widehat{\mathbf{H}}_{LS} = \arg\min_{\{\mathbf{H}\}} |\mathbf{Y} - \mathbf{XH}|^2, \tag{4.3}$$

where $\mathbf{H}$ and $\mathbf{Y}$ are the vectors containing the channel frequency responses and the received signals and are given as follows

$$\mathbf{Y} = [Y_0, Y_1, \dots, Y_{K-1}]^T,$$
$$\mathbf{H} = [H_0, H_1, \dots, H_{K-1}]^T,$$

where K denote the number of subcarriers. The transmitted signal $\mathbf{X}$ is a $K \times K$ diagonal matrix, where the transmitted symbols X_k lie on the diagonal according to their subscript. [28] In this situation the least squares estimate is given simply as [27]:

$$\widehat{\mathbf{H}}_{LS} = \mathbf{X}^{-1}\mathbf{Y} \tag{4.4}$$

The least squares estimate given by Equation (4.4) corresponds to the temporal estimate over one OFDM symbol. It is often pointed out in the papers that when the noise is additive white Gaussian noise, Equation (4.4) is equivalent to the maximum-likelihood estimation [28]. Although matrix inversion is considered a computationally demanding operation, in Equation (4.4) it is not a problem. Since $\mathbf{X}$ is a diagonal matrix its inversion only requires the inversion of the diagonal elements. Computationally, least squares-based approach in single antenna OFDM systems seems intriguing method since it only corresponds to elementwise multiplication of complex vectors.

In addition to least squares estimation, minimum mean square estimation i.e. MMSE, was also considered in [27]. Computationally, MMSE differs significantly from the LS estimate. MMSE assumes that the statistical distribution of the frequency responses is known, and it aims to minimize the error of the expected value of the squared error. MMSE estimate can be calculated as follows

$$\widehat{\mathbf{H}}_{LMMSE} = \mathbf{R}_H \left(\mathbf{R}_H + \frac{1}{\gamma} \mathbf{I} \right)^{-1} \widehat{\mathbf{H}}_{LS}, \tag{4.5}$$

where $\mathbf{R}_H = E(\mathbf{H}\mathbf{H}^H)$ denote for the channel correlation matrix, and γ stands for SNR. In the equation for MMSE estimator a matrix inversion is required, which significantly increases the computational complexity of the method compared to LS. However, it is expected that the increased computational complexity would provide more accurate estimates. [28]

It is possible to reduce the significant gap in the computational complexity between LS and MMSE methods. Modified LS and MMSE estimators were suggested in [27]. These modifications exploited the properties of the channel impulse response. It was noticed that high energy channel taps were found from the first elements of the impulse response. Therefore, it was suggested that in order to reduce the complexity of MMSE estimator, it was decided to exclude the low energy taps from the estimation and rely on the information given by the taps with the higher energy. Additionally, they applied this exact same method to LS estimator structure even it was not necessary from the computational complexity point of view.

Since it was noticed that the first channel taps tend to have a bigger impact due to their relatively higher energy content, it is intuitively beneficial to exclude the lower energy taps from the LS calculation. Since the noise spectrum of AWGN is considered flat over the whole frequency range, by ignoring the low energy taps, presence of the noise could potentially be reduced. [27]

Transmission reliability results for the modified and conventional LS and MMSE estimators in a system using 16-QAM modulation are provided in Figure 4.2 [27]. Especially the modified LS estimator gained much better results in the low SNR region. However, this was achieved with at the expense of computational complexity, since after the exclusion of the non-significant channel taps, the method no longer corresponds to the simple form given in Equation (4.4).

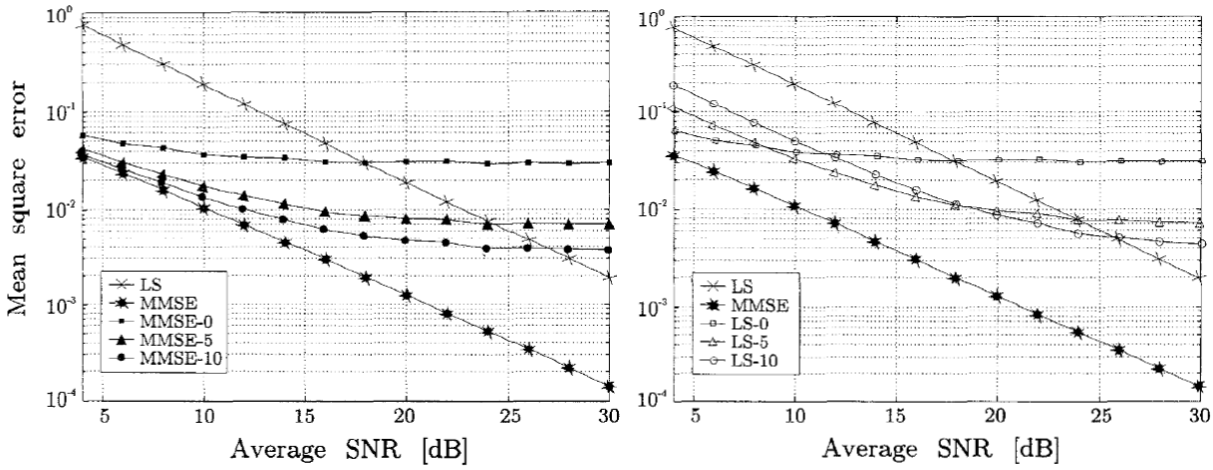

Figure 4.2. Comparison of channel estimation schemes. © [1995] IEEE

According to the discussion above, channel estimation seems to be possible with very low complexity. The least squares estimate given in (3.4) would be a very pleasing method for efficient implementation since it involves only a highly parallelisable multiplication. Although the above results favour complex methods in terms of transmission reliability, the situation is a bit different for beamforming.

In [20] it is pointed out that whether the channel covariance matrix is calculated using a channel matrix with exact values versus one with a slightly erroneous values, the effect on the system performance should be negligible. However, it should be noted that the errors caused by the channel aging are considered harmful [20]. Channel estimation should be performed very frequently so that the aging does not cause problems. Since small inaccuracies are not considered harmful, a low complexity method would provide a good starting point for the efficient implementation of channel estimation. The conventional least squares method offered an incredibly simple estimation scheme. However, beamforming requires a MIMO system. This raises the question whether this simple method could also be utilized in the MIMO systems?

4.3 Estimation in Multiantenna Systems

The previous section introduced the basics of channel estimation in single antenna OFDM systems. It was noticed that it is possible to obtain reasonable estimates using the linear least squares method which offered very low complexity. Since our interest in channel estimation arose from the beamforming techniques, in this chapter, channel estimation in MIMO systems is discussed. At first, by providing a basic picture of the problem, a sort of general approach for the channel estimation in multiantenna system is introduced. In addition to this, we discuss techniques that aim to reduce the complexity of channel estimation.

A narrowband MIMO system is illustrated in Figure 4.3 [16]. There are M_t transmit antennas, M_r receive antennas and the corresponding channel coefficient h_{tr} for the each transmit and receive antenna pairs. This model is often represented using the Equations (3.6) and (3.7) from the previous chapter.

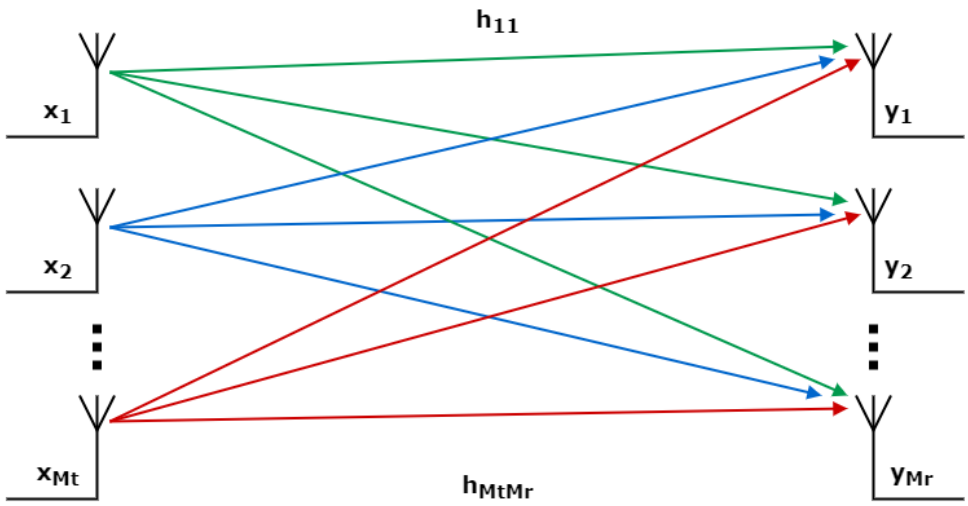

Figure 4.3. Narrowband MIMO system.

The situation has changed quite a bit compared to the conventional single antenna system. If the received symbol $Y_{n,k}^{j}$ at the jth receive antenna is observed, it seems to be a superposition of the transmitted signals from all transmit antennas and it is given by: [28]

$$Y_{n,k}^{j} = \sum_{i=1}^{M_t} H_{n,k}^{j,i} X_{n,k}^{i} + W_{n,k}^{j} \tag{4.6}$$

Since the received signal is now exposed to several channel parameters, the LS method cannot be directly utilized as in the previous section. A straightforward solution would be to perform the channel estimation individually for every transmit antenna by keeping all but one antenna silent. Then during the training period $M_t = 1$ and Equation (4.6) would correspond to the single antenna case in (4.1). This simple solution does not sound promising since it would make the estimation very slow and reduce the spectral efficiency. Clearly the estimation should be performed on all antennas at the same time. [28]

A sort of basic approach for MIMO-OFDM channel estimation is described in [29]. As shown in Equation (4.6), the received symbol at the jth receive antenna can be expressed using the channel frequency response. The channel frequency response for the ith transmit antenna at the kth subcarrier of the nth symbol can be obtained by taking the discrete Fourier transform (DFT) from the channel impulse response as follows [29]

$$H_i[n,k] = \sum_{l=0}^{L-1} h_i[n,l] W_K^{kl}, \tag{4.7}$$

where L denotes the number of non-zero channel taps and $W_K = e^{-j\left(\frac{2\pi}{K}\right)}$. Therefore, the task is to find h_i's which minimize the following function: [29]

$$\sum_{k=0}^{K-1}\left|Y_{n,k}^{j}-\sum_{i=0}^{N_t}\sum_{l=0}^{L-1}X_{n,k}^{i}h_{n,l}^{j,i}e^{-j\left(\frac{2\pi kl}{K}\right)}\right| \tag{4.8}$$

Direct calculation using the above expression will require an inversion of matrix of size $M_tL \times M_tL$ [29]. So, by formulating the problem directly for the MIMO system, the computational complexity appears to be considerable.

Similarly, as with the single antenna systems a method that would ignore the small energy channel taps was suggested for this method as well. This would naturally reduce the size of the matrix, but as the matrix inversion is still required, the complexity is still very high. [28] The complexity of the method can be significantly reduced by careful pilot sequence design. It is shown in [26] that with proper pilot sequence the complexity of the scheme is significantly reduced. It is possible to select a pilot sequence so that the matrix to be inverted is reduced to a diagonal matrix which is much more feasible to work with.

The least squares estimate of channel matrix **H** for a narrowband MIMO system can be obtained as follows [23]

$$\widehat{\mathbf{H}}_{LS} = \mathbf{Y}\mathbf{X}^{\mathrm{H}}(\mathbf{X}\mathbf{X}^{\mathrm{H}})^{-1}, \tag{4.9}$$

where $\mathbf{X}$ is an $M_t \times N_{tr}$ and $\mathbf{Y}$ is an $M_r \times N_{tr}$ matrix. N_{tr} is the length of the training sequence. Calculating the LS estimate directly from Equation (4.9) would be very cumbersome due to matrix inversion. If the transmitted pilot sequences are chosen to be orthogonal to each other, the product between the training symbol matrix and its Hermitian becomes $\mathbf{X}\mathbf{X}^{\mathrm{H}} = \mathrm{N_{tr}}\mathbf{I}$ where $\mathbf{I}$ is identity matrix of size $M_r \times N_{tr}$. Once again, by avoiding the matrix inversion the complexity of the estimation scheme reduced significantly. [23] As a result, the matrix inversion only corresponds to a division by the length of the pilot sequence and (4.9) is reduced to

$$\widehat{\mathbf{H}}_{LS} = \frac{1}{N_{tr}}\mathbf{Y}\mathbf{X}^{\mathrm{H}}, \tag{4.10}$$

which is a matrix multiplication problem.

So far, we have seen a few techniques to facilitate the channel estimation in MIMO systems. Although orthogonal pilot sequences reduced the complexity of the channel estimation, one challenge has been ignored. The size of the matrices given in (4.10) are dependent of the size of the antenna configuration. Since it is expected that the number of antenna elements keeps growing in the future, this sets a prominent challenge towards the implementation. If Equation (4.10) is approached as a traditional matrix multiplication it can be broken down into two operations, multiplications and additions of complex numbers. It seems that calculating $\mathbf{Y}\mathbf{X}^{\mathrm{H}}$ requires $N_tN_rN_{Tr}$ complex multiplications and $N_tN_r(N_{Tr}-1)$ complex additions. Furthermore, a complex multiplication would require four real multiplication with two real additions.

The number of required operations for different antenna configurations is illustrated in Figure 4.4. Although the number of pilot symbols used in this illustration is relatively small, it is clear when the size of the antenna pair doubles, the number of operations quadruples. While graphically, the number of operations seems to increase linearly, it should be noted that the scale is logarithmic, thus the number of operations increases exponentially. Although the result

is not surprising, it causes significant challenges towards the signal processing platforms because both the channel estimation and the beamforming depend on the size of the antenna configuration.

