University of Alberta

Compact and Accurate Hardware Simulation of Wireless Channels for Single and Multiple Antenna Systems

by

Saeed Fouladi Fard

A thesis submitted to the Faculty of Graduate Studies and Research
in partial fulfillment of the requirements for the degree of

Doctor of Philosophy

Department of Electrical and Computer Engineering

Fall 2009
Edmonton, Alberta

Examining Committee

Supervisor: Dr. Bruce Cockburn, Department of Electrical and Computer Engineering

Co-Supervisor: Dr. Christian Schlegel, Department of Computing Science

Examiner: Dr. Vincent Gaudet, Department of Electrical and Computer Engineering

Examiner: Dr. Sergiy Vorobyov, Department of Electrical and Computer Engineering

Examiner: Dr. Mike MacGregor, Department of Computing Science

Examiner: Dr. Steven Wilton, Department of Electrical and Computer Engineering, University of British Colombia

To my parents, Nasrin and Alihossein,

for all their love and support

and

to the memory of my grandfather, Mirzahossein,

for he was a true man

Abstract

The accurate simulation of wireless channels is important since it permits the realistic and repeatable performance measurement of wireless systems. While software simulation is a flexible method for testing hardware models, its long-running simulation time can be prohibitive in many scenarios. Prior to the availability of accurate and standardized channel models, wireless products needed to be verified using extensive and expensive field testing. A far less costly approach is to model the behavior of radio channels on a hardware simulator.

Different channel characteristics should be considered to ensure the faithful simulation of wireless propagation. Among the most important characteristics are the path-loss behavior, Doppler frequency, delay distribution, fading distribution, and time, frequency, and space correlation between fading samples across different antennas. Various fading channel models have been proposed for propagation modeling in different scenarios. A good homogeneous field programmable gate array (FPGA) fading simulator needs to accurately reproduce the propagation effects, yet it also needs to be compact and fast to be effectively used for rapid hardware prototyping and simulation.

In this thesis, new channel models are proposed for the compact FPGA implementation of fading channel simulators with accurate statistics. Compact hardware implementations for physical and analytical fading channel models are proposed that can simulate fading channels with more than one thousand paths on a single FPGA. We also propose design techniques for accurate and compact statistical fading channel simulation of isotropic and non-isotropic scattering in Rayleigh, Rician, Nakagami-m, and Weibull fading channels. Compact FPGA implementations are presented for multiple-antenna fading simulators for geometric one-ring models, two-ring models, elliptical models, and analytical models including the i.i.d. model, and Kronecker, Weichselberger, and VCR channel models. Finally, a fading simulation and bit error performance evaluation platform is proposed for the rapid baseband prototyping and verification of single- and multiple-antenna wireless systems.

Acknowledgements

During the journey through my Ph.D. I was truly lucky to have Prof. Bruce Cockburn as my mentor and supervisor. His maturity, support, and enthusiasm helped me grow and gave me courage to say no to *"the Spirit of Gravity."* His insightful help and guidance significantly improved this work in many aspects.

I would like to express my gratitude to Prof. Christian Schlegel who helped me expand my horizons and welcomed me to the HCDC Lab.

I also want to thank my committee members Prof. Vincent Gaudet, Prof. Sergiy Vorobyov, Prof. Mike MacGregor, and Prof. Steve Wilton for their constructive comments on my dissertation.

I am glad that I had the opportunity of knowing and collaborating with Amirhossein Alimohammad. He has been a true friend to me in the last five years and we have spent a lot of time together working on various projects.

I have learned a lot from my friends in the HCDC and VLSI labs. I had a good time with my friend and they inspired me and made me feel home at the University of Alberta. Amir, Maz, Leendert, John, Tyler, Russel, Eric, Wesam, Geoff, Lukasz, Charmaine, Andrew, Ram, Nima, Anthony, Sheehan, Raj, David, Hesam, Marcel, Behnam, and Majid: It has been great knowing you all. Soon we will be colleagues and I hope we will be in contact and build a great future.

Table of Contents

List of Tables

List of Figures

Nomenclature

List of Acronyms

ACF	Autocorrelation functions
AFD	Average fade duration
AOA	Angle of arrival
AOD	Angle of departure
APS	Angular power spectrum
AR	Autoregressive
ASIC	Application-specific integrated circuit
AU	Arithmetic unit
AWGN	Additive white Gaussian noise
BER	Bit error rate
BPSK	Binary phase shift keying
BQD	Biquad
BRAM	Block random-access memory
CCF	Cross-correlation functions
CDF	Cumulative distribution function
CU	Control unit
DF-I	Direct-form-I
DF-II	Direct-form-II
DOA	Direction of arrival
DOD	Direction of departure
DSP	Digital signal processors

DUT	Design under test
EILPF	Elliptic IIR lowpass filters
FFT	Fast Fourier transform
FIR	Finite impulse response
FOS	First-order sections
FPGA	Field-programmable gate array
GNG	Gaussian noise generator
GSCM	Geometry-based stochastic channel model
GUI	Graphical user interface
HDL	Hardware description language
i.i.d.	Independent and identically distributed
IEEE	Institute of Electrical and Electronics Engineers
IF	Intermediate frequency
IF	Interpolation filters
IFFT	Inverse fast Fourier transform
IFIR	Interpolated finite impulse response
IIR	Infinite impulse response
ILPF	Interpolation lowpass filter
IMLD	Imperfect maximum likelihood decoding
ISI	Inter-symbol interference
ISR	Interrupt service routine
LCR	Level crossing rate
LFSR	Linear feedback shift register
LOS	Line of sight
LTV	Linear time varying
LUT	Look-up table
MA	Moving average
MAC	Multiplier-accumulator

MACL	Media access control
MC	Monte Carlo
MIMO	Multiple-input multiple-output
ML	Maximum-likelihood
MPC	Multipath component
MPS	Multiple parameter set
MSE	Mean square error
MU	Mobile unit
PC	Program counter
PDF	Probability density function
PDP	Power-delay profile
PHY	Physical layer
PNG	Pseudo-random number generator
PSD	Power spectral density
QAM	Quadrature amplitude modulation
RAM	Random-access memory
RF	Radio frequency
ROM	Read-only memory
RP	Random processes
RV	Random variable
RWP	Random walk processes
RX	Receive or Receiver
SER	Symbol error rate
SILPF	Specific interpolation lowpass filters
SIMO	Single-input multiple-output
SISO	Single-input single-output
SNR	Signal-to-noise ratio
SOS	Sum-of-sinusoids

SSF	Spectrum shaping filter
TX	Transmit or Transmitter
VCR	Virtual channel representation
WSS	Wide-sense stationary
WSSUS	Wide-sense stationary uncorrelated scattering

List of Symbols

α	Angle of arrival
$\alpha_\ell(\cdot)$	Angle of arrival from a multipath component
α_n	Angle of arrival
$\Delta\phi$	Phase change
Δd	Change in path length
Δt	Time change
$\delta(\cdot)$	Dirac delta function
$\hat{F}_s$	Initial sample rate (low frequency)
κ	Beam-width parameter
λ	Wavelength
$\mathcal{J}_0(\cdot)$	Zeroth-order Bessel function of the first kind
ω	Normalized frequency (rad/Hz)
$\phi_\ell(\cdot)$	Carrier phase shift from a multipath component
ϕ_o	Random phase of a LOS component
ψ_n	Random phase
$\tau_\ell(\cdot)$	Relative time delay from a multipath component
$\theta[m]$	Random walk process
θ_o	AOA of a LOS component
$\tilde{\psi}$	Average angle of arrival
$\tilde{\varphi}_n$	Random phase (normalized)
φ_n	Random phase
$a_\ell(\cdot)$	Signal amplitude from a multipath component

$B(\vartheta; \varrho; \mathbf{x})$	Barrier function
c	Free-space velocity of an electromagnetic wave
$c(t)$	Complex Rayleigh fading samples (continuous time)
$c(t, \tau)$	Complex multipath fading channel impulse response (Rayleigh)
$c[m]$	Complex Rayleigh fading samples (discrete time)
$c_i(t)$	In-phase (real) component of complex Rayleigh fading samples (continuous time)
$c_i[m]$	In-phase (real) component of complex Rayleigh fading samples (discrete time)
$c_q(t)$	Quadrature (imaginary) component of complex Rayleigh fading samples (continuous time)
$c_q[m]$	Quadrature (imaginary) component of complex Rayleigh fading samples (discrete time)
$E\{\cdot\}$	Mathematical expectation
F_c	Carrier frequency
f_D	Maximum Doppler frequency
F_s	Sample rate
$g(t, \tau)$	Complex multipath fading channel impulse response (Rician)
$G_c(f)$	PSD of isotropic Rayleigh fading samples
G_r	Receiver antenna gain
G_t	Transmitter antenna gain
$H(e^{j\omega})$	Frequency response of a spectrum shaping filter
I	Interpolation factor
$I_m(\cdot)$	The m-th order modified Bessel function of the first kind
K	Rice factor
L	Number of fading paths
L_{loss}	Loss factor
N	Number of sinusoids
N_o	Noise variance
$P_r(\cdot)$	Free space received power

P_t	Transmitted power
$r(t)$	Complex Rician fading samples (continuous time)
$r[m]$	Complex Rician fading samples (discrete time)
$r_i(t)$	In-phase (real) component of complex Rician fading samples (continuous time)
$r_i[m]$	In-phase (real) component of complex Rician fading samples (discrete time)
$r_q(t)$	Quadrature (imaginary) component of complex Rician fading samples (continuous time)
$r_q[m]$	Quadrature (imaginary) component of complex Rician fading samples (discrete time)
$R_{c_i,c_i}(\tau)$	Autocorrelation of the in-phase component of fading samples
$R_{c_i,c_q}(\tau)$	Cross-correlation between the in-phase and quadrature components of fading samples
$R_{c_q,c_q}(\tau)$	Autocorrelation of the quadrature component of fading samples
R_{rms}	Root mean square of a Rayleigh fading process
R_{th}	Threshold level
$S_c(f)$	PSD of non-isotropic Rayleigh fading samples
$S_{c_i}(f)$	PSD of the in-phase component of fading samples
$S_{c_q}(f)$	PSD of the quadrature component of fading samples
T_g	Number of interpolation stages
T_s	Sample period
v	Velocity of the mobile unit
WF	Fraction-length
WL	Word-length

Chapter 1

Motivation and Background

Wireless communication systems are designed to operate over radio channels for a variety of environments and weather conditions. While it is possible to build working hardware prototypes of a system and then field test them in different locations, such an approach will be quite expensive and will not provide useful feedback in the early stages of the design process, when a number of candidate designs must be explored and design changes are easier and less expensive to make. Moreover, propagation conditions are almost impossible to repeat for the comparative analysis of test results. A more practical approach is to create accurate mathematical models for representative radio channels and then base the initial design on these models. Numerous wireless channel models have been proposed to characterize time and/or space-variant propagation environments (e.g., see [15–20]). These channel models have led to different simulator designs that can be efficiently used in the development and accurate error-rate performance evaluation of wireless systems. A channel simulator should mimic the propagation characteristics faithfully since the accuracy of the performance estimation under real world conditions can make the difference between success and failure.

Two major approaches have been widely used to produce statistically accurate sequences of sampled fading variates (i.e., channel attenuation coefficients), namely, shaping the spectrum of Gaussian variates using digital filters and *sum-of-sinusoids* (SOS) based methods. The filter-based technique is attractive for implementing fading channel simulators as it can be customized to accurately reproduce the statistical properties required for simulating fading channels [19]. However, the computationally-intensive multi-rate signal processing required by this technique limits its application in more complex scenarios such as *multiple-input multiple-output* (MIMO) channels. On the other hand, even though the

SOS-based models are substantially less computationally demanding than the filter-based technique, some of the proposed SOS-based fading channel models have inaccurate statistical properties for the unbounded or continuous simulation of time-varying propagation channels.

The demanding performance requirements of wireless applications, along with the increasing computational complexity of baseband algorithms, have greatly increased software-based simulation loads. Therefore, the required run times for the accurate performance evaluation of the most recent low *bit error rate* (BER) baseband algorithms are becoming prohibitively long (e.g., days and even weeks), which makes software-based simulation increasingly impractical. The required run time for accurate fading simulation is an even bigger concern in multiple-antenna or wireless networking scenarios as the number of paths grows rapidly with the number of antennas and/or users. With a large number of antennas/users, the required simulation time becomes prohibitively long for software-based simulators. Although it is much easier to design and implement a fading channel emulator in software than in hardware, hardware-based simulators have been shown to provide several orders of magnitude speed up in performance evaluation over software-based simulators [21, 22], significantly reducing the design time.

Different commercial fading channel emulators are available in the market that accurately reproduce the behavior of propagation environments in the laboratory. They are typically stand-alone units that provide the fading signal in the form of analog or digital samples. They generally require complex hardware consisting of several circuit cards with multiple processors. For example, the NoiseCom MP-2500 Multipath Fading Emulator [23] consists of 11 circuit boards, *radio frequency* (RF) circuitry, cooling fans, and an external computer interface for setting the various parameters of a 12-path frequency-selective fading channel. However, commercially available systems are rather bulky and costly and cannot be used for systems with a relatively large number of antennas and/or users. Some of the main specifications of a number of the available fading channel emulators are listed in Table 1.1. These products are available at prices between $24,000 to $500,000.

A more flexible and cost-effective approach is to implement a real-time channel emulator on a *field-programmable gate array* (FPGA). In the Monte Carlo performance verification of communication systems, the computationally-critical algorithms in the simulation chain (such as the fading channel emulator) can be efficiently implemented on a dedicated hardware device, such as an FPGA. Recent increases in FPGA capacities permit the inte-

Table 1.1: Some Commercially Available Fading Simulators

Model [a]	**I**	**II**	**III**	**IV**	**V**
Number of channels	2	2	2	2	6
Number of paths	12	24	48	6	6
Max. Doppler (Hz)	800	2000	2400	-	340
Fading resolution (Hz)	0.1	0.01	0.05	-	1
Max. delay (ms)	200	2000	10	-	40
Time resolution (ns)	5	0.1	1	1	40

[a] (**I**) Japan Radio Co. NJZ-1600B [24], (**II**) Spirent Communications SR5500 [25], (**III**) Agilent Technologies Inc. N5115A [26], (**IV**) Rohde & Schwarz ABFS [27], (**V**) Ascom Ltd. SIMSTAR [28].

gration of the channel emulator along with a noise generator [29] as well as the transmitter and receiver signal processing blocks onto the same FPGA for rapid prototype design and verification.

Our main goals are (i) to propose accurate SOS-based fading channel models with accurate statistics along with efficient FPGA implementations of these models for simulating fading channels with a large number of simultaneous propagation paths; (ii) to propose filter design techniques for the efficient implementation of fading channels with arbitrary correlation properties; (iii) to propose efficient hardware architectures for especially compact implementations of filter-based fading channel simulators; and (iv) to implement compact realistic single and multiple antenna channel simulators that support emerging standards (e.g., WiMAX, 802.11n, LTE).

1.1 Wireless Fading Channel

In a typical urban area or indoor environment, a direct *line of sight* (LOS) path between the transmitter and the receiver is often absent because of intervening obstacles. Due to the processes of *reflection* (which occurs when a waveform meets an object that is much larger than the signal's wavelength), *diffraction* (which occurs when the surface encountered by the signal has irregularities such as sharp edges), and *scattering* (which occurs when the medium contains a large number of objects nearly the same size as the signal's wavelength) from objects in the path [17], multiple copies of the transmitted signal, called *multipath* signal components or rays, arrive at the receiver via different paths with different *angle of arrivals* (AOAs), time delays, and amplitudes. More importantly, changes in the path length by Δd (due to relative motion between the transmitter and receiver) over a short

time interval Δt causes a phase shift:

$$\Delta\phi = \frac{2\pi\Delta d}{\lambda} = \frac{2\pi v\Delta t\cos\alpha}{\lambda}$$

where v is the relative speed and α is the AOA with respect to the head-on incident direction. Often the transmitter is taken to be fixed in position in which case the relative motion is entirely due to the *mobile unit* (MU). Clearly, as the path length changes by a wavelength λ (30 cm at 1 GHz), the signal phase changes by 2π. Multipath propagation leads to random fluctuations in the amplitude and phase of the received signal due to the movement of the receiver and/or the transmitter over a few wavelengths and for short time durations. Depending upon the relative speed, fades of 30 to 40 dB or more below the mean value of the received signal can occur [30]. The phenomenon of rapid fluctuations of the received signal strength over short distance or short duration is called the *small-scale* fading effect [30]. Typical small-scale fading behavior in shown in Figure 1.1.

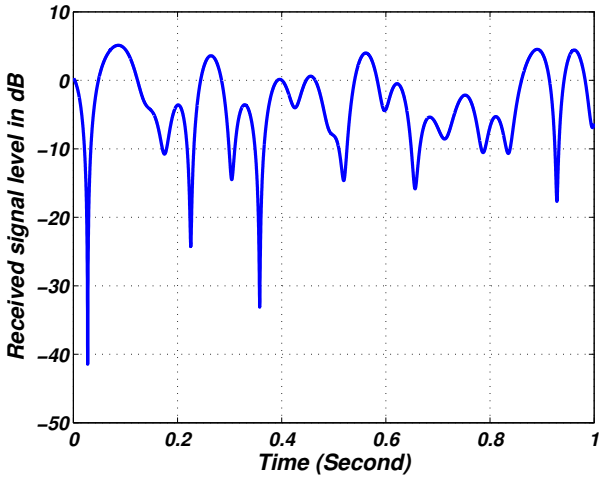


Figure 1.1: Typical simulated Rayleigh fading at the receiver.

If the propagation environment changes or if there is a relative motion of the antennas, the path length and/or geometry changes by Δd and each multipath signal component experiences an apparent shift in frequency, called a *Doppler shift*. The *Doppler frequency* is defined as

$$f_D = \frac{1}{2\pi}\frac{\Delta\phi}{\Delta t} = \frac{v\cos\alpha}{\lambda} = \frac{f_c v\cos\alpha}{c}$$

where f_c is the carrier frequency, $c \approx 3\times 10^8$ m/s is the free-space velocity of the electromagnetic wave, and α is the direction of motion of the mobile with respect to the direction

of multipath signal arrival. The motion of the MU will introduce changes in the signal strength, and hence the apparent channels, at a rate of f_D Hz. For a constant mobile velocity, as f_c increases, the Doppler shift becomes larger. If a sinusoidal signal at the carrier frequency f_c is transmitted, the received signal is spread over a spectrum, called the *Doppler spectrum*, with components lying in the range of $f_c - f_D$ to $f_c + f_D$. If the receiver is moving toward the transmitter, i.e., $-\pi/2 \leq \theta \leq \pi/2$, the Doppler shift is positive (i.e., the apparent received frequency $f = f_c + f_D$ is increased); otherwise, if the receiver reverses direction then the Doppler shift is negative. Relative to the carrier frequency (100s of MHz), the Doppler shift is typically quite small, but relative to baseband frequencies (e.g., 10s of MHz) it can be relatively large.

Fluctuations in the received power are not the only effects of fading. Fading may also affect the shape of the pulse as it is being transmitted through the channel [31, 32]. If the received multipath components are resolvable [33], then multipath effects can result in the broadening of the transmitted pulse, leading to *inter-symbol interference* (ISI), where the pulses of adjacent symbols interfere at the symbol sampling times. It should be noted that the small-scale fading is caused by changes in phase rather than by path attenuation since the path lengths change by only a small amount over small distances. On the other hand, *large-scale* fading causes the received power to vary gradually due to signal attenuation determined by the geometry of the path profile. This phenomenon is affected by prominent terrain contours (hills, mountains, trees, buildings, billboards etc.) between the transmitter and receiver. When such terrain contours are present, the receiver is often said to be *shadowed*. The statistics of large-scale fading provide a way of computing an estimate of the path loss as a function of distance. Large-scale fading may be mitigated by the use of power control, for example, while small-scale fading may require an equalizer that is capable of removing the time-varying ISI introduced by the multipath propagation.

Free space path loss is the signal degradation caused by signal spreading when there is a clear, unobstructed line-of-sight path between the TX-RX pair. The free space received power $P_r(d)$ in the far-field is given by the Friis Free Space Equation [20]

$$P_r(d) = P_t \frac{G_r G_t}{L_{loss}} \left(\frac{\lambda}{4\pi d} \right)^2 \tag{1.1}$$

where P_t is the transmitter power, G_r is the receiver antenna gain, G_t is the transmitter antenna gain, $L_{loss} \geq 1$ is the system loss factor, λ is the wavelength, and d is the distance between the transmitter-receiver pair. Equation (1.1) shows that the received power drops

as the square of the TX-RX separation (or 20 dB/decade with distance).

To evaluate the performance of wireless communication systems in laboratories, a channel simulator must faithfully model both the large-scale and small-scale effects of time-varying propagation environments. Two main approaches are utilized for modeling multipath fading channels: ray-theoretical modeling [34] and impulse-response modeling [35]. The *ray-theoretical model* illuminates essential characteristics of the channel based on geometric propagation theory and physical rays caused by reflections and diffractions. However, the relatively high computational complexity and the typical lack of detailed terrain and building databases make these models difficult to use in practice [19]. By far the most popular channel simulation models are *stochastic parametric models*. In this approach, the channel impulse response is characterized by a set of both deterministic and random parameters. The values of the parameters and the probability distributions governing their behavior are selected according to empirical measurements. A multipath fading channel is commonly modeled as a linear time-varying (LTV) system that is fully described by its impulse response [16, 35–37]. The complex impulse response $c(t,\tau)$ is a low-pass equivalent model of the actual real band-pass impulse response

$$c(t,\tau) = \sum_{\ell} a_\ell(t) e^{j\phi_\ell(t)} \delta\Big[t - \tau_\ell(t)\Big] \tag{1.2}$$

defined as the response observed at time t to an impulse applied at time $t-\tau$, where τ is the delay parameter, t is the time. The ℓ^{th} signal component experiences a different path environment which will determine the amplitude a_ℓ, carrier phase shift ϕ_ℓ, time delay τ_ℓ, AOA α_ℓ, and Doppler shift f_D. In general each of these parameters is time-varying. When a large number of propagation paths exist, the Central Limit Theorem can be applied. In this case, $c(t,\tau)$ can be modeled as a complex Gaussian random process. When the complex Gaussian process $c(t,\tau)$ is zero-mean, the envelope $|c(t,\tau)|$ is Rayleigh-distributed

$$f_{|C|}(c) = \frac{c}{\sigma^2} e^{-c^2/2\sigma^2}, \;\; c \geq 0 \tag{1.3}$$

where $2\sigma^2$ is the variance of the zero-mean complex Gaussian random process $c(t,\tau)$. Figure 1.2 plots the Rayleigh *probability density function* (PDF) in equation (1.3) for different values of σ. Note that the channel model in Equation (1.2) does not consider the AOA of each multipath component. It is usually assumed that the scatterers surrounding the mobile station are about the same height as, or are higher than, the mobile. This implies that the received signal at the mobile antenna arrives from all directions after bouncing from the

surrounding scatterers. Under these conditions, the Gans assumption, namely, that the AOA is uniformly distributed over $[0, 2\pi]$, is valid [36]. The classical Rayleigh fading envelope, with deep fades approximately $\lambda/2$ apart, arises from this model [37].

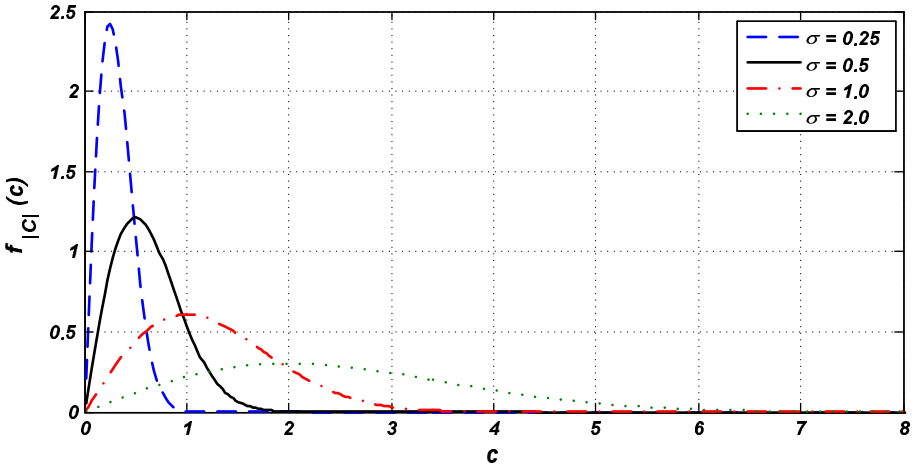


Figure 1.2: Rayleigh probability density function for different values of σ.

When a strong LOS path exists in addition to the scattered paths, then the fading process has a non-zero mean (arising from the LOS signal) and the magnitude of the process becomes *Rician* [19]. This strong component may be a LOS path or a path that goes through much less attenuation compared to the other received components (also called a *non-faded* or *specular* path). The Rician PDF is usually characterized by the ratio of the power of the direct component to the power of the scattered component, where $K(\text{dB}) = 10\log_{10} K$ and the ratio K is called the *Rice factor*. In the presence of a specular path, the fading signal $g(\tau, t)$ can be considered to be the sum of two components: a Rayleigh component $c(\tau, t)$ and a deterministic (in amplitude and phase) component $d(t)$ representing the LOS path as

$$g(\tau, t) = \sqrt{\frac{1}{K+1}}\, c(\tau, t) + \sqrt{\frac{K}{K+1}}\, d(t) \tag{1.4}$$

where the deterministic component $d(t)$ can be written as $d(t) = ae^{j(\omega_d t + \phi_d)}$ where the amplitude, Doppler shift, and phase of the LOS component are denoted by a, ω_d, and ϕ_d, respectively [38]. If the Doppler shift along the LOS path is zero, then the mean value of $d(t)$ is time-invariant.

Another important distribution that has been proposed to model the magnitude of the received envelope is the *Nakagami-m* distribution [39]. Nakagami-m fading is proposed to model fading channels with relatively large time-delay spreads and different clusters of reflected waves [39–41].

In this work, we primarily focus on simulating small-scale fading with the Rayleigh and Rician distributions. Since large-scale fading changes slowly with distance, it can be implemented easily by attenuating the received signal with precomputed coefficients according to the desired models. Later, we will propose a hardware strategy to transform samples with the Rayleigh distribution to samples with the Nakagami-m distribution [42, 43].

1.2 Main Contributions

The contributions of this thesis lie in six main problem areas: *sum-of-sinusoids (SOS) based fading simulation*, *filter-based fading simulation*, *fixed-point filter design*, *simulation of analytical MIMO fading channel models*, *simulation of geometric MIMO fading channel models*, and *fading simulation and bit error rate testing platform*. Here we briefly summarize each of these contributions:

- ***SOS-based fading simulation***

    1. In this thesis we propose three new models for the accurate simulation of Rayleigh and Rician fading [2–4] (see Sections 2.2.1, 2.2.2 and 2.2.3). We also propose compact architectures for the fast and efficient on-chip simulation of wireless fading channels.
    2. We propose a new ultra-compact hardware implementation of an accurate fading simulator in which the fading samples are generated differentially and interpolated with a very compact architecture (see Section 2.2.4). More than one thousand paths can be fitted on a conventional *field-programmable gate array* (FPGA) with this method, with each path generating more than 300 million samples per second. Compared to one of our early designs, the proposed fading simulator is not only 18 times smaller and 50% faster, it can also generate significantly more accurate fading samples.

- ***Filter-based fading simulation***

    1. We propose two architectures for the homogeneous FPGA implementation of filter-based fading simulators [9, 10] (see Sections 3.2). We also propose a multi-stage filter design technique for the efficient hardware simulation of Rayleigh fading channels.

2. We propose an elastic buffer design for the robust implementation of multipath fading simulator that can absorb any clock mismatch between hardware modules [13] (see Section 3.2.3). Another multistage design technique is proposed and multiplication-free filters are used for interpolation. A fixed-point implementation of an example four-path fading simulator on a Xilinx Virtex-II Pro XC2VP100-6 FPGA utilizes only 13.9% of the configurable slices and 2.7% of the on-chip 18×18 multipliers and can generate up to 4×73 million samples per second. This fading simulator was implemented and verified on a Digilent Spartan-III FPGA development board.
3. Two compact filter processors for the efficient simulation of fading channels are proposed [11, 44] (see Section 3.2.1).
4. We also propose the first hardware simulator for non-isotropic Rayleigh fading channels [12] (see Section 3.2.2). Our fixed-point implementation of this simulator on a Xilinx Virtex-II Pro XC2VP100-6 FPGA utilizes only 6.8% of the configurable slices and can generate up to 300 million samples per second.
5. We propose a new transformation-based fading simulator for the compact and efficient implementation of Nakagami-m and Weibull fading channels (see Section 3.4). This fading simulator converts the already available Rayleigh fading samples (from SOS-based or filter-based fading simulators) to Nakagami-m or Weibull fading samples. A new method for the approximation of the transfer function is proposed which is based on hybrid logarithmic-linear segmentation with semi-floating-point curve fitting. Compared to the simple table look-up approach for the approximation of the transfer function, the proposed method provides more than 265x saving in the storage requirements. A fixed-point implementation of this simulator on a Xilinx Virtex-II Pro XC2VP100-6 FPGA utilizes only 1.5% of the configurable slices, five 18×18 multipliers and three block memories and can generate up to 246 million Nakagami-m or Weibull fading samples per second.

- ***Fixed-point filter design***

1. Traditionally, recursive filters are designed with floating-point tools and then converted to fixed-point designs for efficient hardware implementation. This results in unwanted deviations in the filter response and potential numerical

instability due to fixed word-length effects. In addition, to avoid overflow, the designed filter needs to be implemented with a sufficiently large number of bits. We propose a step-by-step technique for the design of stable recursive filters with fixed-point complex and real coefficients [14] (see Section 3.3). Filter design with this technique results in very compact and efficient hardware implementations.

2. The proposed filter design technique can be used to simulate a wide range of fading channel conditions including non-isotropic Rayleigh fading channels and the TGn channel model for the IEEE Standard 802.11n.
3. We propose two methods to reduce the variable range for the compact implementation of recursive filters. The first method involves sorting the poles and zeros in a specific order while the second method involves augmenting auxiliary poles and zeros at certain frequencies.
4. Using the proposed filter design technique results in significant savings in the hardware implementation and substantial increases in the system throughput. In one example, the new filter design technique resulted in a nine times reduction in the number of configurable slices and more than 22 times higher throughput.
5. We implemented a filter processor for the designed filters and verified the hardware generated *power spectral density* (PSD) against the target response.

- ***Simulation of analytical MIMO fading channel models***

1. We propose two new models for the efficient simulation of MIMO fading channels. Unwanted cross-correlations are reduced by adding random-walk processes (see [8, 45]).
2. We propose a compact differential implementation of a MIMO fading simulator [5] (see Section 2.2.4). This fading simulator is very compact and we can simulate more than one thousand paths on a single FPGA.
3. We propose a compact and efficient FPGA fading simulator that can support the simulation of the i.i.d., Kronecker, Weichselberger, and VCR MIMO fading channel models (see Section 4.5). A new stable interpolator structure is proposed for this fading simulator. When implemented on a Xilinx Virtex-5 XC5VLX110-3 FPGA, the matrix processor of this design for a 4×4 MIMO

system utilizes only 1.8% of the configurable slices, two multipliers, and four block memories, and can operate at up to 234.1 MHz. This MIMO fading simulator is also implemented on our bit error rate testing platform.

4. Using the proposed MIMO fading simulator, we measure the bit error performance of an example 2×2 MIMO communication system on our hardware platform. The accuracy of our hardware implementation was verified by comparison with floating-point computer simulations.

- ***Simulation of geometric MIMO fading channel models***

  1. We propose a compact and efficient simulator for the geometric MIMO channel models (see Section 4.4). The new fading simulator can simulate a wide variety of single- and double-bounce geometric MIMO channel models. A fixed-point implementation of this simulator for a 4×4 MIMO channel on a Xilinx Virtex-5 XC5VLX110-3 FPGA utilizes only 6.6% of the configurable slices, two 18×18 multipliers and three block memories and can generate more than 16×324 million samples per second.
  2. We verified the accuracy of our fading simulator by comparing the fixed-point bit-true results with theoretical reference functions. Three common geometric MIMO fading models, namely the one-ring and two-ring models and the geometrical elliptic MIMO fading channel model, were simulated with our fading simulator.
  3. This fading simulator was also implemented on our fading simulation and bit error rate testing platform.

- ***Fading simulation and bit error rate testing platform***

  1. We implemented a fading simulation and bit error rate testing platform for the verification of our fading simulators, and for testing MIMO systems (see Section 4.7). This platform was implemented on a GVA-290 FPGA development board which hosts two Xilinx Virtex-E XCV2000E FPGAs.
  2. The implemented platform can be used to verify single- and multiple-antenna wireless systems. It supports the simulation of various fading channel models including single-antenna models (AWGN, Rayleigh, Rician) and multiple-antenna models (AWGN, Rayleigh, Rician, one-ring, two-ring, geometric el-

liptic, i.i.d., Weichselberger, Kronecker, and VCR). The implemented platform also supports both multipath integer (or tap) delays (up to 1023 taps) and fractional delays.

3. The accuracy of the implemented fading simulation platforms was verified against floating-point computer simulations.
4. We tested a sample 2×2 MIMO system with our platform. The MIMO system under test included an extended Golay code encoder and decoder, a length-16383 random interleaver and de-interleaver, and a maximum likelihood detector.
5. Software and hardware parts required for the bit error rate performance evaluation of the system under test were developed. A detailed software graphical interface was implemented to simplify parameterization of the fading channel simulator and the bit error performance evaluation platform.

1.3 Thesis Outline

The rest of this thesis is organized as follows. Chapter 2 considers modeling and efficient implementation of SOS-based Rayleigh and Rician fading channel simulators. In this chapter, various SOS-based fading channel models are presented and their statistical properties are compared. Three novel fading channel models based on the SOS approach are presented, and FPGA implementation results of the proposed fading simulators are given.

Chapter 3 presents novel design and implementation methods to realize parameterizable fading channel simulators on a single FPGA using digital filters. This chapter discusses compact and efficient implementation of isotropic and non-isotropic Rayleigh and Rician fading channel models. A new technique is proposed for designing recursive filters with real and complex fixed-point coefficients. Also, techniques are proposed for reducing the variable range and thus allowin compact implementation. This chapter also proposes an efficient technique for generating Nakagami-m and Weibull fading samples from the already available Rayleigh fading samples.

Chapter 4 considers the accurate and compact simulation of MIMO wireless channels. This chapter starts with a brief review and classification of various MIMO fading channels. Then, this chapter discusses hardware simulation of single- and double-bounce geometric MIMO channel models. Hardware simulation of analytical MIMO channel models is

discussed next. This chapter also presents our fading simulation and bit error rate testing platform.

Finally Chapter 5, proposes related future work and makes some concluding remarks.

Chapter 2

Sum-Of-Sinusoids Based Fading Simulation

The effective design of wireless communication systems needs faithful modeling and accurate simulation of radio propagation characteristics. Successful implementation of wireless products requires prototyping and extensive field testing possibly involving two or more design iterations in the event that problems are discovered. Accurate modeling of characteristics of the fading channel plays an important rule in this process. For example, the level crossing rate of the envelope of the fading samples provides important information conveying the statistics of burst errors in wireless communication systems [46, 47]. As another example, the average fade duration determines the average length of error bursts, which has a great impact on design and testing of wireless communication systems [46, 47] not only on the physical layer but also on the link and network layers [48]. In addition, the fading correlation properties affect channel estimation (e.g., [49]), prediction and equalization, and the distribution of fading samples impacts the error rates [50].

Sum-of-sinusoids (SOS) based methods are among the most widely used approaches to simulate fading channels [51]. The basic idea behind SOS-based fading channel simulators is that when a sinusoidal carrier is transmitted and subjected to multipath fading, the received signal can be modeled as a superposition of waveforms received from different propagation paths. Since the nature and orientation of obstacles in the wireless channel are not known in advance, the net effect of the received waveforms can be considered to be a stochastic processes. In the SOS approach, the flat-fading process is modeled by superimposing sinusoidal waveforms with amplitudes, frequencies and phases that are selected appropriately to generate the desired statistical properties in the sum signal.

A well-established technique for modeling the behavior of a Rayleigh fading channel

with given temporal correlation properties was introduced by Rice [51]. Rice's model is based on the superposition of weighted sinusoids with random phases. For a large number N of sinusoids, Rice's model can accurately generate samples with the desired statistics; however, the relatively large number of required sinusoids makes this model computationally inefficient.

Various fading channel simulators have been proposed over the last four decades that involve superimposing a sufficient number of sinusoidal waveforms [1, 16, 17, 30, 52–56]. Due to the reasonable computation requirements in this approach, the SOS-based models have been widely used to implement fading channel simulators [57–61]. Moreover, as we will see in Chapter 4, a SOS-based fading simulator can be used to simulate a wide variety of multiple-antenna channel models.

In this chapter we will briefly compare SOS-based models for simulating isotropic fading in single-antenna scenarios. We have made several improvements to the fading channel models to enhance the statistical accuracy and reduce computational complexity. Several SOS-based fading simulators have been implemented in this work. We will briefly present implementation results for these fading channel simulators. Interested readers can refer to our listed publications for more details.

2.1 Background and Related Work

A Rayleigh fading channel is commonly modeled using a complex Gaussian *wide-sense stationary uncorrelated scattering* (WSSUS) process $c(t) = c_i(t) + jc_q(t)$ [15], where the envelope $|c(t)|$ follows the Rayleigh distribution

$$f_{|C|}(u) = 2u\exp(-u^2). \tag{2.1}$$

Two of the most important statistical properties of the fading process are manifested in the *autocorrelation function* (ACF) and the *cross-correlation function* (CCF) between $c_i(t)$ and $c_q(t)$. In a two-dimensional isotropic scattering environment with an omnidirectional receiving antenna at the receiver, the ACF associated with either $c_i(t)$ or $c_q(t)$ is expressed as [62]

$$\begin{aligned} R_{c_i,c_i}(\tau) &= R_{c_q,c_q}(\tau) \\ &= E\{c_i(t)c_i(t+\tau)\} \\ &= \frac{1}{2}\mathcal{J}_0(2\pi f_D\tau) \end{aligned} \tag{2.2}$$

where $E\{\cdot\}$ denotes mathematical expectation, f_D is the maximum Doppler frequency and $\mathcal{J}_0(\cdot)$ denotes the zeroth-order Bessel function of the first kind [63]. The CCF between $c_i(t)$ and $c_q(t)$ is

$$\begin{aligned} R_{c_i,c_q}(\tau) &= R_{c_q,c_i}(\tau) \\ &= 0. \end{aligned} \tag{2.3}$$

Moreover, the *power spectral density* (PSD) associated with either $c_i(t)$ or $c_q(t)$ is the so-called Jakes PSD, which can be written as [64]

$$\begin{aligned} S_{c_i}(f) &= S_{c_q}(f) \\ &= \begin{cases} \frac{1}{2\pi\sqrt{f_D^2 - f^2}} & \text{for } |f| < f_D \\ 0 & \text{elsewhere.} \end{cases} \end{aligned} \tag{2.4}$$

Also, higher-order statistical properties, such as the envelope *level crossing rate* (LCR) and the *average fade duration* (AFD), provide important information for assessing verifying system performance. For a Rayleigh fading channel with the Jakes' PSD, the LCR and the AFD can be shown to be [20]

$$L_{|c|} = \sqrt{2\pi} f_D \rho e^{-\rho^2} \tag{2.5}$$

and

$$T_{|c|} = \frac{e^{\rho^2} - 1}{\sqrt{2\pi} f_D \rho}, \tag{2.6}$$

respectively, where $\rho = R_{th}/R_{rms}$ is the value of the specified threshold level R_{th}, normalized to the root-mean-square value of the fading envelope R_{rms}.

When a strong direct path exists in addition to the scattered paths, then the fading process has a non-zero mean and the magnitude of the process becomes Rician-distributed instead of simply Rayleigh-distributed [19]. This strong component may be a *line-of-sight* (LOS) path (also called a specular path or component) or a path that happens to undergo much less attenuation compared to the other received components. Rician fading is often characterized by the ratio of the power of the direct component to the total power of the scattered components, also known as the Rice factor K. In the presence of a specular path, the complex fading process $r(t)$ can be considered to be the sum of two complex components: a Rayleigh component $c(t)$ and a deterministic component $d(t)$. In this case the fading process is expressed as

$$r(t) = \sqrt{\frac{1}{K+1}}\, c(t) + \sqrt{\frac{K}{K+1}}\, d(t). \tag{2.7}$$

Existing Rician channel models assume that the LOS path is either (a) constant and non-zero [65], or (b) time-varying and deterministic [66], or (c) time-varying and stochastic [67,68]. Since the first two assumptions may not accurately reflect the behavior of the LOS component, without loss of generality we chose the third model. This Rician fading model can be expressed as

$$\begin{aligned} r(t) &= r_i(t) + jr_q(t) \\ &= \sqrt{\frac{1}{K+1}}\, c(t) + \sqrt{\frac{K}{K+1}}\, e^{j2\pi f_D t\cos(\theta_o)+\phi_o} \end{aligned} \tag{2.8}$$

where θ_o and ϕ_o are the AOA and the initial phase of the LOS component, respectively, which are uniformly distributed random variables over $[-\pi, \pi)$.

It can be shown that the *probability distribution function* (PDF) of the amplitude of the fading samples in (2.8) follows the Rician distribution [68]

$$\begin{aligned} f_{|R|}(u) &= 2(1+K)u \times \exp\big[-K-(1-K)u^2\big] \times \\ &\quad I_o\big[2u\sqrt{K(1+K)}\big] \qquad u \geq 0, \end{aligned} \tag{2.9}$$

where $I_o(\cdot)$ denotes the zero-order modified Bessel function of the first kind [63]. Further, for the above Rician fading model the ACF and the CCF are given by [68]

$$\begin{aligned} R_{r_i,r_i}(\tau) &= R_{r_q,r_q}(\tau) \\ &= E\{r_i(t)r_i(t+\tau)\} \\ &= \frac{\mathcal{J}_o(2\pi f_D\tau) + K\cos(2\pi f_D\tau\cos\theta_o)}{(2+2K)} \end{aligned} \tag{2.10}$$

and

$$\begin{aligned} R_{r_i,r_q}(\tau) &= -R_{r_q,r_i}(\tau) \\ &= E\{r_i(t)r_q(t+\tau)\} \\ &= \frac{K\sin(2\pi f_D\tau\cos\theta_o)}{2+2K}, \end{aligned} \tag{2.11}$$

respectively. Also, the LCR and the AFD of the Rician model can be expressed as [68]

$$\begin{aligned} L_{|R|}(\rho) &= \sqrt{\frac{2(1+K)}{\pi}}\rho f_D \exp\big(-K-(1+K)\rho^2\big) \times \\ &\quad \int_0^{\pi} \Big(1+\frac{2}{\rho}\sqrt{\frac{K}{K+1}}\cos^2\theta_o.\cos\alpha\Big) \times \\ &\quad \exp\big(2\rho\sqrt{K(1+K)}\cos\alpha \\ &\quad -2K\cos^2\theta_o.\sin^2\alpha\big)\mathrm{d}\alpha, \end{aligned} \tag{2.12}$$

and

$$T_{|R|}(\rho) = \frac{1 - Q\left(\sqrt{2K}, \sqrt{2(1+K)\rho^2}\right)}{L_{|R|}(\rho)}, \tag{2.13}$$

where $Q(\cdot)$ is the Marcum $Q-$function [69].

Jakes proposed a computationally-efficient model for the simulation of Rayleigh fading channels. Compared to Rice's model, Jakes' model requires a relatively small number of sinusoids [17] whose amplitudes, frequencies, and phases are constant and angles of arrival are equally distributed about the receiver. Over the past three decades, many modifications and improvements to Jakes' model have been proposed (e.g., see [1,54–56,70]). Jakes' original model and its derivatives are generally divided into either deterministic models (i.e., the amplitude, phase, and frequency of each sinusoid are constant values) or stochastic models (i.e., at least one of the three parameters is a random variable) [71]. Both deterministic and stochastic SOS-based models have been used to model Rayleigh [30, 62], Rician [66–68], and Nakagami [72, 73] fading channels. These simulators have been used successfully in the burst error analysis of mobile communication systems (e.g., see [74, 75]).

A channel simulator should generate a discrete time (i.e., $t = mT_s$ where T_s is the sample period) fading process $c[m] = c_i[m]+jc_q[m]$ whose statistical properties are sufficiently close to those of the reference model. Unfortunately, the deterministic Jakes' simulator [62] and related derivatives (either deterministic or stochastic models), have undesirable statistical properties. For example, it is shown in [56] that the deterministic models in [17,70] have different ACFs for the in-phase and quadrature components of the fading process (when they should be identical). Also the CCF between $c_i(t)$ and $c_q(t)$ is not zero (as required by theory), and the outputs of independently initialized pairs of fading processes are not statistically independent.

The deterministic model in [66] is not *wide-sense stationary* (WSS) [68] and the quadrature CCF of the model in [54] can deviate from zero. The mean of of a WSS stochastic process is constant over time and the correlation function of a WSS stochastic process depends only on the time difference [76, p. 388]. These WSS properties are necessary in a fading simulator since they assure rather similar results independent of simulation time.

A deterministic model of the exact Doppler spread [77] as well as the model in [56, 77], requires a relatively large number of sinusoids to successfully produce multiple uncorrelated Rayeligh fading processes [78]. This increases the computational complexity and precludes the use of these models in frequency-selective channels and MIMO scenarios.

Some hardware simulators based on deterministic SOS-based fading channel models have been proposed in the literature (e.g., [53, 79]), despite the fundamental weaknesses that we will describe later.

Stochastic SOS-based models have been used instead of deterministic models to increase the statistical accuracy. In stochastic models, multiple uncorrelated fading sequences can be generated using a relatively small number of sinusoids (e.g., $N = 8$) [80]. There are also several hardware-based implementations of SOS-based fading channel simulators based on stochastic models [6, 7, 60, 61].

The main drawback of the stochastic models (e.g., [1, 81]) is that their statistical properties converge to the reference properties only when averaged over a sufficiently large number of simulation trials and/or when a sufficiently large number of sinusoids are superimposed. Unfortunately computationally-intensive time-averaging implies long simulation times to achieve reliable results. Also, time-averaging requires the *design under test* (DUT) to be fully synchronized with the repeated restarts of the channel simulator, which is not always feasible. In other words, the fading simulator cannot change the fading parameters when the DUT is transmitting packets in the middle of a simulation-based measurement.

Using larger values of N (e.g., $N = 40$) produces greater simulation accuracy over long simulation runs [68], but as mentioned before, this will lengthen the simulation times. When N is a small number, the deterministic and stochastic models are not in general ergodic and the properties of one simulation trial will deviate substantially from the theoretical reference properties. If a channel model is ergodic, then each fading process generated by one simulation trial will have the same statistics. In such a case, only one sample fading process would suffice to characterize the channel. Thus an ergodic model significantly reduces the overall simulation time, requiring only the time-averaging provided by a single simulation run instead of ensemble-averaging [68]. References [78, 82] report the stationarity and ergodicity properties of a deterministic SOS-based model and compare them with those of different classes of stochastic SOS-based models (depending on the randomness or constant behavior of sinusoid parameters).

An SOS-based Rayleigh fading model that has been used in FPGA implementations of fading channel simulators [60, 61] is the Li *et al.* model [56] (henceforth called *Model I*), which can be written in discrete time as follows:

Model I:

$$c_i[m] = \sqrt{\frac{1}{N}} \sum_{n=0}^{N-1} \cos\left(2\pi f_{D_n} T_s m \cos\alpha_n + \varphi_n\right),$$

$$c_q[m] = \sqrt{\frac{1}{N}} \sum_{n=0}^{N-1} \sin\left(2\pi f_{D_n} T_s m \sin\alpha_n + \psi_n\right),$$

where N is some sufficiently large number of sinusoids, m is the discrete-time index, $f_{D_n}T_s$ is the normalized maximum Doppler frequency of the n-th sinusoid, T_s is the sample period, and $\alpha_n = 2\pi n/(4N) + \alpha_o$ is the *angle of arrival* (AOA) of the n-th sinusoid where $0 < \alpha_o < 2\pi/N$ and $\alpha_o \neq \pi/N$. The phases of the n-th sinusoidal components, φ_n and ψ_n, are statistically independent and uniformly distributed random variables over $[-\pi, \pi)$, for all n.

Another well-known SOS-based model that has been utilized in FPGA implementations of fading channel simulators [6, 7] is the model proposed by Zheng *et al.* [1] (henceforth called *Model II*). It is shown in [71] that this fading model is one of the most accurate models for simulating a fading channel for a relatively small number of sine waves. This fading model can be written in discrete time as follows:

Model II:

$$c_i[m] = \sqrt{\frac{1}{N}} \sum_{n=1}^{N} \cos\left(2\pi f_D T_s m \cos\alpha_n + \varphi_n\right),$$

$$c_q[m] = \sqrt{\frac{1}{N}} \sum_{n=1}^{N} \cos\left(2\pi f_D T_s m \sin\alpha_n + \psi_n\right),$$

where $\alpha_n = (2\pi n - \pi + \theta)/(4N)$ and θ, φ_n and ψ_n are mutually independent random variables uniformly distributed over $[-\pi, \pi)$. This model requires a relatively small number of sinusoids (between 8 and 12) to generate relatively accurate correlation and LCR properties. It was concluded in [71] that *Model II* has superior statistical properties compared to the other SOS-based models.

The Rayleigh model proposed by Xiao *et al.* [67] (henceforth called *Model III*) can be written in discrete time as follows:

Model III:

$$c_i[m] = \sqrt{\frac{1}{N}} \sum_{n=1}^{N} \cos\left(2\pi f_D T_s m \cos\alpha_n + \varphi_n\right),$$

$$c_q[m] = \sqrt{\frac{1}{N}} \sum_{n=1}^{N} \sin\left(2\pi f_D T_s m \cos\alpha_n + \psi_n\right),$$

where $\alpha_n = (2\pi n + \theta_n)/N$ and φ_n is the phase of the n-th sinusoid. φ_n and θ_n are statistically-independent and uniformly-distributed random variables over $[-\pi, \pi)$, for all n.

As shown in Figure 2.1, the ACFs of *Models I*, *II*, and *III* all deviate significantly from the reference properties, especially at the larger lags, when the statistics are taken over only one block of generated fading samples. In this figure, all of the above models have been simulated with $N = 8$ sinusoids. For more accurate results using *Model I*, a relatively large number of sinusoids N (e.g., $N = 32$) is required [56], which is not desirable for implementation. Even though these models have serious limitations with respect to matching the theoretical ACF, as shown in Figure 2.2, *Model II* has a much better (i.e., closer to zero) cross-correlation function between the quadrature components compared to *Models I* and *III*.

Compared to the other models, *Model II* can reproduce the theoretical CCF (i.e., that at zero) more accurately with a relatively small number N of sinusoids, as shown in Figure 2.2. However the ACF properties of this model need further improvements for the purpose of an accurate hardware simulator.

The main drawback of the above models is that their statistical properties converge to the desired properties only when averaged over a relatively large number of simulation trials [1]. The statistical properties of a single trial, no matter how long, deviate from the reference properties [78]. If the channel simulator were to be ergodic, then each simulation trial would produce the same statistics. Being able to use only a single trial would significantly reduce the overall simulation time [78]. The simulation results in Figures 2.1 and 2.2 confirm the results in [78] that *Model II*, which has constant amplitudes but random frequencies and phases, is not ergodic. Therefore, *Model II* is also not appropriate for the simulation of wireless systems with continuous (or very long) communications [2]. To generate more accurate results one could frequently reset the random phases in *Model II*. However, this is not desirable in practical channel simulations since it would require simulation restarts and would introduce channel property discontinuities during the trials.

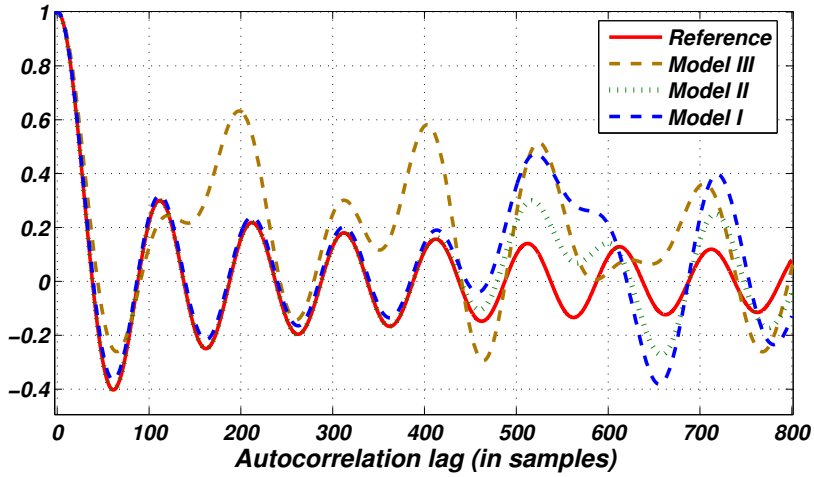


Figure 2.1: Normalized ACF for one block containing 2×10^6 fading samples using the Zheng *et al.*, Xiao *et al.*, and Li *et al.* models with $f_D T_s = 0.01$ for $N = 8$.

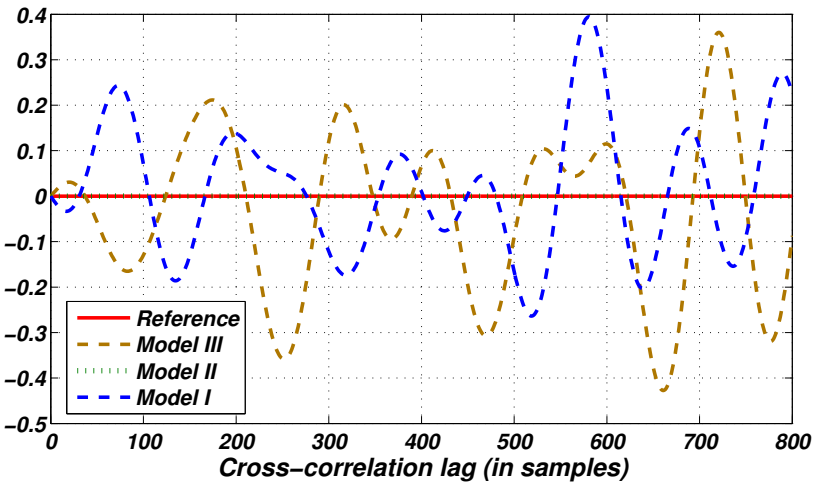


Figure 2.2: CCF for one block containing 2×10^6 fading samples using the Zheng *et al.*, Xiao *et al.*, and Li *et al.* models with $f_D T_s = 0.01$ for $N = 8$.

To overcome these limitations, we propose new fading channel models that replace the angle of arrival and other phase parameters in *Model II* with random walk processes. These modifications significantly improve the statistical accuracy of the generated fading samples. In the following sections, we will provide more details about our modifications and the corresponding implemented SOS-based hardware fading simulators.

2.2 Implementation of SOS-Based Fading Simulators

The SOS-based fading simulators developed in [6, 7] used *Model II* for the FPGA implementation of a fading simulator. This model was chosen for implementation because it could produce the desired channel properties with a relatively small number N of sine waves [71]. The main difference between the hardware implementations in [6] and [7] is in the way the trigonometric functions $\cos(\cdot)$ and $\sin(\cdot)$ are approximated. In [6], these functions are stored in look-up tables with 1024 entries, while the design in [7] calculates these trigonometric functions iteratively.

Comparing the implementation results from [6, Table 1] and [7, Table 1] shows that using table look-up for approximating the basic trigonometric functions is a more efficient approach compared to iterative calculation. Based on these results, in our implementations we approximated the $\cos(\cdot)$ and $\sin(\cdot)$ trigonometric functions using the table look-up approach.

2.2.1 SOS-Based Fading Simulator with Improved Statistics

As mentioned before, one of the drawbacks of SOS-based fading simulators is that their average statistical properties converge over a relatively large number of simulation trials. To overcome this problem, a *multiple parameter set* (MPS) simulation method is proposed in [83]. This technique divides a simulation trial into several frames and randomly generates Doppler frequencies and phases for each frame. For example, to generate 10^7 in-phase and quadrature components, we can divide them into 10^3 frames of length 10^4 samples each, to find time-averaged results. It should be noted that the autocorrelation with the MPS model is zero for time delays that exceed the frame length. Hence, the frame length should be sufficiently large to cover the time delays of interest to get meaningful results. With this method, the performance of Monte Carlo simulation with the *Model II* fading simulator is considerably improved [71].

Unfortunately, the MPS model creates discontinuities in the temporal behavior. As a consequence, the testing of a communication system should be interrupted and re-initialized every time with a new set of random parameters for each trial to ensure accurate modeling of the channel. At the receiver, the channel estimation or carrier recovery must be re-acquired after each draw of random parameters. However stopping and restarting the communication system and channel simulator in this way might not be convenient in many practical cases.

Therefore, *Model II* may not be suitable for emulating a Rayleigh fading channel with accurate time-average statistical properties.

To improve the existing models, Zajić and Stüber [84] proposed a deterministic model which is ergodic. In other words, for their model the average of the fading process parameters over time and the average over the statistical ensemble are the same. However, the autocorrelation of the in-phase and quadrature components do not accurately match the theoretical properties. Zajić and Stüber also proposed a statistical model to overcome this shortcoming of their deterministic model. However, the resulting modified model was no longer ergodic.

In order to have a SOS-based fading simulator whose time average converges to its statistical ensemble average, the random AOA in the model should vary in a reasonable way with time and become a stochastic process. We proposed a new model in [2], based on *Model II*, in which the in-phase and quadrature components of the fading samples are expressed as

Model IV:

$$c_i[m] = \sqrt{\frac{1}{N}} \sum_{n=1}^{N} \cos\left(2\pi f_D T_s m \cos\alpha_n[m] + \varphi_n\right),$$

$$c_q[m] = \sqrt{\frac{1}{N}} \sum_{n=1}^{N} \cos\left(2\pi f_D T_s m \sin\alpha_n[m] + \psi_n\right),$$

where $\alpha_n[m] = (2\pi n - \pi + \theta[m])/(4N)$ and $\theta[m]$ is a stationary stochastic process. In particular, we were inspired by the measurements in [85] to model the behavior of AOA as a random walk process.

Although the AOA changes continuously, the time samples look like a random walk [85]. Moreover, in isotropic scattering, $\theta[m]$ is uniformly distributed over $[-\pi, \pi)$, and also since the AOA does not change rapidly, $\theta[m]$ must be highly correlated. A highly correlated (depending on Doppler frequency) random process was thus required with uniformly distributed samples over $[-\pi, \pi)$. Time-correlation can result in a non-uniform distribution of samples except in a random walk processed with controlled step size. We proposed to use the stochastic random walk process given by Algorithm 1 (from [2]).

The random walk process θ in this algorithm is generated using the process $u[m]$ which has independent samples that are uniformly distributed over $[0, 1)$. The step size δ_o should be chosen to be small enough to model the behavior of highly correlated (or slowly changing) AOAs for different Doppler rates. Some suitable values for δ_o are suggested in [2].

Algorithm 1 The proposed random walk process $\theta[m]$

Initialize $\delta_o = \epsilon \ll 1$, $\theta[0] = U(-\pi, \pi)$;
for $m > 0$ **do**
  $\theta[m] = \theta[m-1] + \delta_o \times u[m]$;
  if $\theta[m] > +\pi$ **then**
    $\theta[m] = +\pi$; $\delta_o = -\delta_o$;
  end if
  if $\theta[m] < -\pi$ **then**
    $\theta[m] = -\pi$; $\delta_o = -\delta_o$;
  end if
end for

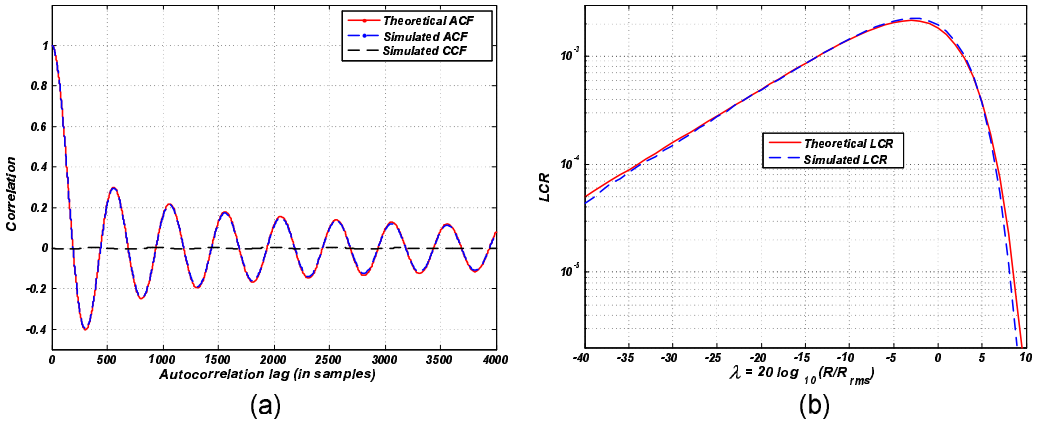


Figure 2.3: (a) Normalized ACF and CCF of 10^7 fading samples generated using *Model IV*. (b) LCR of generated fading samples and the reference LCR.

We performed several numerical simulations to verify the statistical properties of *Model IV*. In one simulation trial, we generated a block of 10^7 fading samples using $N = 8$ sinusoids and measured the time-averaged statistical properties. Figure 2.3 (a) plots the reference ACF along with the ACF and CCF of the samples generated with *Model IV*. As Figure 2.3 (a) shows, the measured ACF accurately matches the reference ACF and the measured CCF is very small. Also, the LCR of the envelope of the generated fading variates and the theoretical LCR for $f_D T_s = 0.002$ are plotted in Figure 2.3 (b). Here again a close match between the measured LCR and the desired LCR from equation (2.5) can be observed. A comparison between *Model IV* and the original *Model II* (from [1]) based on the mean square error of different statistics is provided later in this chapter.

To measure the hardware performance of the new fading simulator, *Model IV* was implemented as a Verilog HDL design and synthesized for three typical Xilinx Virtex-II Pro, Xilinx Virtex-4, and Altera Stratix FPGA devices. The results are summarized in Table 2.1.

Table 2.1: Implementation of the Fading Simulator on Three Different FPGAs

Device	XC4VSX55-11	XC2VP100-6	EP1S80F1508C6
Max. clock freq. (MHz)	195.61	204.75	103.01
Output rate (MSamps/sec)	195	204	103
Slice utilization	2447 (9%)	2444 (5%)	1292 (1%)
18 × 18 MULTs	48 (9%)	48 (10%)	128 (72%)
Number of BRAMs	12 (3%)	12 (2%)	12 (2%)

More details about the FPGA implementation can be found in [2].

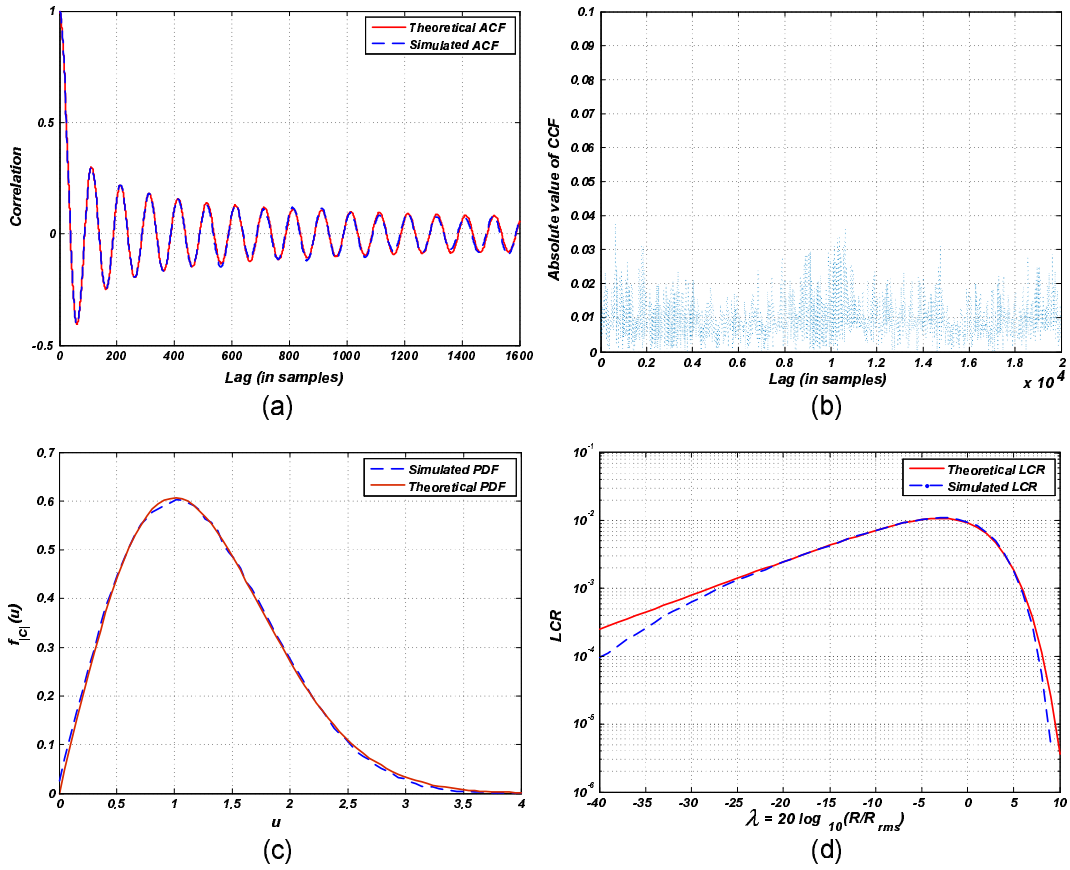


Figure 2.4: Simulation results for a block of 2×10^6 fading samples with $f_D T_s = 0.01$ generated with *Model V* using $N = 8$ sinusoids. (a) Normalized estimated ACF, (b) absolute value of normalized CCF, (c) estimated PDF, and (d) estimated LCR.

2.2.2 Proposed SOS-Based Rician and MIMO Fading Simulator

We now propose new modifications to our model [2] to reduce the cross-correlation levels and to cover Rician fading as well as Rayleigh fading. The new model also has a more efficient hardware implementation [3, 8, 86]. The modified version of *Model IV*, called *Model V*, generates the Rayleigh fading samples as

Model V:

$$c_i[m] = \sqrt{\frac{1}{N}} \sum_{n=1}^{N} \cos\left(2\pi f_D T_s m \cos\alpha_n[m] + \varphi_n[m]\right),$$

$$c_q[m] = \sqrt{\frac{1}{N}} \sum_{n=1}^{N} \cos\left(2\pi f_D T_s m \sin\alpha_n[m] + \psi_n[m]\right),$$

where $\varphi_n[m]$ and $\psi_n[m]$ are independent and stationary *random processes* (RPs). More specifically, compared to *Model IV*, in *Model V* we propose replacing the random phases φ_n and ψ_n with independent stationary *random processes* (RPs). Each random process $\theta[m]$, $\varphi_n[m]$, and $\psi_n[m]$, is updated using [8, Algorithm 1] which is, in essence, Algorithm 1 extended to update $\varphi_n[m]$ and $\psi_n[m]$. These modifications reduce the CCF between the generated streams of fading samples. Low CCF levels are particularly important for simulating independent fading paths for simulating multipath channels (including wide-band frequency selective channels) or MIMO channels.

To measure the statistical properties of the fading samples from *Model V*, we generated one block of 2×10^6 samples with $f_D T_s = 0.01$, and $N = 8$ sinusoids. Figure 2.4 (a) plots the ACF of the generated fading samples and the theoretical reference. Note that the estimated ACF matches the theoretical ACF, even at the larger lags. Figure 2.4 (b) shows the small cross-correlation between two sequences of 2×10^6 generated fading variates. Figure 2.4 (c) and Figure 2.4 (d) show close agreement between the theoretical functions and estimated PDF and the LCR of the fading channel.

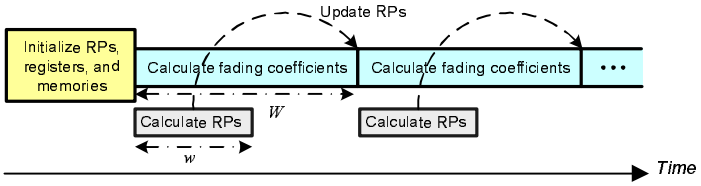


Figure 2.5: Dataflow for updating random processes and calculating fading coefficients.

We used a time-overlapped approach for the compact implementation of *Model V*. In this approach, each simulation trial is divided into shorter intervals, each of length W time units (e.g., clock cycles). To minimize the number of functional units and memories, instead of updating the RP values every clock cycle, their values are updated (with no loss in performance) every w clock cycles, where $1 < w \leq W$. Rather than computing the RPs in parallel, they can be calculated sequentially using time-shared arithmetic resources

but distinct register sets. The updated RP values are copied at the beginning of the next interval to another set of registers where they can be read during the next fading coefficient generation interval. This dataflow schedule is shown in Figure 2.5.

We will now present the computation of the in-phase $c_i[m]$ component of $c[m]$. The quadrature $c_q[m]$ can be evaluated similarly. First, we factor out the 2π and re-write $c_i[m]$ as

$$c_i[m] = \sqrt{\frac{1}{N}} \sum_{n=1}^{N} \cos\Big(2\pi\big((f_D T_s \cos\,\alpha_n[m]) \times m + \tilde{\varphi}_n[m]\big)\Big), \tag{2.14}$$

where $\tilde{\varphi}_n[m] \in [-0.5, 0.5)$. We can also write $\alpha_n[m]$ as

$$\alpha_n[m] = \frac{2\pi(n - 0.5 + \tilde{\theta}[m])}{4N}, \tag{2.15}$$

where $\tilde{\theta}[m] \in [-0.5, 0.5)$. Note that $\theta[m] \in [-\pi, \pi)$ and $\varphi_n[m] \in [-\pi, \pi)$ are rescaled as $\tilde{\theta}[m]$ and $\tilde{\varphi}_n[m]$ in (2.14), respectively, to lie within $[-0.5, 0.5)$.

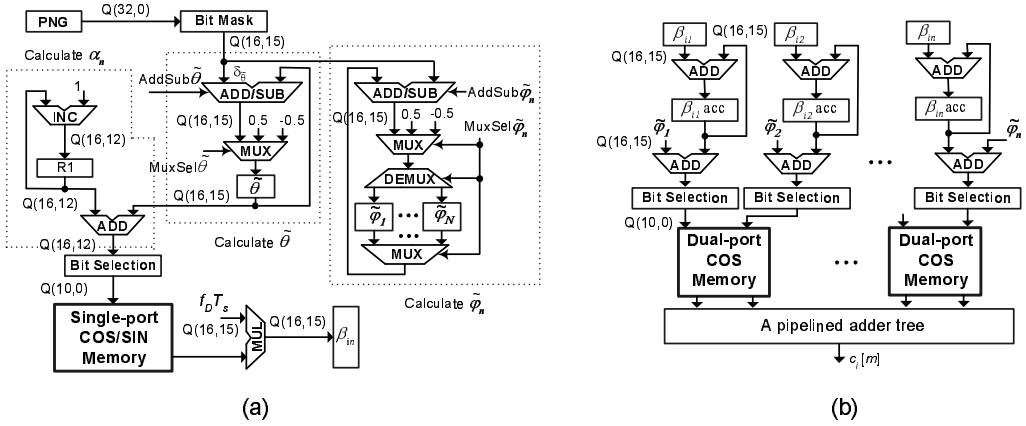


Figure 2.6: (a) Datapath for updating the random processes using shared hardware. (b) Datapath for calculating $(f_D T_s \cos\alpha_n[m]) \times m$ and generating $c_i[m]$.

To update the random walk process $\tilde{\theta}$ (and also the other RPs $\tilde{\varphi}_n$ and $\tilde{\psi}_n$), the random step size δ for each process is calculated by multiplying a generated uniform pseudo-random number (PN) in $(0, 1)$ by a positive value $\xi \ll 1$. Note that the fading variate generator must create long periods of non-repeating propagation conditions to evaluate the error-rate performance accurately and thereby ensure confidence in the test results. To ensure that test conditions do not repeat during test runs, we used a combined multiple-component linear pseudo-random number generator (PNG) [87] that has a very long period and substantially better randomness and correlation properties compared to conventional linear PNGs. To eliminate the multiplication operation, the value of ξ is chosen to be 2^{-d} for some suitable

exponent d. Note that the `Bit Mask` module masks the d most significant bits of the generated PNs and uses the result $\delta_{\tilde{\theta}}$ as a random updating step for the RP $\tilde{\theta}$, as shown in Figure 2.6 (a). The RPs $\tilde{\varphi}_1, \cdots, \tilde{\varphi}_n$ are updated similarly to $\tilde{\theta}$, as shown in Figure 2.6 (a).

According to (2.15), for $n = 1, \cdots, 8$, the lower and upper bounds of α_n are $[0, \pi/16)$ and $[7/16\pi, \pi/2)$, respectively. Therefore we only need to calculate the cosine values (for $c_i[m]$) and the sine values (for $c_q[m]$) of $\alpha_n[m]$ over the interval $[0, \pi/2)$. We uniformly quantize the cosine and sine values within $[0, \pi/2)$ over 512 sub-intervals and store their values in one of the Block RAM (BRAM) memories in the FPGA. This BRAM is used in the 1024×16 configuration with the first 512 words storing the cosine values and the second half storing the sine values, both in the 2's complement fixed-point format `Q(16,15)` (i.e., 16-bit words with the 15 least significant bits dedicated to the fractional part). The `Bit Selection` module selects the 10 least significant bits of the adder output (i.e., the value of α_n) and addresses the `COS/SIN Memory`. The N values of $\cos\,\alpha_n[m]$ are read from the `COS/SIN Memory` in N clock cycles and are multiplied by $f_D T_s$ sequentially to generate N values of $\beta_{i_n} = f_D T_s \cos\,\alpha_n$. The N values of $\beta_{q_n} = f_D T_s \sin\,\alpha_n$ are calculated similarly in the next N clock cycles. Since $f_D T_s \ll 1$ and since the sine and cosine values lie within $[-1, 1)$, β_{i_n} and β_{q_n} lie within $[-1, 1)$ and can be represented in `Q(16,15)`.

Figure 2.6 (b) shows the datapath for calculating $(f_D T_s \cos\alpha_n[m]) \times m$. After calculating $\beta_{i_n}[m] = f_D T_s \cos\alpha_n[m]$ (as shown in Figure 2.6 (a)), integer multiples of β_{i_n} can be obtained using an accumulator instead of a multiplier. Then the $\beta_{i_n}[m] \times m$ values are added to the $\tilde{\varphi}_n$ values and used to address $N/2$ dual-port cosine memories. Note that when calculating the $\cos(2\pi\gamma)$ functions, only the fractional part of the γ is required and the integer part can be ignored. Hence, the adders and registers used in Figure 2.6 (b) are only 16-bit modules. The outputs of the $N/2$ cosine memories are then passed to a pipelined adder tree to compute the in-phase component of $c[m]$. Computation of $c_q[m]$ can be performed simultaneously using a similar datapath to that in Figure 2.6 (b).

The registers and the operation of the functional modules for computing $c_i[m]$ and $c_q[m]$ are controlled using a W-state machine. The value of W depends on the number N of sinusoids, the functional dependency between operations, and also the number of pipeline stages in the longest path in the datapath that updates the RPs. For clarity let us ignore the number of pipeline stages. As shown in Figure 2.6 (a), since we use only one PNG and since updating $\tilde{\theta}[m]$, $\tilde{\varphi}_n[m]$ and $\tilde{\psi}_n[m]$ requires independent noise samples, these RPs

must be updated sequentially in $2N + 1$ clock cycles. After updating $\tilde{\theta}[m]$, values $\beta_{i_n}[m]$ and $\beta_{q_n}[m]$ can be calculated sequentially for N sinusoids in $2N$ clock cycles. Note that register $\tilde{\theta}$ is disabled during the calculation of β_{i_n} and β_{q_n}. The process of updating $\beta_{i_n}[m]$ and $\beta_{q_n}[m]$ takes place in parallel with the updating of $\tilde{\varphi}_n[m]$ and $\tilde{\psi}_n[m]$. Therefore, W can be any integer number greater than or equal $2N + 1$. For $N = 8$ sinusoids, and including several pipeline stages in the design, we chose $W = 32$. Note that register R_1 is initialized to 0.5 at the start (i.e., $n = 1$) of the $\cos\ \alpha_n[m]$ and $\sin\ \alpha_n[m]$ computations.

Table 2.2: FPGA Implementation Results for *Model V*

Device	**XC2VP100-6**	**XC2VP100-6**	**XC2V4000-6**
Model	*Model IV*	*Model V*	*Model V*
Max. clock freq. (MHz)	204.75	201.1	177.99
Output rate (MSamps/sec)	204	201	177
Configurable slices	2444 (5%)	1105 (2%)	1100 (4%)
18 × 18 MULTs	48 (10%)	1 (< 1%)	1 (< 1%)
Number of BRAMs	12 (2%)	9 (2%)	9 (7%)

The implementation results for the above fading simulator based on *Model V* using $N = 8$ sinusoids are summarized in Table 2.2. Our implementation on a Xilinx Virtex-II Pro XC2VP100-6 FPGA, uses 1105 of the 44096 configurable slices (2%), one of the 444 dedicated multipliers ($< 1\%$), and 9 of the 444 on-chip memory blocks (1%) while generating over 200 million 16-bit complex-valued fading samples per second. This table also compares the new implementation results with the FPGA implementation results for *Model IV* (from Table 2.1). As shown in Table 2.2, the previous fading channel simulator based on *Model IV* requires 10% of the dedicated multipliers on the same device. The reason that the previous implementation of *Model IV* (from [2]) uses such a large number of dedicated multipliers is because in order to compute $f_D T_s \times m$, in [2] we used a 48-bit register to accumulate successive multiples of $f_D T_s$ since m, which is a discrete-time index, can be a large integer. Then we used N multipliers to multiply the 12-bit $\cos\ \alpha_n[m]$ with the 48-bit value of $m f_D T_s$. Thus this earlier scheme is certainly not compact enough for larger systems with more than 10 channels given present FPGAs. However, the proposed scheme uses substantially fewer dedicated multipliers, fewer BRAMs, and also fewer configurable slices than the implementation in [2], while the slight decrease in fading generation rate is negligible. In [8], *Model V* has also been used for the simulate a MIMO fading channel with Kronecker model.

In [3], *Model V* is extended to cover Rician fading. The new Rician model (henceforth

called *Model VI*) can be expressed in discrete time as

Model IV:

$$r[m] = r_i[m] + jr_q[m],$$

$$r_i[m] = \big(c_i[m] + \sqrt{K}\cos(2\pi f_D T_s m \cos\theta_o + \phi_o)\big)/\sqrt{K+1},$$

$$r_q[m] = \big(c_q[m] + \sqrt{K}\sin(2\pi f_D T_s m \cos\theta_o + \phi_o)\big)/\sqrt{K+1},$$

where K is the Rice factor (the ratio of the specular power to the scattered power), and θ_o and ϕ_o are the AOA and the initial phase of the LOS component, respectively, which are uniformly distributed random variables over $[-\pi, \pi)$. In *Model VI*, the Rayleigh samples $c_i[m]$ and $c_q[m]$ are defined in as *Model V*.

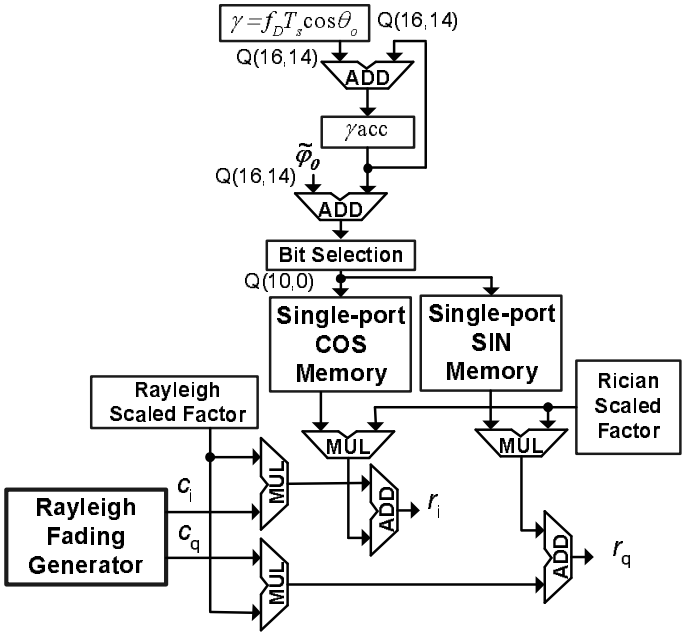


Figure 2.7: Datapath of generating Rician fading variates.

Hardware simulation of Rician fading according to *Model VI* is straightforward. In this model, the contribution of the scattered components (i.e., $c[m]$) is scaled and added to the contribution of the specular component. Here, the Rayleigh fading samples $\{c[m]\}$ are generated with the datapath in Figure 2.6 (b). Figure 2.7 shows the datapath for generating Rician fading variates.

Similar to the Rayleigh fading, the parameters of the cosine and sine functions in *Model VI* can be written as $2\pi(f_D T_s m \cos\theta_o + \tilde{\phi}_o)$, where $\tilde{\phi}_o \in [-0.5, 0.5)$. The integer multiples

of $f_D T_s \cos\theta_o$ are calculated using an accumulator, as shown in Figure 2.7. This value is then added to $\tilde{\varphi}_o$ to address the cosine and sine memories over the domain $[0, 2\pi)$. The outputs of the memories are multiplied by scaled Rice factors that are read from the primary inputs. The results are added later to the normalized Rayleigh coefficients.

Table 2.3: Comparison of the FPGA Implementation Results for *Model IV*, *Model V*, and *Model VI* on a Xilinx Virtex-II Pro XC2VP100-6 FPGA

Model	*Model IV*	*Model V*	*Model VI*
Fading Type	Rayleigh	Rayleigh	Rician
Max. clock freq. (MHz)	204.75	201.1	201.1
Output rate (MSamps/sec)	204	201	201
Configurable slices	2444 (5%)	1105 (2%)	1222 (2%)
18 × 18 MULTs	48 (10%)	1 (< 1%)	5 (1%)
Number of BRAMs	12 (2%)	9 (2%)	11 (2%)

Table 2.3 compares the FPGA implementation results between *Model IV*, *Model V*, and *Model VI* (from [2], [8], and [3], respectively). An FPGA implementation of the Rician fading channel simulator *Model VI* using $N = 8$ sinusoids on a Xilinx Virtex-II Pro XC2VP100-6 FPGA uses 1222 of the 44096 configurable slices (2%), 5 of the 444 dedicated multipliers (< 1%), and 11 of the 444 on-chip memory blocks (2%) while generating 201 million 16-bit complex-valued fading samples per second. Table 2.3 shows that the Rician fading simulator *Model VI* requires four more 18×18 on-chip multipliers and utilizes approximately 11% more configurable slices than the Rayleigh fading simulator *Model V*, but the two new simulators generate fading variates at the same rate. Thus the increased cost of providing Rician fading beyond Rayleigh fading is quite reasonable.

2.2.3 Compact Architecture for Fading Simulation

In [4] we proposed a compact architecture for the simulation of multipath Rayleigh channels based on *Model V*. In a typical wireless communication systems, since the Doppler frequency f_D is significantly smaller than the signal sample rate $F_s = 1/T_s$, the fading samples can be generated at significantly lower rates. Hence, rather than implementing N complex "oscillators," one time-shared datapath can be used for the compact implementation of the N complex "oscillators" in *Model V*.

When resources are shared for a compact hardware implementation, the fading sample generation rate will typically be reduced proportionally. To compensate for the throughput reduction, we utilize a linear interpolator to achieve the desired output sample rate. To use such an interpolator, the signal bandwidth should be small enough that the interpolator

response does not have a significant impact on the statistics of the generated samples. More specifically, if an initial sample rate $\hat{F}_s \geq 32 \times f_D$ is used, a simple linear interpolator can provide more than 80 dB of attenuation on the image signals (introduced by operating at $\hat{F}_s$ instead of F_s) with no significant effects on the desired signal.

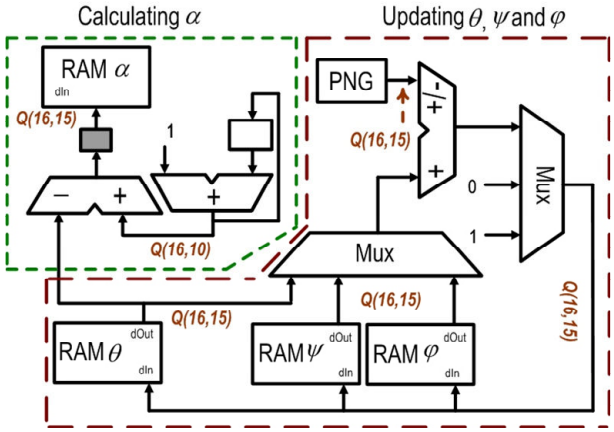


Figure 2.8: Datapath for updating phase angles.

This low-rate fading simulator generates the fading samples in three steps. In the first step, the *random walk processes* (RWPs) $\theta[m]$, $\varphi_n[m]$, and $\psi_n[m]$ are updated. In the second step, low-rate Rayleigh fading samples are generated at a frequency $\hat{F}_s \ll F_s$. Finally, in the third step, the low rate fading samples are interpolated up to F_s. Here we explain how these three steps are processed in the implemented hardware.

2.2.3.1 Updating the RWPs θ, φ_n, and ψ_n

We define $\hat{\alpha}_n[m] = \alpha_n[m]/(2\pi) = (n - \hat{\theta}[m])/(4N)$ where $\hat{\theta}[m] \in [0, 1)$ and thus $\hat{\alpha}_n[m] \in [0, 1/4)$. We also define $\hat{\varphi}_n[m] = (\pi + \varphi_n[m])/(2\pi)$ and $\hat{\psi}_n[m] = (\pi + \psi_n[m])/(2\pi)$ to be in the range $[0, 1)$. Note that shifting a random phase process by π does not change its statistical properties. Correspondingly, when implementing Algorithm 1, we normalize the random walk process to fall within $U(0, 1)$. The resulting datapath that updates the RWPs is shown in Figure 2.8, where the signals are represented in the 2's complement fixed-point format `Q(WL,WF)` (i.e., WL-bit words with the $WF \leq WL$ least significant bits dedicated to the fractional part). The pseudo-random number generator (PNG) generates uniformly-distributed samples. The common size of on-chip memories *RAM* α, *RAM* ϕ, and *RAM* ψ, is $L \times N \times 16$, where $L \geq 1$ is the number of independent faders and each sinusoid parameter is stored in 16-bit 2's-complement format. For exam-

ple, utilizing 18-Kb on-chip block memories on Xilinx FPGAs, each one of the parameters $\hat{\alpha}_n, \hat{\varphi}_n$ and $\hat{\psi}_n$ for $L = 32$ different faders with $N = 32$ can be stored in one BRAM, as shown in Figure 2.8. L different values of $\hat{\theta}$ are also stored in a memory *RAM* θ.

2.2.3.2 Generating the Rayleigh fading process

Based on the discrete-time definition of *Model V*, which operates at a sample rate F_s, we define a slow discrete-time fading signal $\acute{c}_i[m]$ (in-phase component) operating at sample rate $\hat{F}_s$ ($< F_s$) as follows:

$$\acute{c}_i[m] = \sqrt{\frac{1}{N}} \sum_{n=1}^{N} g\big(f(\frac{1}{4} - \hat{\alpha}_n[m])\frac{m}{64} + \hat{\varphi}_n[m]\big) \tag{2.16}$$

where $g(x) = \cos(2\pi x)$ and $f(x) = 64 \times (f_D/\hat{F}_s)\sin(2\pi x)$ for $x \in [0, 1/4)$. We choose $\hat{F}_s \geq 64 \times f_D$, hence the value of $f(x)$ is limited to the range $[0, 1]$. As shown in Fig. 2.9, the values of $f(x)$ for $x \in [0, 0.25)$ are precomputed by uniformly quantizing $f(x)$ over 1024 segments and then storing these values in *ROM* f. Note that the inner cosine function in (2.16) is obtained using the identity $\cos(2\pi x) = \sin(2\pi(1/4 - x))$. For $x \in [0, 1/4)$, $1/4 - x$ can be calculated using the negation *NEG* operation.

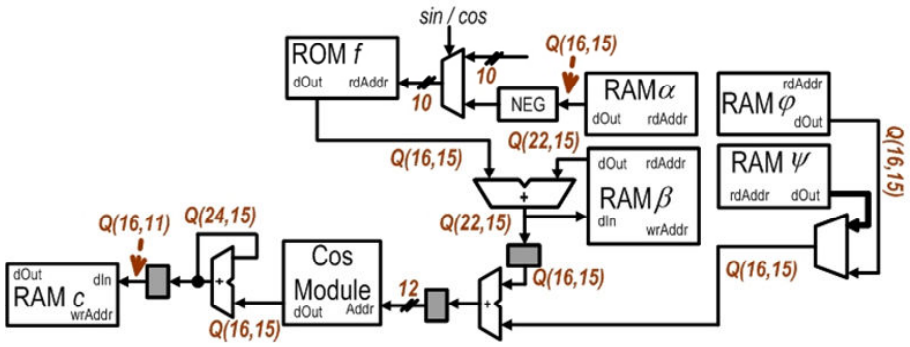


Figure 2.9: Datapath for generating the in-phase component of the fading process.

The repeated multiplication of m and $f(\cdot)$ in (2.16) can be replaced with a running summation as follows:

$$f(\frac{1}{4} - \hat{\alpha}_n[m])m \approx \beta_{in}[m] = \sum_{j=1}^{m} f(\frac{1}{4} - \hat{\alpha}_n[j]). \tag{2.17}$$

Note that $\beta_{in}[m]$ can be written in recursive form as $\beta_{in}[m] = \beta_{in}[m-1] + f(1/4 - \hat{\alpha}_n[m])$ with $\beta_{in}[-1] = 0$. With similar modifications to $\acute{c}_q[m]$, the resulting simplified in-phase

and quadrature components can be written as follows:

$$\acute{c}_i[m] \approx \sqrt{\frac{1}{N}} \sum_{n=1}^{N} g(\beta_{in}[m]/64 + \hat{\varphi}_n[m]), \tag{2.18}$$

$$\acute{c}_q[m] \approx \sqrt{\frac{1}{N}} \sum_{n=1}^{N} g(\beta_{qn}[m]/64 + \hat{\psi}_n[m]), \tag{2.19}$$

where $\beta_{qn}[m]$ is defined recursively as $\beta_{qn}[m] = \beta_{qn}[m-1] + f(\hat{\alpha}_n[m])$ with $\beta_{qn}[-1] = 0$. Our bit-true fixed-point simulations show that the word length of β_{in} and β_{qn} has a significant impact on the output statistics. Through experimentation we found that the `Q(22,15)` format provides enough accuracy for the computation. The $2 \times L \times N$ values of β_{in} and β_{qn} are stored in memory *RAM* β, as shown in Figure 2.9. Also, the *Cos Module* is used to calculate the $g(x)$ function using look-up tables. The first quarter cycle of the cosine function is quantized into 1024 segments and the resulting values are stored in an on-chip BRAM. The value of $g(x)$ over $(0, 1)$ is calculated using the values of the first quarter cycle. The outputs of the *Cos Module* are accumulated to compute scaled values of (2.18) and (2.19), which are then stored in memory *RAM* c.

2.2.3.3 Interpolation

In this step, fading samples generated at $\hat{F}_s$ samples per second are oversampled and interpolated I times to provides samples at the target sample rate $F_s = I \times \hat{F}_s$. The linear interpolator requires the discrete difference between two successive low-frequency samples $y[mI]$ and $y[(m+1)I]$ to generate the interpolated fading samples $y[mI + i]$, where $i = 0, 1, \cdots, I-1$, which can be expressed as

$$y[mI + i] = \frac{(y[(m+1)I] - y[mI])i}{I} + y[mI]. \tag{2.20}$$

To avoid the multiplication and division operations, we use an accumulator and the divisor I is chosen to be a power of two. Thus

$$y[mI + i] = \sum_{j=0}^{i} \frac{y[(m+1)I] - y[mI]}{I} + y[mI], \tag{2.21}$$

and the interpolator is implemented as shown in Figure 2.10. The interpolator contains a 24-bit accumulator and one register that holds the value of the input signal for an interval of I samples.

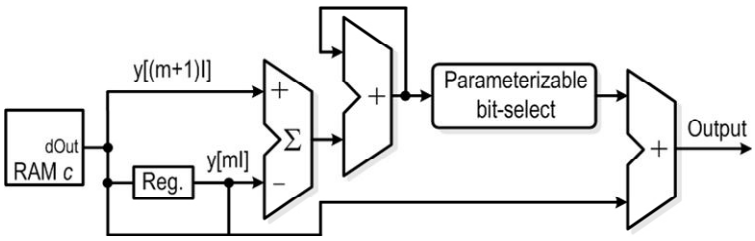


Figure 2.10: Interpolator structure.

An important property of the proposed architecture is that since the same datapath is shared to update the sinusoid parameters and to calculate the superposition of sinusoids, the number N of sinusoids only impacts the size of the memories used to store the sinusoid parameters. The computational complexity is directly proportional to the number L of faders since, for every stream of in-phase and quadrature samples, one instantiation of the interpolator shown in Figure 2.10 is required. For example, the implementation of the low-rate fading simulator with $L = 32$ paths, which requires 64 independent interpolators for the in-phase and quadrature components, uses 6899 (7%) of the configurable slices in a Xilinx Virtex-4 XC4VLX200-11 FPGA. The synthesis results show that 6214 out of 6899 configurable slices (i.e., 91%) used for the entire fading channel simulator are used by the interpolators.

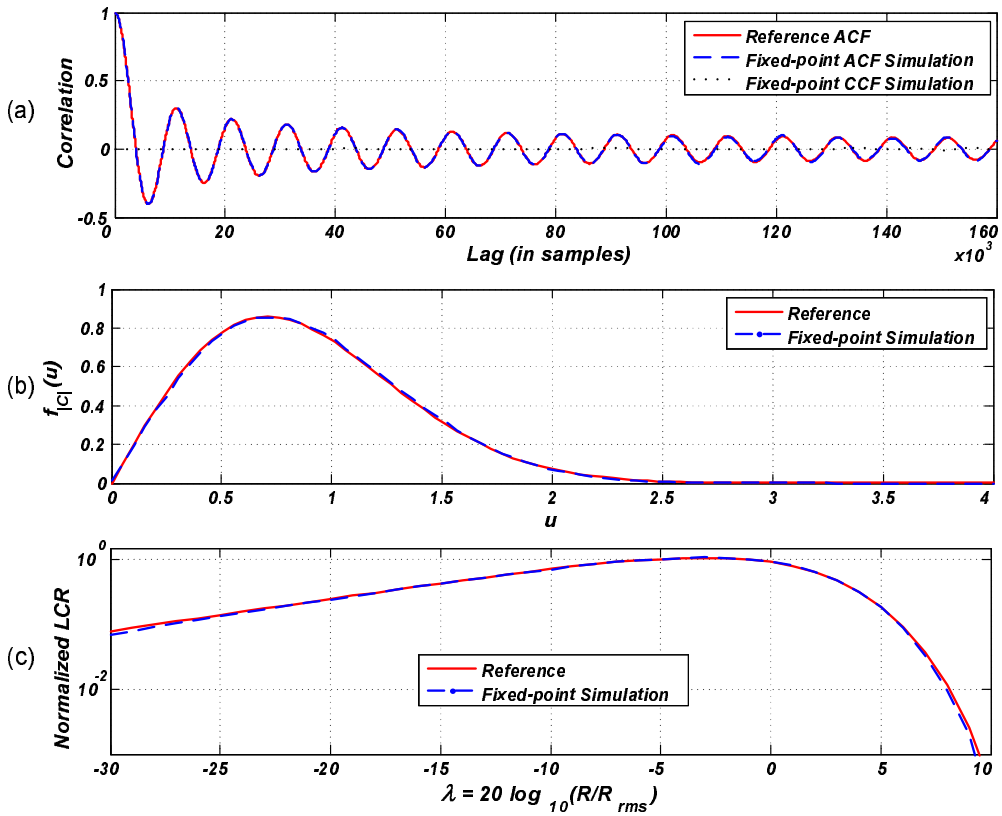


Figure 2.11: (a) Normalized ACF and CCF of the $c[m]$ for one block containing 2×10^6 fading samples with $f_D T_s = 0.0001$ and $N = 16$. (b) PDF of the fading envelope $|c[m]|$. (c) Normalized LCR.

To verify the accuracy of the Rayleigh fading simulator, we compared the bit-true fixed-point simulation results of important statistical properties of the generated fading variates with the theoretical functions. The simulations were performed using $N = 16$ sinusoids. Four major statistical measures, namely ACF, CCF, PDF, and LCR were examined against the theoretical references in equations (2.2), (2.3), (2.1), and (2.5) respectively. The plots in Figure 2.11 show excellent agreement between the fixed-point bit-true simulated results and the corresponding theoretical curves.

Based on the above methodology, we modified this design to simulate Rician fading channels using *Model VI*. For a fairer comparison with other designs, we implemented the fading channel simulator with different numbers L of paths on a Xilinx Virtex-II Pro XC2VP100-6 FPGA. Table 2.4 compares the characteristics of two implementations of the new fading simulator with the fading channel simulators in [59] (implemented on an Altera APEX EP20K1000EBC652-3 FPGA) and [3] (for *Model VI*). Note that the new $L = 32$

Table 2.4: Comparison Between FPGA Implementation Results

Model	A [59]	B [3]	C *NEW*	D *NEW*
FPGA Device	EP20K1000	XC2VP100	XC2VP100	XC2VP100
Fading type	Rayleigh	Rician	Rician	Rician
Number of paths L	16	32	32	64
Number of sinusoids N	16	8	32	32
Max. clock freq. (MHz)	50.0	201.1	238.94	238.94
Output rate	16×1.5	32×201	32×238	64×238
Configurable slices	58%	87%	15%	29%
18×18 MULTs	-	36%	0.2%	0.2%
BRAMs	17%	79%	2.0%	2.0%

path fading simulator is 82% smaller and 18% faster than the previous implementation for *Model VI* despite the fact that the simulator uses four times more sinusoids (and hence has greater statistical accuracy). Moreover, comparing the number of paths L, the number of sinusoids N, and the maximum speed of the designs, the new fading simulator is significantly faster and more efficient than the fading simulator reported in [59] while providing higher statistical accuracy.

2.2.4 High Path Count Rician and MIMO Fading Simulator

In [5], we proposed new modifications to the fading simulator in [4] (represented in Section 2.2.3) and made it significantly more efficient for hardware implementation. The new fading simulator basically generates the discrete difference between fading samples at a low sample rate. A simplified interpolator is then used to provide the target sample rate. The compact implementation of this fading simulator makes it an especially appropriate candidate for simulating fading scenarios with large number of paths, like multipath and MIMO fading channels [5]. Here we explain how this fading simulator is implemented.

2.2.4.1 Modifications

Let's start with equation (2.21). Note than when I is chosen to be a power of two (i.e., $I = 2^k$), the interpolator (2.21) can be implemented without multiplications or divisions. Note also that the interpolator requires the discrete difference between two subsequent low-frequency samples. In addition, since the difference is a linear and time-invariant operation, it can be performed before adding in any Rician specular (i.e., LOS) component. The discrete difference signal for the Rayleigh in-phase component in (2.18) is thus

$$d_i[m] = \sum_{n=1}^{N} \frac{(g(\frac{\beta_{in}[m]}{64} + \hat{\varphi}_n[m]) - g(\frac{\beta_{in}[m-1]}{64} + \hat{\varphi}_n[m-1])}{\sqrt{N}}. \tag{2.22}$$

By substituting (2.22) into (2.21), one can verify that the in-phase Rayleigh fading samples of *Model V* at sample rate F_s can be approximated by

$$\begin{aligned}\hat{c}_i[mI+u] &= \acute{c}_i[0] + \sum_{\acute{m}=0}^{m-1} d_i[\acute{m}] + 2^{-k} d_i[m]u \\ &= \acute{c}_i[0] + \sum_{\acute{m}=0}^{m-1} d_i[\acute{m}] + 2^{-k} \sum_{j=1}^{u} d_i[m], \end{aligned} \tag{2.23}$$

where $k = log_2(I)$, $m = 0, 1, 2, \cdots$, and $u = 0, 1, \cdots, I-1$. Equation (2.23) shows that with the above modifications, the interpolation operation can be simplified to a discrete difference, a shifting operation, and an accumulation (the two summations in equation (2.23) can be implemented with only one accumulator).

Next, the discrete difference of the LOS component is added to the Rayleigh fading samples. The discrete difference signal for Rician samples can be written as (see *Model VI*)

$$s_i[m] = \frac{1}{\sqrt{1+K}} d_i[m] + \sqrt{\frac{K}{1+K}} (g(\lambda[m]) - g(\lambda[m-1])), \tag{2.24}$$

where $\lambda[m] = \sum_{\acute{m}=1}^{m} (\eta/64) + \hat{\phi}_o$, $\hat{\phi}_o = \phi_o/(2\pi)$, and $\eta = 64 \times (f_D \hat{T}_s)\cos(\theta_0)$. The in-phase component of the final Rician samples are interpolated with an accumulator as follows:

$$\begin{aligned}\hat{r}_i[mI+u] &= \acute{r}_i[0] + \sum_{\acute{m}=0}^{m-1} s_i[\acute{m}] + 2^{-k} s_i[m]u \\ &= \acute{r}_i[0] + \sum_{\acute{m}=0}^{m-1} s_i[\acute{m}] + 2^{-k} \sum_{j=1}^{u} s_i[m], \end{aligned} \tag{2.25}$$

where $\acute{r}_i[0] = (\acute{c}_i[0] + \sqrt{K} g(\hat{\phi}_o))/\sqrt{1+K}$. The interpolated quadrature component $\hat{r}_q[mI+i]$, for $m = 0, 1, 2, \cdots$ and $u = 0, 1, \cdots, I-1$, can be calculated similarly.

2.2.4.2 Hardware Model

We now describe an efficient hardware design for an especially compact and high-throughput simulator based on the above simplified fading channel model. Without loss of generality we explain the design for 32 channels with each channel providing Rayleigh fading with $N = 32$ sinusoids. The architecture of this fading simulator consists of two cascaded stages. In the first stage, the complex sinusoids are generated at the sample rate $\hat{F}_s$. Since $\hat{F}_s$ is much slower than the target sample rate F_s, a common data path can be time-shared to interleave the calculations for different paths. The various required waveform parameters

are stored in different random-access memories to allow the samples to be updated using a shared datapath. However, every low-speed stream of fading samples must be interpolated with a dedicated interpolator. This way, the low-speed calculations are performed more efficiently with the least amount of hardware. Note that the final sample rate of the fading simulator depends on the maximum speed of the interpolator if different clock sources are used for the first (wave superposition) and second (interpolation) stages.

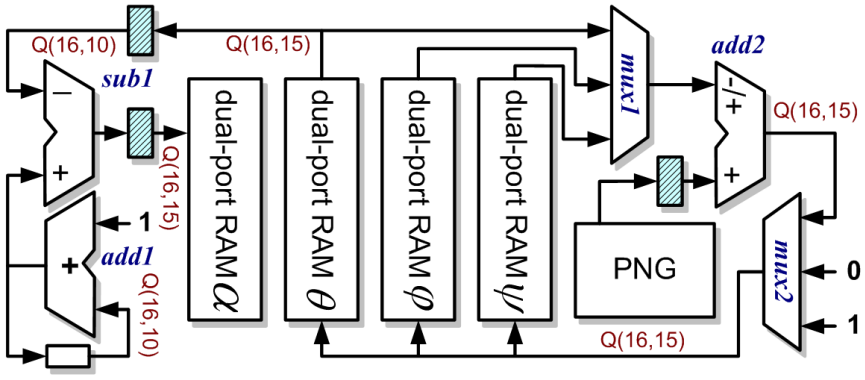


Figure 2.12: Datapath for generating the random phase processes.

Here, a time-overlapped approach is used where one simulation trial is divided into shorter intervals, each of length W clock cycles. To minimize the number of FPGA functional units and memories, instead of updating the RP values every clock cycle, without significant loss in final accuracy their values are updated every w clock cycles, where $1 < w \leq W$. The RPs are calculated sequentially using time-shared arithmetic resources but distinct memory locations. The datapath shown in Figure 2.12 updates the random phases $\hat{\alpha}_n[m]$, $\hat{\varphi}_n[m]$ and $\hat{\psi}_n[m]$. These random parameters are stored in three block memories "RAM α", "RAM φ" and "RAM ψ", respectively, in `Q(16,15)` format each of depth 1024 words. Also, 32 values of $\hat{\theta}$ are stored in the dual-port distributed memory "RAM θ" in `Q(16,15)` format. The random processes $\hat{\theta}$, $\hat{\varphi}_n[m]$ and $\hat{\psi}_n[m]$ are updated according to a modified version of the algorithm described in [3] (see Section 2.2.2).

A fading variate generator must create long periods of non-repeating propagation conditions to evaluate accurately the error-rate performance of the communication system under evaluation. The quality of the modeled conditions depends on the quality of an underlying pseudo-random number generator (PNG). To ensure this, we used a combined linear PNG [87] that has a very long period and substantially better randomness and correlation properties compared to conventional linear PNGs. In addition to updating the ran-

dom phases, the datapath in Figure 2.12 updates the value of $\hat{\alpha}_n[m] = (n - \hat{\theta})/(4N)$ for $n = 1, 2, \cdots, N$.