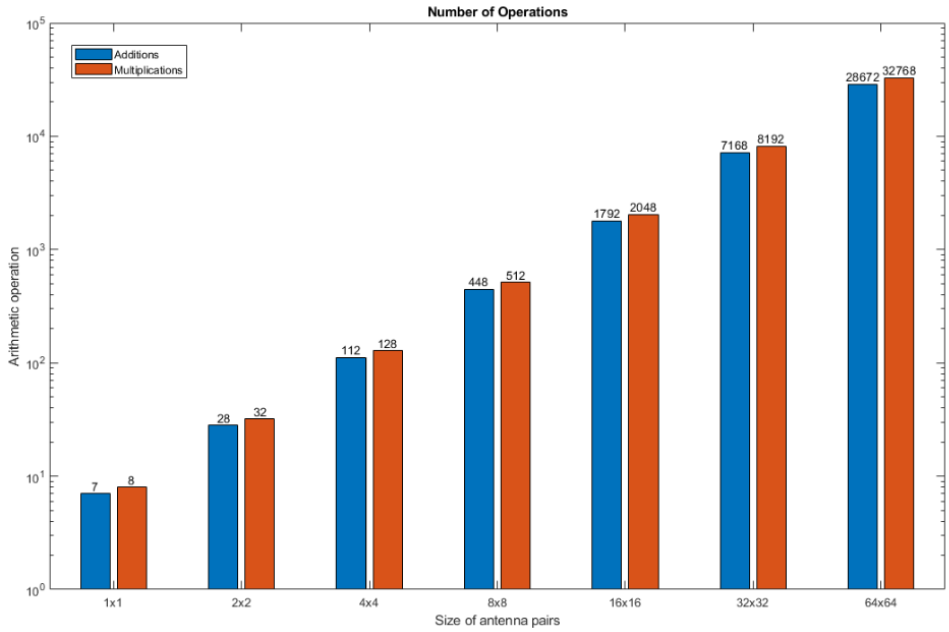

Figure 4.4. Number of operations with respect to antenna elements on logarithmic scale.

There are matrix multiplication algorithms which promise to reduce the computational complexity compared to the conventional approaches. For example, the Strassen's algorithm provides a reduction in the total number of multiplications compared to the conventional method. Strassen's algorithm alongside the other published matrix multiplication algorithms are usually considered useful for very large matrices. Naturally, the practicality of all advanced matrix multiplication algorithms depends on the platform on which they are implemented. For example, the elementary steps in the Strassen's algorithm require very much specific indexing operations. Therefore, fitting it to some specific system architecture can be very challenging and the reduced number of multiplications does not directly translate into better performance.

However, a very straightforward way to increase the efficiency of the estimation is to use a parallel architecture. Already an assumption was made that the complex arithmetic operations were considered as a single operation. This is a totally valid assumption when the algorithm is implemented using an architecture which offers complex arithmetic operations as elementary instructions.

If the algorithm could be implemented by performing multiple operations in parallel, it would significantly reduce the number of required operations to certain extent. How parallelisation of the algorithm affects to the number of operations is illustrated in Figure 4.5. The curve appears to be identical for all degrees of parallelism (DOP). However, the larger DOP postpones the inevitable growth in the number of operations by shifting the curve towards the right side of the figure. Figure 4.5 clearly suggests that by increasing the degree of parallelism the time performance of the system could be enhanced. When the size of the antenna configuration increases the degree of parallelism should also be increased in order to keep the number of operations restrained.

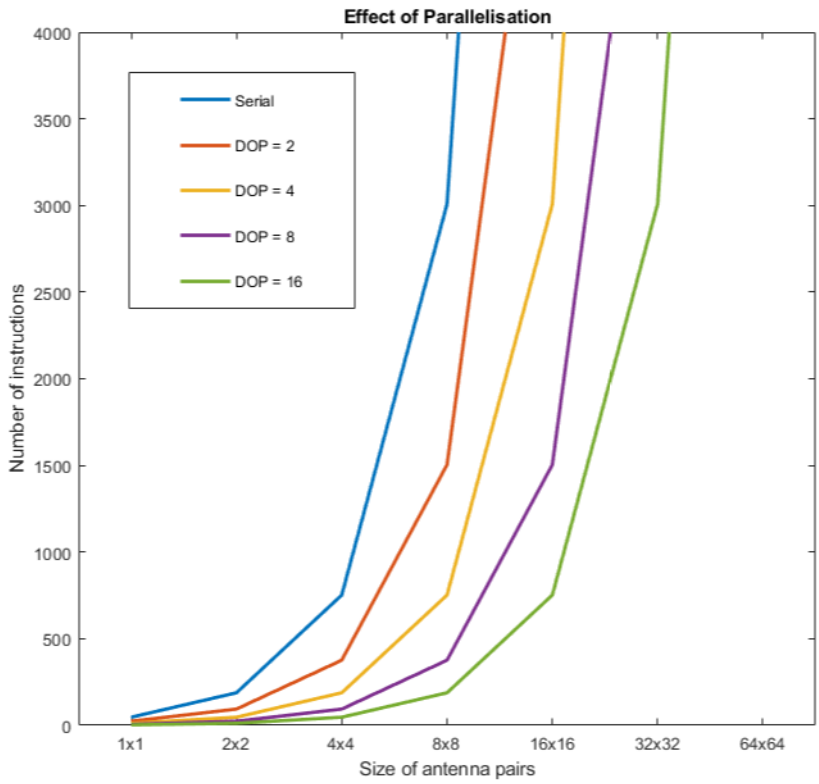

Figure 4.5. Effect of parallelisation to number of operations.

Overall, several generalizations about the LS method exists for channel estimation in MIMO systems. In this chapter the topic was introduced by exploring the basic methods that trade performance to complexity and vice versa. Channel estimation in wireless communications is a very broad topic and we were mainly focused on the basics. For example, the training sequence also affects several other properties than complexity. Careful training sequence design also improves the estimation accuracy [23]. 3GPP specifications for 5G specify an orthogonal pilot sequence called Zadoff-Chu sequence. As pilot sequences play very important role in channel estimation, SRS transmission alongside Zadoff-Chu sequences are discussed in the next subsection.

4.4 SRS Transmission in LTE and 5G

Data transmission in LTE and 5G systems is performed using a grid type of structure which divides the time and frequency resources into fixed size components as shown in Figure 4.6. The time and frequency resources are organized into frames, which in turn consists of slots and subframes. The radio frame can be configured according to several numerologies which are specified in the 3GPP specifications [30]. Numerology parameter determines for example the length of the subcarrier spacing and the number of slots within the frame. The frame structure illustrated in Figure 4.6 corresponds to the configuration $\mu = 0$ from the specifications in [30]. The smallest allocable resource unit is called a resource element (RE). In Figure 4.6 [31] resource element corresponds to a one small square. Resource elements consists of one OFDM subcarrier during one symbol period. Resource elements can be used to carry information by

modulating the subcarrier with some digital modulation scheme such as QPSK or different sized QAM.

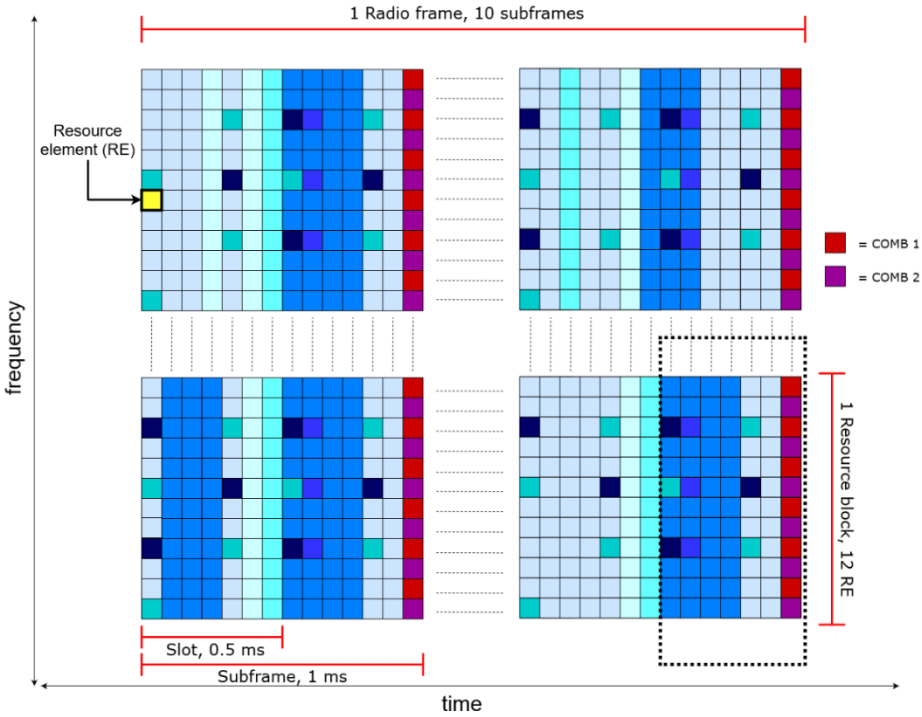

Figure 4.6. Resource grid illustration with SRS transmission.

3GPP specifies a different type of reference signals for NR and LTE which can be used as a pilot signal. One of the specified reference signals is called sounding reference signal (SRS). SRS is a comb type of pilot signal, which means that it is sent on every other subcarrier during one OFDM symbol as shown in Figure 4.6. In addition to this, a transmission using four combs is also specified. This would just simply just increase the number of sequences within the same frequency band thus shortening the length of the sequence. The resource elements highlighted in different shades of blue in Figure 4.6, can correspond to the different physical channels. In uplink these could be for example: PUSCH, PUCCH, PRACH, DM-RS and PT-RS.

SRS is constructed from constant amplitude zero autocorrelation (CAZAC) type of sequences called Zadoff-Chu sequences. These sequences feature an interesting property which allows code division multiplexing (CDM) [32]. Zadoff-Chu sequences are exploited by determining the common root sequence which is modified to form cyclically shifted versions of the same base sequence. Since these cyclically shifted versions of the base sequence are orthogonal with each other, it is possible to multiplex several users for the same frequency resources using the same root sequence [33]. The formula for the base ZC-sequence and the cyclic shifts is given in Section 5.5.3.1 in [34]. If the base sequence is $X = (X_0, X_1, \dots, X_{N-1})$ the modified sequences for up to eight transmit antennas in frequency domain for uth user on kth subcarrier are calculated as follows: [33]

$$X_{u,k} = X_k e^{\frac{j2\pi k m_u}{8}}, m_u \in \{0, \dots 7\} \tag{4.11}$$

The time domain cyclic shift corresponds to the phase shift in frequency domain, denoted by exponential term in (4.11). Furthermore, the length of the generated SRS sequence is equal to

the number of subcarriers used for the transmission i.e. one SRS symbol is mapped to one RE. Since SRS is transmitted in a comb type of pattern it could be viewed as a capacity of 16 transmit antennas per SRS instead [33].

SRS is an application of particular interest since it is used as the pilot signal for acquiring the channel state information in this thesis. On the other hand, also some other specified signal could be used for the acquisition of CSI such as demodulation reference symbol (DMRS). In TDD systems, sounding reference signals sent in uplink can be used to estimate the channel in downlink due to the reciprocal nature of the channel [35]. In addition, since the Zadoff-Chu sequences are orthogonal, they serve as an excellent pilot sequence for complexity reduction purposes.

4.5 Summary

In this chapter we discussed channel estimation. Our interest towards the channel estimation arose from the fact that the beamforming techniques that were used to improve the channel capacity and performance were completely dependent on the channel parameters. We found out that there exists a trade-off between the complexity and the accuracy of the estimation scheme. In addition to this we found out that using orthogonal pilot sequence the complexity of the estimation method was further reduced.

Since channel estimation for beamforming purposes has to be conducted very frequently due to channel aging, using low complexity scheme would help to ensure the system performance. Fortunately, we learned that beamforming does not require an exact estimate which made the low complexity schemes such as the least squares very feasible for beamforming purposes.

Despite the reduced complexity, the number of operations grew exponentially with respect to the number of antennas. This made the estimation very demanding for the large antenna configurations. However, we saw that using parallel computing it was possible to postpone the inevitable growth in the number of operations. Due to this, signal processing architectures are discussed in the next chapter in order to find out how well the benefits from the parallelism can be utilized in action.

5 ALGORITHM ARCHITECTURE MATCHING

In the previous chapter we learned that the mandatory channel estimation for beamforming introduced some additional computationally demanding operations. Although we found that the complexity could be reduced with certain solutions, even the complexity of simple methods was dependent of the size of the antenna configuration. This made estimating large antenna configurations challenging. In this chapter we explore the properties of the different architecture options for the implementation of the channel estimation algorithm. Because parallel computing was found to be an effective way to combat the performance degradation caused by the large antenna configurations, we focus on the efficient parallel architectures and speculate their feasibility for given problem. Although it is said that high performance is possible to achieve with efficient hardware, which exploits high level of parallelism [36], we will learn that exploiting parallelism can prove to be challenging.

We begin this chapter by briefly inspecting the efficiency and basic properties of the common implementation technologies. Since we are interested in efficient parallel architecture, the nature of the hardware accelerator-based solutions is speculated. The implementation was conducted using a digital signal processor specialized to vector processing. Inspired by this, we review the common ways to exploit parallelism in processor architectures from pipelining to the instruction and data level parallelism. We end this chapter with a quick discussion about exposed data path architectures. In a way, these architectures embody the methods to improve efficiency presented in the context of the both implementation technologies and architectures making them intriguing options for more detailed future investigation.

5.1 Implementation Technologies

Implementation technology can significantly affect the efficiency of the solution. By implementing the same architecture or functionality using different implementation technologies, the resulting efficiency can differ between these two implementations. Sometimes the properties of the implementation technology can have so significant impacts that they have to be considered during the system design. Since efficient implementation was the objective of this thesis, only differences in the efficiency of the different technologies are discussed. The main emphasis is on the comparison of application specific integrated circuit (ASIC) and field programmable gate array (FPGA). A well-known distinction between these implementation technologies lies in the programmability of the hardware. FPGA features reprogrammable hardware units whereas the hardware in ASIC is static. Deriving differences between ASIC and FPGA is not too complicated task, but producing comparable results is more challenging.

A comprehensive research on the subject was conducted in [37]. Authors tried to offer a fair comparison between the mediums by identifying the unjust aspects from the previous researches. The research was conducted by implementing several benchmark designs using both FPGA and standard cell. The authors compared the delay, area and power consumption of the benchmark designs. Evaluation metrics were geometric means from the results of different benchmark designs. In order to provide more detailed results, separate evaluations were conducted according to the hardware blocks that the designs used.

The area consumption of FPGA benchmark circuits was considerably higher than ASICs. If FPGA circuit used only generic logic components its area consumption was on average 35-fold higher than equivalent ASIC. However, the significance of the hardware components of higher granularity was noticed. If the benchmark design was able to utilize hardware blocks such as

multipliers and accumulators the gap was reduced to less than 10-fold. According to the delay measurements, FPGA entailed three times worse critical path delay, thus offering slower maximum clock frequency. Also, the dynamic power consumption of FPGA-based benchmarks was also significantly higher. Although it was mentioned that FPGA could improve its results if its resources were used as efficiently as possible, the results speak in favour of the efficiency of the ASIC designs [37].

Table 1. Comparison table for implementation technologies[1]

Metric	FPGA	ASIC
Area[2]	27.5	1
Critical path delay	3.35	1
Dynamic power consumption[3]	11.775	1
Static power consumption[4]	-	-

The results obtained in [37] are summarized in table 1 by averaging over all the geometric means obtained for the different hardware block setups. Benchmark specific results with more detailed explanation can be found from [37]. Since the original results were given as a ratio FPGA/ASIC, they are directly translated to table 2 by assigning value one for ASIC results. Although ASICs provided better performance in each area, FPGAs should not be considered completely useless. High non-recurring engineering (NRE) costs of ASICs in conjunction with their long time to market time makes them expensive and risky [37]. Figure 5.1 [38] summarizes the key differences between the two introduced implementation technologies.

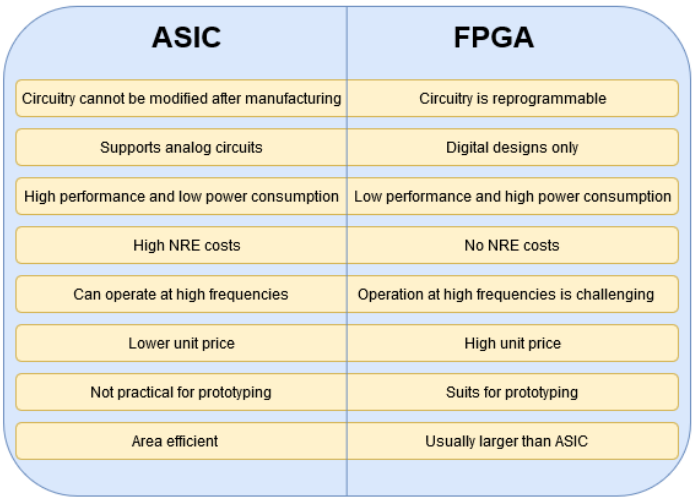

Figure 5.1. FPGA vs ASIC.

[1] If complete results are reviewed from the original paper it is noticed that in some situations FPGA can offer significantly better results than summarized here. This table mainly summarizes the fact that on average ASIC is more efficient solution.

[2] Area gap can be considerably smaller with clever choices. E.g., using hard blocks to implement functionality.

[3] Dynamic power consumption of result for FPGA is a bit unfair. FPGAs clock network is designed to much larger designs than benchmarks given here, therefore it gives relatively larger values here compared to its intended usage scenario [37].

[4] No precise conclusions could be drawn from the static power consumption. [37]

We noticed that the flexibility of the FPGA-solution was acquired at the expense of efficiency. If the gate-level reconfigurability of FPGA is traded to a reconfigurability at the operation level, the efficiency could be improved. Coarse grained reconfigurable arrays (CGRA) use this approach [39]. When implementing some functionality on an FPGA, one needs to identify the Boolean expression of the task, and the expression is built from a cluster of logic ports within look-up-tables. CGRA differs from FPGA by offering building blocks of higher granularity [39].

In CGRA, bit-level operations have been replaced by an array of functional units that perform word level operations, e.g. multiplication of two 32-bit integers. This modification leads to a reduction in area and power consumption as well as reduces the delay significantly. Although this has been achieved with reduced flexibility, the reconfigurability of the hardware has not been lost. [40] In other words, CGRA suggests that using reconfigurable blocks of higher granularity than FPGA, the gap between the metrics presented in the table 2 could be reduced. In this section, we noticed the impact of implementation technology on efficiency. There are trade-offs between efficiency and flexibility that make implementation technology selection challenging.

5.2 Hardware Accelerators

As it was already discussed in the previous chapters, modern cellular communication systems utilizing MIMO technology consists of computationally very intensive tasks with strict timing constraints. In fact, the complexity of the most intensive algorithms has exceeded the capabilities of digital signal processors already during the third-generation mobile network. This piques the interest towards hardware accelerators. [41]

We saw that efficient implementation of matrix multiplication plays an important role in achieving high performance. The benefits of parallel computing were demonstrated in Section 4.3. Parallelisation offered a way to postpone the explosive growth in the number of operations. In many situations, both processors and accelerators are able to provide sufficient data processing capabilities. However, it is said that the abilities to exploit data level parallelism in accelerators, implemented on either FPGA or ASIC, exceeds the capabilities of digital signal processors [41]. Since we have seen our application to benefit from parallel processing of data, the feasibility of hardware accelerators is speculated in this section.

In addition to parallelism, hardware accelerators can offer better throughput and power efficiency than microprocessors. Previously we discussed the efficiency of the implementation technologies. When the efficiency of hardware accelerators is compared to processors, it should be kept in mind that the comparison should be conducted on a microprocessor that implements the functionality. It is said that if some common functionality is implemented as a microprocessor-solution, it offers 100 million operations per watt. FPGAs offer a tremendous 10 to 20-fold improvement and ASICs will even increase the efficiency of FPGAs by 10-fold [42]. According to the above, the differences are significant.

5.2.1 Hardware-Software Partitioning

The splitting of workload between processor and hardware accelerator is called hardware/software partitioning [41]. Sometimes it is easy to identify the need for a hardware accelerator. Since processors can be programmed to perform almost anything, the necessity of hardware

accelerator can be identified from the absolute maximum throughput of the processor. If we think about the situation where our algorithm is expected to perform N multiply-accumulate (MAC) operations at time t. This would mean $\frac{N}{t}$ operations per second. If the available processor guarantees a latency of L clock cycles for the multiply-accumulate operation it is evident that a processor operating at clock frequency C, cannot perform better than $\frac{C}{L}$ operations per second. Naturally, if $\frac{N}{t} > \frac{C}{L}$ the processor cannot even in theory achieve the desired throughput. This raises the question of how a hardware accelerator would handle this?

An extreme opposite using a hardware accelerator would be a parallel combination of N hardware multiply-accumulate units which consists of combinational logic. In theory, this could perform the required operations in a single clock cycle whose length would be tied to the critical path delay of the circuit. The differences between the processors and the hardware accelerators are illustrated in the tables below. The time taken to process data sets of different sizes is shown in table 3. Our made-up processor uses single instruction single data (SISD) type of architecture, thus the total latency grows linearly with respect to the size of the data set. However, the hardware accelerator is composed by increasing the number of 32-bit hardware MAC units proposed in [43]. Although the accelerator can keep the latency fixed, it requires an expansion in the silicon area, which increases the estate and power consumption.

Table 2a. MAC block

Function	**Delay**	**Area**
Multiply and accumulate	$\approx 5ns$	$\approx 2000\ \mu m^2$

Table 2b. Processor A

Instruction	**Latency**	**Clock Frequency**
Multiply and accumulate	4 clock cycles	1 GHz

Table 3. Comparison of execution times

Samples	MAC block (Time/Area)	Processor A (Time/Area)
1	5ns/2000 μm^2	4ns/Fixed
50	5ns/100000 μm^2	200ns/Fixed
100	5ns/200000 μm^2	400ns/Fixed

It must be kept in mind that the above example does not directly translate to a real-world situation. However, it certainly illustrates the parallel processing capabilities of hardware accelerators. Unfortunately identifying the maximum degree of parallelism of the algorithm and implementing it as a hardware accelerator as it is does not necessarily guarantee the optimal throughput in terms of parallelism. When we are dealing with a heterogenous system consisting of several different signal processing platforms, memories and connections, there might be numerous issues which complicate the situation. For example, a spatial locality of data, should be taken into consideration. If the data is aligned poorly and the data access latency is high, the performance of the whole system will eventually be negatively impacted. Even if the operations required for the algorithm itself could be computed efficiently. [41]

In Figure 5.2, a multiplication of three by three matrix is illustrated step by step. Let's say this was accelerated using nine parallel MAC blocks which could calculate the product in three clock cycles as shown by the numbering in Figure 5.2. Furthermore, we assume that our accelerator was fed with direct memory access (DMA) that is able to transfer a block of data from consecutive continuous memory locations.

$$\begin{bmatrix} a_{11} & a_{12} & a_{13} \\ a_{21} & a_{22} & a_{23} \\ a_{31} & a_{32} & a_{33} \end{bmatrix} \times \begin{bmatrix} b_{11} & b_{12} & b_{13} \\ b_{21} & b_{22} & b_{23} \\ b_{31} & b_{32} & b_{33} \end{bmatrix}$$

$$\begin{bmatrix} a_{11}b_{11} & + & a_{12}b_{21} & + & a_{13}b_{31} & & a_{11}b_{12} & + & a_{12}b_{22} & + & a_{13}b_{32} & & a_{11}b_{13} & + & a_{12}b_{23} & + & a_{13}b_{33} \\ a_{21}b_{11} & + & a_{22}b_{21} & + & a_{23}b_{31} & & a_{21}b_{12} & + & a_{22}b_{22} & + & a_{23}b_{32} & & a_{21}b_{13} & + & a_{22}b_{23} & + & a_{23}b_{33} \\ a_{31}b_{11} & + & a_{32}b_{21} & + & a_{33}b_{31} & & a_{31}b_{12} & + & a_{32}b_{22} & + & a_{33}b_{32} & & a_{31}b_{13} & + & a_{32}b_{23} & + & a_{33}b_{33} \\ 1 & & 2 & & 3 & & 1 & & 2 & & 3 & & 1 & & 2 & & 3 \end{bmatrix}$$

Figure 5.2. Parallel matrix multiplication using MAC operations.

If the matrices were stored in memory row by row, in order to use the introduced system, the data would have to be rearranged and duplicated. If we look at the operations during the first clock cycle, due to DMA limitations, the operation cannot be performed directly in this way. The left side operands of the multiplication during the first cycle are the elements from the first column of the matrix A. Since our DMA was able to transfer data row by row this does not directly fit to the system. Although the elements from the matrix B seem to reside in the consecutive locations, two additional copies of the first row of the matrix B should be obtained.

Arranging the data in an appropriate order would naturally require extra operations which would cause a deterioration in performance. It seems that hardware accelerators can provide incredible amount of efficient parallelism. However, the hardware accelerator must be matched with the rest of the system which might decrease the performance gain from parallelism.

5.2.2 Accelerator Architectures

Hardware accelerator architectures can be divided into two different categories, for example. These categories are coarse-grain architecture and its opposite fine-grain architecture [41]. The etymology comes from the level of the granularity of the architectures. An illustration of a coarse grain accelerator architecture is given in Figure 5.3 [41]. Idea of this architecture is to offload all computationally demanding tasks to a separate accelerator unit next to the host processor [41]. As shown in Figure 5.3, the accelerator part is titled to be reconfigurable. This allows the system to be reconfigured to accelerate different types of tasks.

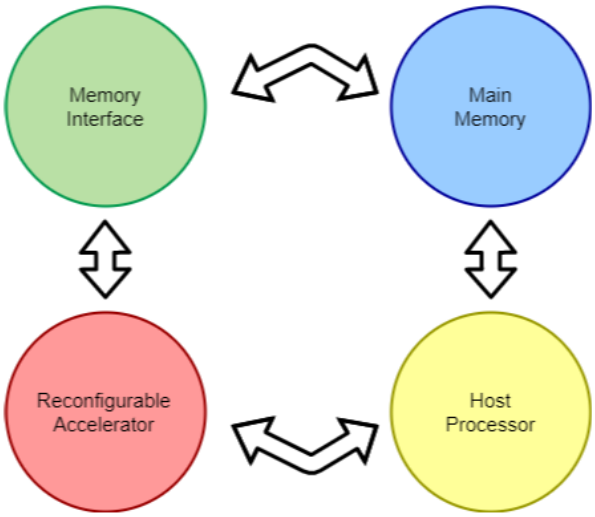

Figure 5.3. Coarse grained accelerator.

Fine-grain accelerator architectures take the opposite approach. Instead of providing a separate unit for acceleration next to the main processor, the fine-grain architecture provides instruction level acceleration by offering instructions that are tailored to perform the accelerated task. This requires modifying the instruction set architecture which can prove to be very challenging [41].