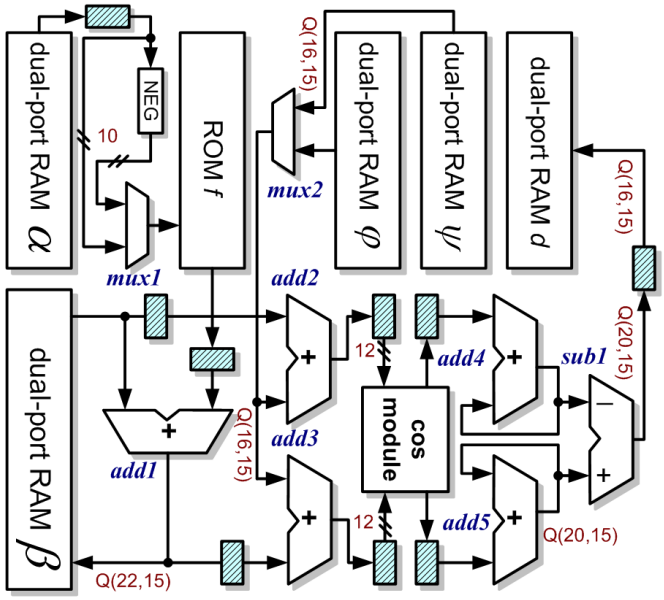


Figure 2.13: Datapath for generating Rayleigh fading samples.

Figure 2.13 shows the datapath for calculating the sequences of differences, $d_i[m]$ and $d_q[m]$. In our example design, $1024 = 32 \times 32$ values of $\hat{\alpha}_n[m]$ are stored in dual-port block memory "RAM α" in `Q(16,15)` format. The function $f(x) = 64 \times (f_D/\hat{F}_s)\sin(2\pi x)$ for $x \in [0, 1/4)$ is precomputed and quantized in `Q(16,15)` format in 1024 steps and stored in "ROM f". To calculate the sine function, the value of $\hat{\alpha}_n[m]$ is passed to "ROM f" after proper bit selection. To calculate the cosine function, the reformatted value $\hat{\alpha}_n[m]$ is first passed through a negation circuit and then passed to "ROM f".

Through extensive fixed-point simulations we found that the choice of fixed-point representation for $\beta_{in}[m]$ and $\beta_{qn}[m]$ has a great impact on the output statistics. Specifically, we found that the `Q(22,15)` fixed-point representation provides enough accuracy for our purposes. $2048 = 2 \times 32 \times 32$ values of $\beta_{in}[m]$ and $\beta_{qn}[m]$ are stored in dual-port block memory "RAM β". The $\beta_{in}[m]$ and $\beta_{qn}[m]$ values are updated using adder "add1" according (2.17) after reformatting the $f(\cdot)$ values from `Q(16,15)` to `Q(22,15)` format. Moreover, $\hat{\varphi}_n$ and $\hat{\psi}_n$ from two-port block memories "RAM φ" and "RAM ψ" are used to compute phases in (2.18) and (2.19) after proper bit selection. In Figure 2.13, "cos module" provides $g(x) = \cos(2\pi x)$ values for two inputs from a look-up table. According to our fixed-point simulations, to ensure acceptable statistical accuracy, the look-up table for $g(x)$

in `Q(16,15)` format should have at least 4096 entries (requiring 12-bit addressing). For a more efficient implementation, only the first quarter cycle of $g(x)$ (i.e., $x \in [0, 1/4)$) is stored in an on-chip block memory. For $x \in [1/4, 1)$, we can find the corresponding values of $g(x)$ based on the values from the first quarter cycle. The outputs of the "cos module" are then passed to accumulators "add4" and "add5" to compute scaled copies of (2.18) and (2.19). The outputs of "add4" and "add5" are then passed to "sub1", which computes the corresponding differences $d_i[m]$ and $d_q[m]$ (see (2.22)) in `Q(20,15)` format. After scaling and proper bit selection, 32 values of $d_i[m]$ and $d_q[m]$ are stored in a distributed memory of depth $64 = 2 \times 32$ in `Q(16,15)` format.

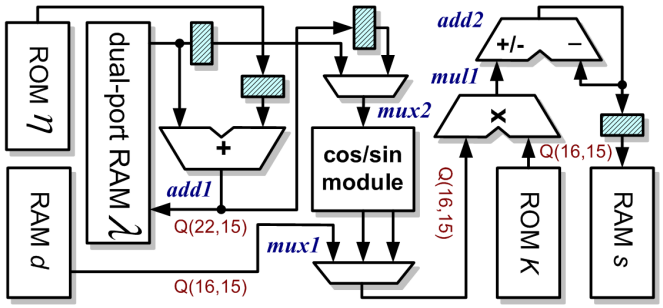


Figure 2.14: Datapath for converting Rayleigh fading samples into Rician fading samples.

The specular component is added to the Rayleigh samples using the datapath shown in Figure 2.14. Here, $N = 32$ values of $\eta = 64 \times (f_D/\hat{F}_s)\cos(\theta_o)$ are stored in single-port memory "ROM η" and used to update the λ values stored in memory "RAM λ" in `Q(22,15)` format. Memory "RAM λ" is initialized with $\hat{\phi}_o$ values. The "sin/cos module" reads the sine or cosine of the λ for the specular component (see *Model VI*) from a look-up table. Moreover, memory "ROM K" holds $64 = 2 \times 32$ values of $1/\sqrt{1+K}$ and $\sqrt{K/(1+K)}$ in `Q(16,15)` format. Multiplier "mul1" performs the four multiplications required to calculate $s_i[m]$ and $s_q[m]$ (see (2.24)). Adder/subtracter "add2" accumulates different components of (2.24) the result of which, after proper bit selection, is stored using format `Q(16,15)` in distributed memory "RAM s" of depth $64 = 2 \times 32$.

The datapath of the interpolator is shown in Figure 2.15. Note that one interpolation branch is dedicated to every in-phase and quadrature stream of samples ($64 = 2 \times 32$ interpolation branches in total). Each interpolation branch consists of a 24-bit accumulator and a register that holds the input signal for an interval of I samples (see (2.21)). Data from memory "RAM s" is read and stored in these registers with specific timing. A decoder

circuit selects which interpolator branch should store the present output data from "RAM s". The interpolator circuit generates the final 32 streams of independent complex Rician fading samples at the desired output sample rate.

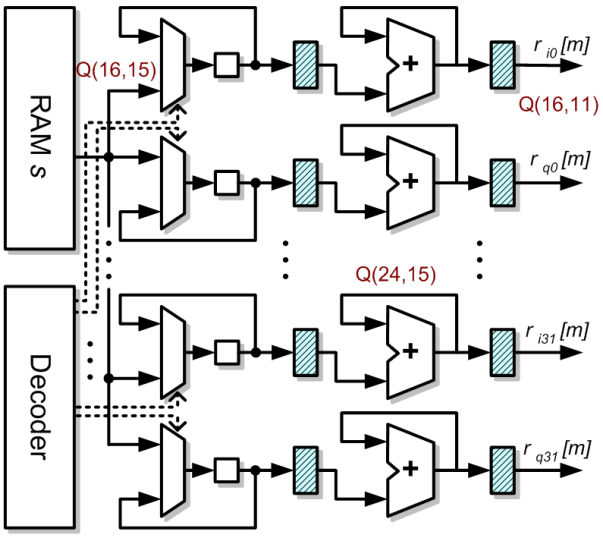


Figure 2.15: Datapath for the interpolation.

2.2.4.3 Results

To demonstrate the performance of this fading simulator, we first implemented a fixed-point bit-true model in the compiled language MEX (C for MATLAB) and generated sequences of fading variables. We simulated different Rayleigh and Rician fading scenarios for a fading channel with $f_D T_s = 0.001$ and $N = 32$ sinusoids.

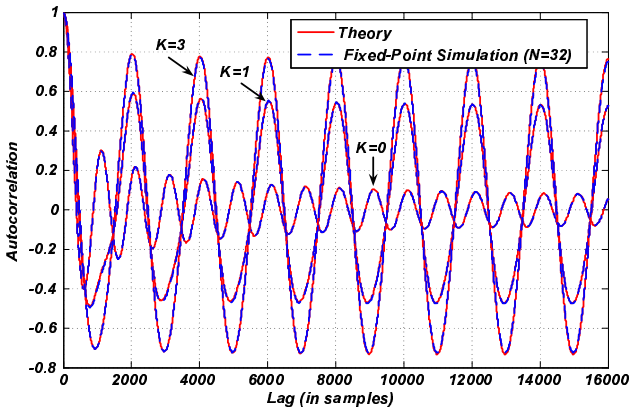


Figure 2.16: Normalized autocorrelation of the generated fading samples (real part) for one block containing 2×10^6 samples generated in a fixed-point simulation with $f_D T_s = 0.001$, $\theta_0 = \pi/3$, and $N = 32$ for $K = 0$ (Rayleigh), 1 and 3.

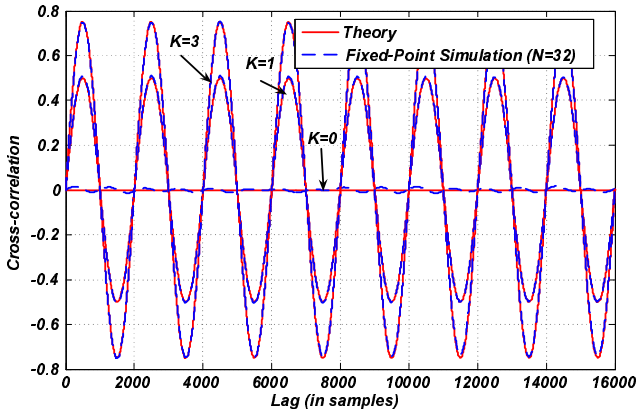


Figure 2.17: Normalized cross-correlation between the in-phase and quadrature components of the generated fading samples for one block containing 2×10^6 samples generated in a fixed-point simulation with $f_D T_s = 0.001$, $\theta_0 = \pi/3$, and $N = 32$ for $K = 0$ (Rayleigh), 1 and 3.

Figure 2.16 demonstrates the autocorrelation for 2×10^6 in-phase and quadrature components of the generated fading samples for Rice factors $K = 0, 1$ and 3. The theoretical reference ACFs and CCFs between the quadrature components of the fading samples are given by equations (2.10) and (2.11), respectively. As Figure 2.16 confirms, there is a close match between the expected analytical autocorrelation plots and the generated fixed-point simulation results. Also, Figure 2.17 plots the cross-correlation between the in-phase and quadrature components of the generated fading samples and the analytical curves. This figure again shows a close match between the fixed-point simulation results and the desired curves.

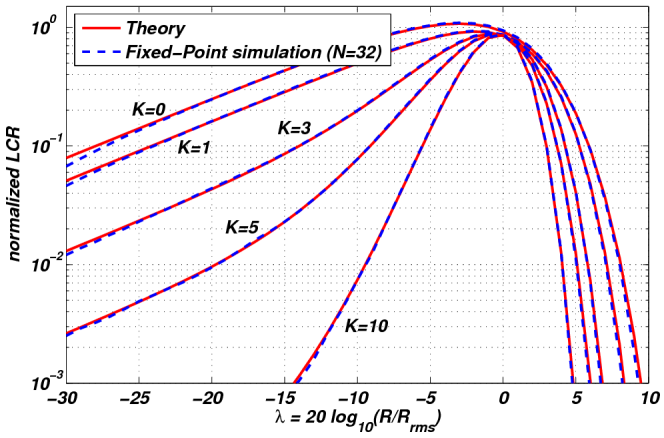


Figure 2.18: Normalized level crossing rate (LCR) function for one block containing 2×10^6 fading samples generated in a fixed-point simulation with $\theta_0 = \pi/3$ and $N = 32$ sinusoids, for $K = 0$ (Rayleigh), 1, 3, 5 and 10.

Figure 2.18 shows the normalized LCR of the amplitude of generated complex fading samples. The LCR is normalized to $f_D T_s$. The theoretical LCR from equation (2.12) is plotted too. Figure 2.18 shows excellent agreement between the theoretical (from equation (2.12)) and fixed-point simulation results for different values of Rice factor K.

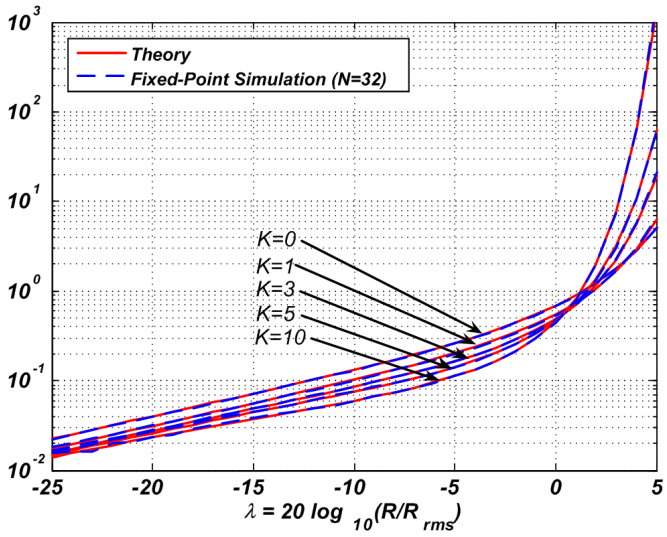


Figure 2.19: Normalized average fade duration (AFD) function for one block containing 2×10^6 fading samples generated in a fixed-point simulation with $\theta_0 = 0$ and $N = 32$ sinusoids, for $K = 0$ (Rayleigh), 1, 3, 5 and 10.

Figure 2.19 shows the normalized AFD of the generated samples in our fixed-point simulation for 2×10^6 samples and different values for Rice factor K. To have distinct curves for illustration, in this simulation the angle of arrival for the specular component is set to $\theta_o = 0$. As this figure shows, the results of our fixed-point simulation closely match the theoretical references from equation (2.13).

Table 2.5: Comparison Between FPGA Implementation Results

Model	B [3]	C [4]	E (*NEW*)
Number of paths L	32	32	32
Number of Sinusoids N	8	32	32
Number of complex waves	256	1024	1024
Max. clock freq. (MHz)	201.1	238.9	224.2
Output rate (MSamp/sec)	32×201	32×238	32×276 [a]
Configurable slices	39104 (87%)	6894 (15%)	2151 (4%)
18 × 18 MULTs	160 (36%)	1 (0.2%)	1 (0.2%)
Number of BRAMs	352 (79%)	9 (2.0%)	9 (2.0%)

[a] The sample rate of the new fading simulator depends on the maximum speed of the interpolator (here 276.7 MHz) if different clock sources are used for the first (wave superposition running at a maximum frequency of 224.2 MHz) and the second (interpolation) stages. If one clock source is used, the maximum sample rate will be 32×224 million complex fading samples per second.

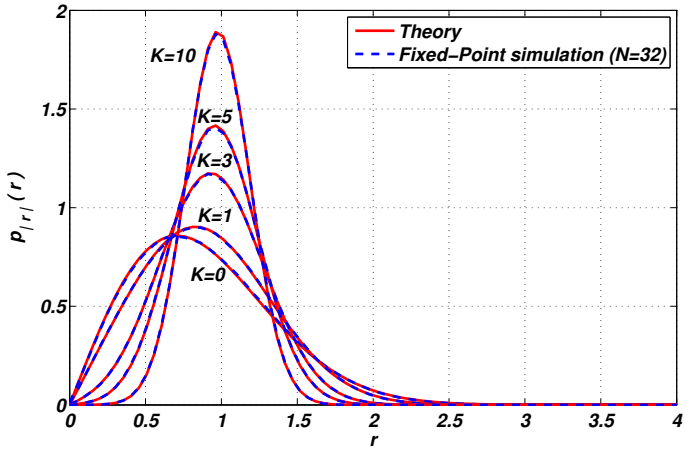


Figure 2.20: Probability density function (PDF) for one block containing 2×10^6 fading samples generated in a fixed-point simulation.

Finally, Figure 2.20 plots the PDF of the amplitude of the generated fading samples. Once again it can be observed that this PDF accurately reproduces its reference value in equation (2.9).

We implemented the new fading simulator on a Xilinx Virtex-II Pro XC2VP100-6 FPGA. As in our example simulator, we configured the hardware to generate 32 independent streams of Rician fading samples using $N = 32$ sinusoids. For each fading stream, our FPGA implementation uses 2151 of the 44096 configurable slices (4%), only one of the 444 dedicated 18×18 multipliers ($< 1\%$), and nine of the 444 on-chip memory blocks (2%). When implemented with one clock source, our fading generator can generate up to 32×224 million 16-bit complex-valued fading samples per second. However, when imple-

mented with two clock sources (for wave superposition and for interpolation), this figure rises to 32×276 million samples per second, which is determined by the maximum speed of the interpolation circuit in Figure 2.15. Table 2.5 summarizes the characteristics of the new fading simulator.

This table also compares the new implementation with the fading channel simulators in [3] (presented in Section 2.2.2) and [4] (presented in Section 2.2.3). Our fading channel simulator in [3] requires 39104 (87%) configurable slices, 160 dedicated 18×18 (36%) multipliers, and 352 (79%) on-chip block memories and provides 32 independent streams of Rician fading samples utilizing $N = 8$ sinusoids and runs at a maximum frequency of 201.1 MHz. By comparison, the new fading simulator is more accurate (32 sinusoids versus 8 sinusoids), 18 times smaller (in terms of number of slices), and 37% faster than the previous design. Also compared to our previous implementation in [4] ((from Section 2.2.3), the new fading simulator is more than 3 times smaller.

The compactness of the new fading simulator arises mainly because of the efficient implementation of the interpolation circuit. Since each in-phase and quadrature component of a fading sample requires an interpolator, the efficiency of the interpolation is directly related to how compactly the fading simulator is implemented. In our latest implementation of an SOS-based fading simulator, the interpolator was reduced to a simple accumulator which in turn resulted in compact and efficient implementation of the fading simulator.

2.3 Accuracy and Efficiency Comparisons

Detailed simulation and implementation results for different proposed models and architectures are provided in their corresponding publications. Here, we provide a comparison between accuracies of different designs in terms of mean square error (MSE) for different statistical characteristics. The MSE is defined as the average square error between the measured statistics and the theoretical targets (from equations (2.2), (2.3), (2.1), (2.5), and (2.6)) over a specified range.

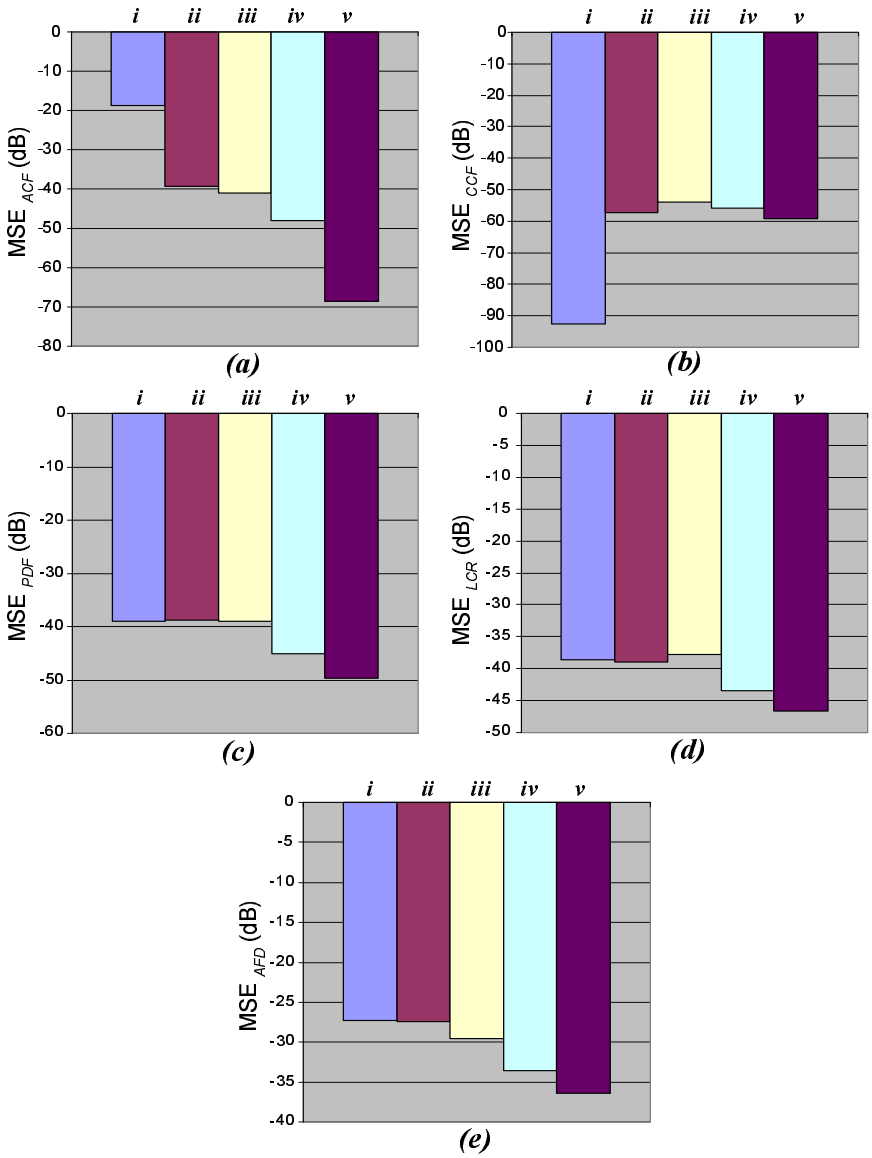


Figure 2.21: Mean square error (MSE) for different statistical measures of the new fading simulators: (a) MSE of ACF, (b) MSE of CCF, (c) MSE of PDF, (d) MSE of LCR, and (e) MSE of AFD. The MSE values are measured over one continuous block of Rayleigh fading samples of length 10^7 with normalized Doppler frequency $f_D T_s = 0.01$. The simulated fading models are (*i*) the model proposed in [1] (see Section 2.1), (*ii*) the model proposed in [2] (see Section 2.2.1), (*iii*) the model proposed in [3] (see Section 2.2.2), (*iv*) the model proposed in [4] (see Section 2.2.3), and (*v*) the model proposed in [5] (see Section 2.2.4).

2.3.1 Accuracy Comparison

We compared the statistical accuracy of different fading simulators. We measured the mean square error of different statistical measures (ACF, CCF, PDF, LCR, and AFD) over one

continuous block of Rayleigh fading samples of length 10^7 with normalized Doppler frequency $f_D T_s = 0.01$. The output statistics of the fading simulators proposed in [1–5] (represented in Sections 2.1, 2.2.1, 2.2.2, 2.2.3, and 2.2.4) were gathered in simulation. Figure 2.21 (a) compares the MSE of the ACF of of different designs. This MSE is measured with over a lag of 3200 samples. As this figure shows, the improved SOS model proposed in [2] (from Section 2.1) provides a 20 dB reduction in ACF-MSE compared to the model in [1] (*Model II* in Section 2.2.1). Also, Figure 2.21 (a) shows that the fading simulator in [5] (see Section 2.2.4) reduces the ACF-MSE by almost 50 dB. Figure 2.21 (b) compares the CCF-MSE for different model. As this figure show, *Model II* (from [1]) has lower CCF than other designs. However, the CCF-MSE in [2–5] is probably small enough (CCF-MSE < -50 dB) for practical purposes. In addition, in Figure 2.21 (c) the PDF-MSE is illustrated and shows that the PDF of our latest simulators are more accurate than the original models. In Figures 2.21 (d) and (e) the LCR-MSE and AFD-MSE of different models are plotted. These figures also show the improvement in the statistics of our fading simulators.

2.3.2 Efficiency Comparison

Figure 2.22 compares the implementation results for different SOS-based fading simulators. All of the fading simulators are implemented in Verilog HDL and synthesized on a Xilinx Virtex-II Pro FPGA XC2VP100-6. The implemented fading simulators are from [2–8] (from Sections 2.2, 2.2.1, 2.2.2, 2.2.3, and 2.2.4). The resource utilization figures are divided up based on the number of generated fading paths.

Figure 2.22 (a) compares the maximum clock frequency of different designs. All of the designs are fully pipelined. The maximum clock frequency among all of these designs belongs to the design in [4] (from Section 2.2.3). Figure 2.22 (b) compares the maximum output sample rate of the different designs [1]. Note that all of the designs are fully pipelined to achieve maximum throughput. As this figure shows, in the designs reported in [2, 3, 6–8] the output rates are similar. However in the two latest designs [4,5] (from Sections 2.2.3 and 2.2.4), the hardware has become faster because of the two-stage design. More specifically, the design reported in [5] can generate up to 276 million samples per second (only one

[1] The fading simulator proposed in [5] is designed in two stages and its output sample rate depends on the maximum speed of the interpolator (here 276.7 MHz) if different clock sources are used for the first (wave superposition running at a maximum frequency of 224.2 MHz) and the second (interpolation) stages. If one clock source is used, the maximum sample rate will be 224 million complex fading samples per second.

path).

Figure 2.22 (c) compares the required number of configurable slices per fading path. As this figure shows, the designs in [4, 5] are the most compact. More specifically, the design in [5] requires only 68 slices per fading path with $N = 32$ sinusoids.

Figures 2.22 (d) and (e) compare the required number of on-chip multipliers and block memories per fading path. As this figure shows, newer designs require fewer and fewer multipliers and less storage per fading path. Finally, Figure 2.22 (f) compares the number of sinusoids N in all of the fading simulators, which could be used as a measure of accuracy.

It can be seen that in the process of this work, the SOS-based fading simulator's design has evolved so that our latest fading simulator is

1. Smaller than all of the other known designs (only 68 slices per path, 1 multiplier and 9 block memories for 32 paths);

2. Faster than the other designs (up to 276 MSamp/Sec on a Virtex-II Pro FPGA and 342 MS/Sec on a Virtex-4 FPGA); and,

3. More accurate than our other designs and all of the designs available in the literature.

2.4 Summary and Conclusions

The sum-of-sinusoids fading channel model is an efficient approach for the software and hardware simulation of fading channels. In this chapter, we first provided a brief overview of different sum-of-sinusoid based fading simulators. We evaluated their advantages and shortcomings. We proposed different models for the accurate simulation of Rayleigh and Rician fading channels. We also proposed efficient architectures suitable for compact and high-throughput hardware implementation. During the progress of this work, the proposed fading channel models became more statistically accurate for the continuous simulation of fading channels. Moreover, the implemented fading simulator architectures became more compact and more efficient.

One of our early SOS-based fading simulators in [2] (discussed in Section 2.2.1) required 2444 configurable slices, 48 multipliers, and 12 block memories on a Xilinx Virtex-II Pro XC2VP100-6 FPGA to generate the complex fading gains of a single path Rayleigh channel with $N = 8$ complex sinusoids, generating 204 million samples per second. However, our latest fading simulator in [5] (presented in Section 2.2.4) requires 2151 config-

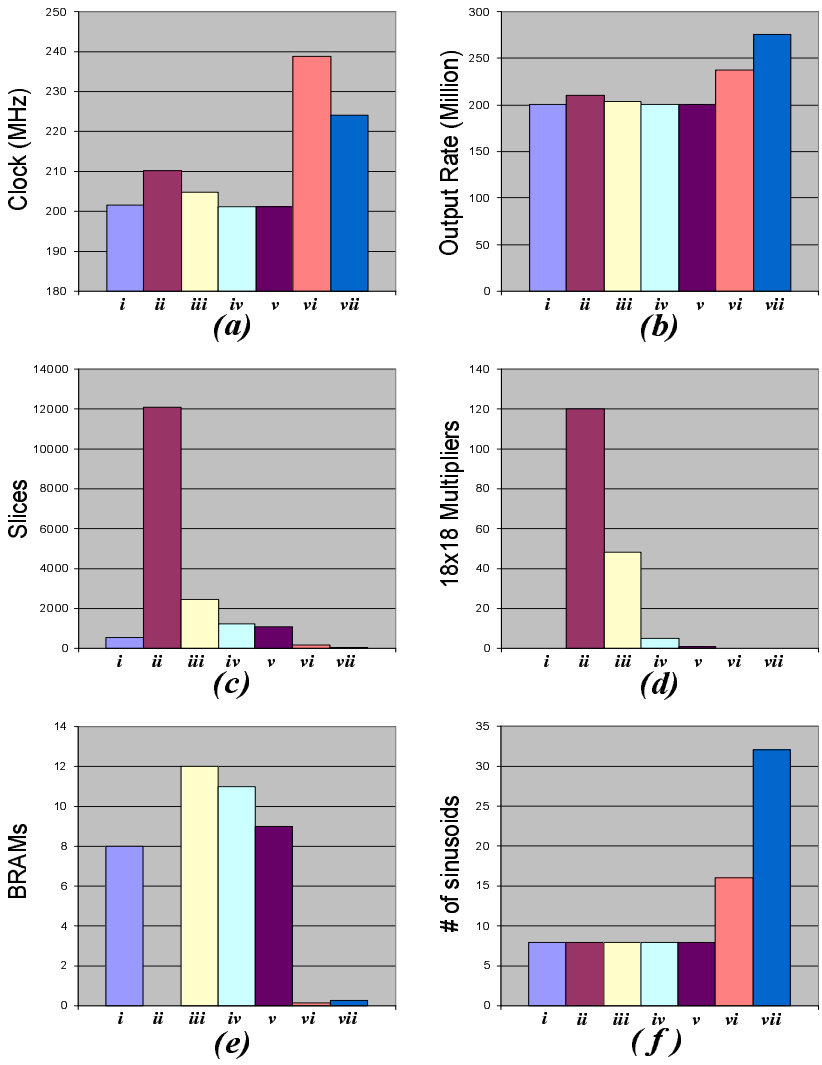


Figure 2.22: Comparison between different implementations of our fading simulators. All of the fading simulators have been implemented in Verilog HDL and synthesized on a Xilinx Virtex-II Pro FPGA XC2VP100-6. The plots show (a) maximum clock frequency, (b) maximum output sample rate, (c) number of configurable slices per generated path, (d) number of utilized on-chip 18×18 multipliers per generated path and, (e) number of utilized on-chip 18-Kb block memories per generated path. The implemented fading simulators are from (*i*) the model proposed in [6] (see Section 2.2), (*ii*) the model proposed in [7] (see Section 2.2), (*iii*) the model proposed in [2] (see Section 2.2.1), (*iv*) the model proposed in [3] (see Section 2.2.2), (*v*) the model proposed in [8] (see Section 2.2.2), (*vi*) the model proposed in [4] (see Section 2.2.3), and (*vii*) the model proposed in [5] (see Section 2.2.4).

urable slices, 1 multiplier, and 9 block memories on the same PFGA device to generate the complex fading gains of a 32 path Rayleigh/Rician fading channel with $N = 32$ complex sinusoids for each path, generating 32×276 million samples per second.

Chapter 3

Filtered-Based Fading Simulation

In addition to the *sum-of-sinusoids* (SOS) method, another well-known approach for generating fading variates for channel simulation is to filter a complex, zero-mean, Gaussian random process with independent samples. The fading process $c(t) = c_i(t) + jc_q(t)$ is obtained by passing the Gaussian variates through a suitable *spectrum shaping filter* (SSF) [18, 88–90]. Under the common assumption of a two-dimensional isotropic scattering environment with an omni-directional receiving antenna at the receiver [62], the power spectral density (PSD) of $c(t)$ has the band-limited U-shaped form, the so-called Jakes' PSD [64]. The filter-based approach can be customized to accurately provide the statistical properties required for simulating different fading scenarios [19].

This chapter presents novel design and implementation schemes for realizing parameterizable fading channel simulators on homogeneous architectures. More specifically, one of the new designs is an accurate filter-based fading channel simulator that is compact enough to be integrated on a single *field-programmable gate array* (FPGA) along with many communication circuits of interest. Several new architectures for efficient hardware implementation of fading simulators are presented.

Also, we address the related and more general problem of designing complex and real *infinite impulse response* (IIR) filters with fixed-point coefficients for compact and stable implementation. These complex filters are required in different applications such as fading simulation with non-isotropic scattering [91] or the implementation of the standard TGn fading channel models for simulating IEEE 802.11n wireless local area networks [92].

Moreover, in this chapter we will present an efficient hardware implementation for the simulation of Nakagami-m and Weibull fading channels. The simulation method is based on transforming Rayleigh fading samples with arbitrary correlation into Nakagami-m or

Weibull distributed samples.

3.1 Background and Related Work

As mentioned earlier, the behavior of multipath fading channels is commonly modeled as a complex Gaussian *wide-sense stationary* (WSS) process $c(t) = c_i(t) + jc_q(t)$ [15]. In a two-dimensional isotropic scattering environment with an omnidirectional receiving antenna at the receiver [62], the *autocorrelation function* (ACF) associated with either $c_i(t)$ or $c_q(t)$ is given by $\mathrm{R}_{c_i,c_i}(\tau) = \mathrm{R}_{c_q,c_q}(\tau) = \mathcal{J}_0(2\pi f_D\tau)/2$. The PSD function of $c(t)$, denoted by $\mathrm{G}_c(f)$, can be written as

$$\mathrm{G}_c(f) = \begin{cases} \frac{1}{\pi f_D \sqrt{1-(f/f_D)^2}} & |f| < f_D \\ 0 & |f| \geq f_D. \end{cases} \tag{3.1}$$

In order to generate the in-phase and quadrature components of fading variates with a particular correlation between variates, we can pass a complex zero-mean and unit-variance white process $n(t) = n_i(t) + jn_q(t)$ through a linear SSF with an appropriate frequency response $H(f)$. A linear filtering operation on the complex Gaussian samples with flat PSD, yields samples that also have a Gaussian distribution, with the power spectrum density $\mathrm{G}_y(f) = \mathrm{G}_n(f)\,|H(f)|^2$, where $\mathrm{G}_n(f)$ is the PSD of the input samples and $\mathrm{G}_y(f)$ is the PSD of the output samples.

The theoretical spectral density of the complex envelope of the signal received by an omni-directional antenna in a Rayleigh fading wireless channel is given by the Jakes' PSD [93]. The required shaping filter must be designed with a frequency response that is equal to the square root of the PSD of the desired fading process (i.e., $|\mathrm{G}_c(f)|^{1/2}$).

3.1.1 Realization of a Spectrum Shaping Filter

The filtering process can be performed in the frequency domain [18, 89] by multiplying the Gaussian samples by $|\mathrm{G}_c(f)|^{1/2}$. Then an *inverse fast Fourier transform* (IFFT) can be applied to the resulting discrete spectrum to obtain time series fading samples [18, 89]. The resulting series is still Gaussian by virtue of the linearity of the IFFT, and it has the desired Jakes' spectrum. The IFFT has a computational complexity of $O(\Gamma \log \Gamma)$ operations, where Γ is the number of time-domain sampled Rayleigh channel coefficients. One major disadvantage of the IFFT method is its block-oriented nature, which requires all channel coefficients to be generated and stored before the data is sent through the channel. This implies significant memory requirements and precludes unbounded continuous transmission,

which is usually preferred in long running characterization applications such as hardware fading simulation.

In the time domain, the SSF can be implemented with either *finite impulse response* (FIR) filters, *infinite impulse response* (IIR) filters or autoregressive (AR) models. The fading channel simulators in [18, 94–102] use FIR filters as the shaping filter while the designs in [4, 9–11, 13, 14, 21, 88, 103–108] use IIR filters. The AR modeling approach has also been proposed for generating fading processes by passing the noise through an all-pole IIR filter [90, 109]. Several important points should be considered when implementing fading channel simulators using FIR and IIR filters on hardware platforms:

• While these approaches might be appropriate for computer simulations, they are not necessarily the best candidates for hardware implementation. For example in the IFFT method, samples are generated with a single *fast Fourier transform* (FFT) operation, and the large storage requirement can result in space-inefficient and costly implementation [109]. To produce samples with accurate statistics, the AR model needs a large filter order, which greatly increases the number of required multiplications per output sample. Also, implementation of the AR fading simulator demands highly accurate floating-point variables, which makes it unappealing for compact fixed-point implementations.

• The degree of the FIR filter is related to the time span of the truncated signal held in the filter and inversely proportional to the Doppler frequency. Specifically, implementation of an extremely narrow-band digital filter with a sharp cutoff and a very large attenuation in the stop-band requires a high-order FIR filter [93, 101]. Meeting the same specifications with an IIR filter typically requires fewer hardware resources than an FIR filter. In fact, utilizing both feedforward and feedback polynomials in an IIR filter permits steeper frequency roll-offs to be implemented for a given filter order than an FIR filter [110].

• An FIR filter has no feedback and is thus inherently stable. However, as the coefficients are quantized in any fixed-point implementation, the resulting numerical error is fed back in the IIR filter, possibly causing instability. Moreover, such effects can cause significant deviations from the expected response. To make sure that the filters are stable under quantization effects, we have designed the filters in fixed-point format using Filter Design Toolbox by MATLAB [111] which offers bit-true implementations of second-order sections with section scaling and reordering to obtain maximum accuracy.

• Although FIR and IIR filter-based approaches might be appropriate for computer simulations of isotropic fading, they are not the best candidates for the hardware simulation

of non-isotropic scattering scenarios, where the *angle of arrival* (AOA) is not uniformly distributed or when the antennas are not omni-directional. Problems arise because the PSD of fading samples in non-isotropic fading is not symmetric in general and requires filters with complex coefficients.

Isotropic scattering refers to the case in which the distribution of the incident directions of the received multipath signals, or AOA, are equally distributed. Assuming two-dimensional isotropic scattering with an omni-directional antenna at the receiver [62], the PSD associated with either the in-phase or quadrature component of a complex fading signal has the well-known Jakes' U-shaped band-limited form [64] with independent in-phase and quadrature samples. However such assumptions have been challenged [112–116] and experimentally demonstrated [117–126] to be inaccurate due to the blockage of some propagation directions and antenna directivity [127], resulting in a nonuniform *probability density function* (PDF) for AOA at the receiver. As discussed in [126], the PDF of the AOA has a great impact on the second-order statistics of the fading process including correlation functions, envelope *level crossing rate* (LCR) and *average fade duration* (AFD). These properties can greatly affect the design and analysis of the wireless systems, not only at the physical layer but also at the link and network layers [48]. In a wireless channel, for example, the average length of error bursts is determined with the AFD. Hence, the AFD of the fading channel plays an important role in design and verification of wireless transceivers.

Several nonuniform PDFs have been proposed in the literature to represent the AOA including the geometrically-based PDFs [128, 129], Gaussian PDF [130], quadratic PDF [131], Laplace PDF [120], cosine PDF [132], and von Mises PDF [91]. The von Mises PDF, which includes the uniform AOA distribution as a special case, is supported with empirical measurements of narrow-band fading channels in [91]. Also it is argued that the von Mises PDF is attractive because it can approximate other non-uniform PDFs and can provide mathematical convenience for analysis [91].

3.1.2 General Channel Model

We assume multipath fading in which the complex envelope of the fading process is

$$\begin{aligned} c(t) &= c_i(t) + jc_q(t) \\ &= \lim_{N\to\infty} \frac{1}{\sqrt{N}} \sum_{n=1}^{N} \alpha_n e^{j(2\pi f_D \cos(\psi_n)+\varphi_n)}, \end{aligned} \tag{3.2}$$

where f_D is the Doppler frequency, ψ_n, $n = 1, ..., N$ are independent and identically distributed (i.i.d.) angles of arrival of the incoming wave at the receiver antenna with distribution $p_\Psi(\psi)$, φ_n, $n = 1, ..., N$ are i.i.d. phases with uniform distribution over $[-\pi, \pi)$, and α_n, $n = 1, ..., N$ are deterministic normalized complex constants that satisfy $\sum_{n=1}^{N} |\alpha_n|^2 = N$. When the scattering encountered in the propagation environment is non-isotropic, the power spectral density (PSD) function associated with $c(t)$ is given by [91]

$$S_c(f) = \frac{e^{\kappa \cos(\tilde{\psi}) \frac{f}{f_D}} \cosh\left(\kappa \sin(\tilde{\psi}) \sqrt{1 - (\frac{f}{f_D})^2}\right)}{\pi I_0(\kappa) \sqrt{1 - (\frac{f}{f_D})^2}}, \tag{3.3}$$

where κ controls the beam-width, $\tilde{\psi}$ denotes the average AOA of the scattered component, and $I_m(\cdot)$ is the m-th order modified Bessel function of the first kind. To obtain (3.3) it is assumed that the AOA of the scattered component is distributed with the von Mises/Tikhonov distribution [133, 134] as follows

$$p_\Psi(\psi) = \frac{\exp[\kappa \cos(\psi - \tilde{\psi})]}{2\pi I_0(\kappa)}, \qquad \psi \in [-\pi, \pi). \tag{3.4}$$

Note that when the beam-width parameter κ is zero, the AOA has a uniform distribution over $[-\pi, \pi)$ and (3.3) reduces to Jakes' "U-shaped" spectrum $S_c(f) = (\frac{\pi}{f_D} \sqrt{f_D^2 - f^2})^{-1}$. The ACF of $c(t)$ can be obtained by taking the inverse Fourier transform of (3.3) as follows

$$R_c(\tau) = \frac{I_0(\sqrt{\kappa^2 - 4\pi^2 f_D^2 \tau^2 + 4j\kappa \cos(\tilde{\psi}) \pi f_D \tau})}{I_0(\kappa)}. \tag{3.5}$$

Another important statistical property of $c(t)$ is the envelope LCR, which is defined as the expected number of envelope crossings per second through a given level ρ with positive slope. It can be shown that the LCR of $c(t)$ is [135]

$$L_{|c|}(\rho) = \frac{\sqrt{I_0^2(\kappa) - I_1^2(\kappa) + \cos(2\tilde{\psi})[I_0(\kappa) I_2(\kappa) - I_1^2(\kappa)]}}{I_0(\kappa)} \times \sqrt{2\pi} f_D \rho \exp(-\rho^2). \tag{3.6}$$

Note that for $\kappa = 0$ (isotropic scattering), (3.6) reduces to the Rayleigh LCR, namely $L_{|c|}(\rho) = \sqrt{2\pi} f_D \rho \exp(-\rho^2)$. Also, for the case where the AOA is modeled by the von Mises PDF, the AFD is

$$T_{|c|}(\rho) = \frac{I_0(\kappa)}{\sqrt{I_0^2(\kappa) - I_1^2(\kappa) + \cos(2\tilde{\psi})[I_0(\kappa) I_2(\kappa) - I_1^2(\kappa)]}} \times \frac{\exp(\rho^2) - 1}{\sqrt{2\pi} f_D \rho}. \tag{3.7}$$

For isotropic scattering ($\kappa = 0$) AFD reduces to $T_{|c|}(\rho) = (\exp(\rho^2) - 1)/(\sqrt{2\pi} f_D \rho)$. To mimic the behavior of a realistic wireless channel, a fading simulator must be able to generate the path gains $\{c(t)\}$ with high accuracy.

Compared to the SOS-based method, the filter-based fading simulation method is much trickier to implement. A filter-based simulator needs to be designed carefully because of possible instability and finite word-length effects when implemented with fixed-point arithmetic. On the other hand, filter-based fading simulation has several advantages over the SOS-based method. First, with the filter-based method, it is possible to simulate a wide range of power spectral densities. Second, the filters can be designed to provide a high level of statistical accuracy. Finally, the generated samples have very accurate Gaussian distribution.

The goal of this work is to make filter-based fading simulation both compact and effective so that it can provide accurate statistics with a hardware complexity close to that of a SOS-based fading simulator.

To achieve this, we first aimed for a compact implementation of a filter-based fading simulator. A new multilayer filter design procedure was proposed. Later, we developed two simple signal processors for performing filter operations. A new filter design technique for designing stable complex filters with fixed-point coefficients was proposed as well. We also proposed an elastic design for the convenient and efficient implementation of candidate fading simulators. Finally, we proposed a compact and accurate filter-based fading simulator in which colored-noise is used instead of white noise. The computational complexity of the new filter-based fading simulator is comparable to a SOS-based fading simulator with 32 sinusoids. In the rest of this chapter, we explain how we implemented in hardware a variety of filter-based fading simulators.

3.2 Implementation of Filter-Based Fading Simulators

We described the first filter-based fading simulators in [9]. To design the SSF, transformation-based filter design was used [88, 136]. Since the Doppler frequency is much smaller than the sample rate, the SSF can be designed to operate at a lower sample frequency $\hat{F}_s \ll F_s$ and later interpolated to reach the target sample rate $F_s = 10$ MHz. In [9] the Doppler frequency is set to $f_D = 2$ KHz and the SSF is designed at sample rate $\hat{F}_s = 20$ KHz. An order 10 IIR filter (5 cascaded biquads) is used to implement the SSF.

An important point is that if the stop-band attenuation of the shaping filter is not sufficiently high, then the out-of-band noise that passes through the filter will degrade the accuracy of the statistics of the generated fading variates. Specifically, since designing a narrow-band filter with a sharp cutoff and large attenuation invariably leads to a high-order

filter, to obtain the closest approximation to the desired frequency response with a relatively small filter order, we only minimized the approximation error in the pass-band of the SSF. The low-pass filters utilized downstream in the interpolator stages can then be designed with extra attenuation over the transition region to ensure a sharp cutoff.

The interpolation is performed in two stages. We designed a two-stage interpolator using two low-pass inverse Chebyshev (Type II) IIR filters [137], one that interpolates by a factor $I_1 = 5$ and one for $I_2 = 100$. The interpolation lowpass filters have a maximum of 0.01 dB attenuation in the pass-band and a minimum of 100 dB attenuation in the stop-band. Also, our Gaussian variate generator described in [138] was used to implement the Gaussian noise source.

We proposed a time-multiplexed resource sharing scheme to implement both the spectrum shaping filter and the first interpolation filter. Sharing hardware in this way achieves the maximum performance with a minimum amount of FPGA resources, leading to an efficient implementation [9]. The operation of the shaping filter and the first *interplation low-pass filter* (ILPF) are bound to only one shared biquad. The second interpolation filter uses a separate set of configurable resources to achieve the target throughput. Our implementation of a 32-bit fading channel simulator on a Xilinx Virtex-II Pro XC2VP100-6 FPGA utilizes 4% of the slices, 19% of the dedicated multipliers, and 2% of the BlockRAMs. The core operates at 50 MHz and generates 12.5 million fading variates per second.

We proposed another implementation of the filter-based fading channel simulator in [10]. Since only signal amplitudes impact the correlation properties, no limitations were imposed on the phase response of the SSF and therefore the inverse Chebychev filters in the previous design were replaced with more efficient elliptic (Cauer) filters [137]. In the hardware implementation, an out-of-order scheduling scheme was utilized to reduce the number of required clock cycles to execute cascaded second-order sections. Also, a new time sharing mechanism was used that resulted in twice the throughput at the same operation frequency. The 32-bit fixed-point FPGA implementation of this fading simulator on a Xilinx Virtex-II Pro XC2VP100-6 utilized 4% of the slices, 20% of the dedicated multipliers, and 10 (2%) of the BlockRAMs and operated at 50 MHz generating 25 million fading variates per second [10].

3.2.1 Real and Complex Filter Processors for Fading Simulation

In [11], we presented a flexible and compact general-purpose filter processor. This processor is a convenient building block for fading simulation as well as other applications. Also, in [44] we proposed a processor for IIR filters with complex coefficients. These two filter processors have similar architectures, so we only describe the real filter processor here.

In an isotropic scattering Rayleigh fading channel, the path gains are modeled using a unit-variance zero-mean complex Gaussian process $c(t) = c_i(t) + jc_q(t)$ [15] with the PSD function

$$S_c(f) = \begin{cases} \frac{1}{\pi\sqrt{f_D^2 - f^2}} & \text{if } |f| < f_D, \\ 0 & \text{otherwise.} \end{cases} \tag{3.8}$$

In this model $c_i(t)$ and $c_q(t)$ are Gaussian-distributed independent stochastic processes. These samples can be generated by passing a stream of independent Gaussian samples through a SSF. For the case of Rayleigh fading, the SSF must have the magnitude $|H(f)| = |S_c(f)|^{1/2}$. In the design of the SSF, since no constraints are imposed on the phase response, we can use IIR filters since they are typically much smaller than their FIR counterparts. However, we must ensure that the designed SSF is stable. Similar to [88, 136] we approximated the desired magnitude response of the SSF with an IIR filter of order $2N_Q$. Here, the response of the SSF is expressed as

$$H(e^{j\omega}) = \prod_{q=1}^{N_Q} g_q \times \frac{1 + b_{1,q}\, e^{-j\omega} + b_{2,q}\, e^{-j2\omega}}{1 + a_{1,q}\, e^{-j\omega} + a_{2,q}\, e^{-j2\omega}}, \tag{3.9}$$

which is equivalent to the magnitude response of N_Q cascaded second-order canonical sections, or biquads (BQDs). In (3.9) note that $b_{1,q}$, $b_{2,q}$, $a_{1,q}$ and $a_{2,q}$ denote the real-valued filter coefficients and g_q denotes the scaling factor of the q-th biquad.

The datapath of a biquad in *direct-form-II* (DF-II) structure [137] is shown in Figure 3.1 (a), where four intermediate variables are stored in four on-chip dual-port memories RAM M1I, "RAM M1Q", "RAM M2I", and "RAM M2Q". Four coefficients are stored in four *read-only memories* (ROMs), "ROM a1", "ROM b1", "ROM a2", and "ROM b2". "AD0" denotes the read address and "AD1" denotes the write address for the memories.

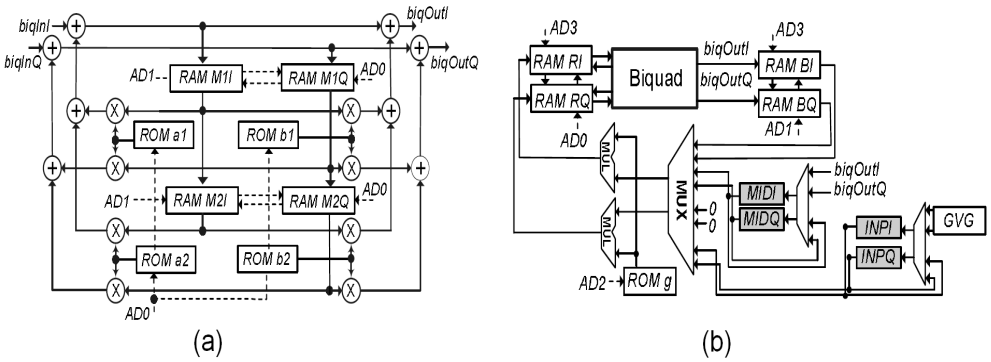


Figure 3.1: (a) Datapath of one biquad and, (b) the filter processor architecture.

Implementation of only one biquad uses 855 configurable slices and utilizes 9% of the dedicated 18×18-bit multipliers available on a Xilinx Virtex-II Pro XC2VP100-6 FPGA. These results confirm that the maximum number of second-order sections that can be implemented on a large contemporary FPGA is limited to 11 due to the relatively large number of high-precision arithmetic units required by each biquad. Instead, we designed a compact filter processor to maximize the throughput with the minimum of FPGA resources. With this processor, the computation of an IIR filter (using N_Q cascaded biquads) can be performed using one compact filter processor.

Figure 3.1 (b) shows the datapath of the filter processor. The memory blocks in this datapath have the same structure as those in Figure 3.1 (a). The core component of the filter processor is a biquad with inputs coming from two memories (in-phase and quadrature), "RAM RI" and "RAM RQ", with read address bus "AD0" and write address bus "AD3". The outputs of biquad are stored in memories "RAM BI" and "RAM BQ", with read address "AD1" and write address bus "AD3". ROM "ROM g" is initialized with the scaling factors of the IIR filters using the "AD2" addressing bus. Two combinational multipliers are used to perform the scaling operation of intermediate values between biquads. The two zero inputs to the 8-input multiplexer are reserved for when filter processor performs zero padding for the interpolators.

The filter processor is sequenced by a microprogrammed controller. A code generator program was developed that receives the specification of shaping filter as inputs and generates a sequence of filter processor microinstructions (microcode). This microprogram is stored in an instruction ROM and is addressed by a *program counter* (PC). To simplify and minimize the hardware, we used a horizontal microcode architecture in which every control bit in the microinstruction drives a control line in the filter processor datapath.

It merits attention that providing a microprogrammed controller makes the proposed filter processor flexible and reconfigurable to perform the arithmetic operations of several IIR filters with different filter lengths, rates, and coefficients. In addition, the design supports the possibility of zero padding, and both up and down sampling. Finally, the programmable filter processor supports the realization of different filter concatenations.

Samples generated at a low sampling frequency need to be over-sampled and passed through lowpass filters in order to obtain the target sample rate. In the fading simulators proposed in the literature, this is normally done using conventional FIR or IIR filters. However, this approach is overly expensive in hardware since the filtering operations are performed at higher sample rates. When the maximum Doppler frequency is much smaller than the sampling frequency, we propose to use a cascade of $l_z = 2$ zero-order hold filters with impulse response $d_P(n) = [P^{-1}, P^{-1}, ..., P^{-1}]_{1\times P}$ where P is the over-sampling rate. Such filters, called *specific interpolation lowpass filters* (SILPF), can be easily implemented without multiplication.

An FPGA implementation of this fading channel simulator on a Xilinx XC2VP100-6 utilizes 2022 configurable slices (4%), 40 dedicated 18×18-bit multipliers (9%), and 10 BlockRAMs. The maximum clock frequency of this fading simulator is 63.4 MHz and it can generate more than 63 million fading samples per second. However, the maximum sample rate can be increased to 300 million sample per second if multiple clock sources are utilized since the maximum clock frequency of the interpolator is 300 MHz.

3.2.2 Non-isotropic Fading Simulation

In [12] we presented the first FPGA-based non-isotropic Rayleigh fading simulator. In the fading model in this paper, it is assumed that the AOA is distributed with the von Mises PDF [91].

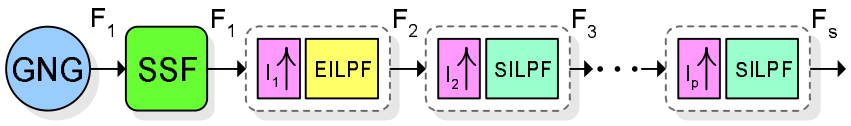


Figure 3.2: Block diagram of the non-isotropic Rayleigh fading channel simulator.

Figure 3.2 shows the block diagram of this non-isotropic fading simulator. To generate the Rayleigh fading process, independent samples of a zero-mean complex Gaussian process, generated by the GNG block in Figure 3.2, and pass through an SSF with a mag-

nitude response equal to the square root of the magnitude of (3.3). In practice, since f_D is much smaller than the sample rate F_s, the designed SSF would have an extremely narrow bandwidth and a very sharp cut-off. However, we can reduce the complexity of the SSF by designing it at a lower sampling frequency, $F_1 \ll F_s$, which also improves the accuracy of the designed filter. Further, the generated samples are interpolated to obtain the target sampling frequency $F_{n+1} = F_1 \times \prod_{j=1}^{T_g} I_j = F_s$, where T_g denotes the number of interpolation stages and I_j is the interpolation factor at the j-th interpolation stage.

In contrast to isotropic fading, the PSD in (3.3) is not symmetric, and hence the filter coefficients are potentially complex [137]. Here the SSF is approximated with

$$H(e^{j\omega}) = \prod_{k=1}^{2N_F} g_k \times \frac{1 - b_{1,k}e^{-j\omega}}{1 - a_{1,k}e^{-j\omega}}, \tag{3.10}$$

where g_k is the positive scaling factor, $b_{1,k}$ and $a_{1,k}$ are the k^{th} complex zero and pole, respectively, and $2N_F$ is the filter order. This filter can be realized as a cascade of $2N_F$ *first-order sections* (FOSs). Figure 3.3 (a) shows the direct-form II realization of an FOS, where the filter coefficients a_1 and $b_1$are complex-valued. For isotropic scattering (i.e., $\kappa = 0$), however, poles and zeros of equation (3.10) appear in complex conjugate pairs and the shaping filter can be implemented using N_F canonic second-order sections. Figure 3.3 (b) depicts the direct-form II realization of a biquad with the real-valued filter coefficients a_1, a_2, b_1, and b_2.

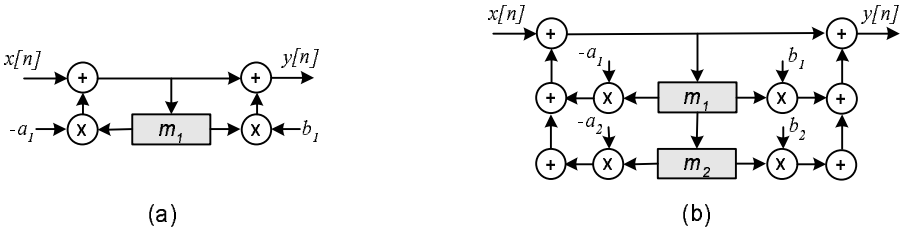


Figure 3.3: (a) Direct-Form II realization of a first-order section. (b) Direct-Form II realization of a second-order section (biquad).

After the shaping filter, the next stage includes zero-padding and lowpass filtering. Since only the amplitude response affects the correlation properties and no restrictions are imposed on the phase response, we used *elliptic IIR lowpass filters* (EILPFs). The lowpass filter has a symmetric frequency response and hence its poles and zeros appear in complex conjugate pairs and therefore this filter can be realized using cascaded biquads. For high sample rate interpolation, we used the same multiplication-free SILPFs mentioned in

Section 3.2.1.

To implement this non-isotropic fading channel simulator on a FPGA, we generated the Gaussian noise samples by adding twelve 16-bit independent uniformly-distributed random variables. These random variables were generated using a 63-bit *linear feedback shift register* (LFSR) with a primitive feedback polynomial to ensure the maximum-length state sequences. Implementation of this noise generator on a Xilinx Virtex-II Pro XC2VP100-6ff1996 FPGA utilizes only 12 (out of 44096) configurable slices and it can generate more than 1.1 million random variables per second.

The operation of the first-order and second-order sections for SSF and EILPF are performed using the datapath in Figure 3.4. Here, the filter coefficients are stored in four on-chip distributed memory blocks "ram aI/a_1", "ram aQ/a_2", "ram bI/b_1", and "ram bQ/b_2". Four intermediate variables are stored in memories "ram mI_1", "ram mQ_1", "ram mI_2", and "ram mQ_2". Note that "adrs0" and "adrs1" are the address lines to the coefficient and intermediate variable memories, respectively. The input Gaussian samples enter this datapath via "MUX 3". The data between successive sections is stored in memory blocks "ram xI" and "ram xQ". This datapath is also capable of performing the zero-padding operation by adding zero inputs via "MUX 3".

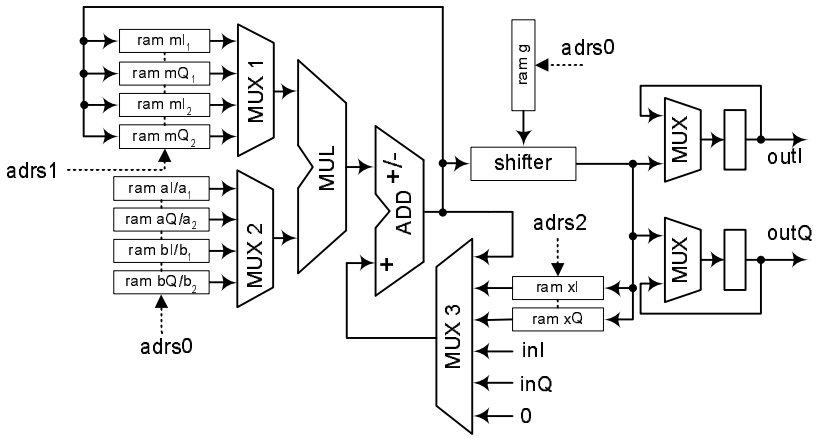


Figure 3.4: Datapath of the SSF and EILPF.

The precision of the fixed-point representation plays an important role in the stability and accuracy of IIR filters. Fixed-point error analysis shows that a 32-bit fixed-point implementation of the fading channel simulator ensures computation accuracy and filter stability. However, to increase the accuracy, a scaling factor g_k was added to each first- or second-order section. The barrel-shifter in Figure 3.4 multiplies/divides the output of each section

by g_k, which is considered to be a power of two. This way, we ensure that the input and intermediate variables of each section remain in the valid range. The number of shifts is stored in "ram g" for each section. In addition, to increase the accuracy, the intermediate variables "ram ml_1", "ram mQ_1", "ram ml_2", and "ram mQ_2" were implemented with 40-bit precision.

An FPGA implementation of the structure in Figure 3.4 uses 1004 of the configurable slices of a Virtex-II Pro XC2VP100-6ff1996, operates at up to 110 MHz, and can perform the operations of 44734 first- or second-order sections per second. Note that the adder and multiplier in this datapath operate sequentially. Also, this design can run IIR filters with complex coefficients of up to order 32 (order 64 for filters with real-valued coefficients). Increasing the filter order is straightforward by adding more storage capacity.

The SILPF can also be efficiently mapped onto hardware by using only addition and shifting operations. Our FPGA implementation of a SILPF with an interpolation factor $I = 4$ and 16-bit precision, utilizes 171 configurable slices on the same device and can operate at up to 300 MHz, generating up to 300 million samples per second.

To verify the performance of this design, we parameterized the design to simulate the Rayleigh fading experienced by the radio signal of a moving receiver traveling at $v = 120$ Km/hr with the carrier frequency $f_c = 1.8$ GHz. The received signal experiences a maximum Doppler frequency of $f_D = 200$ Hz. We also assumed the target sample rate to be $F_s = 10$ MHz.

For both the isotropic and non-isotropic scattering Rayleigh fading cases, we designed the SSF at the sampling frequency $F_1 = 625$ Hz with $N_F = 10$ biquads. The signal is then up-sampled $I_1 = 25$ times and passed through an elliptic lowpass filter with 6 biquads at a sampling frequency $F_2 = 15625$ Hz with pass-band corner frequency $f_{pass} = 203$ Hz, stop-band corner frequency $f_{stop} = 240$ Hz, maximum pass-band ripple $A_{pass} = 0.1$ dB, and minimum stop-band ripple $A_{stop} = 100$ dB.

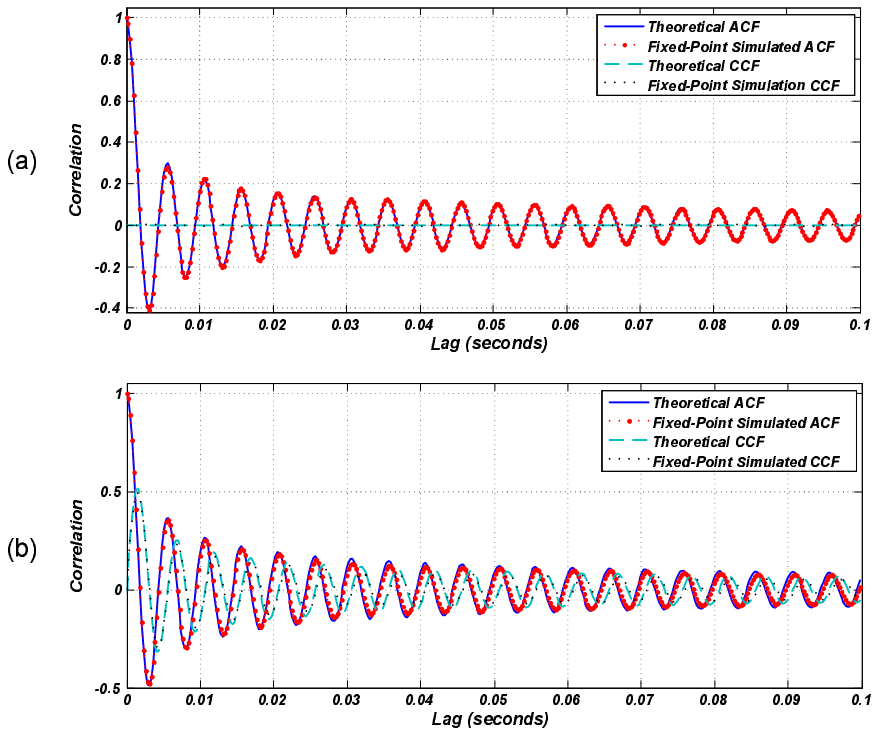


Figure 3.5: Normalized ACF and CCF between real and imaginary components of the generated fading process with $f_D = 200$ Hz and $F_s = 10$ MHz for (a) isotropic ($\kappa = 0$, $\tilde{\psi} = 0$), and (b) non-isotropic ($\kappa = 1$, $\tilde{\psi} = 0$) scattering fading channels.

The signal is then up-sampled with five SILPF blocks with interpolation factors $5, 4, 4, 4$, and 2, respectively, which increase the generated signal rate up to the target sample rate $F_s = 10$ MHz.

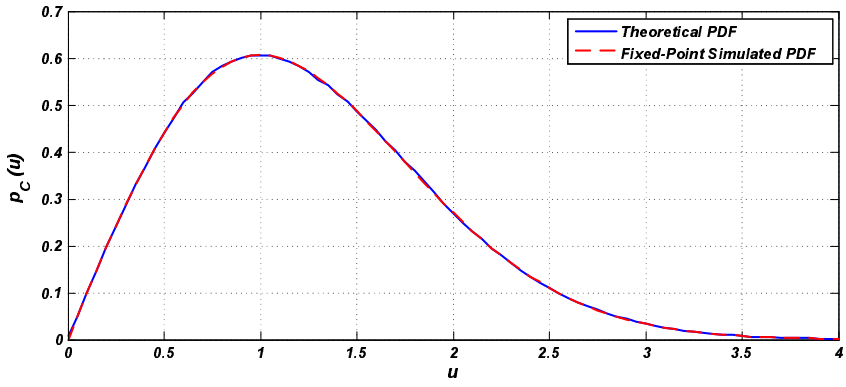


Figure 3.6: PDF of the generated samples for isotropic scattering fading (i.e., $\kappa = 0$, $\tilde{\psi} = 0$).

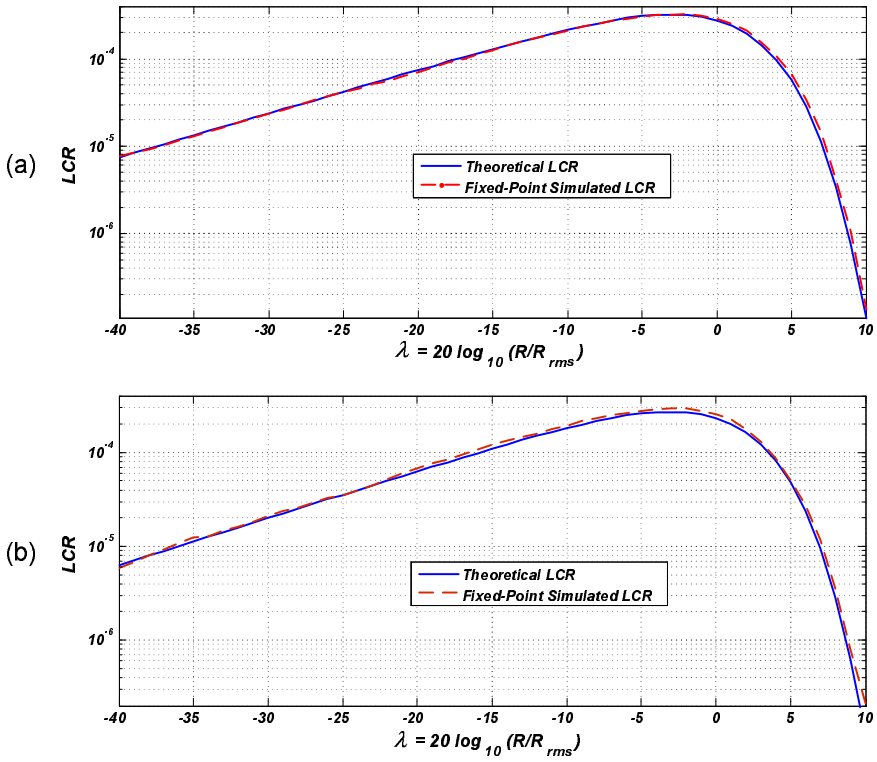


Figure 3.7: Envelope LCR of the generated fading process with $f_D = 200$ Hz and $F_s = 10$ MHz for (a) isotropic ($\kappa = 0$, $\tilde{\psi} = 0$), and (b) non-isotropic ($\kappa = 1$, $\tilde{\psi} = 0$) scattering fading channels.

Figure 3.5 shows the ACF and CCF between the real and imaginary parts of the generated fading process along with the the theoretical references from equation (3.5). These curves were generated by passing 200,000 complex noise samples through the SSF, EILPF and SILPF blocks (or equivalently 3.2 billion generated fading samples). Figure 3.5 (a) plots these statistics for the isotropic scattering case while Figure 3.5 (b) depicts similar statistics for the non-isotropic scattering case. As seen in this figure, there is a close match between the desired statistics and those of the generated results for both the isotropic and non-isotropic cases. Figure 3.6 illustrates the *probability density function* (PDF) of the envelope of the generated samples for isotropic scattering fading. This figure shows that the PDF of the generated samples closely follows the Rayleigh PDF. For the isotropic and non-isotropic cases, the LCR of the generated samples is plotted in Figure 3.7 (a) and Figure 3.7 (b), respectively. To generate these curves, the statistics were measured over 43 million fading samples. These figures also show a close match between theoretical (from equation (3.6)) and simulation results for both cases.

The FPGA implementation of this fading simulator uses one Gaussian noise generator, one SSF, one EILPF (with the same datapath as the SSF), and five SILPF blocks, which together utilize 6.8% of the configurable slices of a Virtex-II Pro XC2VP100-6ff1996 FPGA, operating at a maximum clock frequency of 110 MHz. Increasing the output sample rate up to 300 million samples per second is also possible by adding more SILPF blocks and utilizing multiple clock sources.

3.2.3 Multipath Rayleigh and Rician Fading Simulator

In [13] we proposed a new elastic structure for simulating multipath Rayleigh and Rician fading channels with isotropic scattering. A new architecture was described for efficiently implementing multiple IIR filters. Also, a compact and parameterizable implementation for the interpolator was presented.

Multipath propagation is the situation where the received signal contains different faded copies of the transmitted signal. The effect of the multipath fading on the baseband signal can be modeled with a time-variant linear system with the following impulse response [19]

$$h(t,\tau) = \sum_{n=0}^{N_p-1} \mu_n \left|c_n(t)\right| e^{j\angle c_n(t)} \delta(\tau - \tau_n(t)), \tag{3.11}$$

where N_p is the number of independent paths, μ_n is the average attenuation of the n-th path, and $c_n(t)$ and $\tau_n(t)$ denote the complex gain and delay of the n-th path.

When a *line-of-sight* (LOS) or strong specular component is present, the channel is called Rician. In Rician propagation, the non zero-mean complex path gain can be divided into two components. The first component is the LOS part with normalized average power $|\beta_n|^2 \leq 1$ and the second component is the random scattering part with average power $1 - |\beta_n|^2$. For the n-th path (possibly between an antenna pair), the Rice factor K_n is defined as $K_n \triangleq \frac{|\beta_n|^2}{1-|\beta_n|^2}$. For purely Rayleigh fading channels, $\beta_n = 0$ and hence $K_n = 0$; for Rician channels $|\beta_n| > 0$ and hence $K_n > 0$. To simulate Rician propagation, one can generate the sequence $\{c_n(t)\}$ with the PSD given in equation (3.3) and then attenuate the samples by $\sqrt{1 - |\beta_n|^2}$ to obtain the power of the random scattering component. The total complex path-gain can then be obtained by adding in a scaled LOS component.

Similar to the previous design, this fading simulator consists of a SSF, an EILPF, and several SILPFs. The block diagram of this fading simulator is shown in Figure 3.2. Note that for each of the N_p independent fading paths, the corresponding SSF is approximated by equation (3.9).

3.2.3.1 Filter and Interpolator Design

The interpolation factors in this design are chosen using a different approach compared to the previous designs. The SSF is designed at sample rate F_1, where $4f_D < F_1 \leq 8f_D$. Choosing F_1 in this range satisfies the minimum Nyquist rate while keeping the computational complexity low. In addition, we have the opportunity to exploit power-of-2 interpolation factors to further reduce the hardware complexity and simplify the filter design.

Output samples from the SSF are up-sampled $I_1 = 16$ times and passed through the EILPF. Since the SILPF stages are designed to operate on narrow-band signals, the first interpolation stage is positioned prior to the SILPF stages. Then the samples are passed through T_g successive SILPFs. The i^{th} SILPF interpolates the signal 2^{k_i} times. Based on the processing architecture, the relation between F_1 and the target output sampling rate is $F_s = 16 \times F_1 \times \prod_{i=1}^{T_g} 2^{k_i}$. From here we have $F_1 = 2^{-(4+S_g)} F_s$, where $S_g = \sum_{i=1}^{T_g} k_i$ is an integer value in the range $\log_2(F_s/f_D) - 7 \leq S_g < \log_2(F_s/f_D) - 6$. Based on the maximum interpolation factor $2^{K_{max}}$, where $K_{max} = 4 + \max\{S_g\}$, each SILPF is assigned a specific interpolation factor. The minimum Doppler frequency that can be simulated by this system is

$$f_D^{min} = 2^{-(K_{max}+3)} F_s. \tag{3.12}$$

The maximum Doppler frequency, on the other hand, is dictated by the biquad processor that performs the SSF operations. If the biquad processor requires C_{bq} clock cycles to perform the biquad operations for the in-phase or quadrature part, it can be verified that the maximum achievable Doppler frequency is

$$f_D^{max} = 0.5 \times f_{CLK} \times (C_{bq} \sum_{l=1}^{N_f} N_B(l))^{-1}, \tag{3.13}$$

where f_{CLK} is the biquad processor clock frequency, N_f is the number of filters, and $N_B(l)$ denotes the number of biquads in the l^{th} filter.

3.2.3.2 Elastic Buffers

In our design, consecutive blocks are interconnected with elastic buffers. This reduces the complexity of our hardware design significantly and simplifies interfacing with external (off-the-chip) blocks. Interfacing is simplified because consecutive blocks do not have to be strictly synchronized with respect to instantaneous data throughput. Figure 3.8 shows

the block diagram of this buffer. The elastic buffer is basically a random access memory with two (potentially) asynchronous ports. The elastic buffer works by allowing input data to be written into the memory using the input clock and then read out according to an output clock. The one key constraint for correct operation is that the read time must occur after the write time for any specific word, and not before or concurrently with the write.

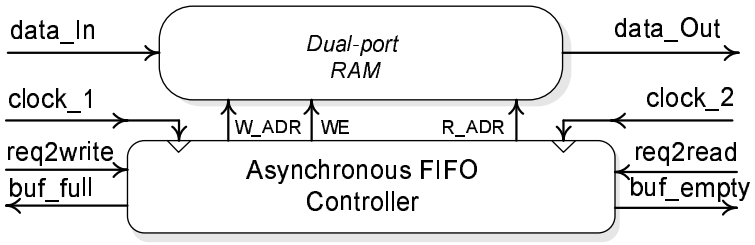


Figure 3.8: Block diagram of the elastic buffer.

To solve this problem, we designed the elastic buffer to operate in two modes. When operating in the handShaking mode, the receiver block requests to read data samples prior to reading from the buffer. When the buffer is empty, the receiver is informed (by asserting buf_empty) to stop requesting until new data arrives. Likewise, the transmitter requests to write into the elastic buffer and it is informed (by asserting buf_full) when the buffer is full. The handShaking mode is used for interfacing blocks that are capable of handshaking. We also use the elastic buffer in the handShaking mode to interconnect consequent blocks in the design of our fading simulator.

The second mode of the designed elastic buffer is called continuous mode. It is used for interfacing external clock domains without handshaking capability. In the continuous mode, the transmitter and the receiver have the same nominal clock frequency. In this mode the elastic buffer absorbs the clock mismatch between the input and local timing. In the continuous mode, if the transmitter is clocked slightly faster than the receiver, it may start to fill the elastic buffer faster than the receiver can drain it. In that case the transmitter re-writes some of memory locations when the buffer is full and therefore some of the data words are dropped. Conversely, if the receiver is running slightly faster than the transmitter, the receiver may clock out more data than have been transmitted. In this mode, as the buffer drains, the elastic buffer inserts some data words by re-reading the last data word. The consequences of repeating read data in this way depend on the application.

When $F_s \gg f_D$, it can be shown that the elastic buffer in the continuous mode does not have a significant effect on the statistics of the generated fading samples. In practical

systems, the sample rate is much higher than the maximum Doppler frequency and therefore immediate fading samples have almost equal values. To show this, we define the difference signal $y_k(t)$ to be the difference between $x(t)$ and $x(t-kT_s)$ for any time $t = nT_s = n/F_s$. Here $y_k(n)$ can be obtained by passing $x(t)$ through a filter with the impulse response $b_k(t) = \delta(t) - \delta(t - kT_s)$. From here, $y_k(t)$ is a zero-mean Gaussian random process with variance

$$\begin{aligned} R_{yy}(0) &= E\{y(t)y(t)^*\}, \\ &= \int_{-\infty}^{+\infty} S_c(f)|1 - e^{-j2\pi f k T_s}|^2 df, \\ &= \mathcal{J}_o(0) - \mathcal{J}_o(2\pi k f_D T_s), \\ &\approx (k\pi f_D T_s)^2, \end{aligned} \tag{3.14}$$

where in (3.14) we used the expansion of the zeroth-order Bessel function

$$\mathcal{J}_o(x) = \sum_{m=0}^{\infty} \frac{(-1)^m x^{2m}}{2^{2m}(m!)^2},$$

to find the approximate value. Equation (3.14) shows that when $F_s \gg f_D$, the power of the difference signal goes to zero and therefore the difference between $x(t)$ and $x(t \pm T_s)$ becomes negligible. Under this condition, if the signal $x(t)$ is sampled out at $\acute{F}_s$ samples per second, we can approximate the sampled signal $\acute{x}(t) \approx x(\eta t)$ where $\eta = \acute{F}_s/F_s$. From here, the ACF of $\acute{x}(t)$ is $R_{\acute{x}(\tau)} = \mathcal{J}_o(2\pi(\eta f_D)\tau)$. Therefore, in the presence of our elastic buffer the clock mismatch between the transmitter and receiver has a similar effect as a slight (probably negligible in most cases) change in the maximum Doppler frequency.

3.2.3.3 Implementation

Figure 3.9 shows a block diagram of the implemented four-path fading emulator. It can generate $N_p = 4$ independent fading processes with $N_f = 4$ filters with different correlation properties. The multiple processes can be used to model, for example, frequency-selective channels or fading channels in multiple-input multiple-output (MIMO) systems. The generated Gaussian samples are passed to the first shared filter processor, which runs the designed SSF from (3.9) in four parallel and independent threads. Each thread of data is then up-sampled 16 times and passed through an EILPF that is implemented using another shared filter processor. Each IIR filter processor is capable of processing eight independent streams of input data. The maximum order of each IIR filter is 16 (eight biquads per filter).

After the EILPF, the data streams are passed to *interpolation filters* (IFs). Each IF includes four configurable SILPFs that are interconnected with elastic buffers. Each IF also contains a terminal elastic buffer for interfacing to external hardware.

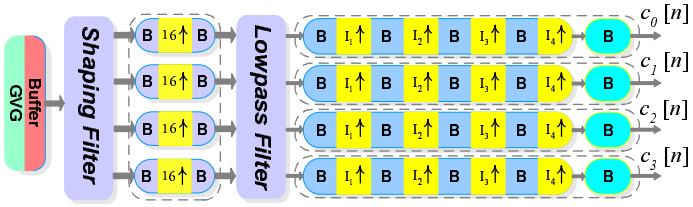


Figure 3.9: Block diagram of the implemented four-path fading simulator

We found it to be useful to re-use an optimized fixed-point processor to perform the operations of the SSF and the EILPF. Figure 3.10 shows the architecture of our biquad processor. The main datapath element is a *multiplier-accumulator* (MAC) that multiplies the in-phase and quadrature data components by real-valued coefficients. The output of each biquad is written into RAM d, which holds the intermediate results. RAM m_1 and RAM m_2, store the complex contents of the biquad memories. The biquad coefficients are stored in RAM a_1, RAM a_2, RAM b_1, RAM b_2, and RAM g. The biquad processor can also add a bias value from RAM β, which is necessary when simulating Rician fading.

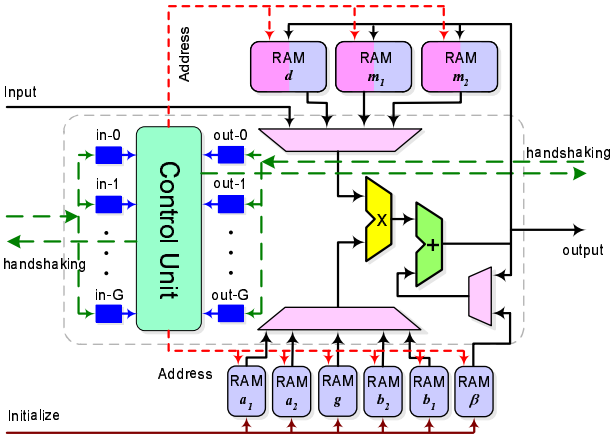


Figure 3.10: Datapath of the biquad processor.

The control sequence for running the cascaded biquads is quite straightforward. However, when emulating multiple paths, where each path could have different filter specifications and different sample rates, flexible implementation of the control unit becomes challenging. In addition, the fading emulator might have to generate samples for more than

one external system with different clocks. To improve the robustness of our biquad processor, two flags are assigned to every thread of cascaded biquads (i.e., for each individual path labeled in- i and out- j in Figure 3.10). These flags govern the data flow through each thread. For example, for the i^{th} thread (path), if in- i is high, then input data is ready to be read. Also, if out- i is high, then data can be written to the next stage. For each thread, the biquad processor keeps executing the biquads unless either of the input or output flags is de-asserted. To prevent the overwriting of unprocessed data, the biquads in each thread are scheduled to be executed from the last biquad to the first.

We synthesized the Verilog HDL of a four-path Rayleigh and Rician fading simulator on a Xilinx Virtex-II Pro XC2VP100-6 FPGA. In this implementation we used 36-bit fixed-point variables in the SSF and EILPF filters. Our implementation utilized 6139 (13.9%) configurable slices, 12 (2.7%) dedicated 18×18 multipliers with maximum clock frequency 73 MHz, generating up to 4×73 million samples per second. In the design of elastic buffers, we used both the rising and falling clock edges for reading the input signals and writing the ouput samples. This complicated the buffer design and lowered the hardware speed. However, this method reduces the complexity of the rest of the design.

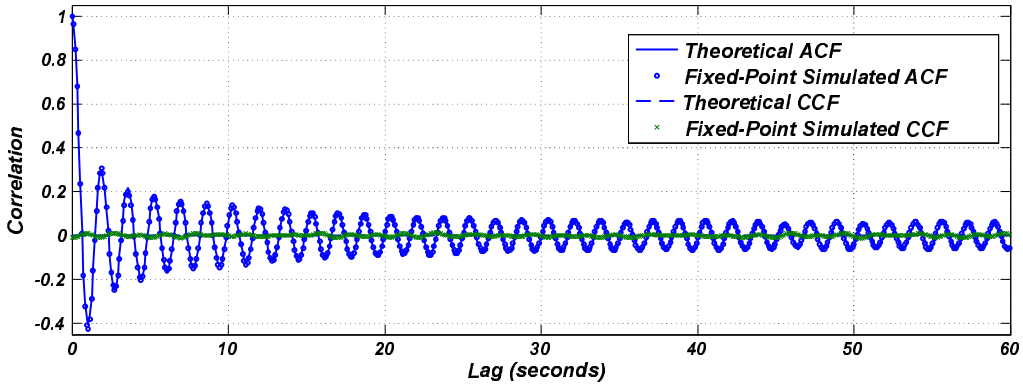


Figure 3.11: Normalized cross-correlation and autocorrelation between real and imaginary components of the generated fading process with $f_D = 0.625$ Hz, $F_s = 2.5$ MHz for 60 seconds.

Figure 3.11 shows the autocorrelation and cross-correlation between real and imaginary components of the generated fading process with $f_D = 0.625$ Hz, $F_s = 2.5$ MHz for 60 seconds. These results were generated using the fixed-point bit-true model of this fading simulator. However, this design was also implemented on a Digilent Spartan-3 board that hosts a Xillinx XC3S1000 FPGA [139]. Figure 3.11 confirms a close match between the desired response and the generated results over up to 60 seconds. In another example we

measured the PDF for the amplitude of the generated samples with $f_D = 200$ Hz and $F_s = 10$ MHz (see Figure 3.12 (a)). Note that the measured PDF accurately matches the Rayleigh PDF. The LCR of the generated fading samples in Figure 3.12 (a) also shows a good match between the reference curve and the generated results.

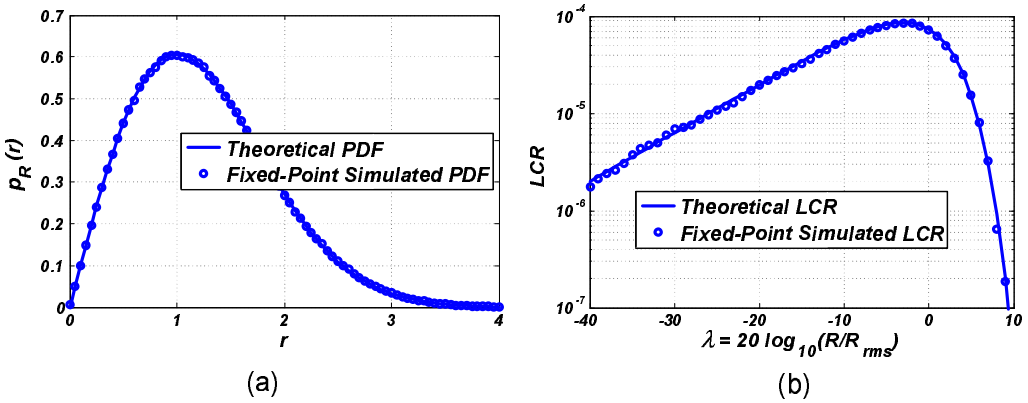


Figure 3.12: (a) Envelope probability density function. The measured and reference PDFs are indistinguishable in this figure. (b) Envelope level crossing rate of the generated fading process with $f_D = 200$ Hz and $F_s = 10$ MHz. In this figure, the solid line represents the theoretical reference and the circles are the measured values.

3.3 Fixed-Point Complex Stable IIR Filter Design

One of the important steps for generating the fading samples is designing the SSF. The bit-precision selected for implementing the SSF plays an important role in the accuracy and stability of the filter implementation. While double-precision floating-point arithmetic is readily available in most software platforms, floating-point arithmetic on an FPGA is significantly slower and less efficient than fixed-point arithmetic.

Conventionally IIR filters are designed with double-precision floating-point variables using available tools like the `iirlpnorm` function in MATLAB which uses the least p-norm method [140]. The filter coefficients are then quantized for fixed-point implementation. To obtain a compact hardware implementation, variables should be implemented with the minimum possible fixed-point word-length. However, reducing the word-length impacts the response, and potentially the stability, of the designed IIR filter.

Here, we address the problem of designing complex and real IIR filters with fixed-point coefficients for compact and stable implementation. Most of the previously proposed filter design techniques do not support complex filters. However, complex filters are required in many applications like fading simulation with non-isotropic scattering (e.g., [91]) or

the implementation of the TGn fading channel models for simulating IEEE 802.11 radio propagation [92].

In the following sections we propose a step-by-step technique for designing real and complex stable IIR filters with complex coefficients. We will also propose some techniques for reducing the number of bits required for fixed-point implementation of these IIR filters.

3.3.1 Filter Design

Several methods exist for designing complex IIR filters. Nonlinear optimization [141–143], linear programming [144], and semidefinite programming [145] have been proposed to design IIR digital filters in the complex Chebyshev sense. Least-squares methods have been applied extensively to design FIR and IIR filters [146–151]. However, ensuring the stability of an IIR filter whose coefficients are obtained by least-squares methods is not straightforward. Many authors have neglected this topic and concentrated on finding close approximations to the desired frequency response. This approach can be used for designing real filters in which the complex poles and zeros appear in complex-conjugate pairs. In this case, all unstable factors in the filter transfer function $1/(1 - s_k e^{j\phi_k} e^{-j\omega})$ can be replaced with their stable counterparts $1/(e^{-j\omega} - s_k e^{j-\phi_k})$ with identical magnitude response. If, however, the complex filter poles and zeros do not appear in complex-conjugate pairs, such as when the filter coefficients are complex, this method cannot be applied.

Different approaches exist for filter stabilization when optimization techniques are used for designing the filter. One method, proposed in [141], is to start the search from a stable point and then control the step size so that the solution never leaves the stable region. This method is computationally intensive and is not easy to implement with traditional optimization procedures. A second method is to make sure that the desired response is minimum-phase [152] or that it has a "large-enough" group delay [153]. These conditions cannot be met in many situations since the flexibility to modify the target frequency response is restricted. In a third method, explicit constraints are imposed on the coefficients of the denominator of the transfer function [154, 155]. This technique, however, has some limitations and affects the filter quality [156]. Finally, in a fourth method, the least-squares cost function is modified so that the minimum always falls in the stable region [149, 156]. In [149], the filter design problem is reformulated using nonlinear transformations so that the final solution always falls in the stable region. In [156] the authors suggest adding a barrier function to the original cost function. Their proposed barrier function is designed

as follows. First they form an all-pole proxy transfer function consisting of all of the filter poles. The barrier function is basically the sum of the squared amplitude of a section of the impulse response of the proxy transfer function. If the filter is unstable, the tail will have (large) non-zero values. This effect is used to avoid filter instability.

In contrast, we propose to augment the least-squares cost function with a specific barrier function to control the location of the poles (and potentially also the zeros) to enforce filter stability. In the proposed approach, the IIR filter is represented as a product of first-order factors

$$H(e^{j\omega}) = A \prod_{k=1}^{\Gamma} \frac{1 - r_k e^{j(\theta_k - \omega)}}{1 - s_k e^{j(\phi_k - \omega)}}, \tag{3.15}$$

where A is a positive scaling factor, $r_k e^{j\theta_k}$ and $s_k e^{j\phi_k}$ are the k-th complex zero and pole, respectively, and Γ is the number of *first-order sections* (FOSs), i.e., the filter order.

Here, we focus on designing IIR filters with a prescribed amplitude response. When the amplitude of the frequency response is symmetric, the poles and zeros of (3.15) appear as complex conjugate pairs and the IIR filter can be implemented using $\Gamma/2$ canonical second-order sections, that is, as *biquads* (BQDs). Note that the poles and zeros are expressed in polar coordinations. This makes it easy to construct a barrier function that increases the cost when any of the poles (or zeros) beyond a certain radius.

Another challenge is to accurately implement the IIR filters in fixed-point arithmetic. When the filter coefficients are quantized, the poles and zeros of the system function typically shift to new positions in the z-plane. This perturbs the frequency response from its intended response. If the designed IIR filter is extremely sensitive to coefficient changes, the resulting filter might not meet the target specifications or the filter might even become unstable.

We assume that the desired amplitude response is represented with $2 \times M$ samples, i.e.,

$$y_i^d = \begin{cases} |P(e^{j2\pi u_i})| & \text{if } 2\pi u_i \text{ is in pass-band} \\ \varepsilon, & \text{otherwise,} \end{cases} \tag{3.16}$$

where $P(e^{j2\pi u_i})$ is the desired response, $u_i \in [-0.5, 0.5]$ is the normalized sampling frequency, and $\varepsilon > 0$ is the attenuation in the stop-band. We introduce a weight vector $\mathbf{v} = [v_1, v_2, ...v_{2M}]^T$ to allow us to emphasize the error minimization for certain frequency bands. We define the column vector $\mathbf{x}$ of length 4Γ containing r_k, s_k, θ_k and ϕ_k for $k = 1, ..., \Gamma$. Similar to the work in [136], we express $H(e^{j\omega}) = A \times F(\mathbf{x}; e^{j\omega})$ where $F(\mathbf{x}; e^{j\omega})$ represents the product of FOSs in (3.15). Next, to find the filter parameters we

define the cost function

$$q(A,\mathbf{x}) \;=\; \sum_{i=0}^{2M-1} v_i \left(\log(A|F(\mathbf{x};e^{j\omega})|) - \log(y_i^d)\right)^2 + B(\vartheta;\varrho;\mathbf{x}), \qquad (3.17)$$

Note that the sum of squared errors on a logarithmic scale is augmented by a parametric barrier function $B(\vartheta;\varrho;\mathbf{x})$. Function $B(\vartheta;\varrho;\mathbf{x})$ is included to keep the poles (and zeros, if necessary) within the unit circle and is defined as

$$B(\vartheta;\varrho;\mathbf{x}) = \sum_{k=k_0}^{2\Gamma} b(\vartheta;\varrho;x_k), \qquad (3.18)$$

where

$$b(\vartheta;\varrho;\tau) = \begin{cases} 0 & \text{if } |\tau| \le \varrho, \\ \vartheta\left(\frac{|\tau|-\varrho}{1-\varrho}\right)^2 & \text{if } \varrho < |\tau| \le 1, \\ \frac{2\vartheta}{1-\varrho}|\tau| - \vartheta\left(\frac{1+\varrho}{1-\varrho}\right) & \text{if } |\tau| > 1. \end{cases} \qquad (3.19)$$

In (3.18), the parameter $\varrho \le 1$ determines an outer boundary for the poles and zeros. When $k_0 = \Gamma + 1$, the barrier function tries to keep the poles within a circle of radius ϱ. Setting $k_0 = 1$, on the other hand, forces both poles and zeros into the same boundary. In addition, ϑ determines how fast the barrier function grows outside of the unit circle. The barrier function (3.18) is useful when designing filters for fixed-point implementation since it can be parameterized to keep the poles and zeros at any desired safe distance from the unit circle. Moreover, using this technique, the quantization noise can be reduced to acceptable levels. It can be shown that the variance of the quantization noise that originates in the k^{th} first-order factor when implemented in *direct-form-I* (DF-I) is

$$\sigma_f^2(k) = \frac{7 \times 2^{-2(\Omega_k - 1)}}{6 \times (1 - s_k^2)}, \qquad (3.20)$$

where Ω_k is the number of bits used to quantize the coefficients (and the intermediate variables) at the k^{th} stage. To derive (3.20) it is assumed that the quantization noise after each multiplier in our model is uniformly-distributed, wide-sense stationary white noise that is uncorrelated with the input signal and the quantization noise in other stages. We also assumed that the samples are truncated and represented in 2's-complement.

The coefficients of the IIR filter are calculated iteratively. At each iteration, the optimum scaling factor A^o is calculated as

$$A^o = \prod_{i=0}^{2M-1} \left(\frac{y_i^d}{|F(\mathbf{x};e^{j2\pi u_i})|}\right)^{\frac{v_i}{\sum_{i=0}^{2M-1} v_i}}. \qquad (3.21)$$

This expression for A^o is found by differentiating (3.17) with respect to A and setting the resulting expression to zero. Next the gradient vector

$$\mathbf{g}(A^o, \mathbf{x}) = [\frac{\partial q(A^o, \mathbf{x})}{\partial x_1}, ..., \frac{\partial q(A^o, \mathbf{x})}{\partial x_{4\Gamma}}]^T$$

is calculated, where the partial derivative of (3.17) with respect to x_k can be expressed as

$$\frac{\partial q(A^o, \mathbf{x})}{\partial x_k} = 2 \sum_{i=0}^{2M-1} \left[\frac{v_i \log(\frac{A^o |F(\mathbf{x}; e^{j\omega_i})|}{y_i^d})}{|F(\mathbf{x}; e^{j\omega_i})|} \times \frac{\partial |F(\mathbf{x}; e^{j\omega_i})|)}{\partial x_k} \right] + \frac{\partial B(\vartheta; \varrho; \mathbf{x})}{\partial x_k},$$

where $\omega_i = 2\pi u_i$ and each partial derivative is given by

$$\frac{\partial B(\vartheta; \varrho; \mathbf{x})}{\partial x_k} = \begin{cases} 0 & \text{if } |x_k| \leq \varrho \text{ or } k < k_0 \text{ or } k > 2\Gamma, \\ \frac{2\vartheta(|x_k| - \varrho)}{(1-\varrho)^2} \text{sign}(x_k) & \text{if } \varrho < |x_k| \leq 1, \\ \frac{2\vartheta}{1-\varrho} \text{sign}(x_k) & \text{if } |x_k| \geq 1, \end{cases}$$

$$\frac{\partial |F(\mathbf{x}; e^{j\omega})|}{\partial r_k} = |F(\mathbf{x}; e^{j\omega})| \frac{r_k - \cos(\theta_k - \omega)}{|1 - r_k e^{j(\theta_k - \omega)}|^2},$$

$$\frac{\partial |F(\mathbf{x}; e^{j\omega})|}{\partial \theta_k} = |F(\mathbf{x}; e^{j\omega})| \frac{r_k \sin(\theta_k - \omega)}{|1 - r_k e^{j(\theta_k - \omega)}|^2},$$

$$\frac{\partial |F(\mathbf{x}; e^{j\omega})|}{\partial s_k} = |F(\mathbf{x}; e^{j\omega})| \frac{\cos(\phi_k - \omega) - s_k}{|1 - s_k e^{j(\phi_k - \omega)}|^2},$$

and

$$\frac{\partial |F(\mathbf{x}; e^{j\omega})|}{\partial s_k} = |F(\mathbf{x}; e^{j\omega})| \frac{-s_k \sin(\phi_k - \omega)}{|1 - s_k e^{j(\phi_k - \omega)}|^2}.$$

The filter coefficients are then found using the ellipsoid algorithm [157, 158]. Note that several optimization algorithms can be used here. We assumed the ellipsoid algorithm here for its simplicity. However, other techniques could be used to speed up convergence. Algorithm 2 summarizes the steps for iterative filter design.

A weight vector $\mathbf{v}$, the desired response $\mathbf{y}^d = [y_i^d]$, and the fixed-point format for different poles and zeros, $\mathbf{\Omega}$, are passed to Algorithm 2. In this algorithm the function $\mathcal{Q}[\mathbf{\Omega}, \mathbf{x}]$ represents the quantization effects that affect each element of $\mathbf{x}$ in the Cartesian coordinate system (coefficients are transferred to Cartesian coordinates, quantized and then transferred back to the polar coordinates). The algorithm starts from a random point $\mathbf{x}_0$ contained within the unit sphere and the initial ellipsoid matrix $\mathbf{E}_0$. The algorithm then searches for the optimal solution within the present ellipsoid of feasible points. This algorithm then converges on the optimal solution by successively reducing the size of the ellipsoid until it is small enough (i.e., the algorithm has converged) or when $|\mathbf{x}_{k+1} - \mathbf{x}_k| < \epsilon$. Note that stable

Algorithm 2 Iterative calculation of the filter coefficients

Require: $\varrho;\ \vartheta; \mathbf{\Omega}; \mathbf{v} = [v_i]$, and $\mathbf{y}^d = [y_i^d]$ for $i = 0, ..., 2M-1$
  Initialize $k = 0$, $\mathbf{x}_0$, $\mathbf{E}_0 = 20\mathbf{I}_{4\Gamma\times 4\Gamma}$
  while $|\mathbf{x}_{k+1} - \mathbf{x}_k| \geq \epsilon$ **do**
    find A_k^o from (3.21)
    find $\mathbf{g}_k = \mathbf{g}(A_k^o; \mathbf{x}_k)$
    $\chi_k = \sqrt{\mathbf{g}_k^T \mathbf{E}_k \mathbf{g}_k}$
    $\tilde{\mathbf{g}}_k = \mathbf{g}_k / \chi_k$
    $\mathbf{x}_{k+1} = \mathcal{Q}\left[\mathbf{\Omega}, \mathbf{x}_k - \frac{1}{4\Gamma+1}\mathbf{E}_k\tilde{\mathbf{g}}_k\right]$
    $\mathbf{E}_{k+1} = \frac{(4\Gamma)^2}{(4\Gamma)^2-1}\left(\mathbf{E}_k - \frac{2}{4\Gamma+1}\mathbf{E}_k\tilde{\mathbf{g}}_k\tilde{\mathbf{g}}_k^T\mathbf{E}_k\right)$
    $k = k + 1$
  end while

real IIR filters can be designed with the above algorithm as well. To design such filters, the sample update is only performed for half of the poles and zeros, and the other half are simply the complex conjugates of the updated samples.

This filter design algorithm can be parameterized to provide a close approximation of the desired response. The weight vector $\mathbf{v}$ can be used to emphasize error minimization in different frequency bands. The filter design procedure can start with a reasonable order for the initial approximation. The filter order can be increased gradually if the desired filter characteristics are not met.

3.3.2 Range Reduction

IIR filters are naturally susceptible to arithmetic overflow and instability due to the inherent feedback. Design and implementation of digital IIR filters must be carried out carefully to avoid such pitfalls. Scaling is commonly used to keep the filter variables in range [159]. However, a poor choice of scaling factor results in loss of signal precision and increase in quantization noise. Another technique is to use more bits to represent intermediate signals. This method, however, cannot be used effectively on *digital signal processors* (DSPs) with a fixed word-length. Moreover, adding extra bits increases resource utilization in FPGA and *application-specific integrated circuit* (ASIC) implementations. We now propose two techniques for minimizing the range of intermediate signals that can be effectively used for reducing the signal range, overflow probability, and resource utilization in ASIC and FPGA implementations.

3.3.2.1 Pole-zero ordering

Considering a small section of an IIR filter, overflows are mainly caused by oscillations around resonant frequencies. Assuming a limited input signal range, we can reduce the signal range by reducing the oscillation magnitude. The oscillation frequency is mainly determined by filter poles. Consider a single FOS of an IIR filter with only one pole at frequency $e^{j2\pi f_{pole}}$. The output range of this section can be reduced significantly if the input signal to this section is attenuated around frequency $e^{j2\pi f_{pole}}$.

In order to use this technique, we need to implement the IIR filter with the DF-I structure so that the input signal experiences a zero before it is affected by the pole. We thus match the filter poles with closest zeros. Moreover, we sort the filter sections according to their pole magnitude in ascending order. Since larger magnitudes are more likely to happen in the last filter stages, the signal precision will be carried through ordered filter stages.

Further, signals are scaled in different stages to assure filter stability. The proposed pole-zero ordering results in effective range control with minimum precision loss.

3.3.2.2 Augmenting auxiliary poles and zeros

As noted earlier, the resonant frequencies of a filter play an important role in filter overflow. The second proposed method for reducing the signal range is to attenuate the input signal around resonant frequencies of an IIR filter $e^{j2\pi f_{Nat}}$ that can potentially result in oscillation or overflow. The imposed distortion can be later compensated with additional poles and/or zeros. However, this technique is applicable only if the input signal does not have a major frequency component around $e^{j2\pi f_{Nat}}$.

One example is implementation of narrow-band lowpass filters with an approximate resonance frequency around DC. If the input signal does not have a DC component, it can be first passed through a highpass filter $D(e^{j\omega}) = 1 - e^{-j\omega}$ (difference) prior to being passed through the filter. The filter output can be later compensated by passing the output signal through the integrator $I(e^{j\omega}) = (1 - \rho e^{-j\omega})^{-1}$. The coefficient $\rho \in [1 - \epsilon, 1)$ is intentionally added here since quantization noise and computational errors can render a perfect integrator (i.e., $\rho = 1$) unstable.

When employed along with pole-zero ordering, the augmented poles and zeros are not included in the ordering process, and instead they keep their position in the DF-I structure, i.e., the augmented zero appears first and the pole appears last. This technique, when used in conjunction with pole-zero ordering and scaling can provide efficient, accurate and compact

implementation of real and complex IIR filters. Note that increasing the word-length is impractical in many DSPs, and addition of poles and zeros is mostly preferred to widening the datapath in ASIC and FPGA implementations. The next section provides illustrative filter design and implementation examples.

3.3.3 Design Examples

To demonstrate the performance and accuracy of our filter design procedure, we designed several fixed-point IIR filters and compared their results with the desired references. We also implemented a filter processor for the designed filters to show the efficiency of this filtering approach. In the following we present some design and implementation examples.

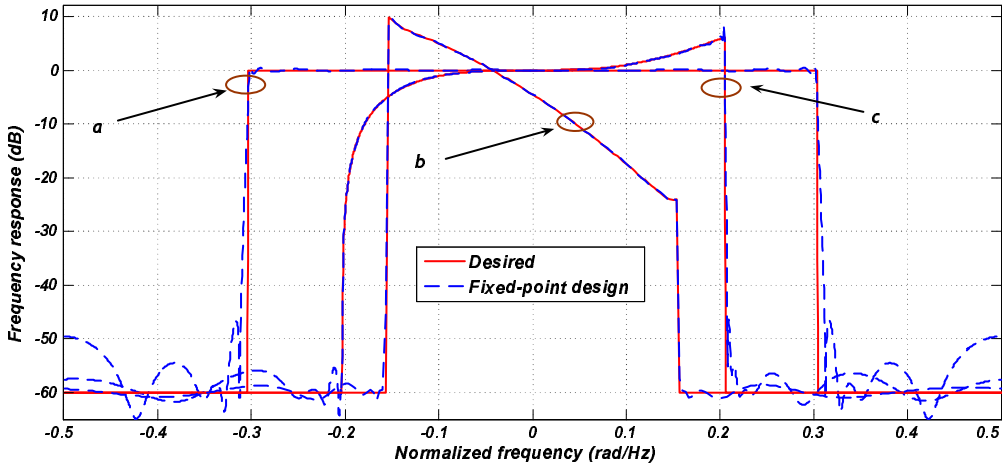


Figure 3.13: Frequency response of designed fixed-point filters and the desired responses in Example 1.