To summarize, hardware acceleration offers an energy efficient option for supporting computationally critical tasks with strict timing constraints. At least the following should be considered when identifying the feasibility of a hardware accelerator: computational complexity, spatial locality of data, data level parallelism as well as task level parallelism [41]. As discussed in this section, hardware accelerated calculations can be used to speed up the signal processing in some situations. Modern telecommunication system consists of beamforming and many other complex signal processing techniques. This requires the integration of multiple hardware accelerator and processor units into a system on a chip (SoC) solution [41]. Naturally this will increase the system cost in terms of money, space and power but clever workload partition would at least provide a faster system with better energy efficiency [41].

The channel estimation algorithm investigated in this thesis would clearly benefit from the huge parallelisation capabilities provided by the hardware accelerators. Because hardware accelerators suffer from the rigidity caused by the lack of programmability, the last sections of this chapter will deal with programmable architectures.

5.3 Programmable Architectures

Hardware accelerators were found to offer an incredible amount of parallelism, but they lacked programmability. The programmability of the processors makes them much more versatile than hardware accelerators. Since we wanted to find an architecture that offers parallelism, some parallel processor architectures are reviewed in this section. First, we take a closer look at how processors handle instructions by introducing the instruction pipeline, which is a very fundamental technique to exploit parallelism in processors. Later in this chapter, we will provide a high-level discussion of a few different parallel architectures that provide either instruction level parallelism or data level parallelism. Instead of making a detailed architecture analysis, the discussion of the parallel architectures focuses on the issues that are central to the implementation. This chapter is concluded by a brief discussion of the exposed data path architectures.

5.3.1 Instruction Pipelining

In Section 5.2 we used a generic SISD processor as an example when we were discussing about the properties of the hardware accelerators. In that example, the hardware solution improved its performance by deploying more MAC units. For the processor, no enhancements were utilized. Since the instruction latency of the processor is reported in clock cycles, the immediate idea to increase the throughput would be to make the clock cycle shorter by increasing the clock frequency. Without going into further details on why increasing the clock frequency is not a good idea, the execution rate of instructions can also be improved by other means as well [44]. In this subsection a very fundamental technique called pipelining is dealt with.

Previously we discussed the instructions without paying any attention to how they actually work and how they are composed. Figure 5.4 [45] illustrates a generic instruction format that

many processors use. Instruction is a binary string that is divided into different parts that provide the information required to perform the given task. Opcode defines the actual operation to be performed, e.g. addition of integers and the operands specify the values used for the operation specified by the opcode. If the opcode would specify an operation that required more than two operands, the instruction word would be longer than illustrated in Figure 5.4. Last part of the instruction specifies the storage address of the result. [45]

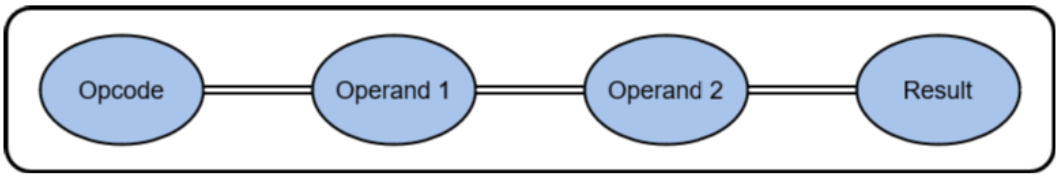

Figure 5.4. Instruction format.

The execution of a single instruction in a reduced instruction set computer (RISC) type of processor is shown in Figure 5.5 [45]. We immediately see that the instruction consists of several elementary operations where each stage takes one clock cycle to complete.

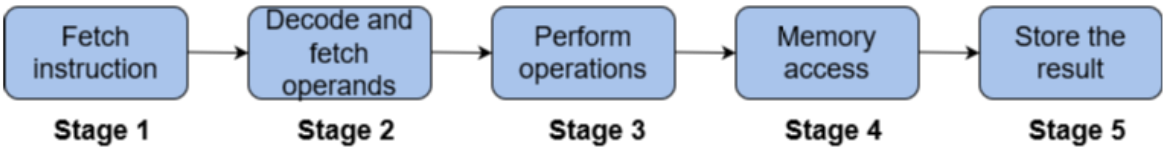

Figure 5.5. Example pipeline of stages to perform instruction.

The term instruction pipeline or execution pipeline refers to a situation, where the operations illustrated in Figure 5.5 are performed in parallel allowing new instruction to enter the pipeline every clock cycle [45]. This is possible because the different stages are able to operate independently. When only one instruction is allowed to enter the pipeline shown in Figure 5.5, it would take five clock cycles to execute the instruction. Since all stages can operate independently, when one instruction has finished stage 1 and moved to stage 2, the next instruction can enter the first stage of the pipeline.

The execution of three instructions is illustrated in Figure 5.6 [44], where I_n denotes the nth instruction and S_m stands for the mth stage. A completely serial execution is shown at the top of the figure and the pipelined execution is illustrated at the bottom part of Figure 5.6. Clearly when the instructions are pipelined and issued every clock cycle, a significant improvement in the throughput is achieved.

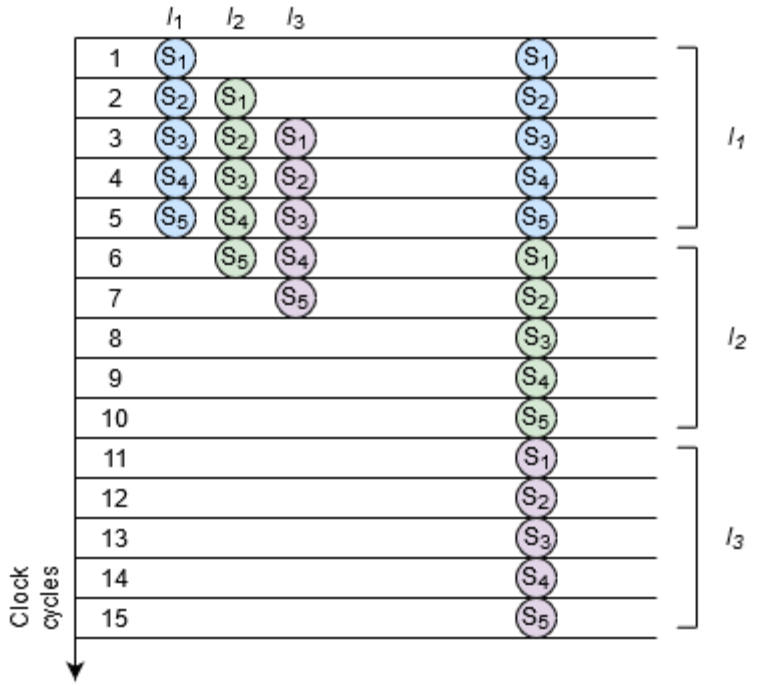

Figure 5.6. Instruction pipelining vs sequential execution.

Expression for the throughput as how many instructions can be executed within a given time unit can be derived from Figure 5.6. By observing the stages shown in Figure 5.6, it is noticed that executing three instruction takes the time taken by one instruction plus two additional clock cycles. If this observation is extended for different setups where m is the number of executed instructions consisting of n stages and the time unit to execute one stage is t the throughput $T(n)$ can be written as [44]:

$$T(n) = \frac{m}{(n + m - 1)t} \tag{5.1}$$

By letting the number of executed instructions m increase, we see that the number of instructions executed toward time unit is given by [44]

$$\lim_{m \to \infty} \frac{m}{(n+m-1)t} = \frac{1}{t}\ , \tag{5.2}$$

therefore, if one stage is processed in one clock cycle, we are able to process one instruction per clock cycle. Pipelined and non-pipelined executions are compared in table 4, where the pipeline used consists of five stages, each of which takes one clock cycle to complete. It is clear that the average execution time towards one instruction approaches to one clock cycle when pipelining making it highly recommended.

Table 4. Execution time of all instructions

Issued instructions	Pipelined	Non-Pipelined
10	14 cycles	50 cycles
50	54 cycles	250 cycles
100	104 cycles	500 cycles
500	504 cycles	2500 cycles

The hardware implementation of pipeline is built in a manner which does not reflect its pipelined structure to the instructions. This means that there is no explicit reference to the pipeline in the instructions. While pipelining provides enhanced throughput with hardware its existence is still good to realise since there can be some dependencies between the instructions or data. [45] For example if instruction I_2 uses a result from I_1 as an operand, by observing Figures 5.5 and 5.6, we notice that when I_2 is decoding the instruction and the fetching operands, I_1 has not yet stored its results to the place where I_2 is looking for them. This situation causes a stall in the pipeline which naturally decreases the performance. Stalls in the pipeline can also occur for example when external storage accesses are performed or branch to a new location occurs. [45]

5.3.2 Instruction Level Parallelism

In the previous section dealt with pipelining. Though the execution stages were processed in parallel, it was possible to issue only one instruction per clock cycle. If we assume that t in Equation (5.2) would be less than 1 clock cycle, clearly the throughput would be further improved. However, if we assume that the operation of logic circuitry is synchronized to a clock signal it is not valid to make the above assumption and something else needs to be considered.

If we want to increase the performance even further, we could issue several instructions simultaneously during one clock cycle. This sort of parallel execution of instructions is possible with the architectures that offer instruction level parallelism (ILP). The operating principle of these multi issue processors compared to a conventional scalar processor using pipelining is illustrated in Figure 5.7 [44]. In principle, a multi-issue architecture provides a significant improvement in the execution rate of instructions. In the previous subsection we were able to reach the rate of one cycle per instruction (CPI). If we are able to issue multiple instructions simultaneously CPI would reduce in the same proportion. [44]

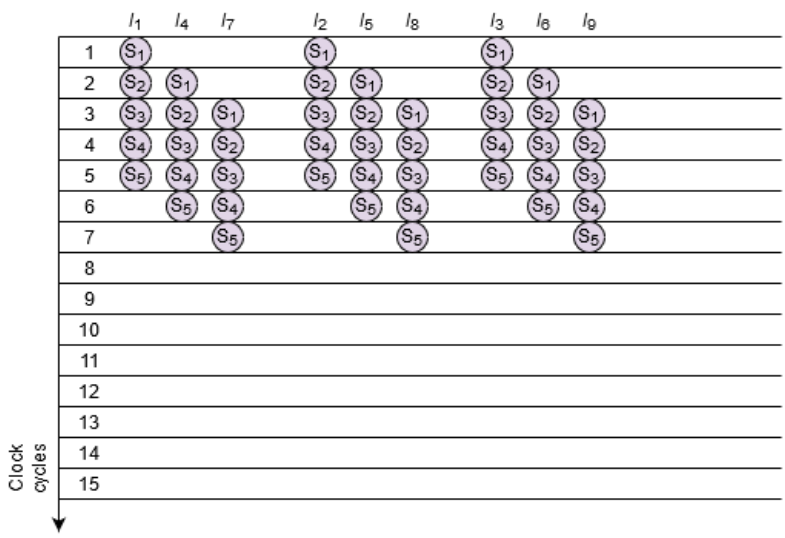

Figure 5.7. Pipelined multi-issue architecture.

Increasing the parallelism by using instruction level parallelism poses some practical challenges. There can be dependencies between the instructions and the data. These dependencies

must be identified in order to make everything operate smoothly. Multi-issue architectures commonly exhibit two different scheduling strategies. These strategies are static scheduling and dynamic scheduling strategies. [44]

Static scheduling strategy is used in an architecture called very long instruction word (VLIW) architecture. The fundamental idea of VLIW architectures is to let the compiler run the instruction dependency analysis for some given source code. During the compilation, the compiler analyses the source code and identifies the dependencies between the instructions. The compiler groups the instructions to be executed in parallel into one very large instruction word. When this very large instruction word is issued, multiple different instructions for different data are executed in parallel as was shown in Figure 5.7. [44] The dynamic scheduling strategy is found from a different architecture called Superscalar architecture. In practice, Superscalar architecture differs from VLIW by handling the instruction scheduling at the hardware level using special hardware instead of the compiler [44].

Different scheduling strategies will require a different type of hardware implementation. A processor implemented with Superscalar architecture performed the scheduling using specific hardware. In general, Superscalar processors will have more complex hardware than VLIW processors. Since the complexity of the dependency analysis grows quadratically with respect to size of the instruction word, one might argue that this might affect to the efficiency of the solution [46]. In fact, it is said that the VLIW architecture can offer the same performance as the Superscalar. However, this is achieved with a reduced hardware complexity [47]. The key differences between the architectures are summarized in table 5. Since the exhaustive scheduling work is performed by the compiler in VLIW architectures, it also means that the compiler design is more difficult, and the portability of the code is harder.

Table 5. Multi-issue architecture comparisons

Metrics	Superscalar	VLIW
Issued instructions M	M > 1	M > 1
Hardware complexity	High	Low
Compiler complexity	Low	High

Since it was said that both architectures can provide the same performance, the VLIW architecture sound more intriguing option due to the lower hardware complexity. Despite the bad portability of VLIW code, it would be the architecture of choice due to the efficiency aspect. Since the implementation in this thesis is not a commercial application targeted for several different types of processors, the code does not need to be very portable in that sense.

Naturally the feasibility of instruction level parallelism depends very much on how it is been implemented. If the architecture allows several complex arithmetic instructions to be grouped into one very long instruction word, it would significantly enhance the performance of the algorithm. Since we have a lot of data to process, by combing the previous feature with parallel data operations such as load, store and move would be a sign of an appropriate instruction level parallelism. In addition to ILP, there are architectures that offer direct data level parallelism which will be shortly discussed in the next subsection. In principle, combining the VLIW architecture with a data parallel architecture would provide a very feasible architecture for our needs.