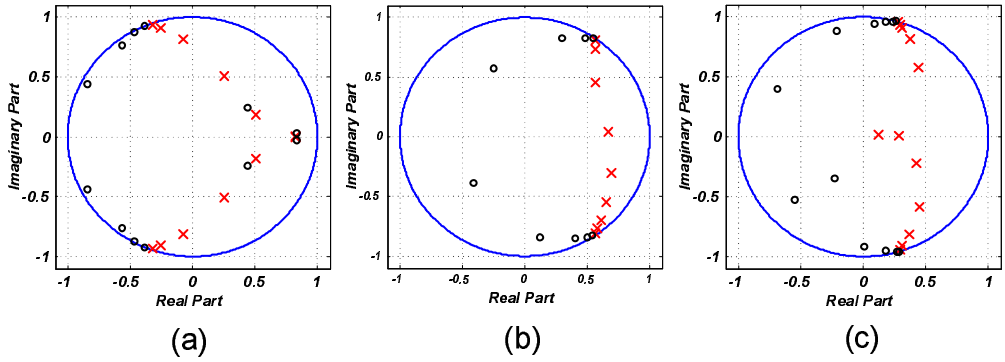


Figure 3.14: Positions of the poles and zeros in the unit circle for the designed filters in Example 1.

Example 1

Table 3.1: Maximum Absolute Signal Range

	1st Section	2nd Section	3rd Section
DF-II	182.22	422.82	421.44
DF-I + ord.	35.43	20.93	9.79
DF-I + ord. + aug.	20.87	14.81	11.04

For the first example, we designed three IIR filters with $\Gamma = 12$ first-order sections. We set the parameters $\vartheta = 5$ and $\varrho = 0.99$, i.e., the poles and zeros are bounded within a circle of radius $r = 0.99$. For all of the first-order sections, the number of bits for representing each coefficient is set to $\Omega = 12$. Figure 3.13 shows the frequency responses of the designed filters (with out-of-band attenuation $\varepsilon = 0.001$) as well as the desired responses. In this figure, filter (a) is a lowpass filter with normalized pass frequency $f_p = 0.3$, filter (b) is a complex spectrum shaping filter for simulating non-isotropic fading with normalized Doppler frequency $f_D = -0.15625$, directivity $\kappa = 5$ (see equation (3.3)), and angle of arrival $\tilde{\psi} = \pi/5$ [91], and filter (c) is a custom complex filter with frequency response $H(f) = \alpha + \beta f^3$, for $|f| \leq 0.2$ and $H(f) = \varepsilon$ for $|f| > 0.2$. As Figure 3.13 shows, the frequency response of the designed fixed-point filters closely match the desired responses. Figure 3.14 shows the position of the poles and zeros for the designed filters. It was verified that all of the poles and zeros were bounded within a circle of radius $r = 0.99$.

To illustrate the effectiveness of our range reduction methods, we simulated an order $\Gamma = 6$ elliptic lowpass filter with sample rate $F_s = 4800$ Hz, $F_{pass} = 1200$ Hz, $F_{stop} = 1500$ Hz, $A_{pass} = 1$ dB, and $A_{stop} = 50$ dB. We measured the maximum absolute range of variables by passing 10^8 uncorrelated zero-mean Gaussian samples through the designed filter. Table 3.1 shows the maximum absolute signal ranges for different filter implementations. The output of each section (biquad) is scaled to lie within $[-1, +1]$. As this table shows, the *direct-form-II* (DF-II) implementation requires the most bits (at least nine bits for integer part). The DF-I implementation of this filter with the proposed pole-zero ordering reduces the signal range significantly. Moreover, augmenting a zero at DC can further reduce the signal range such that the minimum number of integer bits is reduced to five. Note that in this example it is assumed that the input signal (white Gaussian noise) does not have a DC component. In this example, our range reduction technique results in saving four bits in word-length which can significantly reduce the hardware complexity.

To demonstrate the effect of range reduction on hardware complexity, we implemented three filter processors for the three filter designs in Table 3.1. The processors were designed

Table 3.2: Hardware Implementation Results for Different Filter Designs

	Word-length	**FPGA Slices**	**Max Speed**
DF-II	24	2696	174 MHz
DF-I + ord.	20	1768	195 MHz
DF-I + ord. + aug.	18	1164	234 MHz

to work with eight complex filters of order 16. We implemented these filter processors on a Xilinx Virtex-II Pro FPGA [160]. Table 3.2 summarizes the implementation results. As this table shows, due to the reduced word-length, the hardware complexity has been reduced by more than 55% and the maximum operation speed has been increased by more than 34%.

We also implemented a 16-bit version of the designed filter processor on a GVA-290 FPGA development board [161]. In this filter processor, the filter coefficients are presented in 12-bit fixed-point format. As an example, we took the filter coefficients from the sample filter (c) in Figure 3.13. To measure the filter response, white Gaussian noise [138] was filtered. Figure 3.15 shows the power spectrum of the filter output. Comparing the output spectrum with the desired filter response confirms the accuracy of our filter design procedure.

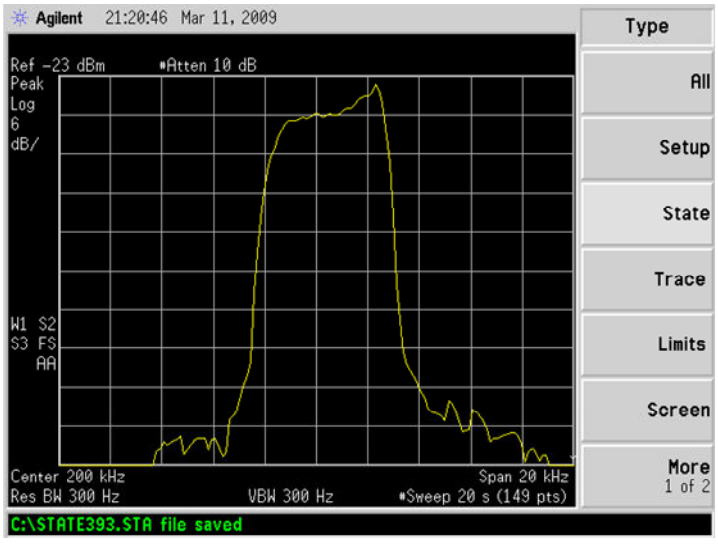


Figure 3.15: Measured power spectrum of the filtered noise.

Example 2

For the second example, using the PSD model in equation (3.3), we designed three SSFs for three different isotropic and non-isotropic fading scenarios (d), (e), and (f). In scenario (d) we generated the complex path gains that simulate an isotropic fading channel with

Doppler frequency $f_D = 9$ Hz. In (e) we simulated a non-isotropic fading system with Doppler frequency $f_D = -18.5$ Hz, beam-width $\kappa = 1$, and AOA $\tilde{\psi} = \pi/4$ rad. Finally in (f) we have $f_D = 2.25$ Hz, beam-width $\kappa = 5$, and AOA $\tilde{\psi} = \pi/3$ rad.

The SSFs for the three fading channels were designed with $\Gamma = 10$ FOSs, and we set $\vartheta = 5$ and $\varrho = 0.99$. For all of the FOSs, the number of bits for representing each coefficient is set to $\Omega = 16$. Figure 3.16 shows the frequency responses of the designed filters (with $\varepsilon = 0.001$) as well as the desired responses. The desired responses are the spectra for the three SSFs with different fading characteristics. As this figure shows, the designed filters accurately produce the desired responses within the pass-band. In the stop-band, the designed filter provides more that 55 dB attenuation for these examples.

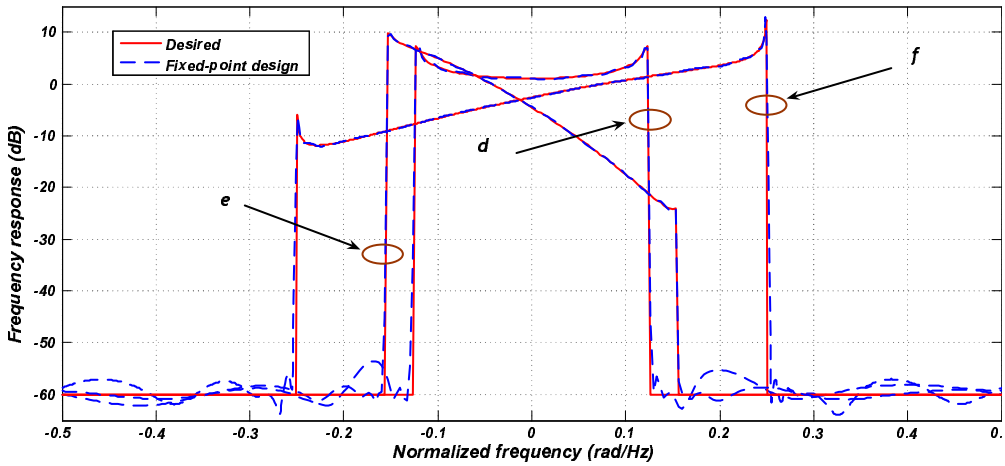


Figure 3.16: Frequency response of the designed SSFs in Example 2. (d) $f_D/F_1 = 0.125$, $\kappa = 0$, and $\tilde{\psi} = 0$, (e) $f_D/F_1 = -0.15625$, $\kappa = 5$, and $\tilde{\psi} = \pi/5$, (f) $f_D/F_1 = 0.25$, $\kappa = 3$, and $\tilde{\psi} = \pi/4$.

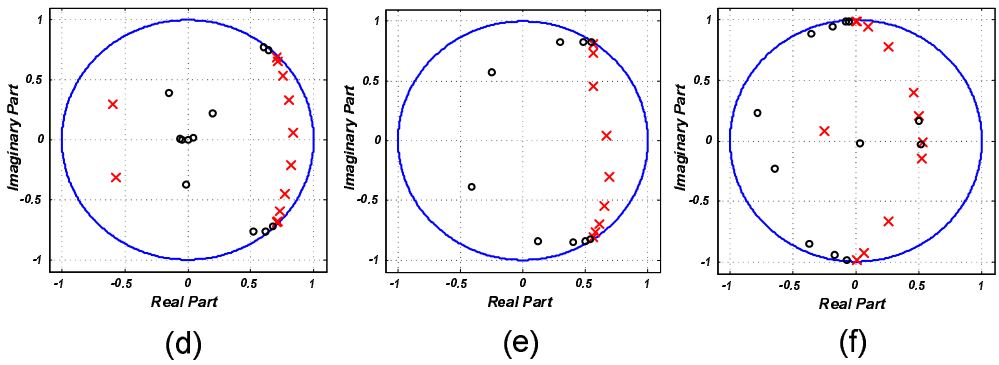


Figure 3.17: Positions of the poles and zeros in the unit circle for the designed SSFs in Example 2. (d) $f_D/F_1 = 0.125$, $\kappa = 0$, and $\tilde{\psi} = 0$; (e) $f_D/F_1 = -0.15625$, $\kappa = 5$, and $\tilde{\psi} = \pi/5$; (f) $f_D/F_1 = 0.25$, $\kappa = 3$, and $\tilde{\psi} = \pi/4$.

Figure 3.17 shows the position of the poles and zeros for the designed filters. Note that all the poles and zeros are located within a circle of radius 0.99.

To ensure the accuracy of our fading simulation, we measured the performance of the fixed-point bit-true model of our hardware fading simulator with the designed filters. The simulations were performed in fixed-point arithmetic, where the filter coefficients were represented with $\Omega = 16$ bit variables and 18-bit variables are used to store the intermediate signals. Also, the target sample rate for the scenarios (d), (e), and (f) was set to $F_s = 40$ MHz.

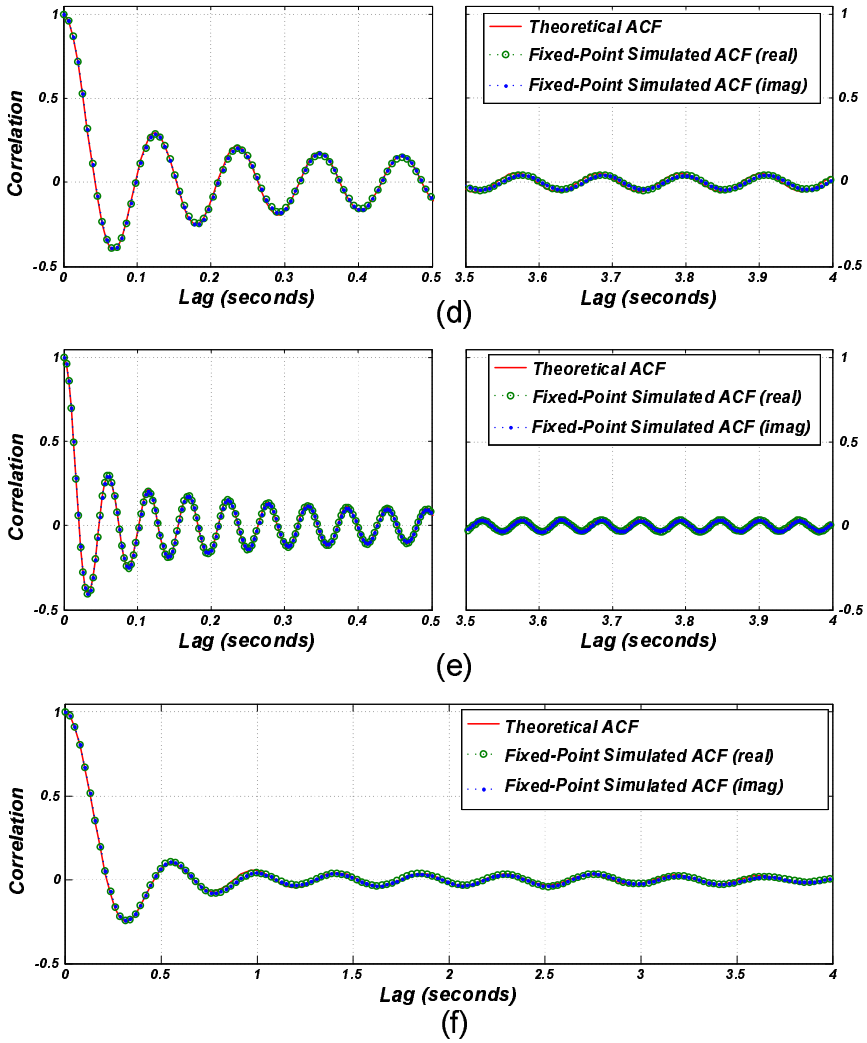


Figure 3.18: Normalized autocorrelation for the real and imaginary components of the generated fading processes in Example 2. (d) $f_D = 9$ Hz, $F_s = 40$ MHz, $\kappa = 0$, and $\tilde{\psi} = 0$ rad, (e) $f_D = -18.5$ Hz, $F_s = 40$ MHz, $\kappa = 1$, and $\tilde{\psi} = \pi/4$ rad. (f) $f_D = 2.25$ Hz, $F_s = 40$ MHz, $\kappa = 5$, and $\tilde{\psi} = \pi/3$ rad.

Figure 3.18 compares the autocorrelation for the real and imaginary components of the generated path gains with the theoretical references. Note that there is a close match between the desired and generated autocorrelations up to 4 seconds (160 million samples). Also, Figure 3.19 plots the cross-correlation between the real and imaginary components of the generated path gains and the reference curves for up to four seconds. This figure confirms a close match between the achieved and desired curves.

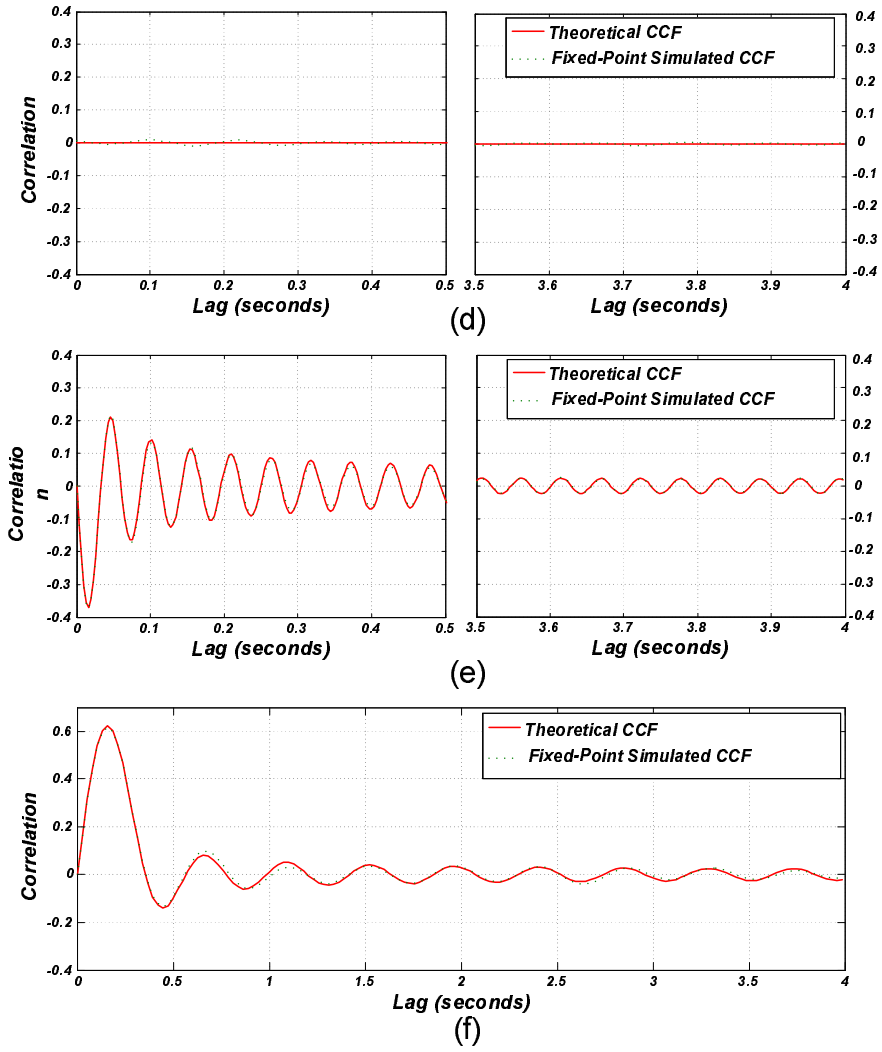


Figure 3.19: Normalized cross-correlation between the real and imaginary components of the generated fading processes in Example 2. (d) $f_D = 9$ Hz, $F_s = 40$ MHz, $\kappa = 0$, and $\tilde{\psi} = 0$ rad, (e) $f_D = -18.5$ Hz, $F_s = 40$ MHz, $\kappa = 1$, and $\tilde{\psi} = \pi/4$ rad. (f) $f_D = 2.25$ Hz, $F_s = 40$ MHz, $\kappa = 5$, and $\tilde{\psi} = \pi/3$ rad.

Figure 3.20 compares the normalized LCR of the amplitude of generated complex path gains in scenarios (d), (e), and (f) with the theoretical LCR from (3.6). The LCR is normalized to $f_D \times T_s$. This figure again confirms excellent agreement between the theoretical and generated curves. In Figure 3.21 the PDF of the amplitude of the generated samples in scenario (f) is plotted. It can be observed that this PDF accurately mimics the Rayleigh PDF. Also the normalized AFD for scenarios (d), (e), and (f) are plotted in Figure 3.22. This figure shows that the simulated AFD matches the reference curve (from equation (3.7)) with good accuracy over a wide range of normalized fading durations.

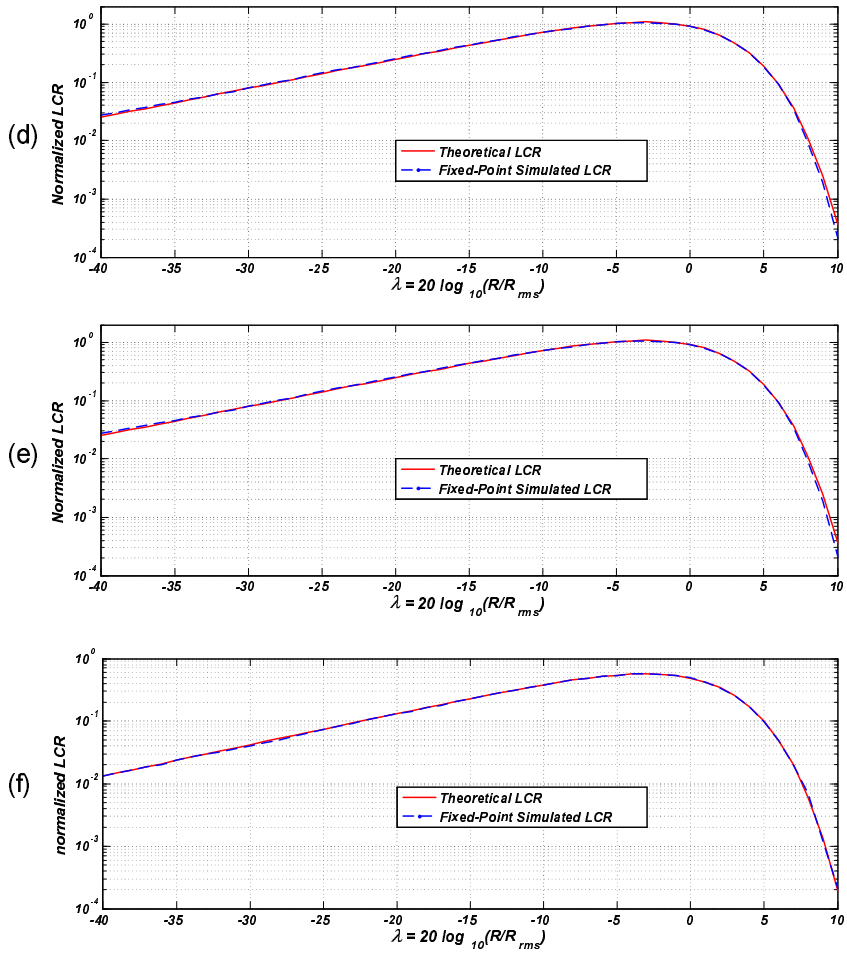


Figure 3.20: Normalized envelope level crossing rate of the generated fading processes in Example 2. (d) $f_D = 9$ Hz, $F_s = 40$ MHz, $\kappa = 0$, and $\tilde{\psi} = 0$ rad, (e) $f_D = -18.5$ Hz, $F_s = 40$ MHz, $\kappa = 1$, and $\tilde{\psi} = \pi/4$ rad. (f) $f_D = 2.25$ Hz, $F_s = 40$ MHz, $\kappa = 5$, and $\tilde{\psi} = \pi/3$ rad.

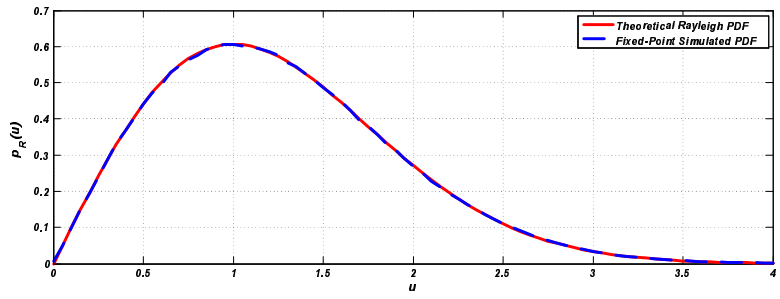


Figure 3.21: Probability density function of the amplitude of the generated fading process for the system (f) with $f_D = 2.25$ Hz, $F_s = 40$ MHz, $\kappa = 5$, and $\tilde{\psi} = \pi/3$ Rad.

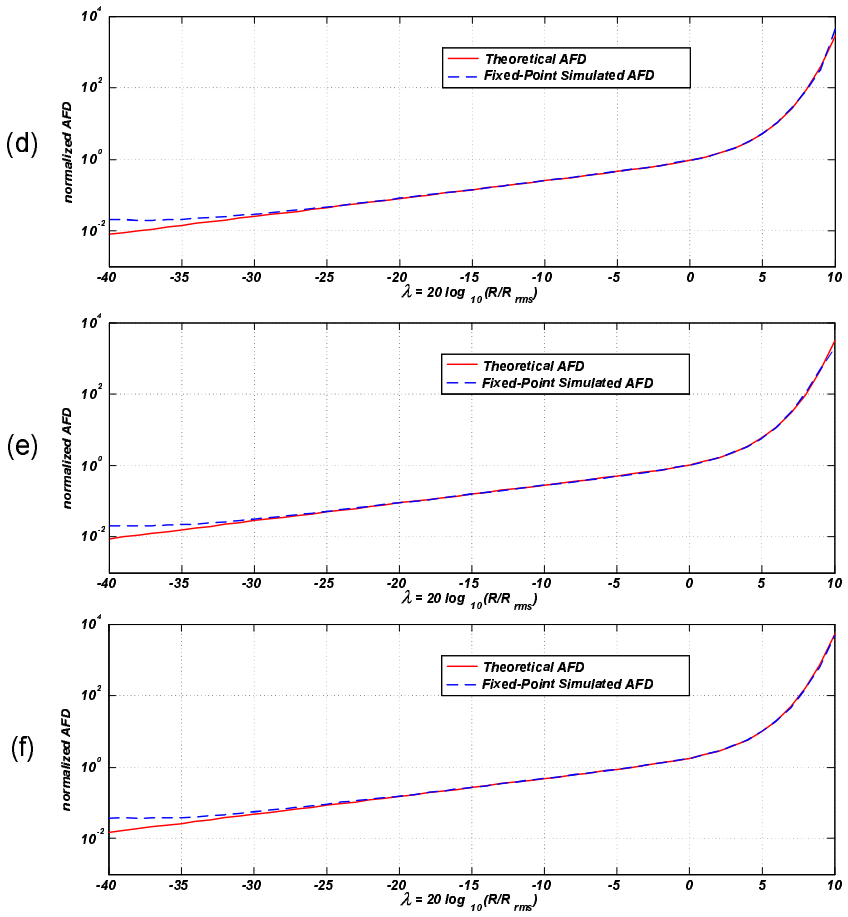


Figure 3.22: Normalized envelope average fade duration of the generated fading processes in Example 2. (d) $f_D = 9$ Hz, $F_s = 40$ MHz, $\kappa = 0$, and $\tilde{\psi} = 0$ rad, (e) $f_D = -18.5$ Hz, $F_s = 40$ MHz, $\kappa = 1$, and $\tilde{\psi} = \pi/4$ rad. (f) $f_D = 2.25$ Hz, $F_s = 40$ MHz, $\kappa = 5$, and $\tilde{\psi} = \pi/3$ rad.

Example 3

For the third example, we designed appropriate filters to simulate the PSD proposed for the IEEE 802.11n indoor wireless fading channel model [92]. In this model, the bell-shaped PSD

$$S(f) = \begin{cases} \frac{1}{1+A\left(\frac{f}{f_D}\right)^2} + \frac{B}{1+C\left(\frac{f-f_{spike}}{f_{spike}}\right)^2}, & \text{if } |f| \leq f_{max} \\ 0, & \text{if } |f| > f_{max} \end{cases} \tag{3.22}$$

is proposed for representing indoor propagation. Here, f_D denotes the maximum Doppler frequency, which is set to 6 Hz or 3 Hz for carrier frequencies of 5.25 GHz and 2.4 GHz, respectively, based on experimental measurements. Also, f_{max} is the maximum frequency component of the Doppler spectrum, which can be set to several times the Doppler frequency [92]. The second term in equation (3.22) corresponds to a Doppler component that

represents a reflection from a moving vehicle as described in model F of [92]. This component is identified with a spike in the PSD at frequency $f_{spike} = \nu_v/\lambda$, where ν_v is the vehicle speed and λ is the signal wavelength. The proposed values for the constants A, B, and C are 9, 0.5, and 90000, respectively [92].

We simulated an indoor fading channel with the above specifications for a system with carrier frequency $F_c = 2.4$ GHz, maximum Doppler frequency 3.0 Hz, and vehicle speed 40.0 km/h, which corresponds to $f_{spike} = 88.9$ Hz. The Doppler spectrum of the designed filter along with the matching reference spectrum are shown in Figure 3.23.

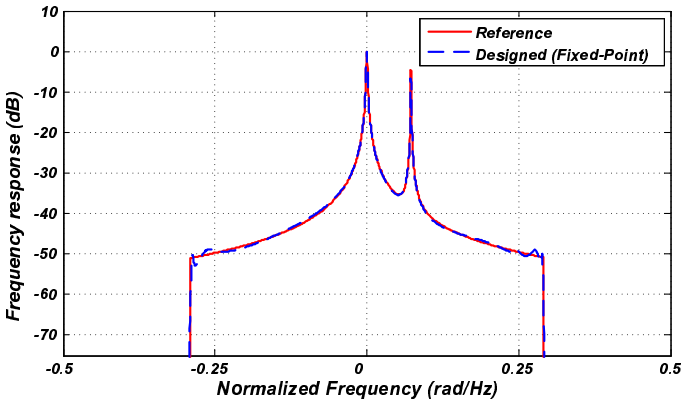


Figure 3.23: Reference and designed "Bell-shaped" Doppler spectra with the Doppler component due to a moving vehicle.

These simulations were also performed in fixed-point arithmetic, where the filter coefficients are represented with $\Omega = 16$ bit variables and 18-bit variables are used to store the intermediate signals. Also, all poles and zeros are constrained to lie within a circle of radius $\varrho = 0.99$ and the target sample rate for all scenarios is $F_s = 40$ MHz. The Doppler spectrum is modeled with a complex filter of order $\Gamma = 4$. Figure 3.24 shows the autocorrelation of the quadrature components of the simulated fading process. The cross-correlation between quadrature components is also plotted in the same figure. Finally, the normalized LCR and AFD of this channel are plotted in Figures 3.25 and 3.26, respectively. As for example 1, the LCR and AFD are normalized to $f_D \times T_s$. Note that compared to the three scenarios in the previous example, the simulated indoor 802.11n model has a higher LCR and lower AFD.

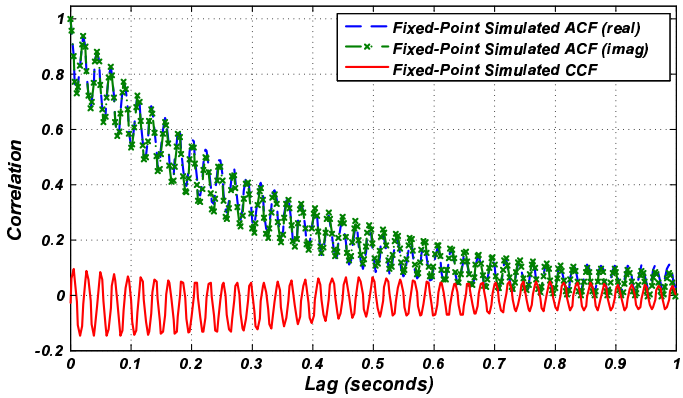


Figure 3.24: Autocorrelation and cross-correlation of quadrature components of the simulated fading process in Example 3.

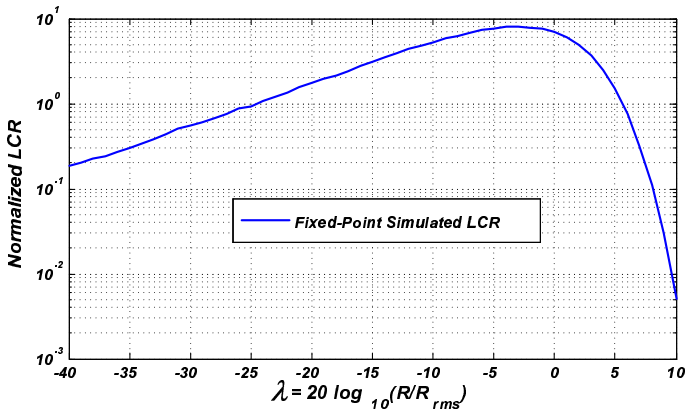


Figure 3.25: Normalized envelope level crossing rate of the system in Example 3.

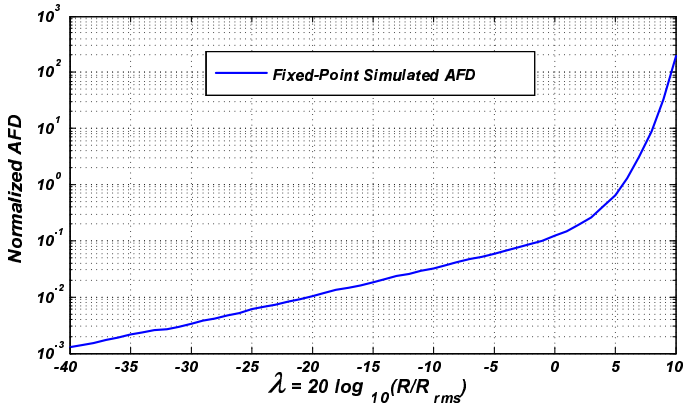


Figure 3.26: Normalized envelope average fade duration of the system in Example 3.

3.3.4 Implementation

To demonstrate the efficiency of the above filter design process, we implemented a hardware fading simulator that exploits a filter processor that is optimized for performing the necessary filtering operations. The hardware was designed to generate fixed-point results that are identical to those produced by our bit-true software simulator. Figure 3.27 shows the datapath of the filter processor that can perform filtering operations using complex coefficients of cascaded FOSs. In Figure 3.27 the coefficients are stored in separate RAMs for $a_Q(k) = s_k \sin(\phi_k)$, $a_I(k) = s_k \cos(\phi_k)$, $b_I(k) = r_k \cos(\theta_k)$, and $b_Q(k) = r_k \sin(\theta_k)$.

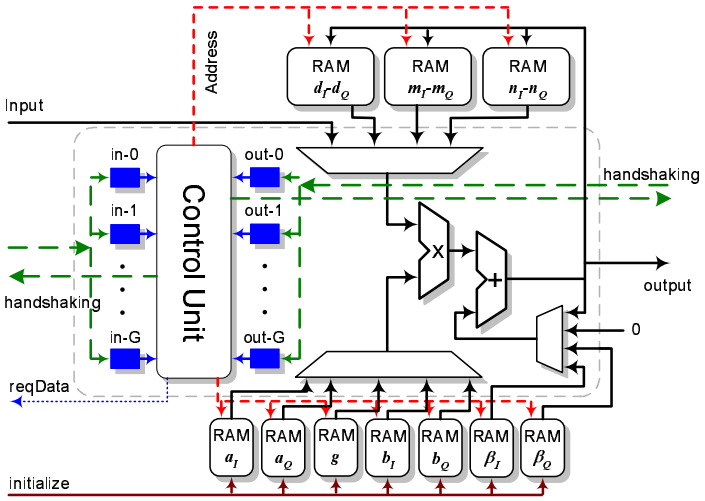


Figure 3.27: Datapath of our filter processor which performs filtering operations using complex coefficients of cascaded first-order sections.

In order to reduce the variable range and the risk of overflow, the poles and zeros are matched in pairs so that each pole appears with the closest zero on the unit circle. The matching process starts with the poles with the largest magnitude since their impulse response has more impact on the variable ranges. Direct-form-I is used for the filter implementations. This way, in each FOS, the signal is first passed through the *moving average* (MA) part of the filter. With zero-pole matching, the signal is attenuated at frequencies close to the pole locations. This way the the variable range, and accordingly the overflow risk, is significantly reduced.

Further, since the filter input has a Gaussian distribution, fixed scaling factors $g(k)$ for each stage are used to maintain the signal magnitude within the representable range. By employing the above techniques, we obtain a compact and efficient implementation with

the minimum number of bits in each variable. To make our design more resilient against overflows and the resulting instability, the adder in the filter datapath in Figure 3.27 is implemented to saturate its output in case of overflow.

The main element in the datapath in Figure 3.27 is a MAC that can multiply the in-phase and quadrature signal components by complex-valued coefficients $a(k) = a_I(k) + ja_Q(k)$ and $b(k) = b_I(k) + jb_Q(k)$. The output of each FOS is written into "RAM $d_I - d_Q$", which holds the intermediate results. "RAM $m_I - m_Q$" and "RAM $n_I - n_Q$", store the complex contents of the two memories in the DF-I implementation of a FOS. The coefficients are stored in "RAM a_I", "RAM a_Q", "RAM b_I", "RAM b_Q", and "RAM g". This datapath can also add bias values from "RAM β_I" and "RAM β_Q", which is necessary when simulating Rician fading. Also, this datapath can be used for performing zero-padding for interpolation, when required.

To implement the required lowpass filter for interpolation, the control unit in the datapath in Figure 3.27 can be slightly modified to perform filtering operations of a real-filter with second-order sections, which requires the same number of multiplications and additions as a complex FOS. For an efficient implementation, additional interpolation can be carried out with the SILPF, as described in Section 3.2.1.

We implemented the datapath in Figure 3.27 and compared the implementation results with implementations in Section 3.2.2 and Section 3.2.3. Table 3.3 compares the implementation results on a Xilinx Virtex-4 XC4VLX200-11 FPGA. All of the designs are configured to process eight independent streams of fading samples and the filter order is set to $\Gamma = 16$. The design in Section 3.2.2 was reconfigured to meet the above requirements (eight instantiations of complex filters of order 16). For a fair comparison, in all designs, the memories are implemented with distributed RAMs (i.e., LUT-based memories) and no dedicated multipliers are used.

Please note that in Table 3.3, the *Design I* (from Section 3.2.3) processes filters with real-coefficients and therefore requires half the memory to store the coefficients and also performs half the arithmetic operations of its complex-valued counterpart. However, the new filter design procedure results in a more compact and efficient design, as shown in Table 3.3. More specifically, compared to the design in Section 3.2.2, the new filter processor utilizes almost nine times fewer configurable slices and can process 22.5 times more samples per second.

Table 3.3: Characteristics of Filter Processors for Fading Channel Simulators

Design	**I** [a]	**II** [b]	**III (NEW)**
Device	XC4VLX200-11	XC4VLX200-11	XC4VLX200-11
Fading	Rayleigh	Rician	Rician
Scattering	Non-isotropic	Isotropic	Non-isotropic
Datapath word-length	40-bits	36-bits	18-bits
Coefficient word-length	32-bits	32-bits	16-bits
Filter order	16	16	16
Filter Coefficients	Complex	Real	Complex
Number of fading paths	8	8	8
Configurable slices	10328	1348	1164
Resource utilization	(11.6%)	(1.5%)	(1.3%)
Max. Clock freq. (MHz)	128	87	234
Output rate (KSamp/Sec) [c]	26	435	585

[a] from Section 3.2.2

[b] from Section 3.2.3

[c] *Design I* requires 16×2459 clock cycles for 8 output samples, *Design II* requires $(16/2) \times 25$ clock cycles per output sample (this design processes real filters) and *Design III* needs $(16) \times 25$ clock cycles per output sample.

3.3.5 Implementation Comparison

Figure 3.28 compares the implementation results for different filter-based fading simulators. All of the fading simulators are implemented in Verilog HDL and synthesized on a Xilinx Virtex-II Pro FPGA XC2VP100-6. The implemented fading simulators are from [9–14] (the results of [14] are from Section 3.3). The resource utilization figures are divided up based on the number of generated fading paths.

Figure 3.28 (a) compares the maximum clock frequency of different designs. All of the designs are fully pipelined. The maximum clock frequency among all of these designs belongs to last design which is based on the filter design technique in Section 3.3. In this implementation, DF-I filter structure is used and the input noise is colored. In addition, an *interpolated finite impulse response* (IFIR) filter [162] is utilized for the first interpolation stage. Figure 3.28 (b) compares the maximum output sample rate for different designs [1]. Specifically, the last design *vi* generates up to 268 million samples per second (only one path). Figure 3.28 (c) compares the required number of configurable slices per fading path. As this figure shows, the design in [12] requires the most slices per path, and the last design *vi* is the most efficient in terms of required number of slices per path. Moreover, design *vi* requires only 278 slices per fading path. Figure 3.28 (d) and (e) compare the required

[1] The fading simulators proposed in [11, 12, 14] are designed in multiple stages and their output sample rate depends on the maximum speed of the final interpolator stage.

Table 3.4: Mean Square Error of Different Statistical Measures

MSE_{ACF}	MSE_{CCF}	MSE_{PDF}	MSE_{LCR}	MSE_{AFD}
-85 (dB)	-82 (dB)	-77 (dB)	-80 (dB)	-75 (dB)

number of on-chip multipliers and block memories per fading path. As this figure show, newer designs require less and less multipliers and storage per fading path.

We also measured the mean square error (MSE) of different statistical measures (ACF, CCF, PDF, LCR, AFD) for design *vi* over one continuous block of Rayleigh fading samples of length 10^8 with normalized Doppler frequency $f_D T_s = 0.001$. For a fair comparison, we used the same measurement parameters as those used for testing the SOS-based fading simulator. Table 3.4 represents the measured MSEs. Comparing these results with the results of the SOS-based simulator (see Figure 2.21) shows that the filter-base method can provide much more accurate results.

3.4 Simulation of Nakagami-m and Weibull Fading Channels

The Nakagami-m distribution (or the m-distribution), is another commonly used model for the simulation and study of fading channels [39]. This distribution can model severe to moderate fading through the parameter m. This distribution is recommended for modeling and simulation of fading channels due to its good fit to data obtained from several radio channels [163–168].

Several methods have been proposed in the literature for the simulation of Nakagami-m fading channels [42, 43, 72, 73, 169–176]. Two Nakagami-m fading simulators based on the sum-of-sinusoids approach have been proposed in [72, 73]. In [169], uncorrelated Nakagami-m samples are first generated using any random number generation method, and then autocorrelation is introduced between Nakagami-m samples by sorting them according to the rank statistics of additional Rayleigh samples with the desired autocorrelation. In [170], a Nakagami fading signal with $m < 1$ is simulated using complex Gaussian processes and square-root-Beta random processes. It has also been proposed to generate Nakagami-m distributed samples from Gamma-distributed samples [171–173] and the Gamma-distributed samples have been correlated using either the Cholesky decomposition of the covariance matrix or the Sim's method [177]. In [174, 175], the effect of Nakagami-m fading on the signal-to-noise ratio at the receiver has been simulated as a finite-state Markov chain. Also in [42, 43, 176] it is proposed to use a transformation for mapping

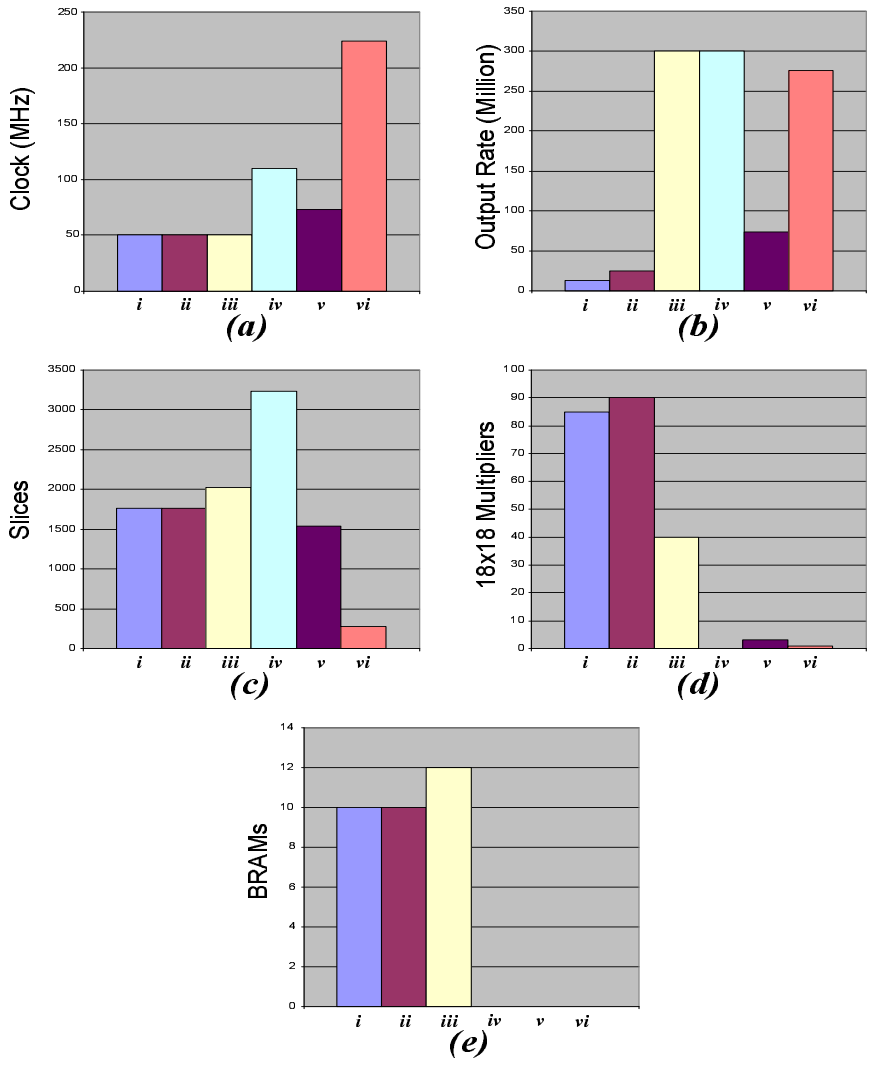


Figure 3.28: Comparison between implementations of different filter-based fading simulators. All of the fading simulators have been implemented in Verilog HDL and synthesized on a Xilinx Virtex-II Pro FPGA XC2VP100-6. The plots show (a) maximum clock frequency, (b) maximum output sample rate, (c) number of configurable slices per generated path, (d) number of utilized 18×18 dedicated multipliers per generated path and, (e) number of utilized on-chip 18 Kb block memories per generated path. The implemented fading simulators are from (*i*) the model proposed in [9] (see Section 3.2), (*ii*) the model proposed in [10] (see Section 3.2), (*iii*) the model proposed in [11] (see Section 3.2.1), (*iv*) the model proposed in [12] (see Section 3.2.2), (*v*) the model proposed in [13] (see Section 3.2.3), and (*vi*) using the model from Section 3.3 (submitted for publication in [14]).

Rayleigh sequences into Nakagami-m sequences.

However, except for the sum-of-sinusoids approach [72, 73], the rest of the above simulators are not appropriate for hardware implementation of a parameterizable continuous-time Nakagami-m fading simulator. This is mainly due to the large memory requirements [169], the block-based nature of the algorithm, [169–173], the high computational complexity [42, 43, 169–173, 176], or the high-precision arithmetic requirements [42, 43, 176]. The sum-of-sinusoids approach on the other hand is not flexible for simulating arbitrary time-correlation properties between fading samples.

Here we want to have a simulation method that can generate any time-correlation between the Nakagami-m fading samples for various values of m. The simulation method also needs to be appropriate for hardware implementation. The Nakagami fading simulator proposed in [42, 43] can simulate different fading scenarios, and the time-correlation between the generated fading samples can be parameterized. This simulation technique is based on transforming Rayleigh fading samples to Nakagami-m fading samples with similar time-correlation properties. However, due to the wide range of variables used in this fading simulator, it can not be used for efficient fixed-point implementation of a Nakagami fading simulator. Based on the fading simulator proposed in [42, 43], here we present a new architecture for the efficient and compact implementation of a parameterizable Nakagami fading simulator. Due to the similarities between the Nakagami-m distribution and the Weibull distribution, with minor modification, the presented technique can be used for the hardware simulation of Weibull fading channels [178–180] as well.

In the following we briefly present the simulation method that was proposed in [42, 43]. Then we present our model for the efficient hardware simulation of Nakagami-m fading channels.

3.4.1 Basic Simulator

The probability density function of the amplitude R of fading samples in a Nakagami-m fading channel can be expressed as [39]

$$f_N(r) = \frac{2m^m r^{2m-1}}{\Gamma(m)\Pi^m} \exp(-\frac{m}{\Pi} r^2), \quad r \geq 0, \tag{3.23}$$

where $\Pi = E\{R^2\}$ is the average fading power ($E\{\cdot\}$ denotes the expectation operator), and $\Gamma(\cdot)$ is the gamma function. In (3.23), the Nakagami fading parameter $m \geq \frac{1}{2}$ which

determines the fading severity is defined as the ratio of moments [181]

$$m = \frac{\Pi^2}{E\{(R^2 - \Pi)^2\}}, \qquad m \geq \frac{1}{2}. \tag{3.24}$$

The Nakagami-m distribution also includes the one-sided Gaussian distribution as a special case when $m = 1/2$ and the Rayleigh distribution when $m = 1$. Values of m in the ranges $[1/2, 1)$ and $(1, \infty)$ correspond to fading more severe than Rayleigh fading and less severe than Rayleigh fading respectively.

The simulation method proposed in [42, 43], is based on transforming of Rayleigh fading samples to Nakagami-m samples. In this method, a zero-mean complex Gaussian fading process $c(t) = c_i(t) + jc_q(t)$ is first generated using a filter-based or sum-of-sinusoid based Rayleigh fading simulator. The envelope of the fading process $r_R(t) = (c_i^2(t) + c_q^2(t))^{1/2}$ is Rayleigh-distributed, and that can be transformed into samples with uniform distribution using the transformation

$$\begin{aligned} u(t) &= F_R(r_R(t)), \\ &= 1 - e^{-\frac{r_R^2(t)}{2\sigma^2}}, \end{aligned} \tag{3.25}$$

where $F_R(\cdot)$ is the *cumulative distribution function* (CDF) of a Rayleigh random variable and σ^2 is the variance of $c(t)$. The uniform random variable $u(t)$ is then transformed to Nakagami-m random variable $r_N(t)$ by the inverse Nakagami-m CDF function as

$$r_N(t) = F_N^{-1}(u(t)), \tag{3.26}$$

where

$$\begin{aligned} F_N(v) &= \int_0^v \frac{2m^m t^{2m-1}}{\Gamma(m)\Pi^m} \exp(-\frac{m}{\Pi}t^2)dt, \\ &= \frac{\gamma(m, \frac{m}{\Pi}r^2)}{\Gamma(m)}, \end{aligned} \tag{3.27}$$

and $\gamma(a, x)$ is the incomplete gamma function. Complex Nakagami fading samples $x(t)$ are then generated using the Nakagami-distributed envelope $r_N(t)$ as

$$\begin{aligned} x(t) &= x_i(t) + jx_q(t) \\ &= r_N(t)\cos(\theta(t)) + jr_N(t)\sin(\theta(t)), \end{aligned} \tag{3.28}$$

where $\theta(t) = \arctan(c_q(t)/c_i(t))$. In [42, 43], authors also propose an approximation for the inverse Nakagami-m CDF

$$F_N^{-1}(u) \approx \eta(u) + \frac{a_1\eta(u) + a_2\eta^2(u) + a_3\eta^3(u)}{1 + b_1\eta(u) + b_2\eta^2(u)}, \tag{3.29}$$

where

$$\eta(u) = \left(\sqrt{\ln \frac{1}{1-u}}\right)^{\frac{1}{m}}. \tag{3.30}$$

For a given Nakagami fading parameter m, the coefficients a_1, a_2, a_3, b_1, and b_2 are calculated to minimize the approximation error.

This method seems attractive for software simulation of Nakagami-m fading channels. However, efficient hardware implementation of this fading simulator in its original formulation can be quite challenging. Due to the extensive use of various mathematical functions and large dynamic range of intermediate variables, it is necessary for this algorithm to be implemented in floating-point arithmetic to sustain the required accuracy. More specifically, calculating (3.28) requires two multiplications, one division, and calculation of $\arctan$, $\sin$, and $\cos$ functions. Also, calculation of (3.29) requires several additions and multiplications, two divisions, square root, $m-$th root, and natural logarithm. Most of these operations result in variables with wide dynamic ranges that cannot be effectively mapped onto a compact hardware with fixed-point arithmetic.

For accurate implementation of this fading simulator, single- or double-precision floating-point data structures are required to support wide dynamic range while maintaining data precision. However, full floating-point arithmetics are often unsuitable for implementation in FPGA or ASIC. Floating-point operators generally have dramatically increased logic utilization and power consumption, combined with lower clock speed, longer pipelines, and reduced throughput capabilities when compared to integer or fixed-point. On the other hand, careless fixed-point implementation can result in excessive quantization noise. Also, issues of truncation, rounding and overflows can render a fixed-point implementation ineffective. In the next section, we reformulate Beaulieu's fading simulator [42, 43] to make it more suitable for fixed-point implementation.

3.4.2 New Nakagami-m and Weibull Fading Simulator

As mentioned in the previous section, in its original formulation, Beaulieu's Nakagami-m fading simulator cannot be efficiently mapped onto hardware. To simplify this Nakagami

fading simulator, we start by rewriting $\sin(\theta(t))$ and $\cos(\theta(t))$ as

$$\begin{aligned}\sin(\theta(t)) &= \frac{c_i(t)}{\sqrt{c_i^2(t)+c_q^2(t)}},\\ \cos(\theta(t)) &= \frac{c_q(t)}{\sqrt{c_i^2(t)+c_q^2(t)}}.\end{aligned}$$

which implies that calculation of $\sin(\theta(t))$ and $\cos(\theta(t))$ requires an inverse square root operation and two multiplications. Since the inverse square root operation is common for both quadrature components, it is possible to factorize it for more compact implementation. From here, Nakagami fading samples can be generated as

$$x(t) = g(r_R^2(t)) \times (c_i(t) + jc_q(t)),$$

where $g(r_R^2(t))$ is the transfer function defined as

$$g(r_R^2(t)) = \frac{F_N^{-1}(1-e^{-\frac{r_R^2(t)}{2\sigma^2}})}{\sqrt{r_R^2(t)}}. \tag{3.31}$$

For hardware implementation, equation (3.31) is evaluated and stored in a *look-up table* (LUT) for a practical range. Since this function $F_N^{-1}(u)$ is continuous, several root-finding algorithms can be used to evaluate (3.31). In particular, we used the bisection method due its simplicity and effectiveness.

Weibull fading channels can be simulated with the same method. The CDF of the Weibull distribution is [180]

$$F_W(r) = 1 - \exp\left\{-\left(\frac{r}{\Pi}\right)^{\beta}\right\}, \quad r \geq 0, \tag{3.32}$$

where the fading parameter $\beta > 0$ corresponds to fading severity. Based on the Weibull CDF, the transfer function for converting Rayleigh samples to Weibull samples is

$$g_W(r_R^2(t)) = \frac{F_W^{-1}(1-e^{-\frac{r_R^2(t)}{2\sigma^2}})}{\sqrt{r_R^2(t)}}, \tag{3.33}$$

$$= \frac{\Pi}{\sqrt{r_R^2}} \times \left(\frac{r_R^2}{2\sigma^2}\right)^{\frac{1}{\beta}}. \tag{3.34}$$

The transfer function $g_W(r_R^2(t))$ can be used to generate Weibull fading samples from Rayleigh-distributed random variates. In the following we present our fading simulation

method for the Nakagami-m method. However, the only difference is in the transfer functions (3.31) and (3.33). Weibull samples can be generated easily by substituting $g(r_R^2(t))$ with $g_W(r_R^2(t))$.

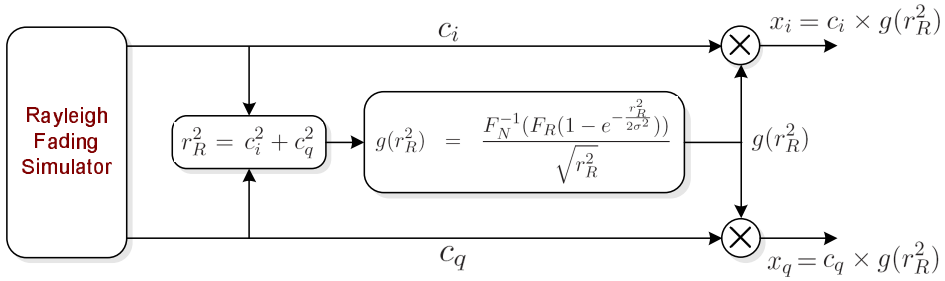


Figure 3.29: Block diagram of the complex Nakagami-m fading channel simulator.

Figure 3.29 shows the block diagram of the Nakagami-m fading simulator. As this block diagram shows, to generate Nakagami samples, first the square envelope of the Rayleigh process r_R^2 is calculated which is used to compute the transfer function $g(r_R^2)$. The in-phase and quadrature components of the Nakagami fading samples are then found by multiplying $c_i(t)$ and $c_q(t)$ by $g(r_R^2)$ respectively.

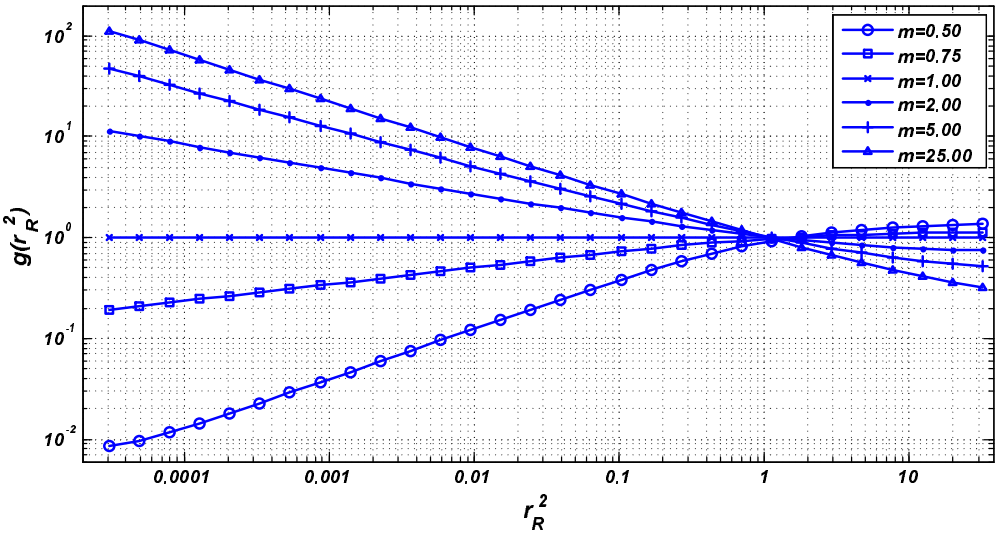


Figure 3.30: Log-Log plot of the transfer function $g(r_R^2)$ for $\Pi = 1$ and different values of m.

The next step is to evaluate the function $g(r_R^2)$. Software simulations showed that accurate calculation of $g(r_R^2)$ is crucial for the accurate simulation of Nakagami-m fading channels. Figure 3.30 plots $g(r_R^2)$ as a function of $r_R^2 \in [2^{-15}, 2^5)$ for different values of m

on a log-log scale. In this figure and in the rest of this section we assume that the average fading power is $\Pi = 1$. For $\sigma^2 = 1$, since $\Pr(r_R^2(t) > 2^5)$ is less than 1.125×10^{-7}, and $\Pr(r_R^2(t) < 2^{-15})$ is less than 1.526×10^{-5}, we focused on evaluating $g(r_R^2)$ over the range $r_R^2 \in [2^{-15}, 2^5)$ without significantly affecting the output statistics (as we will see later). Our simulation results show that for $m \in [0.5, 25]$, and $r_R^2 \in [2^{-15}, 2^5)$, the variations in $g(r_R^2)$ can be as high as four orders of magnitude. Also, $g(r_R^2)$ tends to change faster for small values of r_R^2.

Our simulation results showed that to have an acceptable representation of the transfer function $g(r_R^2)$, we require a minimum resolution of 2^{-15} for the lower part of the range $r_R^2 \in [2^{-15}, 2^5)$. However, no such resolution is required for the upper part of this range since $g(r_R^2)$ changes slowly in this region. On the other hand, since $g(r_R^2)$ has a wide range (2^{-7} to 2^7, see Figure 3.30), at least 14 bits are required for the fixed-point storage of this function. Hence, a linear look-up table needs to store $14 \times 2^{5+15} = 14$ Megabits of information for acceptable representation of $g(r_R^2)$. In other words, it would take at least 797 18-Kbit block memories to store such look-up table!

Instead of using a simple linear look-up table, we calculated $g(r_R^2)$ using linear approximation. In this approach, the range $[2^{-15}, 2^5]$ is divided into small segments and the value of $g(r_R^2)$ over each segment is approximated with a linear equation. For example, over the segment $t \in [t_0, t_1)$, $g(r_R^2)$ is approximated as $\hat{g}(t) \approx \tilde{a}_{t_0}(t - t_0) + \tilde{b}_{t_0}$. We further reduced the number of bits required to store $\{\tilde{a}\}$ and $\{\tilde{b}\}$, with semi floating-point representation. More specifically, over the segment $t \in [t_0, t_1)$, $g(r_R^2)$ is approximated as $\hat{g}(t) \approx 2^{f_{t_0}} \times [a_{t_0}(t - t_0) + b_{t_0}]$, where $a_{t_0} = 2^{-f_{t_0}} \times \tilde{a}_{t_0}$ and $b_{t_0} = 2^{-f_{t_0}} \times \tilde{b}_{t_0}$. Figure 3.31 illustrates this approximation approach.

The storage requirements can be further reduced by non-linear segmentation of the range $[2^{-15}, 2^5)$. We specifically used a hybrid logarithmic-linear segmentation approach for the approximation of $g(r_R^2)$. This hybrid segmentation approach is illustrated in Figure 3.31. As mentioned before, $g(r_R^2)$ changes faster for smaller values of r_R^2. Hence more accurate approximations, i.e., smaller segments, are required for the lower range. On the other hand, since $g(r_R^2)$ changes slowly for larger values of r_R^2, wider segments can be used.

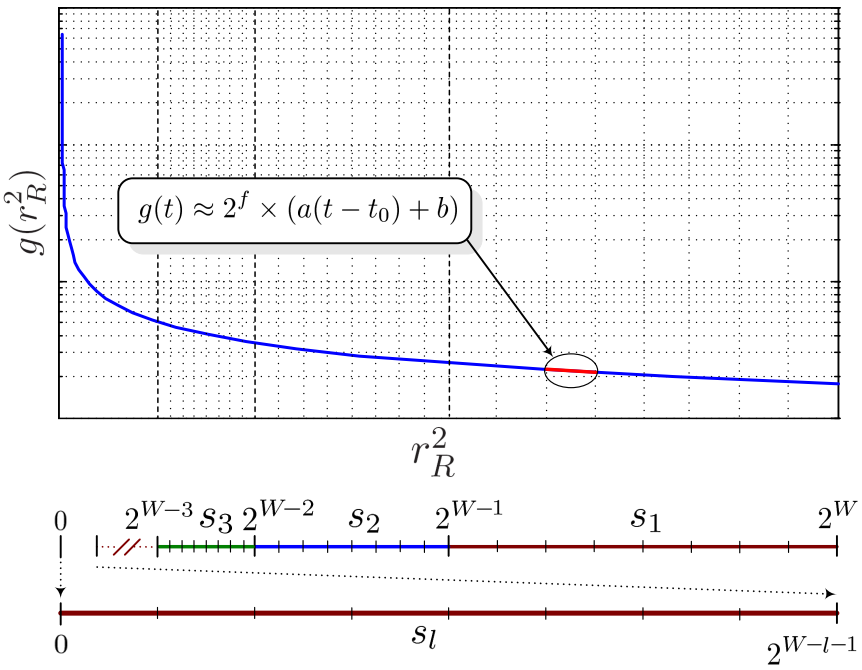


Figure 3.31: Hybrid segmentation and linear approximation of transfer function $g(r_R^2)$.

As Figure 3.31 shows, in this hybrid segmentation approach, the range $(0, 2^W]$ is divided to l logarithmic (i.e., power of 2) segments. Each of these segments are further divided to 2^k linear sub-segments. This segmentation method is particularly convenient for hardware implementation as the logarithmic segment for each sample can be determined using a "leading-1" circuit. Moreover, the next k bits after the "leading-1" can be used to address the linear sub-segment in the memory, and the remaining bits can be used as the argument of the semi floating-point linear function (i.e., $t - t_0$).

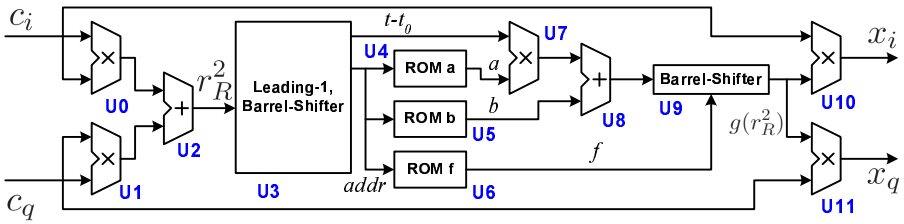


Figure 3.32: Datapath of the transformation-based Nakagami-m and Weibull fading simulator.

Figure 3.32 shows the datapath of the Nakagami-m fading simulator based on the above approach. As this figure shows, the calculated r_R^2 is passed to the "leading-1" detector and "barrel-shifter" `U3` to find the address `addr` of the current hybrid segment and the argument

$t - t_0$. The address for the current segment `addr` is then used to read the values a, b, and f from the corresponding ROMs. Then the value of $g(r_R^2)$ is approximated in `U7`, `U8`, and `U9` as $\hat{g}(r_R^2) \approx 2^f \times [a(t - t_0) + b]$. The quadrature components of the Nakagami fading samples are then calculated as $x_i = c_i \times \hat{g}(r_R^2)$ and $x_q = c_q \times \hat{g}(r_R^2)$ in `U10` and `U11` respectively.

We performed some simulations using the fixed-point bit-true model of the above Nakagami-m fading simulator to verify its accuracy. Based on our fixed-point analysis, we divided the range $[2^{-15}, 2^5)$ into 15 logarithmic segments. Each of the logarithmic segments was further divided into 64 linear sub-segments. The range $[0, 2^{-15})$ was also divided into 64 segments which makes the total number of hybrid segments $(1 + 15) \times 64 = 1024$. The values of a and b were stored in `s18.17` format (i.e., signed 2's compliment 18-bit fixed-point values with 17-bit fraction representing the range $[-0.5, 0.5)$). The values of f were stored in `s5.0` format (i.e., signed 2's complement 5-bit fixed-point integer values representing the range $[-16, 15]$). The input samples c_i and c_q were assumed to be in `s16.11` format. In the hardware implementation, the values of a, b, and f were each stored in a 18-Kbit on-chip block memory.

Figure 3.33 shows the relative approximation error of $g(r_R^r)$ for $m = 10$. The relative approximation error is defined as $e_g(r_R^2) = |1 - \hat{g}(r_R^2)/g(r_R^2)|$. As this figure shows, the presented segmentation and approximation method can accurately mimic the actual transfer function $g(r_R^2)$. In this figure we see that the relative error goes up in the upper region. As we will see in the next simulation results, this increase in the relative error does not have a significant impact on the distribution of the generated Nakagami samples since large values of r_R^2 are less likely and $g(r_R^2)$ goes to zero for such values. Moreover, for very small values of r_R^2, the relative error is higher than the rest of the range, but the effect of this approximation error on the statistical accuracy of the generated Nakagami-m samples is insignificant.

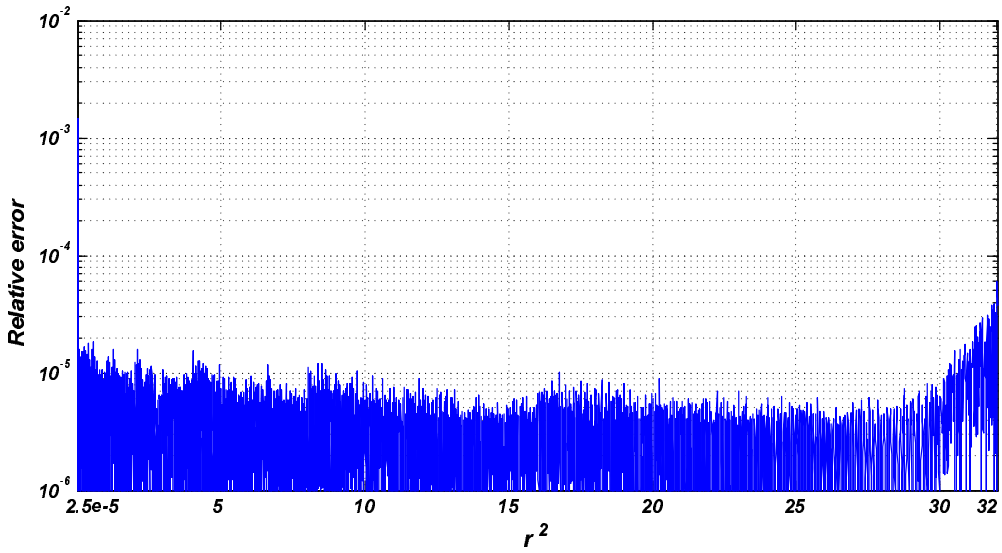


Figure 3.33: Relative approximation error of transfer function $g(r_R^2)$ for $m = 10$.

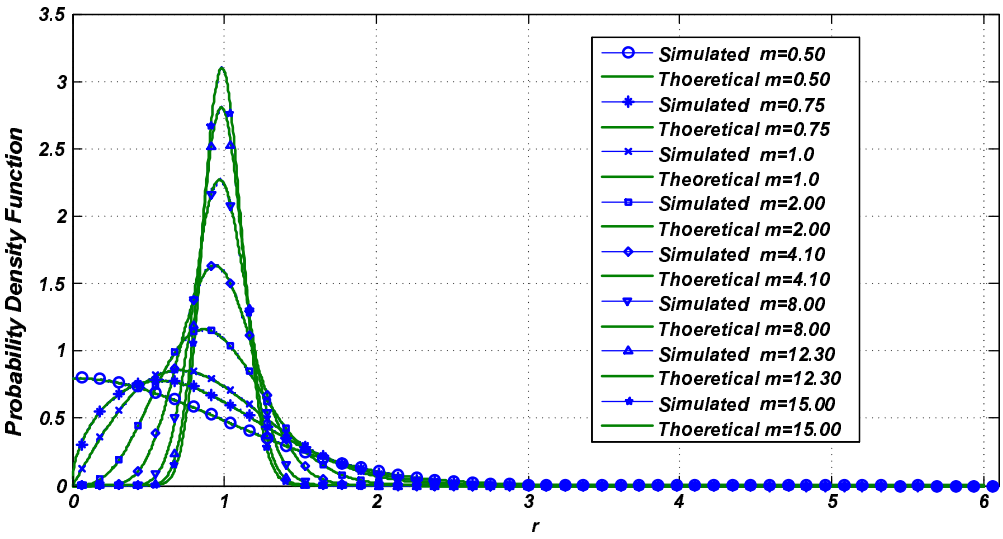


Figure 3.34: Comparison between reference Nakagami-m PDF and the measured PDF of generated samples for different values of m.

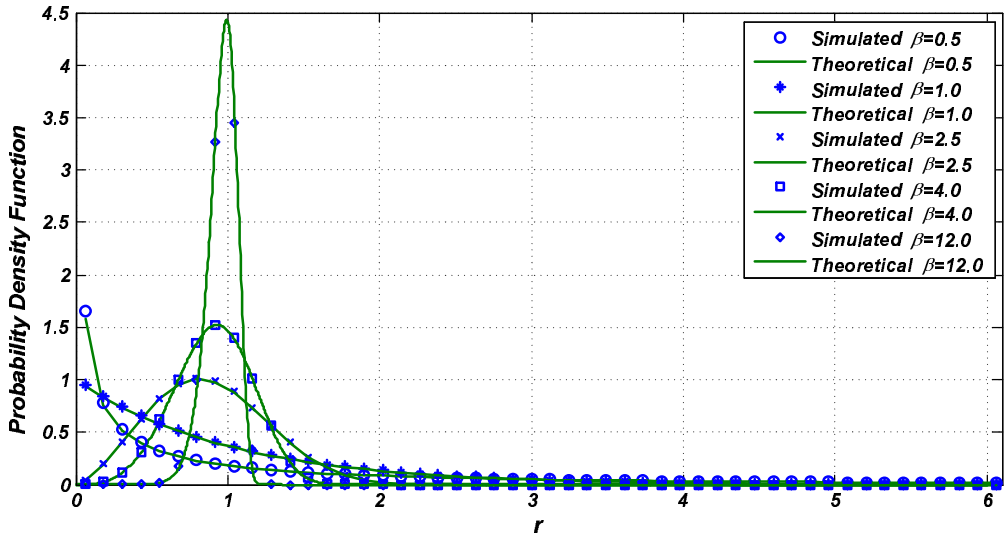


Figure 3.35: Comparison between reference Weibull PDF and the measured PDF of generated samples for different values of β.

To test the statistical accuracy of this design, we generated some fading samples using the bit-true fixed-point model of our Nakagami-m fading simulator and compared the statistical properties of the generated samples against the theoretical references. To generate the Nakagami fading samples, we used isotropic Rayleigh fading samples with the Jakes' PSD (i.e., $\kappa = 0$, and $\tilde{\psi} = 0$ in equation (3.3)) and the Doppler frequency $f_D = 100$ Hz, and sample rate $F_s = 10$ KHz. To generate Nakagami-m samples, the approximation coefficients $\{a\}$, $\{b\}$, and $\{f\}$ need to be precomputed and stored in the corresponding memories for every m.

Figure 3.34 plots the PDF of the generated Nakagami fading samples for different valued of m. The theoretical references from equation (3.23) are plotted as well. As this figure shows, the PDF of the generated samples closely match the reference Nakagami-m PDF which verifies the statistical accuracy of the generated samples. Figure 3.36 compares the autocorrelation of the generated Nakagami-m fading samples and the reference values (from equation (3.5)) for different values of m. This figure also shows a close match between the measured autocorrelation and the reference values. We also generated some Weibull fading samples with our fading simulator. Figure 3.35 compares the PDF of the generated Weibull fading samples with the reference PDF given by

$$f_W(r) = \left(\frac{\beta}{\Pi}\right)\left(\frac{r}{\Pi}\right)^{\beta-1} \times \exp\left\{-\left(\frac{r}{\Pi}\right)^{\beta}\right\}, \quad r \geq 0, \tag{3.35}$$

for different values of β. In this simulation, the average power of the Weibull fading samples

was set to $\Pi = 1$. This figure also shows a close match between the generated PDFs and the reference values.

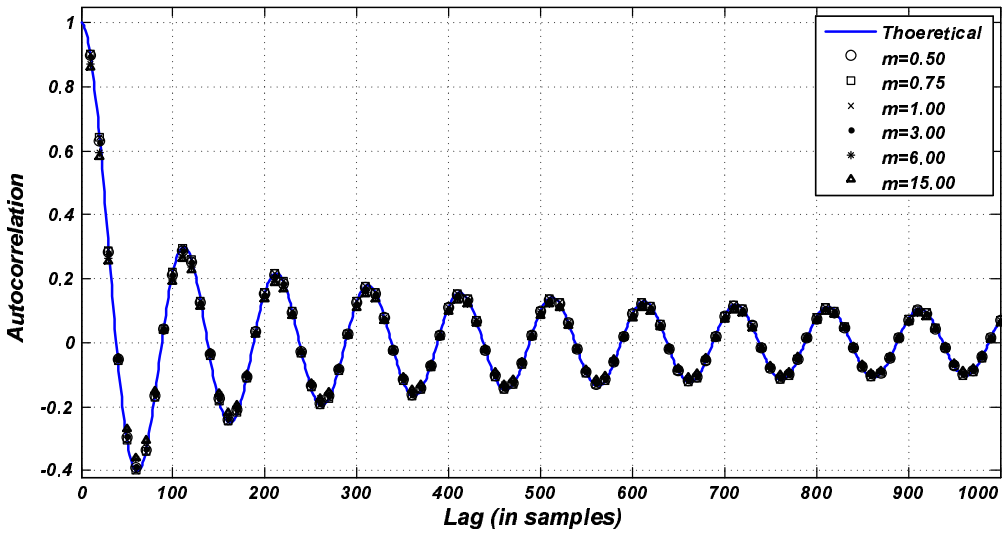


Figure 3.36: Comparison between reference autocorrelation function and that of the generated samples for different values of m.

The normalized autocorrelation function of the envelope of the generated Nakagami-m samples and the reference values are compared in Figure 3.37. It can be shown that the normalized autocorrelation function of the Nakagami-m signal envelope is given by [182, eq. 26]

$$f_{ACF}(\tau) = \frac{{}_2F_1(-\frac{1}{2}, -\frac{1}{2}; m, |J_0(2\pi f_D \tau)|^2)}{\left[\frac{\Gamma(m)}{\Gamma(m+1/2)}\right]^2 \left(\frac{m}{\Pi}\right)} \tag{3.36}$$

where ${}_2F_1(\cdot, \cdot; \cdot, \cdot)$ is the Gaussian hypergeometric function [183]. Figure 3.37 also shows a close match between the generated envelope autocorrelation and the expected values from equation (3.36) which further verifies the accuracy of our implementation.

We also implemented the datapath in Figure 3.32 on a Xilinx Virtex-II Pro XC2VP100-6ff1696 FPGA. To increase the throughput of our FPGA implementation, all of the components of this datapath are pipelined. More specifically, the blocks `U3` and `U9` in Figure 3.32 have 6- and 4-stage pipelines respectively. Our FPGA implementation of this datapath utilizes 652 configurable slices (1.5%), five on-chip 18×18 multipliers (1.1%), three 18 Kbit on-chip block memories (0.7%), and can operate at up to 246 MHz.

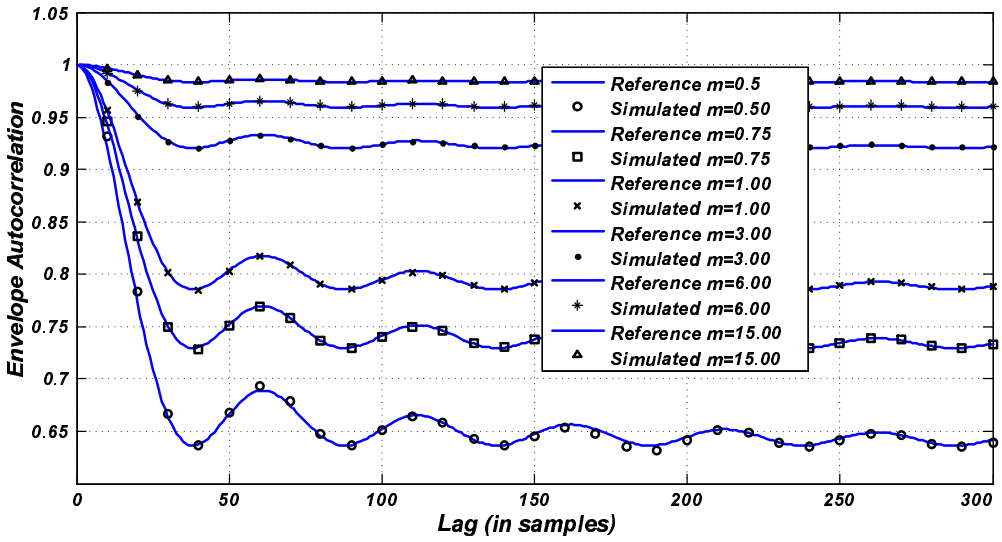


Figure 3.37: Comparison between reference normalized envelope autocorrelation function and that of the generated samples for different values of m.

3.5 Summary and Conclusions

Filter-based simulation can be used to generate fading samples with accurate statistical properties. For a given accuracy, although using IIR filters can be appealing due to their lower computational complexity (in terms of number of computational operations) compared to the FIR filters, the fixed-point implementation of IIR filters can be challenging due to finite word-length effects.

In this chapter we presented several novel computationally-efficient design and implementation schemes for filter-based simulation of isotropic and non-isotropic Rayleigh and Rician fading channels. We also presented a novel technique for designing stable fixed-point filters with real and complex coefficients. We further proposed two techniques for word-length reduction for implementing IIR filters. Using the new filter design technique, Rayleigh and Rician fading samples with arbitrary PSD properties can be simulated. In one example implementation of a non-isotropic Rayleigh fading channel, the proposed filter design technique resulted in a FPGA implementation that was 8.9 times smaller and 22.5 times faster than a previous design that was based on traditional filter design techniques.

We also proposed a computationally-efficient technique for simulating Nakagami-m and Weibull fading channels. The proposed technique was based on transforming Rayleigh fading samples into Nakagami-m and Weibull fading samples. We proposed a technique for the compact hardware implementation of this fading simulator based on a hybrid logarithmic-

linear segmentation of the transfer function. We also proposed a semi floating-point approximation technique which reduced the storage requirements significantly.

Chapter 4

MIMO Fading Channel Overview and Simulation

Multiple-input multiple-output (MIMO) communication systems can offer significant increases in spectral efficiency and link reliability by exploiting multipath propagation. MIMO technology has made significant advances in the past decade and has moved from a purely theoretical blueprint [184, 185] to real-life products (e.g., [186–189]).

Multipath propagation plays a key role in the link capacity increase and diversity gain in MIMO systems. The propagation conditions determine the channel capacity of a MIMO system and hence it is of great importance to characterize and model different MIMO propagation scenarios. Moreover, realistic simulation of signal propagation in MIMO channels is crucial for the design, accurate performance prediction, and verification of MIMO systems. Therefore it is important to have realistic and yet easy-to-use models to understand and reproduce the MIMO propagation effects [190]. Hence, the modeling and simulation of MIMO radio channels has attracted much attention in the literature.

This chapter reviews some of the most important MIMO channel models and presents efficient hardware simulation platforms for the corresponding models.

4.1 Background

In conventional wireless communication systems, one transmit and one receive antenna are used for signal transmission. On the other hand, MIMO systems are equipped with multiple antennas at both the transmit and receive ends. Here we consider an $M \times N$ MIMO system, where $M \geq 1$ and $N \geq 1$ are the number of transmit and receive antennas, respectively. Figure 4.1 illustrates this MIMO system.

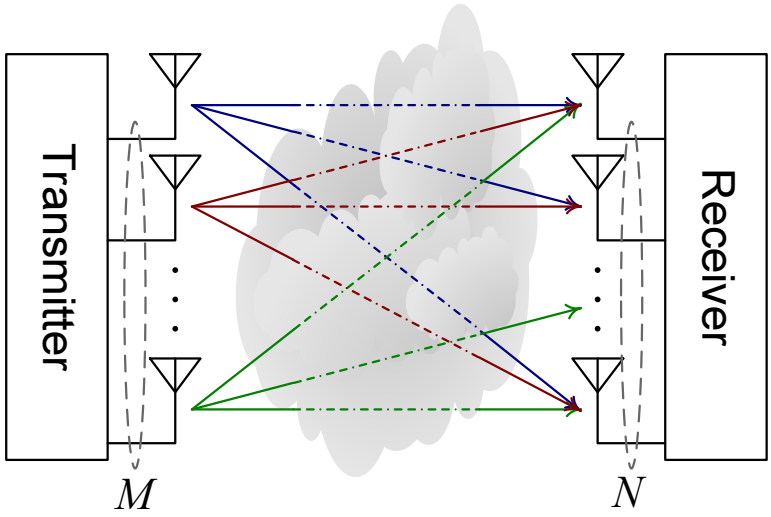


Figure 4.1: Illustration of a MIMO system with M transmit and N receive antenna elements.

The above MIMO channel representation can be used to formulate an overall baseband MIMO input-output relation between the length-M transmit signal vector $\mathbf{s}(t)$ and the length-N receive signal vector $\mathbf{y}(t)$ as

$$\mathbf{y}(t) = \int_{\tau} \mathbf{H}(t,\tau)\mathbf{s}(t-\tau)d\tau + \mathbf{n}(t), \tag{4.1}$$

where the length-N vector $\mathbf{n}(t)$ represents noise and interference. In the representation (4.1), it is assumed that the MIMO channel is linear and time-variant and can be represented by the $N \times M$ channel matrix

$$\mathbf{H}(t,\tau) = \begin{pmatrix} h_{11}(t,\tau) & h_{12}(t,\tau) & \dots & h_{1m}(t,\tau) \\ h_{21}(t,\tau) & h_{22}(t,\tau) & \dots & h_{2m}(t,\tau) \\ \vdots & \vdots & \ddots & \vdots \\ h_{n1}(t,\tau) & h_{n2}(t,\tau) & \dots & h_{nm}(t,\tau) \end{pmatrix}, \tag{4.2}$$

where $h_{ij}(t,\tau)$ denotes the time-variant impulse response between the j-th transmit antenna and the i-th receive antenna. Moreover, it is assumed that the channel matrix (4.2) includes the effects of antennas and frequency response. Note that there is no distinction between different antennas and various polarizations of the same antenna. To model MIMO systems with polarization-diverse antennas we can replace the elements of $\mathbf{H}(t,\tau)$ in (4.2) with polarimetric sub-matrices that describe the coupling between the vertical and horizontal polarizations [191].

If the channel is time-invariant, the channel matrix will be independent of time t and hence we have $\mathbf{H}(t,\tau) = \mathbf{H}(\tau)$. Further, if the signal bandwidth is rather small so that the

channel frequency response can be approximated as a frequency-flat channel, there would be just one single tap, i.e., $\mathbf{H}(\tau) = \mathbf{H}$. In this case (4.1) simplifies to

$$\mathbf{y}(t) = \mathbf{H}\mathbf{s}(t) + \mathbf{n}(t). \tag{4.3}$$

4.1.1 Model Classification

The most commonly used MIMO channel model in the literature is the narrow-band and spatially-independent and identically distributed (i.i.d.) Gaussian channel model. The i.i.d. channel model is an idealized assumption where the entries of the channel matrix are modeled as independent complex Gaussian random variables (see for example [184]). This model corresponds to a so-called "rich scattering" scenario. In this model it is assumed that there is an infinite number of randomly and uniformly located ideal scatterers, which form a uniform scattering medium. Moreover, in the the i.i.d. model the antenna elements are considered to be ideal field sensors with no size and no coupling between the elements in the transmit and receive antenna arrays [192]. For a single-user system with M transmit and N receive antennas, independent scattering from each transmit antenna to each receive antenna provides approximately $\min(M, N)$ separate channels and hence the capacity scales linearly with $\min(M, N)$.

Although the i.i.d. model is convenient for analytical studies, it is too idealized as it does not consider many propagation characteristics. Most propagation environments result in spatial and temporal correlations, which are ignored by the i.i.d. model. Also, modern wireless communication technology targets high data-rate applications over wide-band channels. Hence several sophisticated models for MIMO channels and propagation have been proposed.

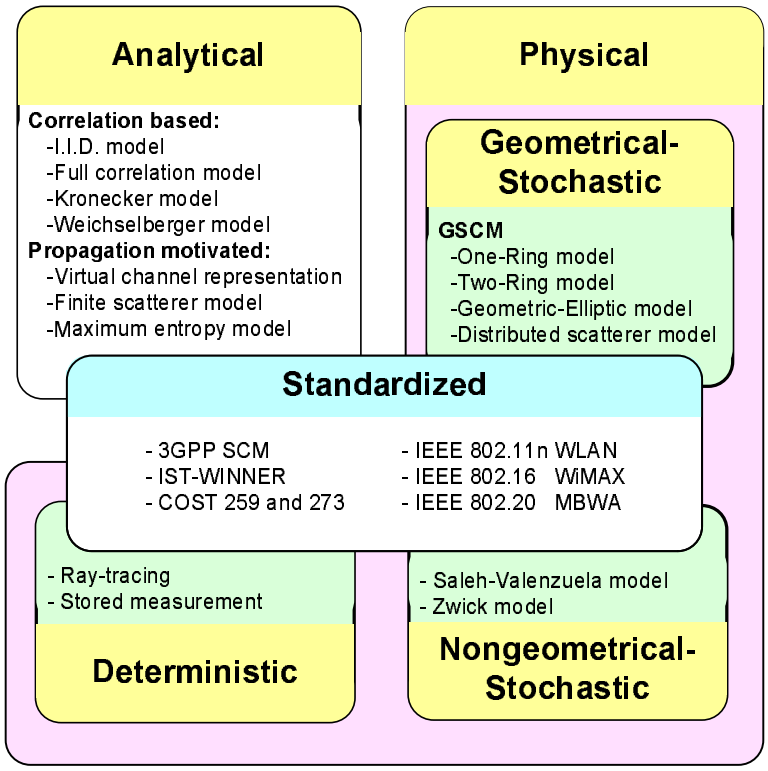


Figure 4.2: MIMO fading channel model classification.

Different MIMO channel models can be distinguished based on different criteria like time and frequency response or modeling approach. For example, considering the bandwidth of the system, a fading channel can be *frequency-flat* (narrow-band, or rather small delay spread) or *frequency-selective* (wide-band, or large delay spread). In the wide-band models (e.g., [193–199]) different frequency subbands can have different channel responses, while the narrow-band models (e.g., [200–207]) assume that the channel has the same response over the entire bandwidth.

Furthermore, fading channel models can be categorized based on the modeling approach. More specifically, a MIMO channel model could be *analytical* or *physical* [208, 209]. Analytical and physical channel models can also be subdivided into different categories based on their channel characterization approach. Model-based classification of some of the MIMO fading channel models is summarized in Figure 4.2.

Analytical channel models characterize the MIMO channel response in a mathematical/analytical fashion. In analytical models, physical aspects of wave propagation are not directly considered. Instead, the channel impulse response is expressed in terms of a complex matrix with a specific structure. Due to the mathematical convenience of analysis, these

models are popular in analytic studies of MIMO systems (see for example [200, 210–214]).

In contrast to analytical MIMO channel models, physical models characterize the fading channel based on the physical characteristics of wave propagation or actual measurements. A physical MIMO channel model considers the relative location of transmitter, receiver, and scatterers in the propagation media. More complex physical models can be used to accurately mimic the wave propagation effects. In the past decade, several physical MIMO channel modeling approaches have been proposed in the literature (see for example [204, 206, 215–225]).

Physical and analytical MIMO channel models can be further divided into sub-categories. Analytical models are either inspired by the correlation between channel samples or by the propagation mechanism in the fading channel. The i.i.d. model [184], full correlation model [200, 223], Kronecker model [200, 212–214], and the Weichselberger model [226], are among the analytical models that try to model a MIMO fading channel based on the correlation between fading samples $h_{ij}(t, \tau)$. The virtual channel representation (VCR) model [227, 228], finite scatterer model [210], and the maximum entropy model [229, 230], on the other hand, are propagation-motivated analytical models. Later in this chapter, we will explain some of these models in more detail.

Similarly, physical MIMO channel models can be divided into *deterministic models*, *nongeometrical-stochastic models*, and *geometrical-stochastic models*. In deterministic models the signal propagation is fully characterized by deterministic model parameters either from actual measurements using *channel sounders*, or ray-tracing through computer simulations. Deterministic models can provide site-specific but accurate channel characteristics that can be used for network planning. Examples of deterministic MIMO channel models can be found in [222, 231–235].

Nongeometrical-stochastic models on the other hand, characterize the fading channel using statistical parameters. In these models, the fading channel is described either as individual *multipath components* (MPCs) or as clusters of MPCs. More specifically, the Zwick model [236] treats each MPC component individually and independent of other MPCs. On the other hand, the Saleh-Valenzuela model [237] and the extended Saleh-Valenzuela model [215, 238] assume that MPCs are grouped in clusters. The idea of forming MPC clusters was inspired by the observation of temporal clusters in propagation delay [237] based on which Saleh and Valenzuela proposed a doubly-exponential decay process in radio signal power for indoor propagation. The proposed doubly-exponential decay process

is further integrated in other channel models including the TGn channel model [92] of IEEE 802.11.

In contrast to deterministic models, in which the location of scatterers is prescribed in a database, in a geometry-based stochastic channel model (GSCM) the location of scatterers is chosen according a specific distribution. GSCM channel models are based on the *double-directional representation* [239, 240] in which the propagation channel is represented as a number of propagation paths (or sub-paths) each characterized by a *direction-of-arrival* (DOA), a *direction-of-departure* (DOD) and a propagation time which is directly related to the signal attenuation. The GSCM model parameters can be chosen randomly from distributions that are derived from geometry or from measurements. Given the model parameters, the channel impulse response is then calculated using a simplified ray-tracing procedure.

GSCM has several advantages [241] over other channel models as it can reflect different real-life propagation effects. GSCM considers the propagation geometry and reproduces large-scale as well as small-scale fading effects by the superposition of incoming waves from individual scatterers. GSCM also includes the mobility effects between the transmitter and receiver and can easily be modified to simulate shadowing effects that happen on the propagation paths. Moreover, the power-delay profile (PDP) and the angular power spectrum (APS) are modeled conveniently for any distribution of scatterers.

The main advantage of GSCM is the possibility of simulating a large number of channels by changing channel parameters. The first GSCM model was proposed in 1973 in [242], where the scatterers are placed on a ring around the base-station and it is assumed that only single scattering occurs. Several GSCM models have been proposed in the literature since then [129, 225, 243–247]. The main differences between these models lie in the number of bounces allowed, in the trajectory of the sub-paths, and in the use of clusters.

Single-bounce scattering models assume that only one interacting object occurs between the transmitter and receiver. Different distributions have been proposed for the scatterer locations. The simplest model assumes that scatterers are distributed uniformly in space. This model is too simplistic and hence does not reflect the real propagation effects. In [225, 243] is it proposed to place the scatterers randomly around the mobile station. Further, in [248] a one-sided Gaussian distribution with respect to the distance from the mobile-station is considered.

The single scattering assumption makes the channel simulation convenient. In a single-bounce scattering model, all paths consist of two subpaths connecting the scatterer to the

transmitter and receiver respectively. A line-of-sight (LOS), or specular component may be present if the transmit and receive antennas can "see" each other. Every subpath is characterized by its direction-of-departure (DOD), direction-of-arrival (DOA), and the propagation time which in turn determines the subpath attenuation according to a power law.

The single-bounce assumption, however, limits the degree of freedom in channel modeling as the location of a scatterer completely determines the DOD, DOA, and delay. In many indoor and outdoor environments in which propagation involves multiple reflections and diffractions, multiple-bounce scattering can happen for which DOA, DOD and delay are completely decoupled [249–251].

Multiple-bounce scattering can be used for more accurate channel modeling. In multiple-bounce scattering more than one interacting object exists on the path between the transmitter and receiver, which can result in complete decoupling between DOA, DOD, and delay. Simulation of multiple-bounce scattering can be further simplified by incorporation of the *equivalent scatterers* concept [191]. Equivalent scatterers are basically substitute single-bounce scatterers that replace multiple-bounce path by mimicking their DOA and power. This concept is incorporated in the COST 259 channel model [191].

This method, however, is not effective for modeling MIMO channels as the DOD cannot be characterized with single-bounce virtual scatterers. In [216] it is proposed to use a double-bounce scattering approach for modeling MIMO channels.

In this thesis we focus primarily on simulating single- and double-bounce geometric models. More specifically, among all of the geometric models we target hardware implementation of the one-ring [206, 217, 218] , two-ring [219, 220], and the geometric elliptical model [221], as they provide clear insight into single- and double-bounce geometric MIMO fading channel and provide analytical background for verification of our hardware implementation. However, extension of this work to other physical models, particularly different GSCM models, is straightforward.

Moreover, we will present our hardware implementation of the most well-known analytical channel models. The implemented analytical channel models are the i.i.d. model [184], the Kronecker model [200, 212–214], the Weichselberger model [226], and the VCR model [227, 228]. The implemented fading simulator can also be easily extended to simulate the finite-scatterer model [210] and the maximum entropy model [229, 230]. We also discuss hardware simulation of of the TGn channel model for the IEEE 802.11 standard [92].

4.2 Physical Models: Single-Bounce Scattering

Single-bounce scattering models are a subset of the GSCM models in which it is assumed that the transmitted signal reaches the receiver either directly or after bouncing off one scatterer. Different distributions have been proposed for the scatterer on the way from the transmitter to the receiver [201, 206, 217, 218, 221, 225, 243, 248]. In this section we present the one-ring and the geometric elliptical channel models. We will present our hardware fading simulator for the single-bounce scattering models later in Section 4.4.

4.2.1 System Parameters

To present different geometric channel models we consider a $M \times N$ MIMO system with M and N omnidirectional antennas at the transmitter (or base-station) and the receiver (or mobile-station) respectively. Figure 4.3 illustrates the geometric representation of this system.

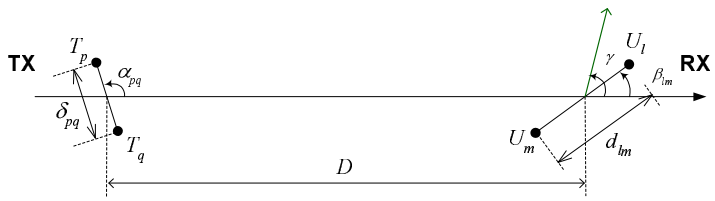


Figure 4.3: Geometrical representation of a $M \times N$ MIMO system.

In this figure, D is the distance between the transmitter and the receiver. In this model, the receiver is assumed to be at the center of the Cartesian coordinate system. The p^{th} and q^{th} TX antenna elements are denoted by T_p and T_q in Figure 4.3 and the l^{th} and m^{th} RX antenna elements are shown as U_l and U_m. Moreover, it is assumed that the antenna spacing in the transmitter for antenna elements p and q, $1 \leq p \leq q \leq M$, is δ_{pq}. Similarly, it is assumed that the antenna spacing in the receiver for antenna elements l and m, $1 \leq l \leq m \leq N$, is d_{lm}. The angle α_{pq} denotes the tilt angle between the p^{th} and q^{th} TX antennas. Similarly, the angle β_{lm} denotes the tilt angle between the l^{th} and m^{th} RX antennas. Further, it is assumed that the the transmitter and the local scatterers are fixed while the receiver is moving in the γ direction.

4.2.2 One-Ring Model

The one-ring model assumes that compared to the receiver, the transmitter is elevated and thus not obstructed by the local scatterers. On the other hand, the receiver is surrounded by local scatterers that are distributed over a ring. Figure 4.4 illustrates the geometric one-ring model. In this figure, R^{RX} is the radius of the ring of scatterers around the receiver, and S_i represents the i^{th} scatterer.

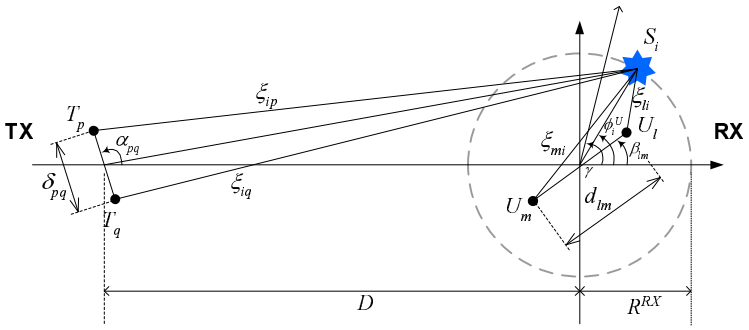


Figure 4.4: Geometrical representation of a MIMO channel with the one-ring model.

The visual representation of this model in Figure 4.4 can be used for deriving the channel gains between transmit and receive antennas. In the presence of a LOS component, the channel gain $h_{lp}^{1R}(t)$ between the p^{th} transmit (TX) antenna and the l^{th} receive (RX) antenna is

$$h_{lp}^{1R}(t) = h_{lp}^{DIF-1R}(t) + h_{lp}^{LOS}(t), \tag{4.4}$$

where $h_{lp}^{DIF-1R}(t)$ represents the contributions of the diffuse components (scattering) and $h_{lp}^{LOS}(t)$ denotes the contribution of the LOS component. Figures 4.4 and 4.5 illustrate the propagation paths for the diffuse and the LOS components, respectively. Let us assume that the total transmitted power through the p^{th} TX antenna and the l^{th} RX antenna link is Ω_{lp}. Using the model representation of Figure 4.4, the contribution of the diffuse component to the channel gain is [206]

$$h_{lp}^{DIF-1R}(t) = \sqrt{\frac{\Omega_{lp}}{K_{lp}+1}} \frac{1}{\sqrt{N_S}} \sum_{i=1}^{N_S} g_i \upsilon_{lpi}(\phi_i^U) \times \exp\{j(2\pi f_i t + \theta_i)\}, \tag{4.5}$$

where

$$\upsilon_{lpi}(\phi_i^U) = \exp\{-\frac{j2\pi}{\lambda}(\xi_{ip}(\phi_i^U) + \xi_{li}(\phi_i^U))\}, \tag{4.6}$$

and

$$f_i = \cos(\phi_i^U - \gamma) \times f_D. \tag{4.7}$$

Moreover, using the representation of Figure 4.5, the contribution of the LOS component to the channel can be written as [206]

$$h_{lp}^{LOS}(t) = \sqrt{\frac{\Omega_{lp} K_{lp}}{K_{lp}+1}} \times \exp\{j(2\pi \cos(\omega_p - \gamma) f_D t - \frac{2\pi \zeta_{lp}}{\lambda})\}. \tag{4.8}$$

Parameters used in the above equations are described here. In (4.5) and (4.8), K_{lp} denotes the Rice factor between the p^{th} TX antenna and the l^{th} RX antenna link, which is the ratio of the LOS component power to the diffuse component power, i.e.,

$$K_{lp} = \frac{\left| h_{lp}^{LOS}(t) \right|^2}{E\left\{ \left| h_{lp}^{DIF-1R}(t) \right|^2 \right\}}, \tag{4.9}$$

where $E\{\cdot\}$ denotes the expectation operator. Moreover, N_S is the number of independent scatterers, g_i is the amplitude of the received wave from the i^{th} scatterer such that $\sum_{i=1}^{N_S} E\{g_i^2\} = N_S$ for $N_S \to \infty$. Also, θ_i denotes the phase shift introduced by the i^{th} scatterer, ξ_{ip} and ξ_{li} are the distances of the i^{th} scatterer from the p^{th} TX antenna and the l^{th} RX antenna respectively as shown in Figure 4.4. Based on this geometric model, the distances ξ_{ip} and ξ_{li} are functions of ϕ_i^U, the direction of arrival (DOA) of the wave traveling from the i^{th} scatterer toward the user, λ is the wavelength, $j^2 = -1$, and f_D is the maximum Doppler frequency. Moreover, in (4.8) ζ_{lp} denotes the distance between the p^{th} TX antenna and the l^{th} RX antenna and ω_p is the approximate direction-of-arrival of the LOS path as shown in Figure 4.5. The set $\{g_i\}_{i=1}^{\infty}$ consists of independent and positive random variables with finite variances, independent of $\{\theta_i\}_{i=1}^{\infty}$. Moreover, it assumed that $\{\theta_i\}_{i=1}^{\infty}$ are independent and identically-distributed (i.i.d.) random variables uniformly distributed over $[-\pi, \pi)$.

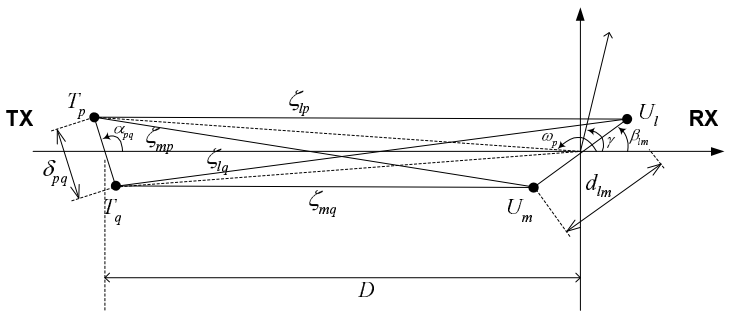


Figure 4.5: The LOS path in the MIMO channel.

4.2.2.1 Correlation Properties

Based on the above model, the space-time correlation properties between different path gains can be derived. More specifically, we are interested in the cross-correlation between $h_{lp}^{1R}(t)$ and $h_{mq}^{1R}(t)$. This cross correlation is defined as

$$\rho_{lp,mq}^{1R}(t,\tau) = \frac{E\left\{h_{lp}^{1R}(t)h_{mq}^{1R*}(t+\tau)\right\}}{\sqrt{\Omega_{lp}\Omega_{mq}}}, \tag{4.10}$$

where $(\cdot)^*$ denotes the complex conjugate operation. Since $h_{lp}^{DIF-1R}(t)$ and $h_{mq}^{DIF-1R}(t)$ are zero-mean stochastic processes (according to (4.5), they are superpositions of zero-mean complex sinusoidal stochastic processes), it can be verified that the cross-correlation $\rho_{lp,mq}^{1R}(t,\tau)$ can be decomposed into two parts, 1) the cross-correlation between the contributions of the diffuse components and 2) the cross-correlation between the contributions of the LOS components, i.e.,

$$\rho_{lp,mq}^{1R}(t,\tau) = \rho_{lp,mq}^{DIF-1R}(t,\tau) + \rho_{lp,mq}^{LOS}(t,\tau), \tag{4.11}$$

where

$$\rho_{lp,mq}^{DIF-1R}(t,\tau) = \frac{E\left\{h_{lp}^{DIF-1R}(t)h_{mq}^{DIF-1R*}(t+\tau)\right\}}{\sqrt{\Omega_{lp}\Omega_{mq}}}, \tag{4.12}$$

and

$$\rho_{lp,mq}^{LOS}(t,\tau) = \frac{E\left\{h_{lp}^{LOS}(t)h_{mq}^{LOS*}(t+\tau)\right\}}{\sqrt{\Omega_{lp}\Omega_{mq}}}. \tag{4.13}$$

The space-time cross-correlation functions can be calculated based on the geometric representation of the one-ring model. Moreover, accurate closed-form approximations for these functions can be found if the transmitter and the receiver are widely spaced, and the antenna spacing is much smaller than the distance from scatterers, i.e., $D \gg R^{RX} \gg \max(\delta_{pq}, d_{lm})$. These assumptions are supported by different measurements conducted in different locations and frequencies (see for example [17, 224, 242, 252–255]). Based on these assumptions, the approximate cross-correlation between h_{lp}^{DIF-1R} and h_{mq}^{DIF-1R} can be written as [206]

$$\begin{aligned}\rho_{lp,mq}^{DIF-1R}(t,\tau) &= \rho_{lp,mq}^{DIF-1R}(\tau) \\ &\approx \frac{1}{\sqrt{(K_{lp}+1)(K_{mq}+1)}}\int_{-\pi}^{+\pi} \upsilon_{lp}(\phi^U)\upsilon_{mq}^*(\phi^U) \\ &\quad \times \exp\{-j2\pi f_D \cos(\phi^U-\gamma)\tau\} f(\phi^U)d\phi^U. \end{aligned} \tag{4.14}$$

To derive (4.14) it is assumed that the number N_S of scatterers is large enough that $E\{g_i^2\}/N_S$ can be approximated as $E\{g_i^2\}/N_S \approx f(\phi_i^U)d\phi^U$, where $f(\phi^U)$ is the probability density function (PDF) of the AOA seen by the receiver. Moreover, the total distance from the p^{th} TX antenna to the l^{th} RX antenna in (4.6) has been expressed as a function of the angle of arrival ϕ^U in (4.14). Similarly, the total distance from the q^{th} TX antenna to the m^{th} RX antenna has been expressed as a function of the angle of arrival ϕ^U.