5.3.3 Parallel Processing of Data

Instruction level parallelism was found to provide a better instruction execution rate. If we look at Equation (4.10) in more detail, we notice that it consists of two arithmetic operations, multiplication and addition. Therefore, only by focusing on the useful arithmetic operations, ILP architecture would require a significant number of parallel arithmetic instructions in order to improve the calculation itself. In other words, since the matrix multiplication itself consists only of these elementwise elementary operations, we could issue a single instruction e.g. multiplication, to several data elements instead of issuing several multiplication instructions each for one data element. Coincidentally, such an architecture exists, and it is called single instruction multiple data (SIMD) architecture.

In SIMD architecture processors, a single instruction is issued for an entire vector of data. In typical SIMD architecture, this operation is implemented using a vector computation unit with vector wide register files. [48] The practical idea behind SIMD architecture is illustrated with an example which extends the MAC operations of table 2a to a SIMD architecture. Let's say we have a SIMD processor with 16 element sized vector register files. The total time taken to process a different number of data elements using the aforementioned SIMD processor and the familiar SISD processor from the previous examples is shown in table 6.

While the comparison is again somewhat straightforward, it illustrates the elementary functionality of the SIMD processor quite well. First of all, we immediately see that the time taken to process one data element is equal for the both architectures. This illustrates that SIMD architecture offers no benefit for problems without data level parallelism. Since our algorithm is shown to benefit from this type of parallelism, the SIMD architecture seems a promising option. It is evident that the SIMD architecture offers a significant improvement for the larger data sets as shown in table 6. Although this requires an assumption that all data lanes are utilized to boost up the performance of the algorithm. For example, if we processed 32 data elements each of which would depend on the result of the previous one, SIMD would offer no benefit due to the highly sequential nature of this problem. Although our algorithm was identified to benefit from this type of parallelism, exploiting the single instruction multiple data scheme is not as straightforward as in this example.

Table 6. Comparison of execution times and fundamental difference between SISD and SIMD

Data elements	SIMD processor	Scalar processor
1	4ns	4ns
16	4ns	64ns
32	8ns	128ns

Memory operations in SIMD architecture operate on vectors of data, similarly as the arithmetic instructions in the above example. Generally, a load operation is specified using a base address from which n consecutive elements are loaded into the vector register file. Since the instructions are issued for the whole vector register file, exploitation of the parallelism is highly dependent from the data alignment and the memory operations. It is highly unlikely that the vector register files could be efficiently filled from arbitrary memory locations. In order to align the data to the instructions, if the data is not already aligned in a proper vector format, memory usage might be inefficient due to necessary padding of data. [48]

An alternative option is to conduct a data permutation and reordering operations on the processor. This solution clearly introduces some overhead and can reduce the performance improvement provided by the parallel processing. [48] Previous issues strongly emphasize the fact

that the increased degree of parallelism does not directly translate to a better performance. Even if the algorithm itself would contain the required amount of data parallelism, the architecture might not be able to provide the parallelism in an optimal way.

Furthermore, although VLIW introduced a challenging instruction scheduling task, this task was handled by the compiler. The data vectorization in the SIMD architectures is considered a difficult task to offload for the compiler and is thus left to programmer's responsibility [48]. The vectorization can lead to a substantial amount of intrinsic functions. With data parallel architectures the programming efforts can be significantly higher than with a conventional scalar processor. The key parameters of the instruction and data parallel properties of the discussed architectures are summarized in table 7. In these sections we focused on the fundamental qualitative details of the discussed architectures on a general level. We highlighted the features and challenges of the architectures that were utilized or faced during the implementation of the algorithm. Naturally these architectures exhibit several other details and differences that were left out of the discussion. The differences between VLIW and SIMD are discussed in more detail in [48].

Table 7. Summary of key characteristics

Feature	Superscalar	VLIW	SISD	SIMD
Issued Instructions	M > 1	M > 1	M = 1	M = 1
Data per instruction	N = 1	N = 1	N = 1	N > 1

In principle, the SIMD architecture was found to provide a suitable kind of parallelism for our problem. However, we saw that the efficient utilization of this architecture can be challenging due to strict data alignment requirements of the vector register files. In practice, an architecture that combines the properties from the both SIMD and VLIW would sound feasible option for our algorithm. VLIW could allow instructions to be executed in parallel which in turn could help with the other mandatory operations beside the arithmetic. Pipelining the data parallel memory operations, data reordering operations and arithmetic operations would most likely offer better performance than these schemes alone.

5.4 Exposed Data Path Architectures

Earlier in this chapter we stated that some decisions can improve the energy efficiency of the signal processing platform. In Section 5.1 we saw an interesting statement that the poor efficiency of the FPGA-based solutions could be improved by utilizing a building blocks of higher granularity. In addition to this, it was said that the efficiency of the VLIW architecture was achieved by allocating more responsibility to the compiler. From the high-level perspective, the efficiency of the exposed data path architectures has indeed been improved by making use of the techniques already mentioned. By taking the hardware efficiency and the compiler workload even further, exposed data path architectures aim for remarkable energy efficiency.

At first, we briefly introduce an architecture called transport triggered architecture (TTA). TTA embodies a new programming philosophy based on data moves making it look like a programmable hardware accelerator. In research articles and papers TTA based designs have been used to implement techniques that are used in wireless communications with promising results, such as in [49]. In conclusion, we briefly introduce the Stanford ELM architecture. Publications on ELM are particularly interesting because the authors have analysed the energy consumption of the processor in more detail.

5.4.1 Transport Triggered Architecture

In [50] the author claims that the ILP architectures especially VLIW suffers from the high complexity which is considered problematic. The author introduces the transport triggered architecture based on the complexity analysis conducted in [50]. It is summarized that the improvements trade the complexity by transferring it from the hardware to software. Interestingly this is the same philosophy already seen between the Superscalar and VLIW architectures but only taken further. The fundamental difference between VLIW and TTA comes from the connections between the register files and functional units. In VLIW, every functional unit is directly connected to a register file, whereas in TTA, functional units and register files are connected to a bypass network which is used as a data transfer media. For example, size and power consumption of register file scales up with the number of ports [48], and one way to reduce complexity in TTA is to reduce the number of register file ports by connecting different elements via the bypass network [50].

A simple TTA processor is illustrated in Figure 5.8 [51]. Different functional units are connected to different components via a bypass network which is represented by the black horizontal lines. Programming of TTA processors happens by issuing a data moves to the different ports of the functional units. Each functional unit has three ports which are input, output and trigger ports. The triangles represent the direction of the port and the distinction between the input and trigger port is done using a cross inside the square. Data moves are performed by the instruction fetch unit that is capable of issuing one instruction per lane for each clock cycle. [51]

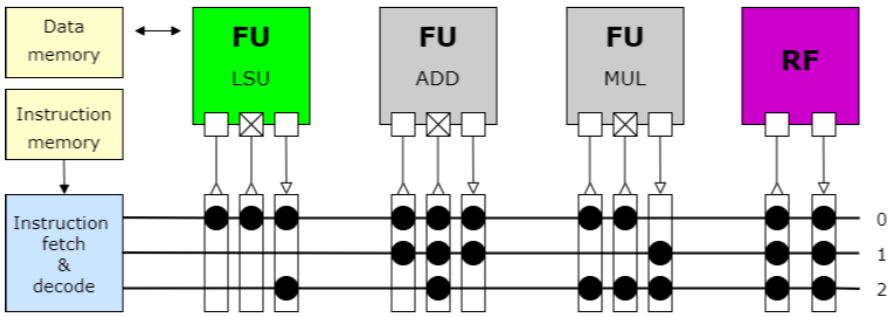

Figure 5.8. Simple transport triggered architecture. © [2020] IEEE

Execution happens when data is moved to the triggering port of the corresponding functional unit. Assuming our instruction format for the data moves is mov src, dst then, for example, adding the result of the multiplication unit to the value from the data memory could be executed as follows [51]:

```
// Executed during one clock cycle

mov LSU.out ADD.in; // Move data to the input port of the adder functional unit

mov MUL.out ADD.trig; // Move data to the trigger port of the adder functional unit

mov ADD.out RF.in; // Move the previous result from the adder to the register file
```

Since the functional units are independent, adding and removing of the units is done very easily. Therefore, TTA offers a potential platform to develop application specific instruction set processors (ASIP) due to its scalability. [52] In principle, TTA seems to continue the same trend introduced with the VLIW architecture, i.e. reduce the hardware complexity and allocate responsibilities to the compiler. TTA provides an interesting setup for the instruction scheduling because it not only schedules the instructions but also the data transfers. This naturally allows the compiler to perform a very fine grain tuning. In addition to this, though implementation technologies and architectures should not be confused with each other, TTA exhibits some similarity to CGRA in terms of the granularity offered by the functional units. This raises the question of whether CGRA could be used to implement a highly modular and efficient TTA.

5.4.2 Stanford ELM

Stanford ELM is another exposed processor architecture that also aims for a good energy efficiency. In [53] the authors of ELM analysed the energy consumption of embedded processors and found out that 70% of the energy is consumed for supplying the data and issuing the instructions, whereas only 6% of the total energy was consumed by the important arithmetic. This architecture also provides so called exposed pipeline, i.e. no hidden stages from programmer point of view during execution. This makes the instruction pipeline very different compared to the conventional RISC pipeline that was introduced at the beginning of this section. This enables an explicit management of the instructions and data movements. Once again, this creates new possibilities to gain control over the execution. [54] A dramatic improvement in energy efficiency was achieved when this architecture was compared to a processor which was derived from a commercial RISC processor. The proposed ELM architecture achieved a tremendous 23-fold energy efficiency compared to its RISC competitor. The architecture and the measurements are discussed in more detail in [54] and [53].

5.4.3 Summary

In this chapter, we saw that parallelism can be achieved in a number of different ways. We demonstrated the operation of a hardware accelerator and saw that, at least in principle, by using a hardware accelerator it is possible to achieve an immense amount of parallelism. Although the hardware accelerators provided efficient parallelism, they generally lack programmability which makes them inflexible. In addition, we saw that an enormous degree of parallelism does not necessarily improve the performance in the same proportion.

The programmable architectures introduced in this chapter provided parallelism in two different ways. Because our algorithm seemed to benefit from a large amount of data level parallelism, the SIMD architecture was found to be a particularly interesting option. Although the parallelism offered by SIMD was appropriate for our problem, exploiting it was found to be challenging. In the SIMD architectures, the instruction was executed on a vector of data located in a vector register file (VRF). This made the performance highly dependent on the vector register file and data operations. Naturally the size of the VRF sets a limit on the number of data elements in the instruction. In addition, the data must appear in the correct order within the vector register file. Since the load and store instructions can be very inflexible, SIMD is also dependent on the data alignment in the memory. By extending the SIMD architecture with VLIW, we could achieve more flexibility by pipelining successive load, store, reordering and

arithmetic instructions using instruction level parallelism obtained from VLIW. Although this would make things a bit more complex, this sort of architecture would provide useful features, above all with programmability. The exposed data path architectures provided an interesting alternative to achieve a low complexity highly energy efficient parallel architecture for future investigation.

6 IMPLEMENTATION AND EVALUATION

In this chapter we describe the implementation of the channel estimation algorithm for beamforming and present the results. Our goals were to provide an implementation that was well matched to the used architecture and to investigate the applicability of the used processor for the given task. The processor was identified as a VLIW+SIMD type architecture which made it a promising candidate for the given task.

As architectures are always implementation-specific, numerous implementations were made to gain a better understanding of the processor architecture. In this chapter we present the implementations based on the three different approaches. All three approaches were used to implement the channel estimation algorithm for different antenna configurations. Each implementation performs an estimation on the same test data that represents a fixed bandwidth. The approaches and the results are presented later in this chapter. Since the estimation method has been found to be suitable for this purpose, we only focus on the time performance of the implementation and do not evaluate the accuracy of the estimation scheme.

The implementations were written and designed using a very highly intrinsic C-programming language. Assembly language-based implementations were left out of the scope. It is reasonable to assume that the processor manufacturer provides a powerful compiler with comprehensive intrinsic functions which together are able to provide a high performance. Furthermore, we focused the investigation on the hard real-time parts of our algorithm. This adjustment made the comparison between the different implementations more feasible and it was considered in the discussion on the practicality of the solutions.

6.1 Implementation

In this section we present the different approaches that were used to implement the algorithm. The main difference between the different approaches is how they exploit the data level parallelism. As expected, the exploitation of data level parallelism proved to be challenging. In our processor, a large proportion of data level parallelism was distributed across the different independent cores. This made the already rigid SIMD architecture even more rigid. In addition to the previous, memory accesses and consequent data reordering also formed a significant difference between the approaches. Besides the differences, all approaches tried to utilize the necessary and most powerful architecture-specific instructions.

The exact architectural details of the processor and approaches will not be exposed in this section. The reason for this is that some of these approaches or the used processor could potentially be used in a future Nokia system. Since the efficient exploitation of data level parallelism is highly system dependent, to avoid exposing too detailed information about the system the discussion on the topic is limited.

6.1.1 Approach A

Approach A implemented the algorithm in two separate functions. Both functions exploited the data level parallelism for the matrix multiplication by adapting to the data alignment. By adapting to the data alignment, we mean that no overhead was introduced due to the data reordering that was discussed in the previous chapter. While this method avoided the additional data rearrangement operations, the number of useful arithmetic operations was not as small as possible.

Approach A performs the matrix multiplication in the same manner as the approach C as show in Figure 6.2.

Due to the system specific and architectural reasons Approach A could not fully utilize the SIMD capacity since it was divided into two separate tasks. Because the implementation of the algorithm in two separate parts is not necessary, this method introduces excess load and store operations compared to the other two approaches. In theory, two separate tasks could be pipelined using multiple units of the given processor. If this was the case, the execution time would approach to the cycle count of the slower one.