The cross-correlation expression in (4.14) holds for any distribution of AOA. The von Mises/Tikhonov distribution [133, 134] has been used in the literature to model the AOA (see for example [91, 206, 221, 256–258]). Also it has been argued that the von Mises PDF is favorable because it can approximate other non-uniform PDFs and can provide mathematical convenience for analysis [91]. Moreover, for the empirical justifications of the von Mises distribution for the AOA the reader may refer to [91, 259]. The PDF of the von Mises distribution is given by [133]

$$f(\phi^U) = \frac{\exp\{\kappa\cos(\phi^U - \mu)\}}{2\pi I_0(\kappa)}, \quad \phi^U, \mu \in [-\pi, \pi), \tag{4.15}$$

where $I_0(\cdot)$ denotes the zeroth-order modified Bessel function of the first kind, $\mu \in [-\pi, \pi)$ is the mean direction of AOA and the parameter $\kappa \geq 0$ controls the beam-width [91]. For $\kappa = 0$ the von Mises PDF becomes a uniform distribution over $[-\pi, \pi)$ while for $\kappa = \infty$ the PDF becomes a Dirac delta function $\delta(\phi^U - \mu)$.

Using the von Mises distribution for AOA the cross-correlation for the diffuse component can be approximated as [206]

$$\begin{aligned}\rho_{lp,mq}^{DIF-1R}(\tau) \approx \frac{\exp\{jc_{pq}\cos(\alpha_{pq})\}}{\sqrt{(K_{lp}+1)(K_{mq}+1)}} \times I_0\Big(\Big\{&\kappa^2 - a^2 - b_{lm}^2 - c_{pq}^2\Delta^2\sin^2(\alpha_{pq}) \\ &+2ab_{lm}\cos(\beta_{lm}-\gamma) + 2c_{pq}\Delta\sin(\alpha_{pq}) \times [a\sin(\gamma) - b_{lm}\sin(\beta_{lm})] \\ &-j2\kappa[a\cos(\mu-\gamma) - b_{lm}\cos(\mu-\beta_{lm}) \\ &-c_{pq}\Delta\sin(\alpha_{pq})\sin(\mu)]\Big\}^{\frac{1}{2}}\Big)/I_0(\kappa),\end{aligned} \tag{4.16}$$

where $a = 2\pi f_D\tau$, $b_{lm} = 2\pi d_{lm}/\lambda$, $c_{pq} = 2\pi\delta_{pq}/\lambda$, and $\Delta = \tan^{-1}(R^{RX}/D)$. Moreover in (4.16), α_{pq} denotes the angle of the p^{th} and q^{th} TX antenna pair. Similarly, β_{lm} is the angle of the l^{th} and m^{th} RX antenna pair as shown in Figure 4.4.

Moreover, assuming that the distance D between the transmitter and the receiver is much larger than antenna spacing at the transmitter and receiver, i.e., $D \gg \max(\delta_{pq}, d_{lm})$, the cross-correlation for the LOS component can be approximated as [206]

$$\rho^{LOS}_{lp,mq}(\tau) \approx \sqrt{\frac{K_{lp}K_{mq}}{(K_{lp}+1)(K_{mq}+1)}} \times \exp\{ja\cos(\gamma) - jb_{lm}\cos(\beta_{lm}) + jc_{pq}\cos(\alpha_{pq})\}. \quad (4.17)$$

It is important to note that the one-ring model reduces to the non-isotropic Rayleigh fading model [91] when single antenna elements are deployed at both transmitter and receiver, i.e. when $M = N = 1$ and $d_{11} = \delta_{11} = 0$, and no line of sight is present, i.e., $K_{11} = 0$. In this case, the autocorrelation between fading samples $\{h_{11}(t)\}$ is

$$\rho(\tau) = \frac{I_0\left(\sqrt{\kappa^2 - 4\pi^2 f_D^2\tau^2 + j4\pi\kappa\cos(\mu)f_D\tau}\right)}{I_0(\kappa)}. \quad (4.18)$$

When scatterers are uniformly distributed around the receiver, i.e., $\kappa = 0$, the correlation model in (4.18) reduces to Clark's temporal correlation $\mathcal{J}_0(2\pi f_D\tau)$ [17] where $\mathcal{J}_0(\cdot)$ denotes the zeroth-order Bessel function of the first kind.

4.2.3 Geometric Elliptical Model

In the one-ring and the two-ring channel models it is assumed that the difference between the propagation delays from from each TX antenna to each RX antenna is negligible. This approximation can be useful when modeling narrow-band channels. However, for modeling wide-band channels, the difference between propagation delays should be considered.

Unlike the one-ring and the two-ring models, the geometric elliptical model can be used for simulating wide-band channels as well as narrow-band channels. In the elliptical model, each ellipsoid represents the scatterers that result in a specific propagation delay. Frequency-selective wide-band fading channels can be modeled by multiple ellipsoids of scatterers, each representing a specific propagation delay.

The geometric elliptical model was first proposed in [129] for the micro- and pico-cell environments with low mount TX and RX antennas. In this model it is assumed that multi-path scattering can happen anywhere around the TX and RX. The geometric elliptical model was further extended to single-input multiple-output (SIMO) channels in [260]. A wide-band geometric elliptical model for wide-band MIMO channels was proposed in [221]. Another geometric elliptical model has been reported in [261].

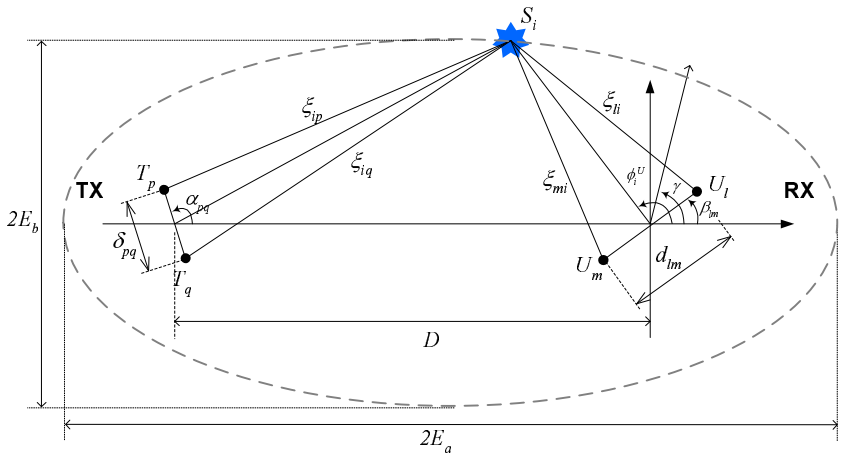


Figure 4.6: Geometric elliptical scattering model for an $M \times N$ MIMO channel with local scatterers S_i lying on an ellipse.

Figure 4.6 illustrates the geometric elliptical scattering model. This figure illustrates that all local scatterers S_i, $i = 1, ..., N_S$, are located on an ellipse, where the transmitter and the receiver are located at the focal points. This ellipse can be associated with a certain path length (or propagation delay) between the transmitter and the receiver. The major axis half length and minor axis half length are denoted by E_a and E_b, respectively. The distance between the two focal points is $D = 2(E_a^2 - E_b^2)^{1/2}$ which is the distance between the transmitter and the receiver.

Similar to the one-ring model, the channel gain $h_{lp}^E(t)$ between the p^{th} TX antenna and the l^{th} RX antenna can be decomposed into the contributions of the diffuse components, $h_{lp}^{DIF-E}(t)$, and the contribution of the LOS component $h_{lp}^{LOS}(t)$ (see equation (4.4)). For the wireless link between the p^{th} TX antenna and the l^{th} RX antenna we will denote the Rice factor and the total transmitted power by K_{lp} and Ω_{lp}, respectively. The LOS path between the TX and RX antennas is similar to that shown in Figure 4.5, hence providing the same equation for the LOS component (equation (4.8)).

Figure 4.6 shows the propagation path for the diffuse component (i^{th} subpath) in the geometric elliptical channel model. The diffuse component of channel gains between the TX and the RX antennas can be derived from the visual presentation in Figure 4.6. Comparing the geometric elliptical model in Figure 4.6 with the one-ring model in Figure 4.4, we can expect the same equation for the contribution of the diffuse component in equation (4.5)

i.e.,

$$h_{lp}^{DIF-E}(t) = \sqrt{\frac{\Omega_{lp}}{K_{lp}+1}} \frac{1}{\sqrt{N_S}} \sum_{i=1}^{N_S} g_i \upsilon_{lpi}(\phi_i^U) \times \exp\{j(2\pi f_i t + \theta_i)\}. \tag{4.19}$$

Both the one-ring and the geometric elliptical models are single-bounce channel models and therefore can be represented with the same equation. The only difference between these two models is in the distribution of scatterers.

For the one-ring model, the scatterers are distributed on a ring around the receiver and hence the location of the scatterers in the Cartesian coordinate system can be expressed in terms of the angle-of-arrival ϕ^U as

$$\begin{array}{l} S^{X-1R}(\phi^U) = R^{RX} \times \cos(\phi^U), \\ S^{Y-1R}(\phi^U) = R^{RX} \times \sin(\phi^U). \end{array} \tag{4.20}$$

Finally, the location of scatterers for the geometric elliptical model can be calculated as

$$\begin{array}{l} S^{X-E}(\phi^U) = R^E(\phi^U) \times \cos(\phi^U), \\ S^{Y-E}(\phi^U) = R^E(\phi^U) \times \sin(\phi^U), \end{array} \tag{4.21}$$

where

$$R^E(\phi^U) = \frac{4E_a^2 - D^2}{4E_a + 2D \times \cos(\phi^U)}. \tag{4.22}$$

4.2.3.1 Correlation Properties

Similar to the one-ring model, since $h_{lp}^{DIF-E}(t)$ and $h_{mq}^{DIF-E}(t)$ are zero-mean stochastic processes, the channel gain cross-correlation can be decomposed into two parts

$$\rho_{lp,mq}^{E}(t,\tau) = \rho_{lp,mq}^{DIF-E}(t,\tau) + \rho_{lp,mq}^{LOS}(t,\tau), \tag{4.23}$$

where

$$\rho_{lp,mq}^{DIF-E}(t,\tau) = \frac{E\left\{h_{lp}^{DIF-E}(t) h_{mq}^{DIF-E^*}(t+\tau)\right\}}{\sqrt{\Omega_{lp}\Omega_{mq}}}, \tag{4.24}$$

and $\rho_{lp,mq}^{LOS}(t,\tau)$ is the cross-correlation between the contributions of the LOS components from equation (4.17). However, to the best of my knowledge, no closed-form equation has been found for the general case of cross-correlation between the contributions of the diffuse components $\rho_{lp,mq}^{DIF-E}(t,\tau)$ for the geometric elliptical channel model. Assuming that $\max_{p,q} \delta_{pq} \ll E_a - D/2$ and $\max_{l,m} d_{lm} \ll E_a - D/2$, the cross-correlation between the contribution of the diffuse components can be approximated as

$$\begin{aligned} \rho_{lp,mq}^{DIF-E}(t,\tau) &= \rho_{lp,mq}^{DIF-E}(\tau) \\ &\approx \frac{1}{\sqrt{(K_{lp}+1)(K_{mq}+1)}} \int_{-\pi}^{+\pi} \upsilon_{lp}^{E}(\phi^U) \upsilon_{mq}^{E\ *}(\phi^U) \\ &\quad \times \exp\{-j2\pi f_D \cos(\phi^U - \gamma)\tau\} f(\phi^U) d\phi^U, \end{aligned} \tag{4.25}$$

where

$$\upsilon_{lp}^{E}(\phi^{U}) = \exp\left\{-\frac{j2\pi(\xi_{p}^{E}(\phi^{U})+\xi_{l}^{E}(\phi^{U}))}{\lambda}\right\}, \tag{4.26}$$

$$\xi_{p}^{E}(\phi^{U}) = \sqrt{(T_{p}^{X}-S^{X-E}(\phi^{U}))^{2}+(T_{p}^{Y}-S^{Y-E}(\phi^{U}))^{2}}, \tag{4.27}$$

$$\xi_{l}^{E}(\phi^{U}) = \sqrt{(U_{l}^{X}-S^{X-E}(\phi^{U}))^{2}+(U_{l}^{Y}-S^{Y-E}(\phi^{U}))^{2}}. \tag{4.28}$$

For a single-antenna system (i.e., $M = N = 1$), if the AOA is distributed with the von Mises PDF, it can be verified that the autocorrelation between fading samples is given by equation (4.18). Similar to the one-ring model with single TX and single RX antennas, when the AOA is uniformly distributed over $[-\pi, \pi)$, the autocorrelation for the geometric elliptical model reduces to Clark's temporal correlation $\mathcal{J}_0(2\pi f_D \tau)$ [17].

4.3 Physical Models: Multiple-Bounce Scattering

Multiple-bounce scattering models are another subset of the GSCM models in which it is assumed that the transmitted signal bounces off multiple objects (scatterers) before reaching the receiver.

Multiple-bounce scattering models can be more accurate than single-bounce models since they characterize the reality of multipath propagation effects more accurately. One of the effects in MIMO channels is the so-called "keyhole-" or "pinhole-effect" in which the actual channel capacity is much less than the anticipated capacity even though the received signals at the antenna elements are uncorrelated [201, 262–264]. It happens when the scattering around the TX and RX is such that each scatterer at the RX "sees" the TX scatterers effectively as the same point source.

Double-bounce scattering models can be used for the convenient simulation of multiple-bounce scattering. A double-bounce scattering approach is proposed in [216] for the modeling of MIMO channels. The double-bounce scattering can provide complete decoupling between the DOA, DOD, and propagation delay. Further it can be used for modeling different MIMO channel scenarios including the keyhole-effect [216].

In this work we focus on simulating the two-ring MIMO channel model which has been used by several authors to characterize local scattering at both the TX and RX sides [201, 219, 220, 225, 258, 265–267]. Different variations of the two-ring model have been proposed. The main difference between these models is in single- or double-scattering, in the distribution of local scatterers, and in the modeling of relative movement between

the transmitter, the receiver, and the local scatterers. In this work we consider the general case of double-bounce two-ring scattering where the transmitter and the local scatterers around the transmitter and receiver are immobile while the receiver is moving compared to the rest of the system. Other scattering channel models can be simulated with a similar methodology.

In this section we present the two-ring channel model. In Section 4.4 we will discuss hardware simulation of this channel model.

4.3.1 Two-Ring Model

The two-ring model assumes that both the transmitter and the receiver are surrounded by scatterers. This can be the case for indoor wireless communications or in outdoor scenarios where the TX and RX antennas are not mounted high enough not to be obstructed by local scatterers. An illustration of the two-ring double-bounce model is shown in Figure 4.7. Note that in this model, each ray is reflected twice.

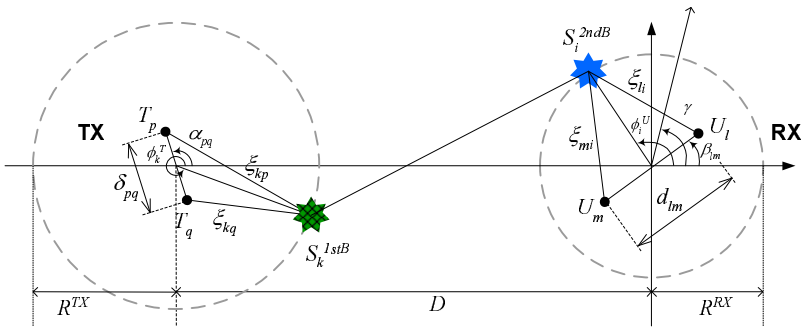


Figure 4.7: Geometrical representation of a MIMO channel with the two-ring model.

Here we assume that N_{S1} TX side (or first-bounce) scatterers are distributed on a circle of radius R^{TX} around the transmitter. Similarly, N_{S2} RX side (or second-bounce) scatterers are distributed on a ring of radius R^{RX} around the receiver. Based on the model presented in Figure 4.7, the diffuse component of the channel gain can be written as

$$h_{lp}^{DIF-2R}(t) = \sqrt{\frac{\Omega_{lp}}{K_{lp}+1}} \frac{1}{\sqrt{N_{S1}N_{S2}}} \sum_{k=1}^{N_{S1}} \sum_{i=1}^{N_{S2}} g_{ik} \upsilon_{lpik}(\phi_i^U) \times \exp\{j(2\pi f_{ik} t + \theta_{ik})\}, \tag{4.29}$$

where

$$\upsilon_{lpik}(\phi_i^U) = \exp\{-\frac{j2\pi}{\lambda}(\xi_{kp}(\phi_i^U) + \xi_{ik}(\phi_i^U) + \xi_{li}(\phi_i^U))\}, \tag{4.30}$$

and

$$f_{ik} = \cos(\phi_i^U - \gamma) \times f_D. \tag{4.31}$$

Similar to the previous models, here K_{lp} and Ω_{lp} denote the Rice factor and the total transmitted power over the wireless link between the p^{th} TX antenna and the l^{th} RX antenna, respectively. In equation (4.29) g_{ik} denotes the amplitude of the received wave through the k^{th} first-bounce and i^{th} second-bounce scatterer and we have $\sum_{k=1}^{N_{S1}} \sum_{i=1}^{N_{S2}} g_{ik}^2 = N_{S1}N_{S2}$. Similar to the single-bounce case, θ_{ik} is the random phase shift introduced jointly by the k^{th} first-bounce and i^{th} second-bounce scatterer pair. Moreover, ξ_{kp} denotes the distance from the p^{th} TX antenna to the k^{th} first-bounce scatterer, ξ_{ik} is the distance between the k^{th} first-bounce and the i^{th} second-bounce scatterers, and ξ_{li} denotes the distance from the i^{th} second-bounce scatterer to the l^{th} RX antenna.

When the transmitter is not moving compared to the scatterers, the observed Doppler frequency from the incoming wave through the k^{th} first-bounce and the i^{th} second-bounce scatterers is only a function of the AOA of this subpath ϕ_i^U (i.e., from the second-bounce scatterer) as shown in equation (4.31). However, if the transmitter is moving, the observed Doppler frequency from this subpath can be written as [265, 266]

$$f_{ik} = \cos(\phi_k^T - \gamma^{TX}) \times f_D^{TX} + \cos(\phi_i^U - \gamma^{RX}) \times f_D^{RX}, \tag{4.32}$$

where ϕ_k^T is the AOD from the transmitter to the k^{th} first-bounce scatterer, γ^{TX}, and γ^{RX} denote the directions of movement for the transmitter an receiver, and f_D^{TX} and f_D^{RX} denote the maximum Doppler frequencies at the TX and RX sides. Assuming mobility of the transmitter can be useful for simulating mobile-to-mobile channels. However, without loss of generality, in the rest of this work we assume that the transmitter is immobile.

Moreover, the LOS path between the p^{th} TX antenna and the l^{th} RX antenna is similar to that shown in Figure 4.5, hence providing the same equation for the LOS component (equation (4.8)).

4.3.1.1 Correlation Properties

Like the one-ring model, the cross-correlation between the channel gains $h_{lp}^{DIF-2R}(t)$ and $h_{mq}^{DIF-2R}(t)$ can be decomposed into two parts

$$\rho_{lp,mq}^{2R}(t,\tau) = \rho_{lp,mq}^{DIF-2R}(t,\tau) + \rho_{lp,mq}^{LOS}(t,\tau), \tag{4.33}$$

where

$$\rho_{lp,mq}^{DIF-2R}(t,\tau) = \frac{E\left\{h_{lp}^{DIF-2R}(t)h_{mq}^{DIF-2R^*}(t+\tau)\right\}}{\sqrt{\Omega_{lp}\Omega_{mq}}}, \tag{4.34}$$

and $\rho_{lp,mq}^{LOS}(t,\tau)$ is given by equation (4.17). As before, K_{lp} and Ω_{lp} are the Rice factor and the total transmitted power over the wireless link between the p^{th} TX antenna and the l^{th} RX antenna. Based on equations (4.29), (4.30), and (4.34) we can write

$$\begin{aligned}\rho_{lp,mq}^{DIF-2R}(t,\tau) &= \rho_{lp,mq}^{DIF-2R}(\tau) \\ &= \frac{1}{N_{S1}N_{S2}(K_{lp}+1)(K_{mq}+1)}\sum_{k=1}^{N_{S1}}\sum_{i=1}^{N_{S2}}E\{g_{ik}^2\}\times \\ &\quad \exp\Big\{-j\frac{2\pi}{\lambda}(\xi_{kp}+\xi_{li}-\xi_{kq}-\xi_{mi}) \\ &\quad -j2\pi f_{ik}\tau\Big\}.\end{aligned} \tag{4.35}$$

Assuming an infinite number of scatterers around the transmitter and receiver, i.e., $(N_{S1}\rightarrow\infty)$ and $N_{S2}\rightarrow\infty)$, $E\{g_{ik}^2\}/(N_{S1}N_{S2})$ can be approximated as $E\{g_{ik}^2\}/(N_{S1}N_{S2}) \approx f(\phi^T)f(\phi^U)d\phi^T d\phi^U$ where $d\phi^T$ and $d\phi^U$ are the AOD and AOA, and $f(\phi^T)$ and $f(\phi^U)$ denote the PDFs of AOA and AOD. Based on this assumption, the summations in equation (4.35) can be expressed in integral form as

$$\begin{aligned}\rho_{lp,mq}^{DIF-2R}(\tau) &= \frac{1}{(K_{lp}+1)(K_{mq}+1)}\int_{-\pi}^{\pi}\int_{-\pi}^{\pi}\exp\Big\{-j\frac{2\pi}{\lambda}\big(\xi_p(\phi^T) \\ &\quad +\xi_i(\phi^U)-\xi_q(\phi^T)-\xi_m(\phi^U)\big) \\ &\quad -j2\pi\cos(\phi^U-\gamma)f_D\tau\Big\}f(\phi^T)f(\phi^U)d\phi^T d\phi^U.\end{aligned} \tag{4.36}$$

The cross-correlation expression in (4.36) can be calculated for different AOA and AOD distributions. Here we use the von Mises/Tikhonov PDF [133, 134] for the AOA and the AOD distributions. More specifically, we consider

$$f(\phi^T) = \frac{\exp\{\kappa_T\cos(\phi^T-\mu_T)\}}{2\pi I_0(\kappa_T)}, \quad \phi^T,\mu_T\in[-\pi,\pi), \tag{4.37}$$

and

$$f(\phi^U) = \frac{\exp\{\kappa_U\cos(\phi^U-\mu_U)\}}{2\pi I_0(\kappa_U)}, \quad \phi^U,\mu_U\in[-\pi,\pi), \tag{4.38}$$

as the PDFs of the AOD and AOA respectively. In (4.37), μ_T is the mean AOD and κ_T controls the AOD beam-width. Similarly, in (4.38) μ_U is the mean AOA and κ_U is the AOA beam-width control parameter. Note that for $\kappa_T = 0$ ($\kappa_U = 0$) the von Mises distribution

reduces to the uniform distribution over $[-\pi, \pi)$. Finally, when $\kappa_T \to \infty$ ($\kappa_U \to \infty$), the von Mises distribution becomes the Dirac delta function at μ_T (μ_U).

Assuming that $\delta_{pq} \ll R^{TX}$ and $d_{lm} \ll R^{RX}$ the closed-form expression for equation (4.36) is given by [219]

$$\rho_{lp,mq}^{DIF-2R}(\tau) = \frac{\rho_{lp,mq}^{DIF-2R-T}(\tau) \times \rho_{lp,mq}^{DIF-2R-U}(\tau)}{(K_{lp}+1)(K_{mq}+1)}, \tag{4.39}$$

where

$$\rho_{lp,mq}^{DIF-2R-T}(\tau) = \frac{I_0\left(\sqrt{\kappa_T^2 - c_{pq}^2 + 2j\kappa_T c_{pq}\cos(\alpha_{pq}-\mu_T)}\right)}{I_0(\kappa_T)}, \tag{4.40}$$

and

$$\begin{aligned}\rho_{lp,mq}^{DIF-2R-R}(\tau) \;=\; & \frac{1}{I_0(\kappa_U)} \times I_0\Big(\big[\kappa_U^2 - a^2 - b_{lm}^2 + 2ab_{lm}\cos(\beta_{lm}-\gamma) \\ & +2j\kappa_U b_{lm}\cos(\beta_{lm}-\mu_U) \\ & -2j\kappa_U\cos(\gamma-\mu_U)\big]^{\frac{1}{2}}\Big),\end{aligned} \tag{4.41}$$

where $a = 2\pi f_D \tau$, $b_{lm} = 2\pi d_{lm}/\lambda$, and $c_{pq} = 2\pi\delta_{pq}/\lambda$. Notice that the cross-correlation expression in (4.39) is separable into TX correlation (4.40) and RX correlation (4.41) parts. This property has been considered in several articles [201, 212, 223, 263, 268].

4.4 Hardware Simulation of Geometric Models

Let us start with the hardware implementation of one-ring and geometric elliptical single-scatterer channel models. In discrete time (i.e., $t = nT_s$ where T_s is the sample period and n is a non-negative integer) the LOS component of the channel gain between the p^{th} TX antenna and the l^{th} RX antenna is (see (4.8))

$$h_{lp}^{LOS}[n] = \sqrt{\frac{K_{lp}}{K_{lp}+1}} \times \exp\{j(2\pi\cos(\omega_p-\gamma)f_D n - \frac{2\pi\zeta_{lp}}{\lambda})\}, \tag{4.42}$$

and the contribution of the diffuse components in the same link can be written as (see (4.5) and (4.19))

$$h_{lp}^{DIF}[n] = \frac{1}{\sqrt{N_S}}\sum_{i=1}^{N_S}\frac{g_i\exp\{j(2\pi f_i n + \upsilon_{lpi} + \theta_i)\}}{\sqrt{K_{lp}+1}}. \tag{4.43}$$

Hardware implementation of a fading simulator that generates samples according to equations (4.42) and (4.43) is straightforward. The fading process between each antenna pair (here the p^{th} TX antenna and the l^{th} RX antenna) is basically the superposition of a

number of complex sinusoidal oscillators with random, but stored, phases. These oscillators can be implemented separately, or with a shared datapath. Considering the limited range of the maximum Doppler frequency, it is more efficient if the fading samples would be generated with a shared datapath.

As an example, consider a $M \times N = 4 \times 4$ indoor narrow-band MIMO system and assume that the maximum Doppler frequency is limited to $f_D = 20$ Hz. Note that the maximum Doppler frequency for indoor environments is rather small (e.g., 6 Hz in the IEEE 802.11n channel model). Moreover, assume that the samples are generated at a low sample rate, say $\hat{F}_s \geq 16 \times f_D$ and then interpolated using dedicated interpolators. A fading simulator needs to process $(M \times N) \times (N_S + 1)$ complex oscillators for the generation of diffuse and LOS components (see equations (4.42) and (4.43)). Using an architecture that processes the samples serially, and assuming that each oscillator requires two clock cycles for processing (one cycle for the in-phase and one cycle for the quadrature component), calculating all $M \times N$ fading samples of one period ($\hat{T}_s = 1/\hat{F}_s$ seconds) requires a minimum clock frequency of $F_{clk}^{min} = 2 \times (M \times N) \times (N_S + 1) \times \hat{F}_s$ Hz. For example, for $M = N = 4$, $N_S = 128$, $f_D = 20$, and $\hat{F}_s = 16 \times f_D = 320$ Hz, a serial-processing architecture requires a minimum clock frequency of $F_{clk}^{min} = 1,320,960$ Hz to generate all of the 4×4 complex fading samples.

As long as the maximum clock supported by the designed architecture is higher than the minimum clock frequency F_{clk}^{min}, the designed architecture can generate the fading samples in real-time. Note that with careful design, maximum speeds of 100 MHz to 200 MHz are achievable with the most recent FPGAs. When the maximum clock speed of the designed architecture is much higher than F_{clk}^{min}, the extra clock cycles can be used for either generating more fading samples (e.g., simulating wide-band channels, or multiple channels), increasing the number of scatterers N_S, or increasing the maximum Doppler frequency f_D.

Similar to the single-bounce scattering fading, for generating fading samples in double-bounce scattering models we need to superimpose a number of complex oscillators. However, simulating double-bounce scattering channels may take more computational power as we need to process $N_{S1} \times N_{S2} + 1$ complex oscillators instead of $N_S + 1$ oscillators. For example, for $M = N = 4$, $N_{S1} = 32$, $N_{S1} = 64$, $f_D = 20$, and $\hat{F}_s = 16 \times f_D = 320$ Hz, the above serial-processing architecture requires a minimum clock frequency of $F_{clk}^{min} = 20,971,520$ Hz to generate all of the 4×4 complex fading samples of one period. Fortunately for a reasonable number of first- and second-bounce scatterers (N_{S1} and

N_{S2}), the minimum clock frequency F_{clk} is less than the the maximum clock speed of the fading simulator. However, for a large number of first- and second-bounce scatterers, one can always employ parallel blocks of fading simulators.

Here we want to design a compact architecture that generates the complex sinusoidal waves and superimposes them with appropriate gains for the generation of fading samples. Moreover, the phases for each of the complex oscillators need to be stored and updated. To make our fading simulator as compact as possible, we want to share our datapath for all of the oscillators. As shown by the above example, spending two clock cycles per complex oscillator is a practical assumption (in terms of clock budget) and can result in significant area savings.

4.4.1 Sample Generation

In a typical wireless communication scenario, the maximum Doppler frequency f_D is significantly smaller than the signal sample rate $F_s = 1/T_s$. This allows us to design the fading simulator at a much lower sample rate and thereby reduce the required hardware resources. The resulting low-rate signal can then be interpolated to achieve the desired output sample rate. Here we assume that the channel gains are generated at $\hat{F}_s = F_s/I$ samples per second and then up-sampled by the interpolation factor I.

Our goal is to implement a pipelined architecture for fading simulation that spends two clock cycles for processing each oscillator in each period. For each oscillator the required operations are 1) reading the corresponding phase from memory, 2) calculating $\sin$ or $\cos$ of the phase, 3) scaling with corresponding g_i, 4) superposition of waves, 5) applying Rice factors, 6) updating each oscillator phase for the next period. In the following we explain how our hardware generates the fading samples.

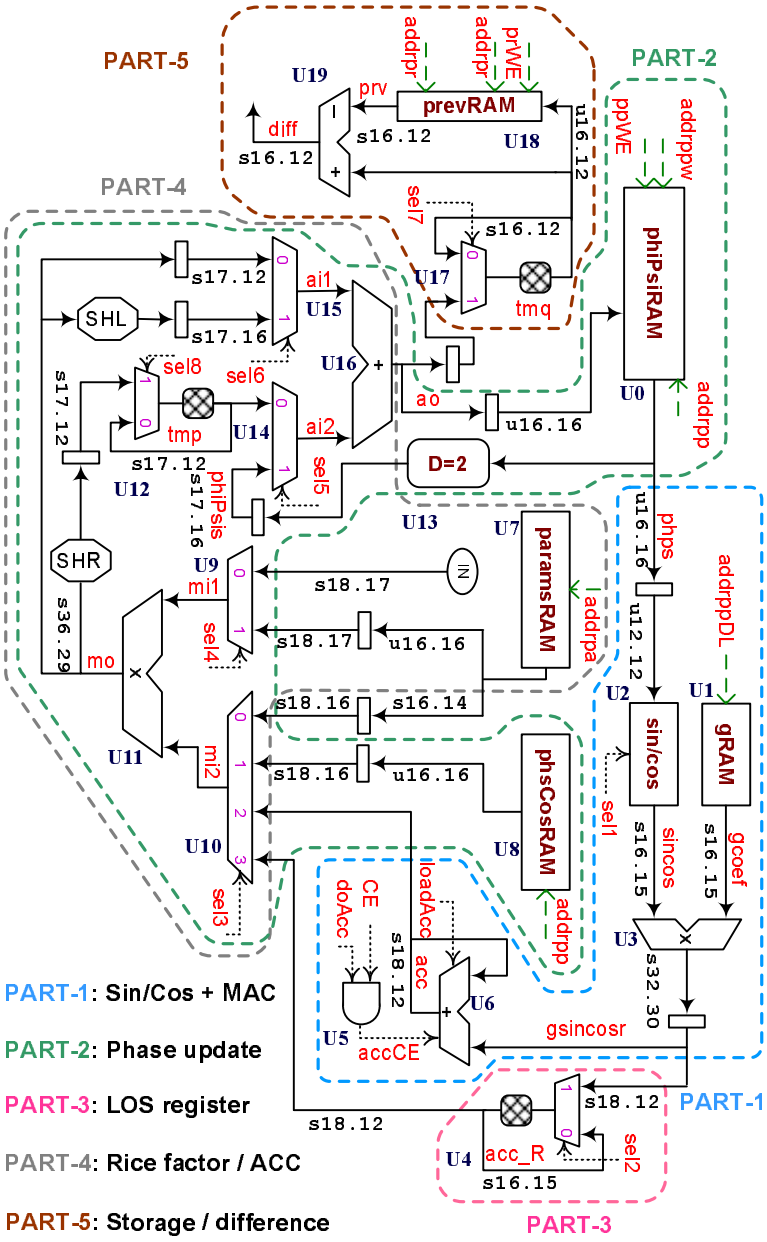


Figure 4.8: Datapath of the MIMO fading samples generator for simulating geometric channel models.

Figure 4.8 shows the datapath of the proposed fading simulator. In each cycle, this datapath calculates the fading gains in two steps. In the first step, the in-phase (or real) part of the channel gains are calculated and in the second step, the quadrature (or imaginary) parts are processed. Moreover, the phases are updated in the second step of each cycle (this will be explained later).

Let us start with equation (4.43). The contribution of the diffuse components to the

channel gain in quadrature form at sample rate $\hat{F}_s$ can be written as

$$\begin{aligned} h_{lp}^{DIF-I}[n] &= C \times T_{lp} \sum_{i=1}^{N_S} g_i \times \mathrm{cs}(\phi(l,p,i,n)), \\ h_{lp}^{DIF-Q}[n] &= C \times T_{lp} \sum_{i=1}^{N_S} g_i \times \mathrm{sn}(\phi(l,p,i,n)), \end{aligned} \tag{4.44}$$

where $C = 1/\sqrt{N_S}$, $T_{lp} = 1/\sqrt{K_{lp}+1}$, $\mathrm{cs}(\phi) = \cos(2\pi\phi)$, $\mathrm{sn}(\phi) = \sin(2\pi\phi)$, and $\phi(l,p,i,n) = I \times f_i \times n + (\upsilon_{lpi} + \theta_i)/2\pi$ is the path-phase for each of the received waves from different scatterers at any time $n \geq 0$. Further, the path-phase can be calculated recursively using the following equation

$$\phi(l,p,i,n) = \begin{cases} \phi(l,p,i,n-1) + \hat{f}_i & \text{for } n > 0, \\ (\upsilon_{lpi} + \psi_i)/2\pi & \text{for } n = 0, \end{cases} \tag{4.45}$$

where $\hat{f}_i = I \times f_i$. Note that in (4.44), $\mathrm{sn}(\phi)$ and $\mathrm{cs}(\phi)$ are periodic functions with period 1, hence the integer part of the path-phase can be discarded when calculating (4.45) without affecting the results. Using extensive computer simulation we found that the phases $\phi(l,p,i,n)$ need to be stored with 16-bit accuracy. Since the values of $\phi(l,p,i,n)$ are limited to the range $[0,1)$, the phases are stored in `phiPsiRAM` (see `U0` in Figure 4.8) in `u16.16` format, i.e., 16-bit unsigned values with 16-bit fraction.

For more clarity, the proposed datapath in Figure 4.8 has been divided into five parts. In this figure, `PART-1` calculates the summations $\sum_{i=1}^{N_S} g_i \times \mathrm{cs}(\phi(l,p,i,n))$ and $\sum_{i=1}^{N_S} g_i \times \mathrm{sn}(\phi(l,p,i,n))$. The stored phases $\phi(l,p,i,n)$, are passed to the module `U2` where $\mathrm{cs}(\phi(l,p,i,n))$ and $\mathrm{sn}(\phi(l,p,i,n))$ are calculated. The sine and/or cosine are calculated in the module `U2`. In this module the sine/cosine of the input is approximated using a look-up table that contains the first quarter cycle of a sine wave. The coefficients $\{g_i\}$ are stored in `U1` in `s16.15` format (i.e., 16-bit signed values with 15-bit fraction). Further, the multiplier `U3` with the accumulator `U6` calculate $\sum_{i=1}^{N_S} g_i \times \mathrm{cs}(\phi(l,p,i,n))$ and $\sum_{i=1}^{N_S} g_i \times \mathrm{sn}(\phi(l,p,i,n))$ after proper format adjustments. The fixed-point data formats shown in Figure 4.8 are chosen based on extensive computer simulation.

In `PART-2` of the datapath shown in Figure 4.8 the phases $\phi(l,p,i,n))$ are updated according to equation (4.45). The initial path-phase values $\phi(l,p,i,0) = (\upsilon_{lp} + \theta_i)/2\pi$, for $l = 1,..,N$, $p = 1,...,M$, $i = 1,...,N_S$ are stored in `phiPsiRAM`. These phases are updated as the time index advances. The maximum Doppler frequency, f_D, is passed to this circuit from input `IN`. Moreover, the values $\{\cos(\omega_i - \gamma)\}$ for $i = 1,...,N_S$ are stored in `phsCosRAM` or `U8`. These two values are multiplied using `U11` to obtain $f_D \times \cos(\omega_i - \gamma)$. As will be explained later, we set the interpolation factor I to be a power of two. The output

of the multiplier is then shifted to obtain $\hat{f}_i = I \times f_D \times \cos(\omega_i - \gamma)$ which is used to update the path-phases according to equation (4.45). Further, a copy of the current phase (through `U13` and `U14`) is passed to the adder `U16` where it is added to $\hat{f}_i$ (through `U15`). Finally, the updated path-phase value is stored in `phiPsiRAM` for the next cycle.

Note that `PART-1` and `PART-2` operate in parallel. In each cycle, the summations $\sum_{i=1}^{N_S} g_i \times \mathrm{cs}(\phi(l,p,i,n))$ and $\sum_{i=1}^{N_S} g_i \times \mathrm{sn}(\phi(l,p,i,n))$ are calculated in `PART-1` for every TX and RX antenna ($p = 1, ..., M$, $l = 1, ..., N$) and for every scatterer ($i = 1, ..., N_S$). The path-phases $\phi(l,p,i,n)$ are updated simultaneously in `PART-2`. The operations of these parts require $M \times N \times N_S$ clock cycles for either the in-phase or quadrature components.

The calculated summations in `PART-1` are then scaled in `PART-4` by $C \times T_{lp}$ to provide the in-phase and quadrature components in equation (4.44). For a large number N_S of scatterers the coefficient $C \times T_{lp} = 1/\sqrt{N_S(K_{lp}+1)}$ is a small number. The scaling is performed using shifting and multiplication operations. More specifically, the coefficient C can be decomposed into $C = 2^{-K} \times \tilde{C}$, where $K = \lfloor log_2(1/C) \rfloor$, and $\tilde{C} = 2^K \times C$. Defining $\tilde{T}_{lp} = T_{lp} \times \tilde{C}$, we have $C \times T_{lp} = 2^K \times \tilde{T}_{lp}$. The values of $\tilde{T}_{lp}$ for $l = 1, ..., N$ and $p = 1, ..., M$ are stored in `paramsRAM` (or `U7`) in `u16.16` format and are passed to the multiplier $U11$ through the multiplexer `U9` in Figure 4.8. These values are then used to scale the summations calculated in `PART-1`. A right-shift operation (K times) after the multiplier `U11` finishes the calculation of the diffuse samples in equation (4.44). Further, the diffuse samples are stored in the register `U14` until they are added to the LOS samples in the adder `U16`.

The LOS samples are computed in a similar way. From equation (4.42) the LOS samples in quadrature form at sample rate $\hat{F}_s$ can be written as

$$\begin{aligned} h_{lp}^{LOS-I}[n] &= W_{lp} \times \mathrm{cs}(\psi(l,p,n)), \\ h_{lp}^{LOS-Q}[n] &= W_{lp} \times \mathrm{sn}(\psi(l,p,n)), \end{aligned} \tag{4.46}$$

where $W_{lp} = \sqrt{K_{lp}/(K_{lp}+1)}$, $\psi(l,p,n) = I \times f_p \times n + \eta_{lp}/2\pi$, $f_p = \cos(\omega_p - \gamma) \times f_D$, and $\eta_{lp} = -2\pi\zeta_{lp}/\lambda$. The path-phases $\psi(l,p,n)$ are updated recursively in a similar way to equation (4.45) with a different initialization value $(1 - \eta_{lp}/2\pi)$. The path-phases $\psi(l,p,n)$, for $l = 1, ..., N$ and $p = 1, ..., M$ are stored in `phiPsiRAM` and updated in `PART-2`. Moreover `PART-1` calculates $\mathrm{cs}(\psi(l,p,n))$ and $\mathrm{sn}(\psi(l,p,n))$, which are stored in the register `U4` in `PART-3`. Also the values of W_{lp} for $l = 1, ..., N$ and $p = 1, ..., M$ are stored in `paramsRAM` (or `U7`), which are passed to the multiplier `U11` through the

multiplexer U9. This multiplier then calculates the LOS sample (according to equation (4.46)). The LOS sample (from U15) is then added to the diffuse sample (from U14) in the adder U16 to obtain the final in-phase or quadrature component of the channel gain. The generated channel gains are yet to be interpolated, which will be discussed in the following.

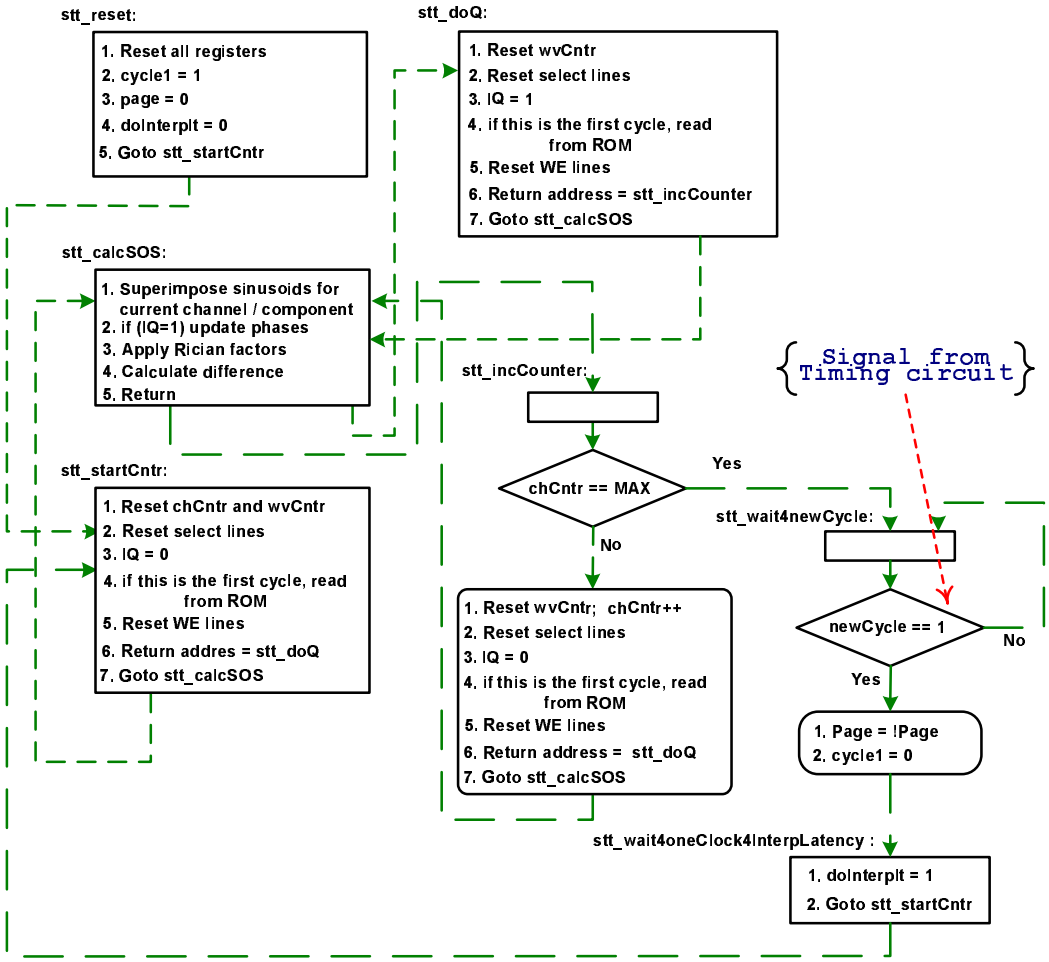


Figure 4.9: Control flow diagram for the control-unit of the fading simulator.

Figure 4.9 shows the simplified state machine of the control unit (CU) that manages the datapath in Figure 4.8. After reset, this state machine starts at stt_reset. In this state, the register cycle1 is set to 1 indicating the first period of the sample generation. In the first period, the initial phases are read from read-only memory (ROM) blocks. Note that these ROMs are not shown in Figure 4.8 for simplicity. Finally, another register, page, is set to zero. As will be explained later, the interpolators require the difference signal to be kept "latched" for every period. That is why each interpolator keeps two copies of the difference signal, called page0 and page1. When the fading simulator is working on

`page0` (`page1`), it informs the interpolators to use the value stored in `page1` (`page0`) since the value store in `page0` (`page1`) is volatile and unreliable.

Further, in the initial state `stt_reset`, the interpolators are inactivated by setting the register `doInterplt=0`. After initialization, the state machine goes to the `stt_startCntr` state to initialize the datapath in Figure 4.8 for calculating the fading samples and updating the phases. In this fading simulator, two counters are used for processing and addressing the fading samples. The `chCntr` counter is used for addressing the $M \times N$ channels and the `wvCntr` is used for counting $N_S + 1$ complex oscillators (waves). The quadrature components of each oscillator are distinguished with the `IQ` register. When `IQ=0` the in-phase component is being calculated and `IQ=1` means that the quadrature component is under process.

The `stt_startCntr` state resets the channel counter, the oscillator counters, the memory write-enable lines and the select lines. After initializations for the starting of each period, this state jumps to the `stt_calSOS` state for calculating the in-phase component of the current channel (channel 0). However, before going to the next state, the return address is set to `stt_doQ` so that after calculation of the in-phase component for the current channel, the required initializations are made for calculating the quadrature component.

Since `IQ` is set to `0`, the `stt_calSOS` state calculates the in-phase component for the current channel and then goes to the `stt_doQ` state. This state sets `IQ=1` and initializes the `wvCntr` register for the calculation of the quadrature component of the current channel. Before calling `stt_calSOS`, however, the return address is set to `stt_incCounter`. The control then moves to the `stt_calSOS` state for the calculation of the quadrature component for the current channel. The `stt_calSOS` state also updates the oscillator phases for the next period (only for the current channel).

After calculating the in-phase and quadrature components of the fading sample for channel 0 and updating the corresponding phases, the state machine goes to the `stt_incCounter` state where it initializes the registers for calculating the in-phase and quadrature components of fading samples for another channel by increasing the channel counter.

When all of the channels have been processed (i.e., the fading samples for one period of all $M \times N$ channels have been generated and the corresponding phases have been updated), the state machine goes to the `stt_wait4newCycle` state. As mentioned before, the datapath in Figure 4.8 needs a certain number of clock cycles to process all of the fading samples of one period. After generating all of the fading samples for one period (i.e., $\hat{T}_s$

seconds), the control unit waits until the interpolators are done with the interpolation. After the end of each period (signaled from the timing circuit by `newCycle=1`), the control unit informs the interpolators to use the newly generated data by changing the page. The control unit also makes sure that the updated phases are read from random-access memories (RAMs) not ROMs by setting `cycle1=0`. Moreover, after the first period, the interpolators are informed to start by setting `doInterplt=1`. From here, the fading samples are generated in the same fashion in the following periods.

4.4.2 Interpolation

To simplify the hardware implementation, we constrain the interpolation factor I to powers of 2, i.e., $I = 2^P$. Using a linear interpolator, the interpolated fading samples $\tilde{h}_{lp}[n]$, $n \geq 0$, at the times $n = 2^P k + u$, $k \geq 0$, $u = 0, ..., 2^P - 1$ can be written as

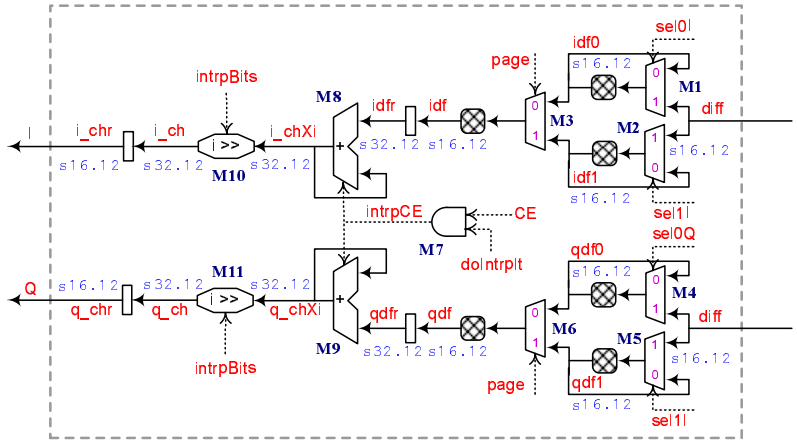
$$\tilde{h}_{lp}[2^P k + u] = h_{lp}[0] + \sum_{z=0}^{k-1} d_{lp}[z] + 2^{-P} d_{lp}[k] u \tag{4.47}$$

where

$$d_{lp}[z] = \begin{cases} h_{lp}[2^P(z+1)] - h_{lp}[2^P z] & \text{for } z \geq 0, \\ 0 & \text{otherwise,} \end{cases} \tag{4.48}$$

denotes the difference between subsequent fading samples at sample rate $\hat{F}_s = 2^{-P} F_s$. Note that the expression $2^{-P} h_{lp}[k] u$ in (4.47) can be calculated using shifting and running-sum operations and therefore the linear interpolator can be conveniently implemented using a simple accumulator and shifter.

In `PART-5` of Figure 4.8, the quadrature components of $b_{lp}[z]$ are calculated. More specifically, in `PART-5` the low frequency channel gains are stored in the memory `prevRAM` and the difference between the current sample and the previous sample for each TX and RX pair (see equation (4.48)) is calculated and passed to the linear interpolator for generating the final channel gains according to (4.47).



(a)

(b)

chSel	wDiff	page	IQ	sel0I	sel1I	sel0Q	sel1Q
0	0	0	0	0	0	0	0
0	0	0	1	0	0	0	0
0	0	1	0	0	0	0	0
0	0	1	1	0	0	0	0
0	1	0	0	0	0	0	0
0	1	0	1	0	0	0	0
0	1	1	0	0	0	0	0
0	1	1	1	0	0	0	0
1	0	0	0	0	0	0	0
1	0	0	1	0	0	0	0
1	0	1	0	0	0	0	0
1	0	1	1	0	0	0	0
1	1	0	0	0	1	0	0
1	1	0	1	0	0	0	1
1	1	1	0	1	0	0	0
1	1	1	1	0	0	1	0

Figure 4.10: (a) Datapath of the quadrature linear interpolator, and (b) control signals for the linear interpolator.

Figure 4.10 (a) shows the datapath of the implemented linear quadrature interpolator. As the fading simulator calculates the difference samples, it informs the corresponding interpolator to latch the updated value in the page register that is not in use. A decoder (decoding `chCntr`) generates the selection signal for individual interpolators and the fading simulator informs the target interpolator by rising the `wDiff` flag. Depending on the state of the `IQ` register, the updated difference is latched into the inactive (under process by fading generator, not the interpolator itself) register for the in-phase or quadrature path. Figure 4.10 (b) summarizes the operations of the selection lines in a table.

As mentioned above, the interpolator is just a simple accumulator performing a running-sum operation. Moreover, the operation of scaling by 2^{-P} in equation (4.47) needs to be performed with the two barrel shifters `M10` and `M11` for the in-phase and quadrature paths, respectively.

So far we explained the hardware implementation of the geometric fading simulator. In the following part we will present the simulation results.

4.4.3 Simulation Results

To ensure the accuracy of the geometric fading simulator, we first implemented a fixed-point bit-true model in MEX (C for MATLAB) and generated sequences of fading variables. Specifically, to verify our datapath we simulated a MIMO channel with three different models, 1) the one-ring model, 2) the geometric elliptical channel model, and 3) the double-bounce two-ring model. We compared our bit-true fixed-point results against the floating-point results to check the accuracy of our fixed-point model.