6.1.2 Approach B

Because the disadvantages of Approach A were the partial usage of the SIMD capacity, excess load and store operations and inefficient matrix multiplication, Approach B was designed to tackle these issues. When Approach B was designed, we encountered the exact same challenges that were discussed in the context of SIMD in the previous chapter. Efficient data accesses proved challenging in Approach B. Although the time required to arrange the non-real time data was excluded from the operation, operations with the real time data caused issues.

The matrix multiplication of Approach B is shown in Figure 6.1. We notice that the vectors contain copies of the same matrix elements. It is also noticed that none of the vectors consists of consecutive matrix elements. Since the data access mechanism of our processor was very similar to the one introduced in Section 5.2, this approach required a lot of data rearrangement operations which in turn reduced the benefits gained from the efficient matrix multiplication. Approach B is a real-world example of the challenges discussed in the previous sections.

Approach B made full use of the available SIMD capacity thus making the most of the data level parallelism. Additionally, it implemented the algorithm within one function, so unnecessary load and store operations were avoided. In a nutshell, this approach tried to minimize the number of operations required for the matrix multiplication. Although in terms of iterations this approach was able to provide a massive reduction of 62.5%, we will see from the results that a greedy strategy like this is not the best one.

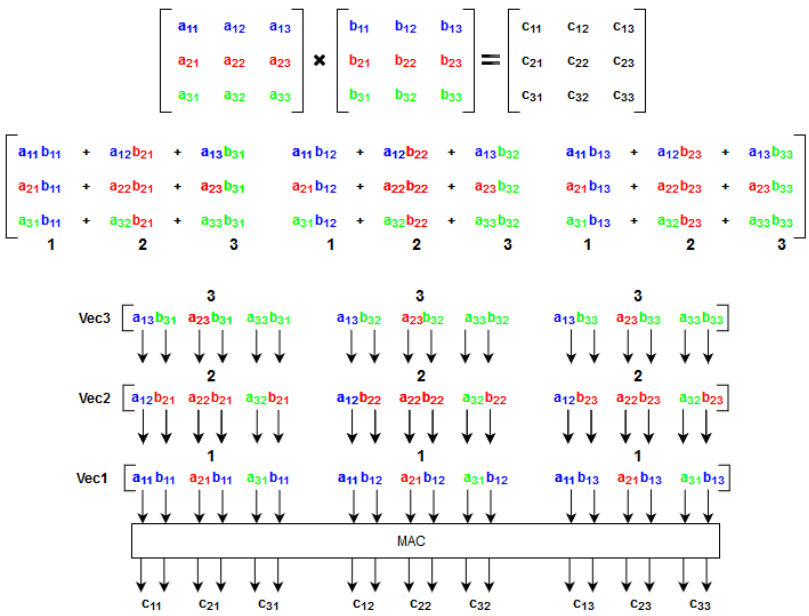

Figure 6.1. Illustration of approach B.

6.1.3 Approach C

So far, we have seen two very different approaches. Approach A tried to adapt to the prevailing system state, but it did not pay enough attention to the specific features of the architecture. On the other hand, Approach B greedily tried to get rid of the inefficiencies of Approach A. We will see from the results that the radical idea behind Approach B cannot offer the best performance thus Approach C is introduced.

Approach C can be seen as a fusion of Approaches A and B. It combines the strengths of the both approaches. Because the inefficient use of the registers by Approach A had been criticized, Approach C was implemented in a single function to avoid the inefficiencies. Since the matrix multiplication used in Approach B was proved to be challenging, Approach C utilized the same technique as Approach A. As shown in Figure 6.2, an extra operation is introduced compared to Approach B, but the data happened to be in more suitable order thus no re-ordering operations were required. Although Approach C is using the same matrix multiplication technique as Approach A, it is still able to utilize the full SIMD capacity due to the other changes. In a nutshell, the approach C tries to effectively benefit from the architecture specific properties while still considering the limitations imposed by the system.

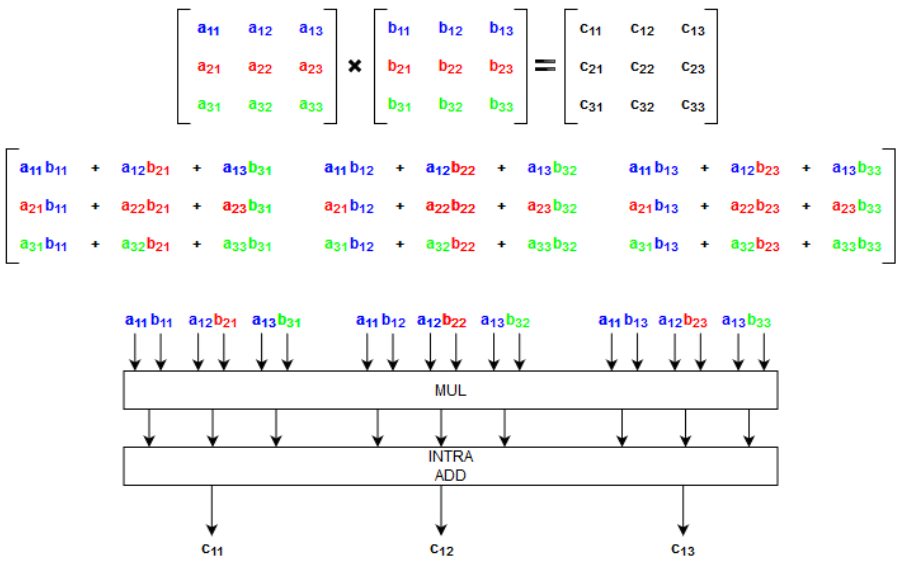

Figure 6.2. Calculation of the first row using approach C.

6.2 Evaluation

In the previous chapters we have emphasized the time-critical nature of the system, therefore the evaluation will focus on the cycle count comparison. At first, we compare how well the different approaches perform when evaluated against antenna configurations of varying sizes. Next, we compare how the performance of individual approaches changes between the different antenna configurations. Measurements were performed on antenna configurations consisting of eight transmitter antennas and two, four or eight receiving antennas.

The cycle count readings were obtained using an integrated development environment (IDE) that provided a simulation environment with a comprehensive profiling tool. This IDE was

provided by the processor manufacturer. The compiler, which was also provided by the processor manufacturer, offered different optimization levels. If the lowest optimization was used, the compiler did not form any VLIW-instructions at all. Therefore, an evaluation of non-optimized code was considered meaningless and all the comparisons were conducted using the second highest optimization level. In addition to the implementations presented in this chapter, implementations were made for numerous other antenna configurations as well. These results have been omitted because they did not provide any additional findings and the best performance can be found among the implementations shown below.

6.2.1 Elementary Comparison

Before we evaluate the different approaches, it is good idea to make sure that the used processor is suitable for this problem. The processor was evaluated using Approach B to implement the channel estimation algorithm for an antenna configuration of one receive and eight transmitting antennas. The cycle count reading obtained from this implementation was compared to a non-vectorized implementation which was compiled and executed using our processor and some i5 processor. It should be kept in mind that the purpose of this comparison is to confirm the feasibility of our processor and by no means to prove another as unsuitable.

Results for the elementary comparison are presented in Figure 6.3. The results are presented in relation to the fastest implementation. Results look promising since the vectorized implementation of Approach B on our processor achieved a 75-fold and 202-fold cycle count performance compared to the other ones. It is very likely that this algorithm could achieve a significantly higher performance on i5 processor, if it was optimized for it. However, the goal of this comparison was to ensure that the processor we used did not perform exceptionally poorly. For this purpose, the results seem promising. It is worth mentioning that even i5 could outperform the used signal processor, commercial processors introduce undesirable features such as very bad energy efficiency and high price.

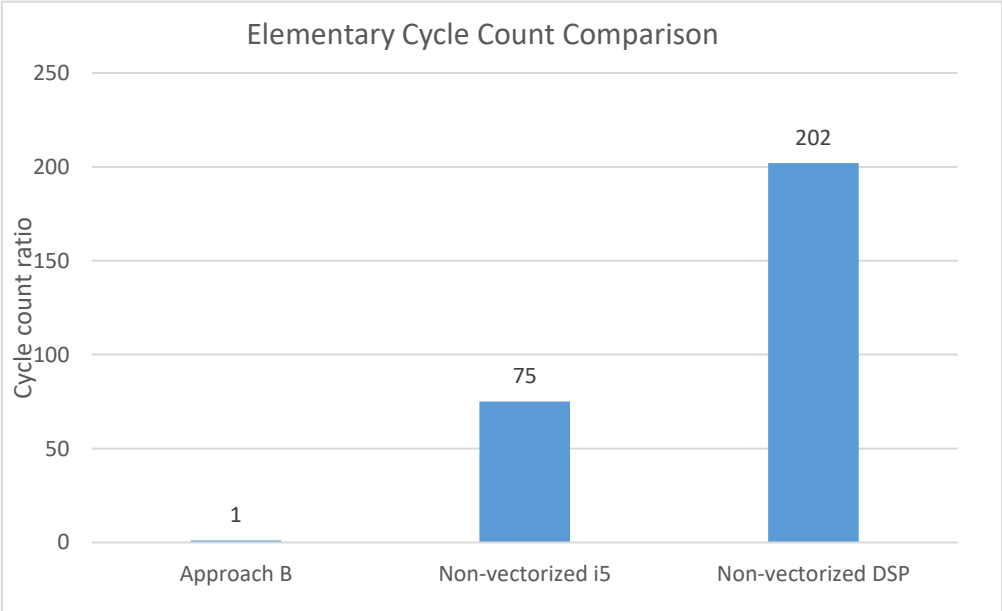

Figure 6.3. Performance comparison between different platforms.

6.2.2 2RX Comparison

In this subsection we evaluate the implementations for two receive antennas. The results for each approach are given as a cycle count ratio with respect to the fastest implementation as shown in Figure 6.4. We see that the implementation using Approach B offered the best performance while the one using Approach A offered the worst one. If Approach A cannot be executed in parallel, it clearly offers the worst performance. Even if it was pipelined according to the previous assumptions, its performance would still be the worst, not to mention that the above style would use twice as much hardware resources.

Although Approach B required some data re-ordering operations, it seems that they have not built up too much overhead with respect to the gains achieved from the fast matrix multiplication. It is still noteworthy that Approach B is very picky with the non-real time data. For this comparison, the non-real time data was provided already in the optimal form. As a result, Approach B at least partially borrows its performance from elsewhere making it less attractive.

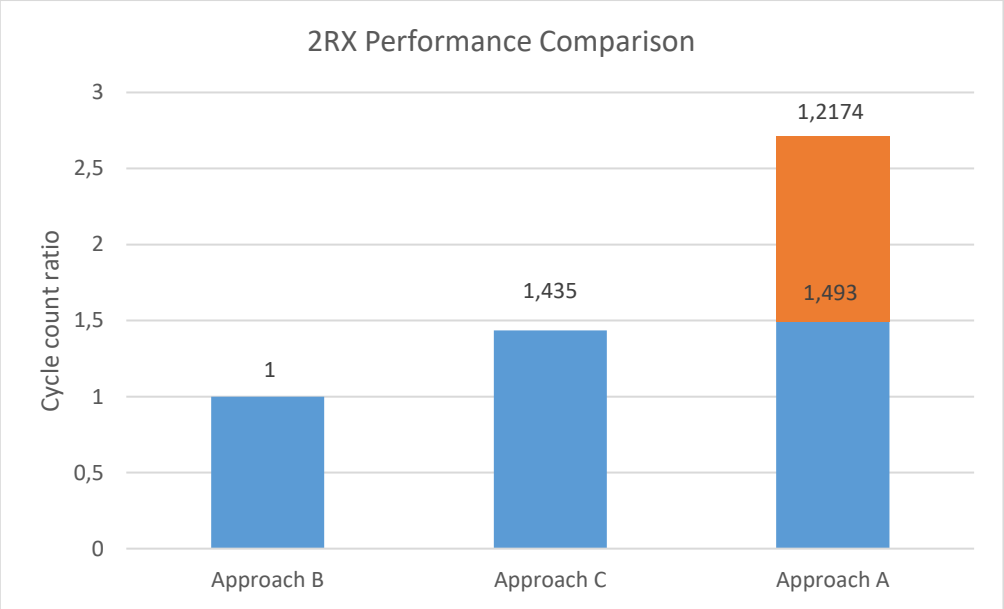

Figure 6.4. Cycle count comparison of 2RX implementations.

6.2.3 4RX Comparison

The same measurements as above were also performed for the antenna configuration comprising four receive antennas. Results for these implementations are presented in Figure 6.4. We see that the implementation using Approach C has performed better than the one using Approach B. This is a positive result due to practical reasons since we criticized Approach B for its ambiguity. Most likely the main reason for the better performance of Approach C is that it does not involve the data rearrangement whereas Approach B requires. The mandatory data reordering in Approach B benefitted from the instruction level parallelism. This result indicates that the compilers ability to produce efficient VLIW instructions is impaired. Because the number of operations and data increases as the number of antennas increases, if the reordering operations cannot be efficiently parallelised, performance will begin to decline significantly.

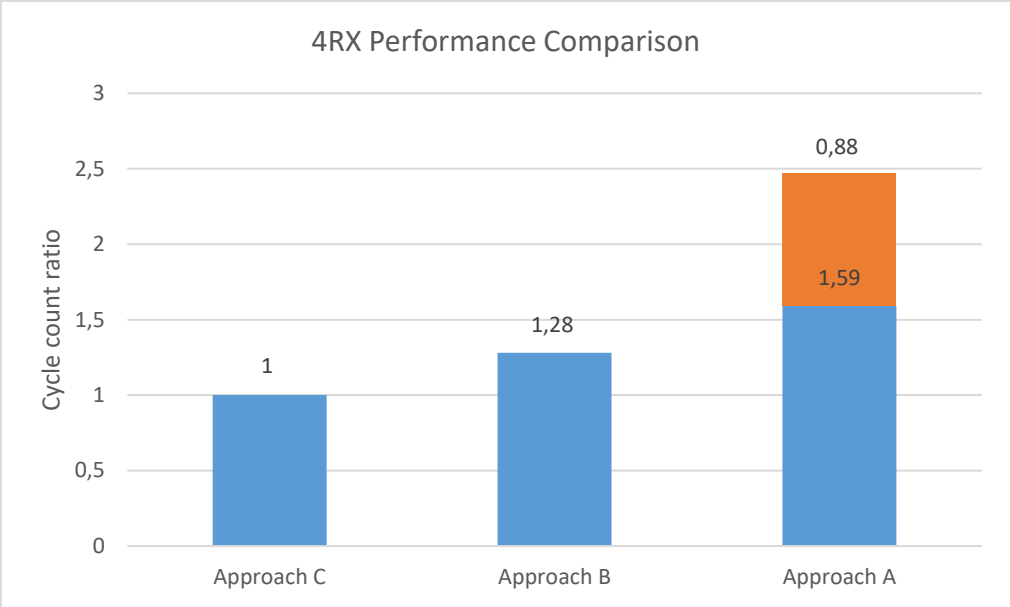

Figure 6.4. Cycle count comparison of 4RX implementations.

6.2.4 8RX Comparison

The last comparison was conducted for the eight receive antennas. Again, identical measurements were performed as before, and the results are shown in Figure 6.5. Clearly, the order has not changed from the previous figure. This is a positive result because Approach C was considered the most practical solution of all, while Approach B was considered the most impractical. Interestingly, the difference between the approaches B and C has increased compared to the previous figures. This confirms the assumption made in the previous subsection that Approach B does not parallelise as efficiently as the others. Approach B is a good example of the challenges addressed in the previous chapter. We were able to utilize the parallel resources for the efficient matrix multiplication, but these benefits were negated due to time consuming data reordering and access operations. These results strongly suggest that it is important to pay attention to the limitations posed by the system and not just focus on identifying the ability of the algorithm to benefit from parallelism.

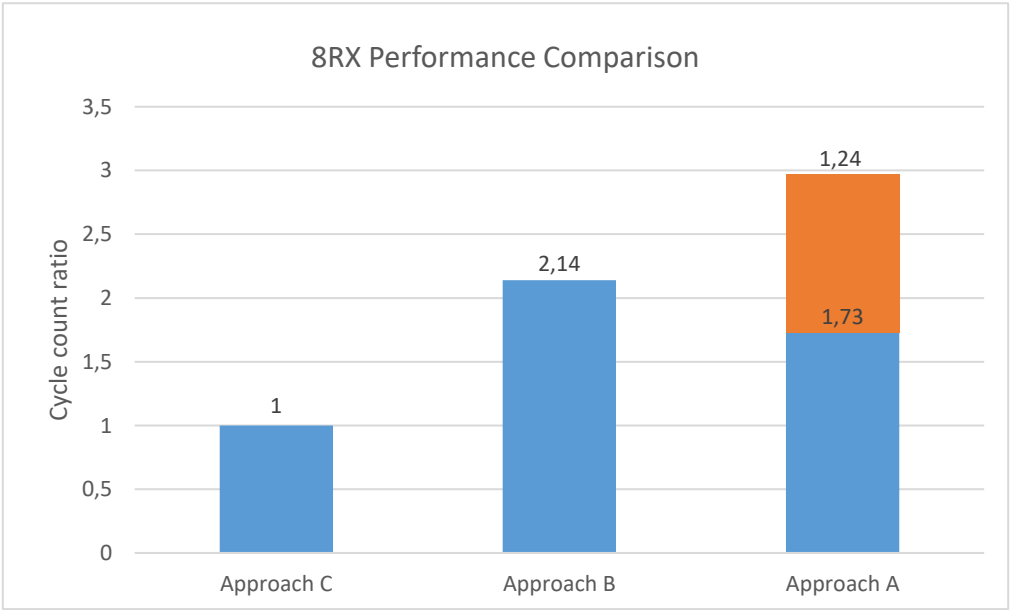

Figure 6.5. Cycle count comparison of 8RX implementations.

6.2.5 Implementation Specific Comparison

In this subsection we compare how the performance of single approach changes between the different antenna configurations. The objective of these measurements was to evaluate how well parallelism could be utilized as the number of antennas increased. Since these measurements indicate how the execution time changes with respect to the number of antennas, these comparisons will provide useful information about the processors capabilities to handle larger antenna configurations. One of the main goals of this thesis was to evaluate the functionality of this processor for different antenna configurations. The results given in this subsection provide an approach specific behaviour for different antenna configurations which reflects the processors ability to handle large antenna configurations. In addition to this, the implementation with the best performance is also found out.

The performance of implementations using Approach A is shown in Figure 6.6. Because the difference in the number of operations required between adjacent implementations is twofold, the results for the implementations using Approach A suggests that the processors resources are most efficiently utilized with a configuration of four receive antennas.

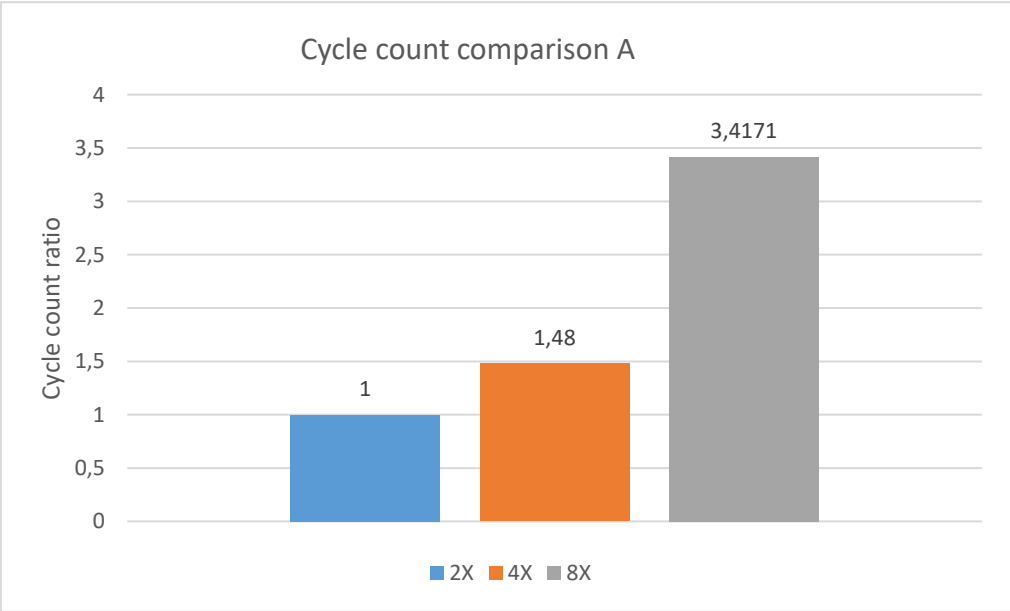

Figure 6.6. Comparison of cycle counts between different antenna sizes in approach A.

A similar comparison for the implementations using Approach B is provided in Figure 6.7. The increase in the number of clock cycles is enormous compared to the previous one. Despite the best performance for the smallest antenna configuration, its performance is significantly decreased for the larger antenna configurations. Approach B is not able to exploit parallelism as efficiently as the others. This development was no surprise since the fast degradation in the performance of Approach B was already perceived in the earlier comparisons. Executing the two-antenna implementation sequentially would achieve an identical cycle count with the four-antenna configuration. This suggests that the performance degradation using this approach starts already after the smallest antenna configuration.

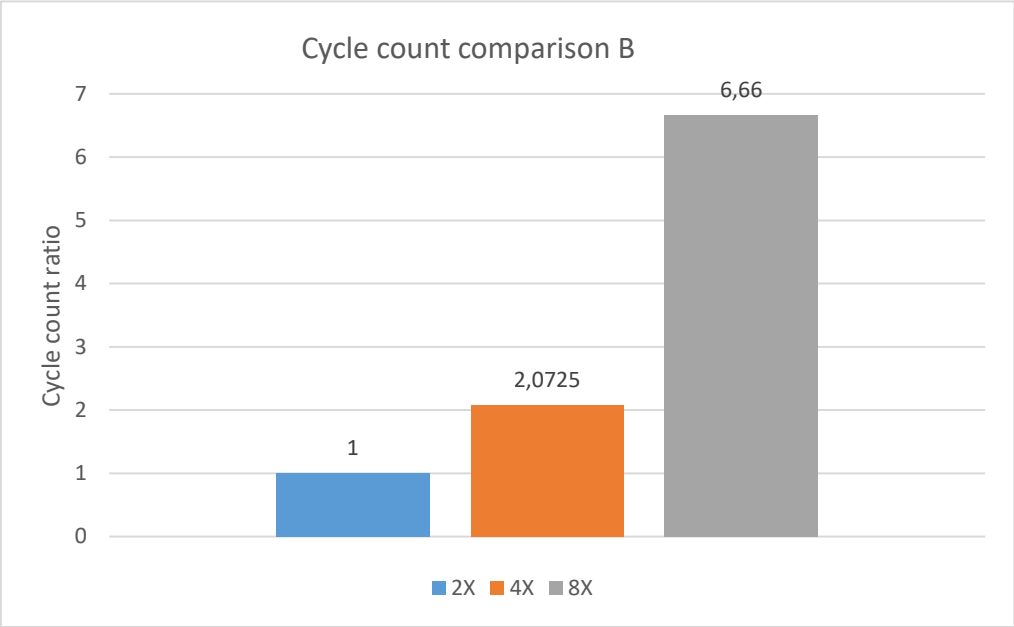

Figure 6.7. Comparison of cycle counts between different antenna sizes in approach B.

In conclusion, the results for the last implementations are given in Figure 6.8. The implementation for the four receive antenna configurations represents the best match with the architecture. Although the number of operations has doubled, the performance has barely changed. The transition from the four receive antennas to eight receive antennas implies that the most efficient utilization of the processor resources had already been achieved earlier. Approaches A and C exhibit very similar behaviour when transitioning from four receive antennas to eight.

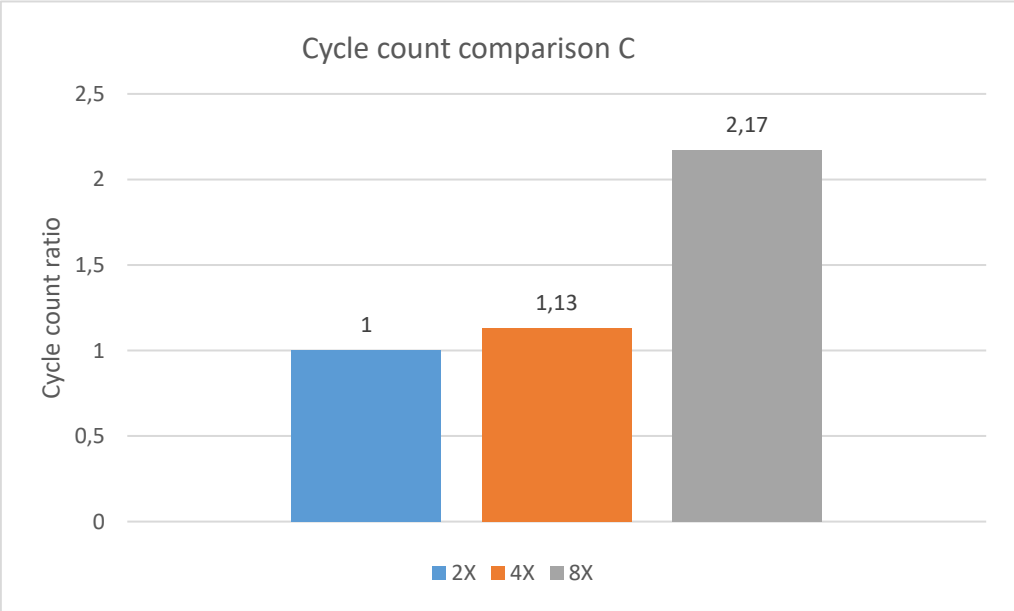

Figure 6.8 Comparison of cycle counts between different antenna sizes in approach C.

6.3 Summary

In this chapter we presented three different implementation approaches and used these approaches to implement the channel estimation for different antenna configurations. The results show that by greedily minimizing the operations required for the matrix multiplication does not lead to the best result. This was because the data were not in the appropriate order for this approach. The data reordering operations introduced overhead which began to grow significantly with larger antenna configurations.

The best performance was achieved by fully utilizing the data level parallelism of the processor and adapting the implementation according to the availability of data. Although this approach introduced some additional arithmetic for the matrix multiplication, efficient data accesses without extra overhead guaranteed the best performance.

Furthermore, the results provided useful information on the suitability of the processor for larger antenna configurations in the future. Results of the approach C show that the best benefit from parallelism was obtained with four receive antennas. If the implementation for 16 receive antennas was conducted, the difference in the time performance would be significantly higher than it was with the previous antenna configurations. Efficiency of the implementation deteriorates for larger antenna configurations which poses a challenge for the future systems.

7 DISCUSSION

In Chapter 4 we stated that the time performance of the channel estimation could be improved by using a parallel architecture. We explored some parallel architectures in Chapter 5 and came to the conclusion that suitable parallel architectures exist but it is very unlikely that they could be utilized as efficiently as we demonstrated in the fourth chapter. Due to the challenges discussed in Chapter 5, several implementations using slightly different approaches were implemented, in order to find out the best way to exploit the architecture specific properties.

It can be said that the results reflected the prior assumptions quite accurately. We suggested that efficient data accesses play an important role in achieving high performance. The best performance was achieved when the implementation considered the system and architecture specific limitations and properties. If the implementation greedily tried to minimize the number of required arithmetic operations for the matrix multiplication itself, the performance degraded very quickly for larger antenna configurations. To sum things up, the best performance was achieved by exploiting the available data level parallelism by adapting the operations to the system specific limitations and condensing the number of required operations by efficiently utilizing the architecture specific properties.

At this point we can delimit the discussion to Approach C since it clearly offered the best performance. It is evident that the best performance was achieved when 4RX-antennas were processed. Although it was slightly slower than the 2RX version, the increase in the total cycle count with respect to the increase in the number of operations was negligible. The transition from four to eight receive antennas introduced significantly larger difference in the time performance than the transition from two antennas to four antennas. In fact, both the number of receive antennas and cycle counts doubled. This suggests that the available parallel resources of this processor were completely utilized. If this processor is used in conjunction with Approach C, problems will emerge for large antenna configurations in the future.

Although we faced some challenges with parallelism using Approach B, it does not mean that this algorithm would not benefit from the increased degree of parallelism. Approach C, for example, could directly benefit from a three times higher degree of parallelism. In addition to this, by increasing the available instruction level parallelism by providing duplicates from the already existing instructions would further enhance the time performance of the algorithm.

As a summary, the proposed platform offers suitable features in conjunction with decent amount of data and instruction level parallelism. These features seem to be adequate for the scenarios seen in this thesis. However, the results show that the processors capabilities are already fully utilized. For the future massive MIMO systems, the signal processing platform for the given task should be reconsidered.

If the current scheme is also utilized for the channel estimation in the future where massive antenna arrays cause channel matrices to be enormous, several other things could also be reconsidered. If the matrices are large enough, one approach could be to investigate the suitability and the system requirements for applying some matrix multiplication algorithm for solving this problem. Another option would be to increase the parallelism provided by the architecture. A hardware accelerator or a TTA-based processor could also be a viable option for achieving parallelism. These options could potentially offer more parallelism and energy efficiency, which is also a particularly important parameter in the current and future networks.

There are digital signal processors on the market that features up to 128 MAC-units. This type of processor could be an interesting option for the future. Determining this would, of course, require a comprehensive investigation. Furthermore, the whole channel estimation method could be reconsidered. Channel estimation could also be a potential topic for a machine

learning application. Instead of estimating the channel from inserted pilots it might be possible to train a neural network for recognizing the channel conditions from the pilots. Neural network applications would also be a very interesting research topic and they could also benefit from the type of processor mentioned above.

8 SUMMARY

In the future, the current antenna configurations are expected to be replaced by configurations with an even greater number of antenna elements. Rapidly growing number of the antenna elements will set challenges towards the implementation of channel estimation. The results show that the best efficiency was already achieved with the configuration consisting of four receive antenna elements. This observation suggests that more parallel resources are required if the desire is to process larger antenna configurations efficiently.

Implementing the current channel estimation algorithm with novel architecture is not a straightforward task. Parallel architectures are programmed using highly intrinsic code which is most likely not portable between two architectures. Transition to a new architecture would naturally require a significant amount of programming work. Alternatively, the current architecture could be efficiently utilized by dividing the large antenna configurations into smaller groups of four antennas. This approach would enable the reuse of the existing code however, the number of processors would be higher. If a new architecture is considered, an extensive investigation should be conducted between the architectures in order to identify the most efficient solution. In addition, the whole estimation scheme could be reconsidered. The suitability of neural networks and other machine learning techniques for channel estimation would be an interesting topic for future research.

We saw that the performance of the channel estimation could be improved by using parallel architecture. Although, utilizing parallelization in large signal processing systems can prove to be challenging. Nevertheless, the amount of parallel resources must grow in the future if the current channel estimation method is desired to be implemented efficiently in large antenna configurations.

9 REFERENCES

[1] Gupta A. & Jha R. K. (2015) A Survey of 5G Network: Architecture and Emerging Technologies. IEEE Access (Volume: 3), p. 1206-1232

[2] 3GPP The Mobile Broadband Standard (Accessed 3.2.2020), LTE-Advanced, URL: https://www.3gpp.org/technologies/keywords-acronyms/97-lte-advanced

[3] Silvén O. & Jyrkkä K. (2007) Observations on Power-Efficiency Trends in Mobile Communication Devices. EURASIP Journal on Embedded Systems, p.1-10

[4] 3GPP TS 22.261 V17.1.0 (2019-12) Service requirement for the 5G system (Release 17)

[5] 3GPP (Accessed 14.1.2020) About 3GPP Home URL: https://www.3gpp.org/about-3gpp/about-3gpp

[6] 3GPP TR 21.915 V15.0.0 (2019-09) Summary of Rel-15 Work Items (Release 15)

[7] ITU-R Radiocommunication Sector of ITU (2015) Recommendation ITU-R M.2083.0 (09/2015) IMT Vision – Framework and overall objectives of the future development of IMT for 2020 and beyond

[8] Boccardi F., Heath R. W., Lozano A., Marzetta T. L. & Popovski P. (2014) Five disruptive technology directions for 5G IEEE Communications Magazine (Volume:52, Issue: 2, February 2014), p. 74-80.

[9] Andrews J. G., Buzzi S., Choi W., Hanly S. V., Lozano A., Soong A. C. K. & Zhang J. K. (2014) What Will 5G Be? IEEE Journal on Selected Areas in Communications (Volume: 32, Issue: 6, June 2014). p. 1065-1082

[10] Pozar D. M. (2012) Microwave engineering. Wiley 4th ed.

[11] Shannon C. E. (1949) Communication in the Presence of Noise, Proceedings of the IRE (Volume: 37, Issue: 1, Jan. 1949). p. 10 – 21

[12] Roh W., Seol J-Y., Park J., Lee B., Lee J., Kim Y., Cho J., Cheun K. & Aryanfar F. (2014) Millimeter-wave beamforming as an enabling technology for 5G cellular communications: theoretical feasibility and prototype results. IEEE Communications Magazine (Volume:52, Issue:2, February 2014) p. 106-113

[13] Khan F. & Pi Z. mmWave mobile broadband (MMB): Unleashing the 3-300GHz spectrum. 34th IEEE Sarnoff Symposium.

[14] Khan F. & Pi Z. (2011) An introduction to millimetre-wave mobile broadband systems, IEEE Communications Magazine (Volume:49, Issue:6, June 2011) p. 101-107

[15] Rappaport T. S., Sun S., Mayzus R., Zhao H., Azar Y., Wang K., Wong G. N., Schulz J. K., Samimi M. & Gutierrez F. (2013) Millimeter Wave Mobile Communications for 5G Cellular: It Will Work! IEEE Access (Volume: 1) p. 335-349

[16] Goldsmith A. (2005) Wireless Communications. Cambridge University Press

[17] 3GPP TS 38.201 V15.0.0 3GPP TSG RAN NR Physical layer; General description (release 15)

[18] Keysight Technologies, Inc. (Accessed 22.1.2020) Concepts of Orthogonal Frequency Division Multiplexing (OFDM) and 802.11 WLAN URL: http://rfmw.em.keysight.com/wireless/helpfiles/89600b/webhelp/subsystems/wlan-ofdm/Content/ofdm_basicprinciplesoverview.htm

[19] Molisch, A. F. (2011) Wireless communications, Chichester Wiley 2011.

[20] Väisänen N. (2018) Beamforming Techniques for Optimizing Channel Capacity in Wireless Communications.

[21] Tse D. N. C. & Zheng L. (2003) Diversity and multiplexing: a fundamental trade off in multiple-antenna channels. IEEE Transactions on Information Theory (Volume:49, Issue: 5, May 2003) p. 1073-1096

[22] Winters J. H. (1998) Smart Antennas for Wireless System, IEEE Personal Communications

[23] Heath R. W. Jr. (2017) Introduction to Wireless Digital Communication: A Signal Processing Perspective, First Edition. Prentice Hall, March 2017

[24] Costa M. H. (1983) Writing on Dirty Paper IEEE Transactions on Information Theory (Volume: 29, Issue: 3, May 1983) p. 439-441

[25] Wiesel A., Eldar Y. C., Shamai S. (2008) Zero-Forcing Precoding and Generalized Inverses. IEEE Transactions on Signal Processing (Volue:56, Issue: 9, Sept. 2008) p. 4409-4418

[26] Li Y. (2002) Simplified channel estimation for OFDM systems with multiple transmit antennas. IEEE Transactions on Wireless Communications (Volume: 1, Issue: 1, Jan 2002) p. 67-75

[27] van de Beek J-J., Edfors O., Sandell M., Wilson S. K. & Börjesson P. O. (1995) On Channel Estimation in OFDM Systems. 1995 IEEE 45th Vehicular Technology Conference. Countdown to the Wireless Twenty-First Century.

[28] Liu Y., Tan Z., Hu H., Cimini L. J. & Li G. Y. (2014) Channel Estimation for OFDM. IEEE Communications Surveys & Tutorials (Volume: 16, Issue: 4, Fourth quarter 2014)

[29] Li Y., Seshadri N. & Ariyavisitakul S. (1999) Channel Estimation for OFDM Systems with Transmitter Diversity in Mobile Wireless Channels. IEEE Journal on selected areas in communications.

[30] ETSI TS 138 211 V15.2.0 (2018-07) 5G; NR; Physical channels and modulation

[31] Penttinen J. T. J. (2019) 5G Explained, Wiley

[32] Chu D. (1972) Polyphase codes with good periodic correlation properties (Corresp.), IEEE Transactions on Information Theory (Volume: 18, Issue: 4, July 1972) p.531-532

[33] Bertrand P. (2011) Channel Gain Estimation from Sounding Reference Signal in LTE, 2011 IEEE 73rd Vehicular Technology Conference.

[34] 3GPP TS 36.211 V16.0.0 TSG RAN E-UTRA Physical channels and modulation (Release 16)

[35] Zhou B., Jiang L. & Zhao S. (2012) Sounding Reference Signal Design for TDD LTE-Advanced System. 2012 IEEE Wireless Communications and Networking Conference (WCNC)

[36] Abd-El-Barr M. & El-Rewini H. Advanced Computer Achitecture and Parallel Processing. John Wiley 2005

[37] Kuon I. & Rose J. (2007) Measuring the Gap Between FPGAs and ASICs. IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems (Volume: 26, Issue: 2, Feb. 2007)

[38] Sigenics (Accessed 6.2.2020) ASIC Vs FPGA: Which On Do You Choose? URL: https://www.sigenics.com/blog/fpga-vs-asic-which-one-is-better

[39] Ansaloni G., Bonzini P. & Pozzi L. (2010) EGRA: A Coarse Grained Reconfigurable Architectural Template. IEEE Transactions on Very Large Scale Integration (VLSI) Systems (Volume:19, Issue: 6, June 2011)

[40] Mei B., Lambrechts A., Mignolet J.-Y., Verkest D. & Lauwereins R. (2005) Architecture exploration for a reconfigurable architecture template. IEEE Design & Test of Computers (Volume: 22, Issue: 2, March-April 2005)

[41] Bhattacharyya S. S., Deprettere E. F., Leupers R. & Takala J. (2010) Handbook of Signal Processing Systems. Springer.

[42] Coussy P. & Morawiec A. (2008) High-Level Synthesis, From Algorithm to Digital Circuit. Springer.

[43] Anitha R., Deshmukh N., Agarwal P., Sahoo S. K., S. Karthikeyan & Reglend I. (2015) A 32 BIT MAC unit design using Vedic multiplier and reversible logic gate. 2015 International Conference on Circuits, Power and Computing Technologies.

[44] Abd-El-Barr M. & El-Rewini H. Fundamentals of computer organization and architecture. Wiley 2005.

[45] Comer D. (2017) Essentials of computer architecture. CRC Press, Taylor & Francis Group 2017

[46] Smith J. E. & Sohi G. S. (1995) The Microarchitecture of Superscalar Processors. Proceedings of the IEEE (Volume: 83, Issue: 12, Dec. 1995)

[47] Dabrowski A., Pawlowski P. & Marciniak T. (2007) Parallel digital signal processing using multi-issue instructions. Signal Processing Algorithms, Architectures, Arrangements, and Applications SPA 2007.

[48] Hunter H. & Moreno J. (2003) A New Look at Exploiting Data Parallelism in Embedded Systems.

[49] Shahabuddin S., Silvén O. & Juntti M. (2017) ASIP Design for Multiuser MIMO Broadcast Precoding

[50] Corporaal H. (1999) TTAs: Missing the ILP complexity wall, Journal of Systems Architecture, Volume 45, Issues 12-13, June 1999, p. 949-973

[51] Hautala I., Boutellier J. & Silvén O. TTADF: Power Efficient Dataflow-Based Multicore Co-Design Flow. IEEE Transaction on Computers (Volume: 69, Issue: 1, Jan. 1 2020)

[52] Corporaal H. (1994) Design of Transport Triggered Architectures, Proceedings of 4th Great Lakes Symposium on VLSI

[53] Dally W. J., Balfour J., Black-Shaffer D., Chen J., Harting R. C., Parikh V., Park J. & Sheffield D. (2008) Efficient Embedded Computing, Computer (Volume: 41, Issue: 7, July 2008)

[54] Balfour J., Dally W., Black-Schaffer D., Parikh V. & Park J. (2008) An Energy-Efficient Processor Architecture for Embedded Systems. IEEE Computer Architecture Letters (Volume: 7, Issue: 1, Jan. 2008)