We simulated a 2×2 MIMO system in which the TX-RX pair are 860 meters apart. The TX and RX antenna arrays are positioned at a 90 degree angle from the horizon. Also the RX is assumed to be moving in the $\gamma = 60$ degree direction. We set the maximum Doppler frequency $f_D = 10$ Hz, sample rate $F_s = 10,000$ Hz, wavelength $\lambda = 20$ cm, and Rice factor $K_{11} = K_{12} = K_{21} = K_{22} = 0$. Finally, we used isotropic scattering and omnidirectional antennas at TX and RX sides (i.e., $\kappa = \kappa_T = \kappa_U = 0$) and we set the antenna TX and RX antenna spacing to $\lambda/2 = 10$ cm.

The fading samples were generated at sample rate $\hat{F}_s = 312.5$ Hz. The interpolator is set to increase the sample rate 32 times to provide the final sample rate of $F_s = 10,000$ samples per second.

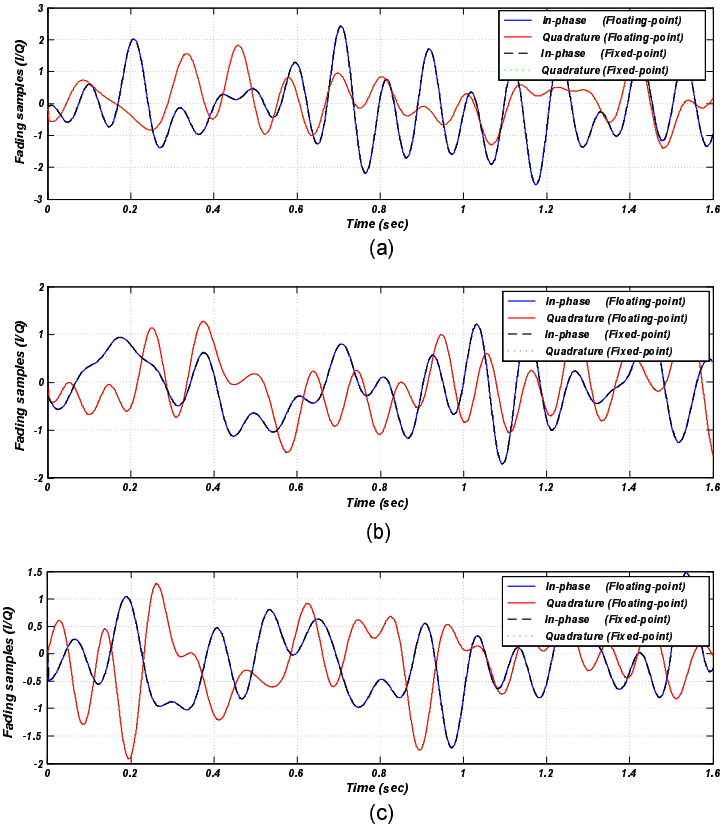


Figure 4.11: Fading samples generated using (a) the one-ring model, (b) the geometric elliptical model and, (c) the two-ring model.

In the one-ring model, $N_S = 128$ scatterers surround the receiver uniformly on a ring of radius $R^{RX} = 30$ meters. Figure 4.11 (a) shows the fading samples $h_{11}[n]$ (channel gain from the first TX antenna to the first RX antenna) that are generated with both floating-point and fixed-point models. This figure shows good agreement between the fixed-point and the floating-point results.

Figure 4.11 (b) shows the fading samples $h_{11}[n]$ that are generated using the geometric elliptical channel model. Both floating-point and fixed-point results are shown. In this model, $N_S = 128$ scatterers surround the transmitter and the receiver on an ellipsoid with the major axis of $2a = 1460$ meters and the minor axis of $2b = 1182$ meters (the transmitter and the receiver are on the focal points of this ellipsoid). This figure also shows good agreement between the fixed-point and the floating-point results.

We also simulated a double-bounce scattering fading channel with the two-ring channel model. In this model, we uniformly distributed $N_{S1} = 32$ scatterers around the transmitted on a circle of radius $R^{TX} = 30$ meters. Also we assumed that $N_{S2} = 64$ scatterers surround

the receiver uniformly on a ring of radius $R^{RX} = 30$ meters. Figure 4.11 (c) shows the fading samples $h_{11}[n]$ that are generated using this model. Here too, the fixed-point results show complete agreement with the floating-point model.

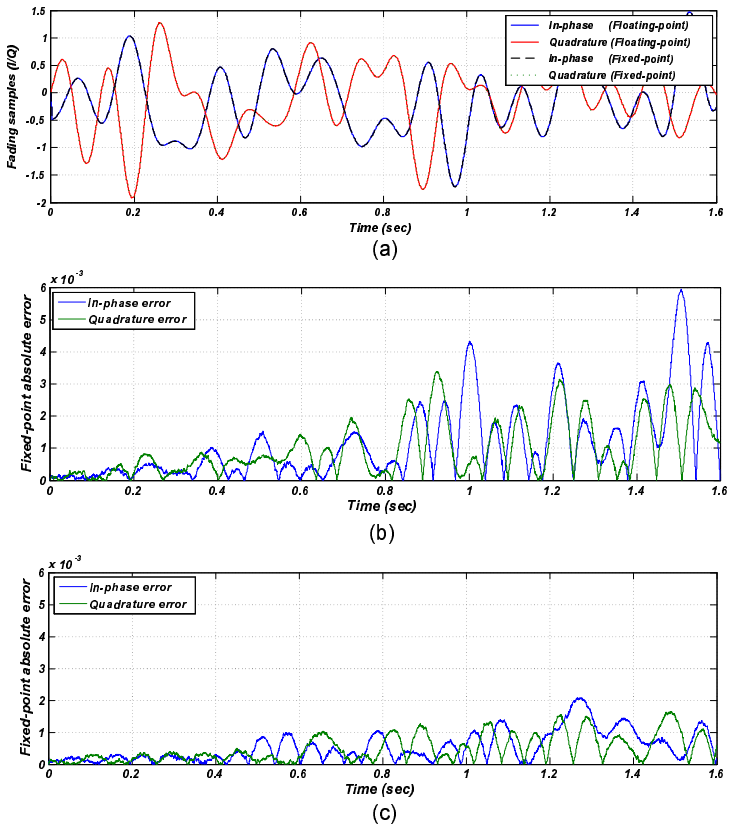


Figure 4.12: (a) Fading samples generated using the two-ring model. (b) Absolute error between fixed-point results and the floating-point results when phases are rounded towards zero. (c) Absolute error when phases are rounded towards the nearest integer.

Figure 4.12 plots the absolute error between the floating-point and the fixed-point results for the two-ring model. Since the oscillator phases are updated using running summations (see equation (4.45), and the description of `PART-2` in Section 4.4), the quantization noise can accumulate in the oscillator phases over time. This in turn causes the fixed-point results to drift away from the floating-point results.

A major contributor to the quantization noise in the oscillator phases is the rounding operation that happens when calculating $f_D \times \cos(\omega_i - \gamma)$ (see the description of `PART-2` in Section 4.4). The result of the multiplication is a 36-bit value that needs to be rounded to a 17-bit value according to the datapath shown in Fig 4.8. The simplest rounding method is to discard the extra fraction bits. Since the phases are positive values, discarding the extra

bits can be interpreted as rounding towards zero. Figure 4.12 (b) plots the absolute error between the fixed-point results and the floating-point results for this case. The original fading samples are plotted in Figure 4.12 (a) for reference. As Figure 4.12 (b) shows, the magnitude of the peaks of the absolute error increases with time. Rounding the 36-bit multiplier outputs to the nearest integer on the other hand can reduce the quantization effects as shown in Figure 4.12 (c). However, rounding to the nearest integer is not necessary as the slow phase drifts do not affect the statistics of the generated fading samples significantly.

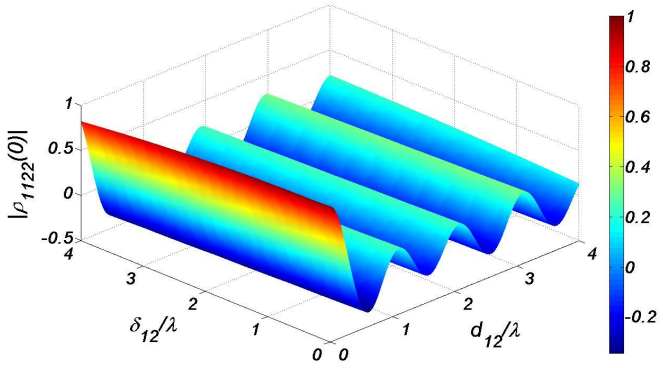


Figure 4.13: Cross-correlation between $h_{11}[n]$ and $h_{22}[n]$ versus transmitter and receiver antenna separation. Results are obtained from fixed-point computer simulation of a MIMO channel with the one-ring channel model.

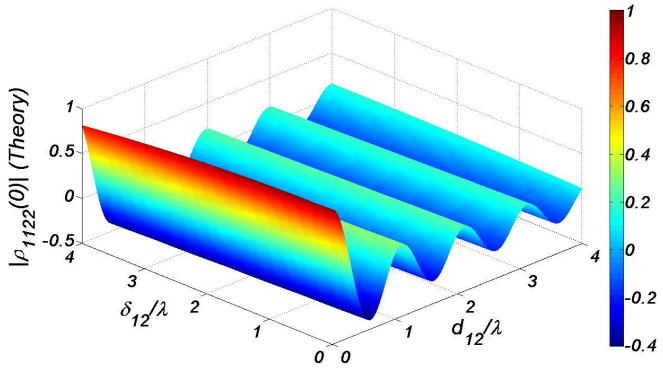


Figure 4.14: Theoretical approximation of the cross-correlation between $h_{11}[n]$ and $h_{22}[n]$ for the one-ring channel model plotted versus antenna separation at the transmitter and the receiver.

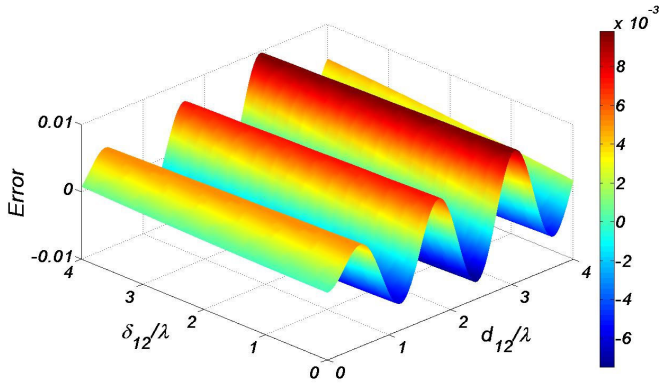


Figure 4.15: Difference between the measured cross-correlation and the theoretical approximation of the cross-correlation between $h_{11}[n]$ and $h_{22}[n]$ for the one-ring channel model.

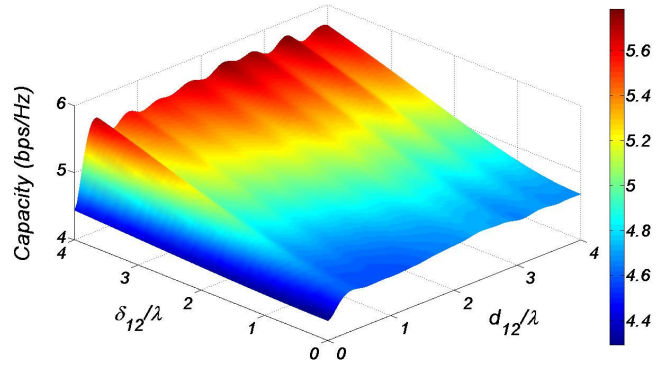


Figure 4.16: Estimated ergodic capacity of a 2×2 MIMO channel versus antenna separation at the transmitter and receiver. The MIMO fading channel is simulated with the one-ring channel model.

To check the statistical accuracy of our fading simulator hardware we compared the measured cross-correlations with their corresponding analytical approximations. In this simulation the transmitter and the receiver of a 2×2 MIMO system are set $D = 860$ meters apart and the antenna tilt at the receiver and transmitter is $\alpha_{12} = \beta_{12} = 90$ degrees. Further, the maximum Doppler frequency is $f_D = 10$ Hz and samples are generated at $F_s = 1000$ samples per second. We set the wavelength $\lambda = 20$ cm, and Rice factor $K_{11} = K_{12} = K_{21} = K_{22} = 0$, and $\kappa = \kappa_T = \kappa_U = 0$.

The one-ring model was simulated with $N_S = 256$ scatterers uniformly distributed around the receiver on a circle of radius $R^{RX} = 30$ meters. Figure 4.13 shows the cross-correlation between $h_{11}[n]$ and $h_{22}[n]$ versus transmitter and receiver antenna separation. The simulation results are obtained using our fixed-point hardware model. Figure 4.14 shows the approximated theoretical cross-correlation from equation (4.11), (4.16), and

(4.17). Figure 4.15 plots the difference between the theoretical approximation and the measured cross-correlation. As this figure shows, there is an insignificant difference between the expected cross-correlation and the generated results.

Moreover, Figure 4.16 plots the estimated ergodic capacity of the above one-ring $M \times N = 2 \times 2$ MIMO channel. The ergodic capacity is expressed as [185]

$$E\{C\} = E\left\{\log_2 \det\left(\mathbf{I}_N + \frac{\sigma}{M}\mathbf{H}\mathbf{H}^*\right)\right\} \text{ bps/Hz}, \tag{4.49}$$

where $\mathbf{I}_N$ is an $N \times N$ identity matrix, and σ is the signal-to-noise ratio. In this simulation the signal-to-noise ratio is set to 15 dB and the ergodic capacity is plotted versus antenna separation at the transmitter and receiver. Note that for a single antenna system ($M = N = 1$) the ergodic capacity is approximately 4.32 bps/Hz. As mentioned before, the one-ring channel model can be related to a scenario where the base-station (transmitter) is not obstructed while the mobile-station (receiver) is surrounded by scatterers. As this figure shows, in a one-ring fading channel mode increasing the antenna spacing at the mobile-station does not increase the channel capacity significantly, while increasing the antenna spacing at the base-station affects the channel capacity considerably.

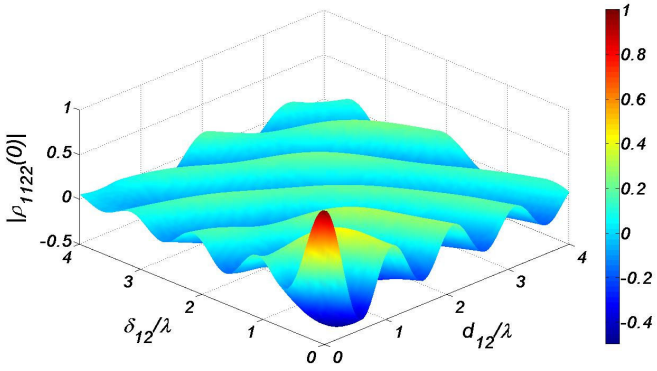


Figure 4.17: Cross-correlation between $h_{11}[n]$ and $h_{22}[n]$ versus transmitter and receiver antenna separation. Results are obtained from fixed-point computer simulation of a MIMO channel with the geometric elliptical channel model.

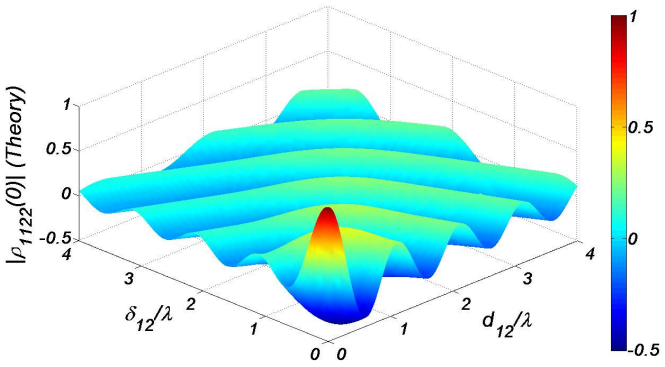


Figure 4.18: Theoretical approximation of the cross-correlation between $h_{11}[n]$ and $h_{22}[n]$ for the geometric elliptical channel model plotted versus antenna separation at the transmitter and the receiver.

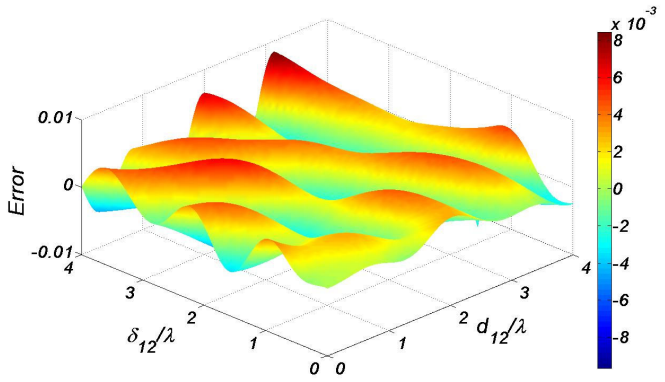


Figure 4.19: Difference between the measured cross-correlation and the theoretical approximation of the cross-correlation between $h_{11}[n]$ and $h_{22}[n]$ for the geometric elliptical channel model.

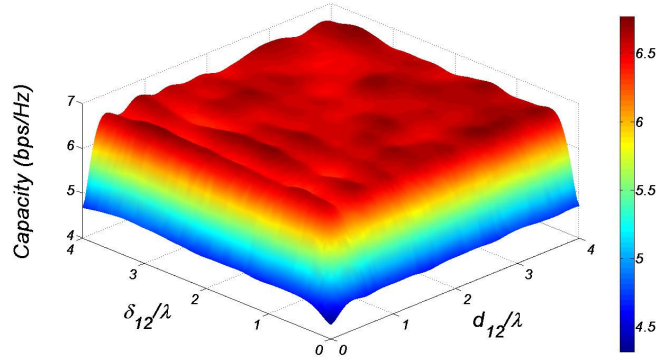


Figure 4.20: Estimated ergodic capacity of a 2×2 MIMO channel versus antenna separation at the transmitter and receiver. The MIMO fading channel is simulated with the geometric elliptical channel model.

We also verified our fading simulation platform by simulating a 2×2 MIMO fading channel with the geometric elliptical model. The system parameters are chosen like in the above MIMO system. In this model, $N_S = 256$ scatterers surround the transmitter and the receiver on an ellipsoid with the major axis of $2a = 1460$ meters and the minor axis of $2b = 1182$ meters. Figure 4.17 shows the measured cross-correlation between $h_{11}[n]$ and $h_{22}[n]$ versus transmitter and receiver antenna separation. The theoretical cross-correlation approximation is shown in Figure 4.18 and the difference between the measured cross-correlation and the expected results is illustrated in Figure 4.19. The theoretical approximation of the cross-correlation was found by numerical evaluation of equations (4.23), (4.17), and (4.25). As Figure 4.19 shows, the generated fixed-point samples have accurate statistics.

Figure 4.20 plots the ergodic capacity of the above MIMO channel model versus antenna spacing at the transmitter and receiver. As this figure shows, when both transmitter and receiver are surrounded by scatterers, unlike the one-ring model, increasing the antenna spacing at either side can help increasing the channel capacity.

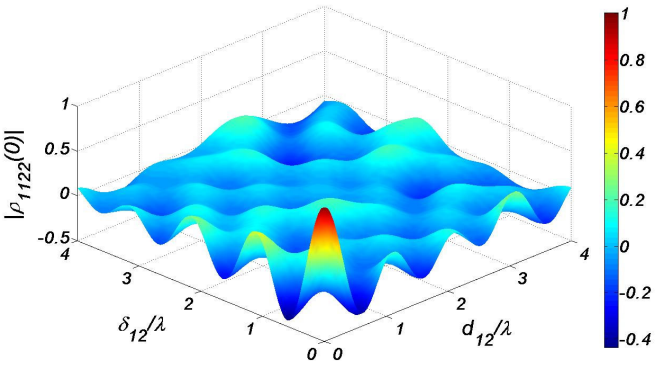


Figure 4.21: Cross-correlation between $h_{11}[n]$ and $h_{22}[n]$ versus transmitter and receiver antenna separation. Results are obtained from fixed-point computer simulation of a MIMO channel with the two-ring channel model.

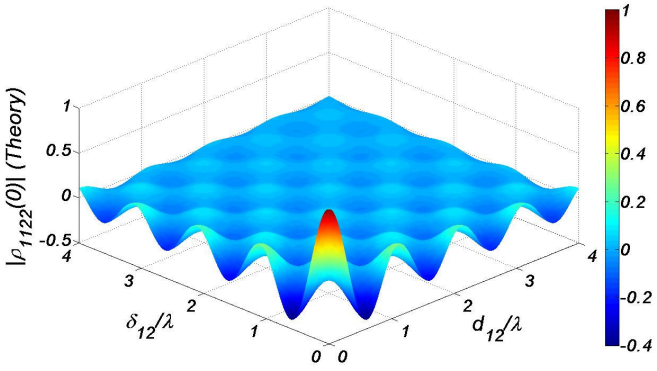


Figure 4.22: Theoretical approximation of the cross-correlation between $h_{11}[n]$ and $h_{22}[n]$ for the two-ring model plotted versus antenna separation at the transmitter and the receiver.

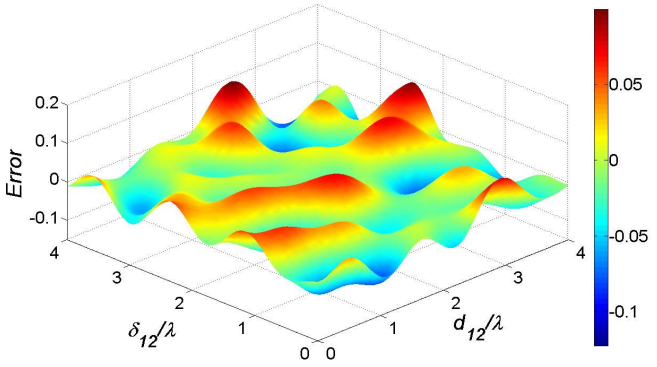


Figure 4.23: Difference between the measured cross-correlation and the theoretical approximation of the cross-correlation between $h_{11}[n]$ and $h_{22}[n]$ for the two-ring channel model.

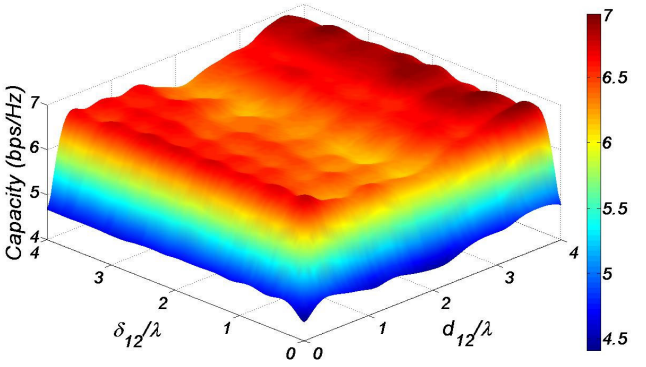


Figure 4.24: Estimated ergodic capacity of a 2×2 MIMO channel versus antenna separation at the transmitter and receiver. The MIMO fading channel is simulated with the two-ring channel model.

We also simulated a 2×2 MIMO channel with the two-ring channel model. In this model, we uniformly distributed $N_{S1} = 128$ scatterers around the transmitted on a circle of radius $R^{TX} = 30$ meters. Also we assumed that $N_{S2} = 128$ scatterers surround the receiver uniformly on a ring of radius $R^{RX} = 30$ meters. The rest of the system parameters are chosen like the above system parameters. Figure 4.21 plots the measured cross-correlation between $h_{11}[n]$ and $h_{22}[n]$ versus transmitter and receiver antenna separation. The theoretical approximation for the cross-correlation from equations (4.33), (4.17), (4.39), (4.40), and (4.41) is plotted in Figure 4.22. The difference between the simulated cross-correlation and the theoretical approximation is plotted in Figure 4.23. Comparing the result in figures 4.21 and 4.22 shows some difference between the measured results and the expected cross-correlation. The difference is mainly because of the low number of simulated scatterers.

The approximate equation for the cross-correlation between fading samples in (4.39), (4.40), and (4.41) is based on the assumption of continuous distribution of an infinite number of scatterers. However, implementation complexity of simulating the two-ring model grows prohibitively large as the number of first- or second-bounce scatterers increases.

The simulation of the fixed-point model for double-scattering scenarios is extremely computationally intensive as it needs to generate $100 \times 100 = 10000$ cross-correlations point for each plot. We require a large number of channel samples (here we used 10000 sample) for reliable estimation of the cross-correlation. Moreover, $(M \times N) \times (N_{S1} \times N_{S2} + 1)$ complex sinusoids need to be scaled and superimposed for calculation of each set of fading samples. With $N_{S1} = 128$ and $N_{S2} = 128$ scatterers and for $M = N = 2$ antennas, our software model required twenty four days to generate 6.554×10^{12} complex sinusoids, and estimate the correlations on a dual-core 3.6 GHz Intel Xeon processor. Note that our fixed-point library includes kernels written in the 80386 32-bit machine language and is extremely fast (80386 Assembly language embedded in C code and linked to MATLAB as a library of MEX files). Software simulation of double-bounce scattering becomes very time consuming as the implementation complexity grows rapidly with the number of scatterers.

Figure 4.20 plots the ergodic capacity of the above two-ring MIMO channel model versus antenna spacing at the transmitter and receiver. As this figure shows, proper antenna spacing at both transmitter and receiver can increase the channel capacity in this scenario.

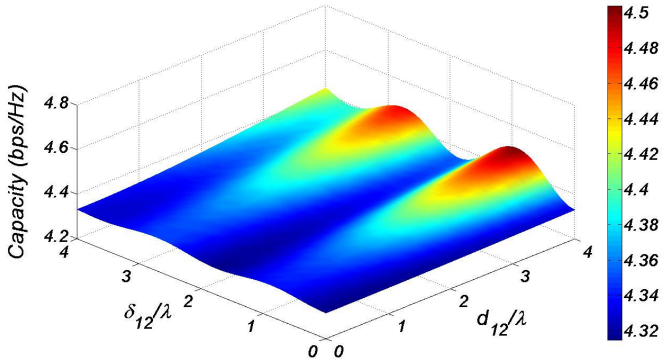


Figure 4.25: Illustration of the impact of the keyhole effect on the channel capacity. This figure shows the estimated ergodic capacity of a 2×2 MIMO channel versus antenna separation at the transmitter and receiver.

To show the impact of the keyhole effect on the capacity of a MIMO channel, we used our fading simulator to simulate a double-bounce scattering case where there is no LOS path between the transmitter and receiver, and the transmitted signal bounces off a couple of far scatterers (i.e., $N_{S1} = 2$) before reaching the receiver by bouncing off $N_{S2} = 256$ near scatterers. The far scatterers are located $R^{TX} = 150$ meters from the transmitter at $\mu_T = 140$ degree angle with the directivity parameter $\kappa_T = 80$ (narrow beam). Also, $N_{S2} = 256$ near scatterers are uniformly distributed on a circle around the receiver. The rest of the system parameters are set like the above systems. Figure 4.25 shows the impact of the keyhole effect on the capacity of the 2×2 MIMO system. As this figure shows, increasing the antenna separation between the transmitter or receiver does not improve the channel capacity significantly. Please note that the ergodic channel capacity for a single antenna system at the same signal-to-noise ratio (15 dB) is approximately 4.32 bps/Hz. As this figure shows, when the keyhole effect happens, the capacity of the multiple-antenna channel does not show significant improvement over single-antenna channels.

4.4.4 Hardware Implementation Results

We implemented our new fading simulator for a 4×4 MIMO system on a Xilinx Virtex-5 XC5VLX110-3 FPGA. We configured the hardware to generate 16 streams of channel gains. Table 4.1 presents the implementation results of our fading simulator. In the implemented hardware, we set the maximum number of scatterers to $N_S = 128$. However more scatterers can be added easily by increasing the storage capacity. The implemented hardware can be configured to simulate a wide variety of propagation conditions and chan-

nel models. Our FPGA implementation uses only 4597 configurable slices, two multipliers (DSP48E), and three on-chip 36-Kb block memories. Moreover, Table 4.1 compares the new fading simulator with two MIMO fading simulators. As this table shows, the new fading simulator is faster and much smaller than the designs reported in [59] and our previous fading simulator in [8], despite the fact that the new design simulates more scatterers and is capable of simulating more fading model types. The accuracy, speed and compactness of the proposed design makes it an appropriate simulator for hardware verification of MIMO systems.

Table 4.1: Comparison of Implementation Results

Design	**I (from [59])**	**II (from [8])**	**III (NEW)**
MIMO Model	IID	IID	IID, GSCM
$M \times N$	4×4	4×4	4×4
N_S	16	8	128
FPGA	EP20K1000-3	XC2VP100-6	XC5VLX110-3
Max. Clock	50 MHz	201.1 MHz	324.5 MHz
Output rate	16×1.5	$2 \times 16 \times 201$	16×324.5
# of slices	22576 (58%)	41198 (93%)	4597 (6.6%)
# of MULTs	–	272 (61%)	2 (3%)
# of BRAMs	(17%)	288 (65%)	3 (2.0%)

We used the designed fading simulator for testing a 2×2 MIMO system on a GVA-290 FPGA board [161]. This board hosts two Xilinx Virtex-E XCV2000E FPGAs. The testing platform includes a pseudo-random data source and a data sink for the bit error rate (BER) performance measurements. The fading channel simulator implemented on this prototyping platform can be parameterized in real-time using the host computer to simulate various propagation conditions.

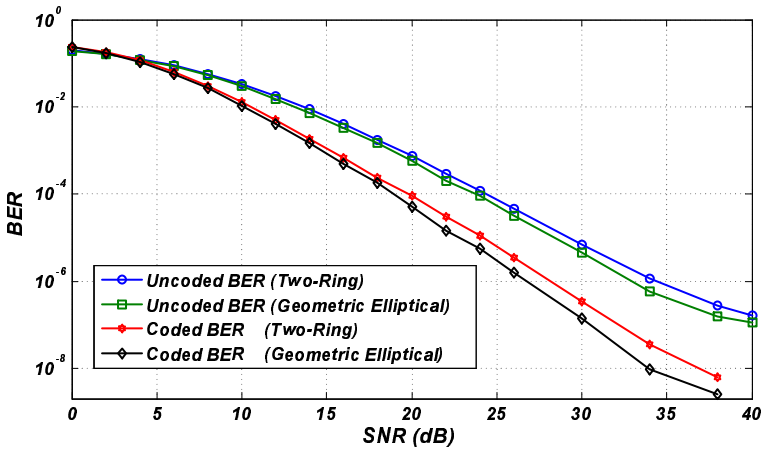


Figure 4.26: Bit error rate performance of a 2×2 MIMO system measured using the FPGA-based fading simulator.

Using our testing platform we measured the BER performance of a 2×2 MIMO system. The transmitter under test utilizes an extended Golay channel-code, an interleaver of length 16383, and a 4-QAM modulator. The receiver includes a maximum-likelihood (ML) detector with perfect channel state information, a de-interleaver and a decoder for the extended Golay code. The implemented MIMO system will be explained in more detail later in this chapter.

Figure 4.26 shows the hardware-generated uncoded and coded BER performance of this MIMO system under the geometric elliptical and two-ring MIMO channel models when the antenna spacing at the transmitter and the receiver is $0.5 \times \lambda$. We simulated a 2×2 MIMO system in which the receiver is placed $D = 860$ meters away from the transmitter. The transmit and receive antenna arrays are positioned at a 90 degree angle from the horizon. Also the receiver is assumed to be moving in the $\gamma = 60$ degree direction. We set the maximum Doppler frequency $f_D = 10$ Hz, wavelength $\lambda = 20$ cm, Rice factor $K_{11} = K_{12} = K_{21} = K_{22} = 0$, and $\kappa = \kappa_T = \kappa_U = 0$.

In the geometric elliptical model, $N_S = 128$ scatterers surround the TX and RX uniformly on an ellipsoid with the major axis of $2a = 1460$ meters and the minor axis of $2b = 1182$ meters. Also, for the two-ring model, $N_{S1} = 32$ and $N_{S2} = 64$ scatterers uniformly surround the TX and RX, respectively, on two circles of radius $R_{TX} = R_{RX} = 30$ meters. The implemented fading simulator shows the difference in the performance of the system under test for the two different propagation scenarios.

4.5 Analytical Models

In contrast to physical models that characterize physical aspects of wave propagation, analytical channel models express the MIMO channel impulse response in terms of a complex matrix with a specific structure. Due to the convenience of analysis, these models are popular in analytic studies of MIMO systems (see for example [200, 210–214]).

Analytical channel models try to mimic different aspects of a MIMO fading channel. Correlation-based analytical models try to generate fading samples with specific correlation properties [184, 200, 212–214, 223, 226]. Other analytical fading channel models are inspired by the propagation effects in the MIMO fading channel [210, 227–230].

The simplest MIMO channel model is the i.i.d. channel model which is characterized by channel coefficients that are i.i.d. zero-mean complex Gaussian random variables. This idealized channel model allows tractable theoretical results, such as the ergodic channel capacity [184, 185]. The i.i.d. MIMO channel model corresponds to a so called "rich scattering" narrow-band scenario. In practice, however, the channel coefficients between different transmit-receive antenna pairs show correlation due to clustered scattering in realistic environments, antenna response and antenna spacing. In realistic conditions, the capacity of MIMO channels can be substantially lower depending on the level of correlation [200, 227].

Assuming that the narrow-band MIMO fading channel is described as a stationary and zero-mean complex Gaussian process, we can fully characterize the MIMO fading process with its second-order statistics [76]. If $\mathbf{H}$ is an $N \times M$ matrix, then we use $\mathrm{vec}(\mathbf{H}) = [\mathbf{h}_1^T, ..., \mathbf{h}_M^T]^T$ to denote the $NM \times 1$ vector formed by stacking the columns of $\mathbf{H}$ (here $(\cdot)^T$ denotes the transpose operation). Using this notation, the full correlation matrix is expressed as [200, 223, 226]

$$\mathbf{R}_H = E\{\mathbf{h}\mathbf{h}^H\}, \tag{4.50}$$

where $\mathbf{h} = \mathrm{vec}\{H\}$, and $(\cdot)^H$ denotes the conjugate transpose (a.k.a. Hermitian transpose) operation. Based on the zero-mean complex Gaussian assumption, the multivariate Gaussian fading process is fully described by its PDF [76] given by

$$f(\mathbf{h}) = \frac{1}{\pi^{MN} \det\{\mathbf{R}_H\}} \exp\left\{-\mathbf{h}^H \mathbf{R}_H^{-1} \mathbf{h}\right\}. \tag{4.51}$$

The most general way of generating the zero-mean complex Gaussian MIMO channel impulse response $\mathbf{H}$ is by

$$\mathbf{H} = \mathrm{unvec}\{\mathbf{R}_H^{1/2}\mathbf{g}\}, \tag{4.52}$$

where $\mathbf{g}$ is an $MN \times 1$ vector of unit-variance and zero-mean i.i.d. complex Gaussian variables, and $\mathbf{R}_H^{1/2}$ denotes any arbitrary matrix square root satisfying $\mathbf{R}_H^{1/2}(\mathbf{R}_H^{1/2})^H = \mathbf{R}_H$. For example, $\mathbf{R}_H^{1/2}$ can be calculated as $\mathbf{R}_H^{1/2} = \mathbf{U}\mathbf{\Sigma}^{1/2}\mathbf{U}^H$ where $\mathbf{U}$ and $\mathbf{\Sigma}$ are obtained by eigenvalue decomposition of $\mathbf{R}_H$, i.e., $\mathbf{U}\mathbf{\Sigma}\mathbf{U}^H = \mathbf{R}_H$.

Note that the main drawback of the above model, also known as the full correlation model, is its complexity. In this model, $(MN)^2$ parameters are required to fully characterize the correlation between fading samples, however, a direct relation between elements of $\mathbf{R}_H$ and the physical propagation effects is not clear. Several different analytical models have been proposed in the literature to simplify the full correlation model and provide a direct interpretation of the elements of $\mathbf{R}_H$ (e.g., the Kronecker and the Weichselberger channel models [194, 212–214, 223, 226, 263, 269]).

In the following we briefly present the Kronecker, Weichselberger, and virtual channel representation channel models. We will also discuss efficient efficient hardware implementation of a MIMO fading channel simulator that supports the above models.

4.5.1 Kronecker Model

The Kronecker model is one of the analytical MIMO fading channel models that is widely used in the research community [194, 200, 212–214, 223, 263, 269]. This model was proposed by [223] in the framework of the IST SATURN project [270] and has also been used in the IEEE 802.11n TGn channel model [92].

The Kronecker model assumes that the spatial correlation at the transmitter end is independent of the spatial correlation at the receiver end (see equations (4.39), (4.40), and (4.41) as a justification), hence the full correlation matrix $\mathbf{R}_H$ can be well approximated as the Kronecker product of the correlation matrices of both ends, i.e.,

$$\mathbf{R}_H = \mathbf{R}_{TX} \otimes \mathbf{R}_{RX}, \tag{4.53}$$

where $\otimes$ denotes the Kronecker product, $\mathbf{R}_{TX} = E\{\mathbf{H}^T\mathbf{H}^*\}$ is the TX correlation matrix where $(\cdot)^*$ is the conjugate operator, and $\mathbf{R}_{RX} = E\{\mathbf{H}\mathbf{H}^H\}$ denotes the RX correlation matrix. The assumption (4.53) can be used to simplify the fading channel simulation. Under this assumption it can be shown than the (4.52) simplifies to

$$\begin{aligned} \mathbf{H} &= \mathrm{unvec}\{(\mathbf{R}_{TX} \otimes \mathbf{R}_{RX})^{1/2}\mathbf{g}\}, \\ &= \mathbf{R}_{RX}^{1/2}\mathbf{G}(\mathbf{R}_{TX}^{1/2})^T, \end{aligned} \tag{4.54}$$

where $\mathbf{G} = \text{unvec}\{\mathbf{g}\}$ is an $N \times M$ matrix with i.i.d. zero-mean unit-variance complex Gaussian elements.

Besides the simplicity of channel matrix synthesis, the other advantage of the Kronecker model over the full correlation model is the reduced number of MIMO channel model parameters (from $(MN)^2$ to $M^2 + N^2$). Moreover, the spatial correlation properties of the channel can be estimated separately at the transmitter and receiver ends. Note that the Kronecker model cannot be used for the accurate simulation of MIMO fading channels with single-bounce scattering because the Kronecker simplification in equation (4.53) removes any dependence between the DOD and the DOA which is a basic characteristic of MIMO channels with single-bounce scattering.

4.5.2 Weichselberger Model

Compared to the Kronecker model, the Weichselberger channel model removes the separability restriction and allows for coupling between AOD and AOA [226, 271]. This model was proposed based on channel measurements and has been verified by other researchers in indoor and outdoor-to-indoor communication scenarios [272–274].

The Weichselberger model represents the MIMO channel impulse response matrix as

$$\mathbf{H} = \mathbf{U}_{RX}(\tilde{\mathbf{\Omega}}_W \odot \mathbf{G})\mathbf{U}_{TX}^T, \tag{4.55}$$

where $\odot$ denotes the Schur-Hadamard product, $\mathbf{G}$ is an $N \times M$ matrix with i.i.d. zero-mean and unit-variance complex Gaussian elements, and $\mathbf{U}_{RX}$ and $\mathbf{U}_{TX}$ are the receive and transmit eigenbasis given by the eigenvalue decomposition of $\mathbf{R}_{RX}$ and $\mathbf{R}_{TX}$ respectively, i.e.,

$$\begin{aligned} \mathbf{R}_{RX} &= \mathbf{U}_{RX}\mathbf{\Lambda}_{RX}\mathbf{U}_{RX}^H, \\ \mathbf{R}_{TX} &= \mathbf{U}_{TX}\mathbf{\Lambda}_{TX}\mathbf{U}_{TX}^H. \end{aligned} \tag{4.56}$$

Moreover, in equation (4.55) $\tilde{\mathbf{\Omega}}_W$ denotes the element-wise square root of the $N \times M$ coupling matrix $\mathbf{\Omega}_W$ which has non-negative and real-valued elements. The coupling matrix $\mathbf{\Omega}_W$ can be obtained by

$$\mathbf{\Omega}_W = E\left\{(\mathbf{U}_{RX}^H\mathbf{H}\mathbf{U}_{TX}^*) \odot (\mathbf{U}_{RX}^T\mathbf{H}^*\mathbf{U}_{TX})\right\}. \tag{4.57}$$

Overall the Weichselberger channel model provides a good trade off between simplicity of the model and the correct characterization of correlation between TX and TX antenna elements [226].

4.5.3 Virtual Channel Representation Model

In contrast to the Weichselberger model that expresses the MIMO fading channel in the eigenspace, the virtual channel representation (VCR) model [227] uses the beamspace in channel modeling. Inspired by the double-directional channel representation [240], the VCR model partitions the AOD and AOA angular range into discrete virtual angles. The number of virtual angles is determined by number of antenna elements at each end. For the M-element antenna array at the transmitter, M virtual angles are selected in such a way that the M steering vectors are orthonormal. At the transmitter side, these orthonormal steering vectors form the $M \times M$ unitary steering matrix $\mathbf{A}_{TX}$. For the N-element antenna array at the receiver side, an $N \times N$ unitary steering matrix $\mathbf{A}_{RX}$ is built in a similar way. The VCR model is expressed as [227]

$$\mathbf{H} = \mathbf{A}_{RX}(\tilde{\mathbf{\Omega}}_V \odot \mathbf{G})\mathbf{A}_{TX}^H, \tag{4.58}$$

where the $N \times M$ matrix $\tilde{\mathbf{\Omega}}_V$ represents the amplitude coupling between the corresponding virtual angles of the TX and RX ends. Here $\tilde{\mathbf{\Omega}}_V$ is the element-wise square root of the $N \times M$ power coupling matrix $\mathbf{\Omega}_V$ whose real and non-negative elements determine the power coupling between the TX and the RX virtual directions. Further, the power coupling matrix for the VCR model can be expressed as

$$\mathbf{\Omega}_V = E\left\{(\mathbf{A}_{RX}^H \mathbf{H} \mathbf{A}_{TX}^*) \odot (\mathbf{A}_{RX}^T \mathbf{H}^* \mathbf{A}_{TX})\right\}. \tag{4.59}$$

The VCR channel model can be easily interpreted based on the geometrical propagation. There is a direct link between the rank of $\tilde{\mathbf{\Omega}}_V$ and the channel capacity [227]. Also, the diversity level of each sub-channel is directly related to the number of virtual RX angles that couple with each virtual TX angle [227].

4.6 Hardware Simulation of the Analytical Models

The above analytical MIMO channel models can be simulated by introducing specific correlation between zero-mean i.i.d. Gaussian samples. These i.i.d. fading samples can be generated either with the fading simulators presented in the previous chapters or with the MIMO fading channel simulator presented in Section 4.4. To generate i.i.d. Gaussian samples with the fading simulator presented in Section 4.4, we can distribute the scatterers isotropically around the receiver.

Assuming that $\mathbf{G}$ is the $N \times M$ matrix with zero-mean i.i.d. Gaussian samples (i.e., spatially independent rich-scattering MIMO flat fading channel), the Kronecker MIMO fading channel model can be expressed as

$$\mathbf{H} = \mathbf{UGV}, \tag{4.60}$$

where $\mathbf{U} = \mathbf{R}_{RX}^{1/2}$ and $\mathbf{V} = (\mathbf{R}_{TX}^{1/2})^T$ (see equation (4.54)). Moreover, the Weichselberger model and the VCR model can be written as

$$\mathbf{H} = \mathbf{U}(\mathbf{W} \odot \mathbf{G})\mathbf{V}, \tag{4.61}$$

where for the Weichselberger model (see equation 4.55)

$$\begin{array}{l} \mathbf{U} = \mathbf{U}_{RX}, \\ \mathbf{V} = \mathbf{U}_{TX}^{T}, \\ \mathbf{W} = \tilde{\mathbf{\Omega}}_{W}, \end{array} \tag{4.62}$$

and for the VCR model (see equation 4.58)

$$\begin{array}{l} \mathbf{U} = \mathbf{A}_{RX}, \\ \mathbf{V} = \mathbf{A}_{TX}^{H}, \\ \mathbf{W} = \tilde{\mathbf{\Omega}}_{V}. \end{array} \tag{4.63}$$

Also, in addition to the above three models, we can express the finite scatterer channel model [210] and the maximum entropy channel model [229] with equation (4.61).

4.6.1 Sample Generation

Here we want to implement a pipelined architecture for the efficient calculation of equations (4.60) and (4.61). The implemented architecture needs to 1) receive the elements of matrix $\mathbf{G}$ from the previous stage, which is the MIMO fading sample generator that was introduced in Section 4.4.1, 2) perform the matrix operations of either equation (4.60) or equation (4.61), and 3) pass the generated samples to the next stage for interpolation.

As mentioned in the previous section, the fading sample generator in Figure 4.8 generates the difference between the low-frequency fading samples, i.e.,

$$\mathbf{D}[z] = \left[d_{lp}[z]\right] = \mathbf{G}[z+1] - \mathbf{G}[z]. \tag{4.64}$$

according to equation (4.48). The difference samples are then passed to the linear interpolators for up-sampling. To introduce correlation between the i.i.d. fading samples, we can perform the matrix calculations (4.60) and (4.61) on the high-frequency samples. Alternatively, to implement an efficient and compact hardware we can perform the matrix

operations on the low-frequency fading samples and later up-sample the resulting streams with appropriate interpolators. To use linear interpolators, we need to generate the difference between the fading samples,

$$\mathbf{E}[z] = \left[e_{lp}[z]\right] = \mathbf{H}[z+1] - \mathbf{H}[z]. \tag{4.65}$$

Moreover, due to the linearity of the basic matrix multiplication we can verify that

$$\begin{aligned} \mathbf{UG}[z+1]\mathbf{V} - \mathbf{UG}[z]\mathbf{V} &= \mathbf{U}(\mathbf{G}[z+1] - \mathbf{G}[z])\mathbf{V} \\ &= \mathbf{UD}[z]\mathbf{V}, \end{aligned} \tag{4.66}$$

and since the Schur-Hadamard product is a linear operation too, we can write

$$\begin{aligned} \mathbf{U}(\mathbf{W} \odot \mathbf{G}[z+1])\mathbf{V} - \mathbf{U}(\mathbf{W} \odot \mathbf{G}[z])\mathbf{V} &= \mathbf{U}(\mathbf{W} \odot (\mathbf{G}[z+1] - \mathbf{G}[z]))\mathbf{V} \\ &= \mathbf{U}(\mathbf{W} \odot \mathbf{D}[z])\mathbf{V}. \end{aligned} \tag{4.67}$$

Equations 4.66 and 4.67 imply that instead of performing the matrix operations (4.60) and (4.61) on the original i.i.d. fading samples, we can use the difference samples. Therefore, the difference sample from the datapath in Figure 4.27 can be directly used for simulating analytical channel models. We will later increase the sample rate of individual streams using linear interpolation.

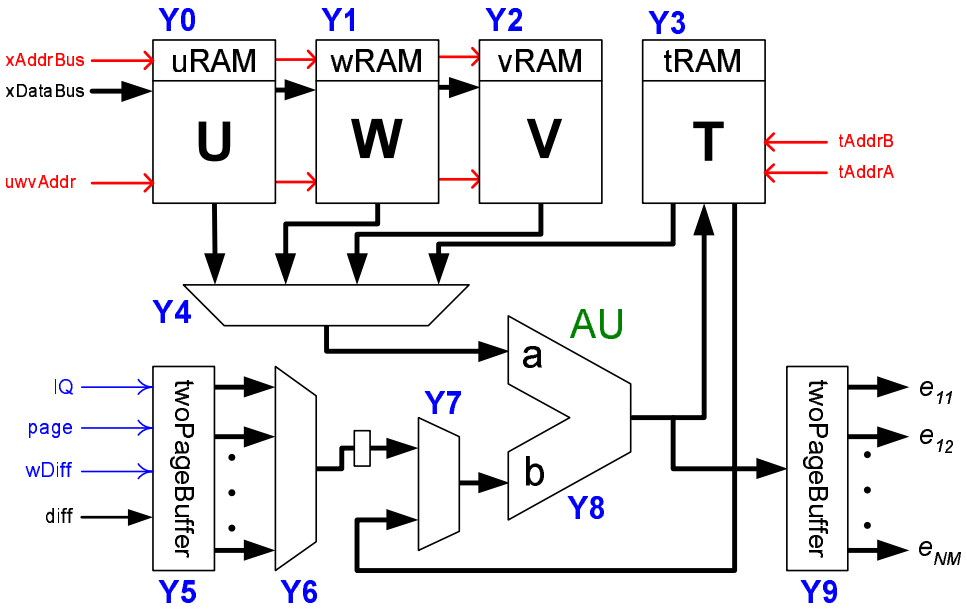


Figure 4.27: Datapath of the pipelined architecture for performing the matrix operations of equations (4.60) and (4.61).

Figure 4.27 shows the datapath of the implemented architecture for performing the matrix operations. For convenience of presentation, this architecture will be loosely referred to as the "matrix processor" henceforth. However, this architecture is not a true processor as the control unit does not read instructions from a program memory. Instead, the control unit is implemented using a set of low-level microinstructions for the elementary matrix operations including the basic matrix summation, matrix product, and the Schur-Hadamard matrix product. This architecture can be converted to a complex matrix processor by making specific changes to the control unit. However designing an efficient compiler that could exploit the pipelining capabilities of this processor is out of the scope of this thesis.

Instead of designing a true processor and performing the matrix operations with individual instructions, we developed specific subroutines of microinstructions to perform the basic matrix operations. These subroutines are developed to perform these matrix operations efficiently as they exploit the pipelining capabilities of the datapath in Figure 4.27. Moreover, the developed subroutines can be parameterized to perform the basic matrix operations for different matrix dimensions (M and N). Also, the control unit can be programmed to perform these matrix operations is any order.

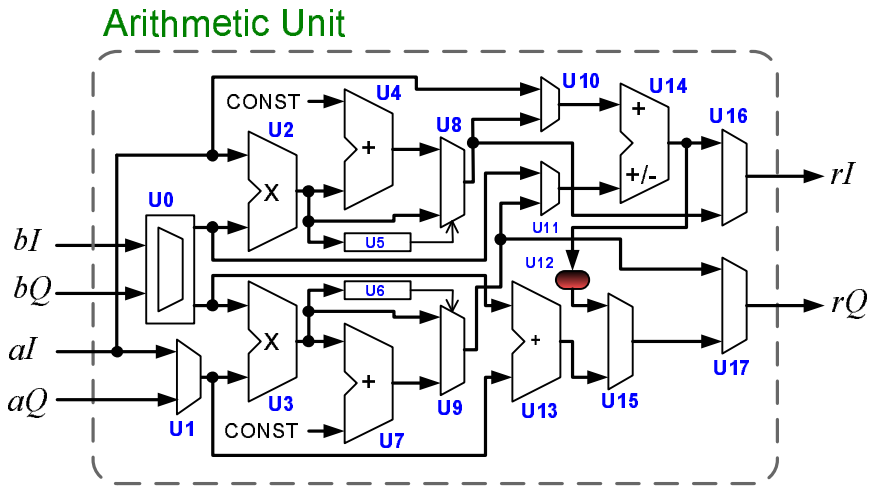


Figure 4.28: Datapath of the pipelined arithmetic unit for performing the basic complex operations.

The core of this matrix processor is an arithmetic unit (AU) that is designed for the basic complex arithmetic operations. Figure 4.28 shows the datapath of this AU that can calculate complex products, complex additions, and real-by-complex products. In other words, for the two complex inputs $aI + jaQ$ and $bI + jbQ$, the AU in Figure 4.28 can

calculate $rI + jrQ = (aI + jaQ) \times (bI + jbQ)$, $rI + jrQ = (aI + jaQ) + (bI + jbQ)$, and $rI + jrQ = aI \times (bI + jbQ)$.

The main components of this AU are the 18-bit multipliers `U2` and `U3` that operate on the quadrature parts. Moreover, the 36-bit output of the multiplier `U2` (`U3`) is rounded to the nearest 18-bit value by the modules `U4`, `U5`, and `U8` (`U6`, `U7`, and `U9`). As we will see later, the rounding operation is necessary to reduce the the quantization noise and stabilize the linear interpolators.

When the addition operation is selected, the in-phase parts of the complex inputs (i.e., aI and bI) are routed to the adder/subtracter `U14` either through the multiplexer `U10`, or through the multiplexers `U0` and `U11`. Also, the quadrature parts of the complex inputs (i.e., aQ and bQ) are routed to the adder `U13` through the multiplexers `U0` and `U1`. The output samples $rI = aI + bI$ and $rQ = aQ + bQ$ are then sent to the output via multiplexers `U16` and `U17`. All of the above operations are pipelined and when the pipeline is full (for complex addition), the AU can perform one complex addition per clock cycle.

For the real-by-complex product, the real input aI is passed to the multipliers `U2` and `U3` where it is multiplied by the bI and bQ respectively. After rounding, the results $rI = aI \times bI$ and $rQ = aI \times bQ$ are passed to the output through multiplexers `U16` and `U17`. Note that the above operations are pipelined as well and when the pipeline (for real by complex product) is full, the AU needs only one clock cycle for every product.

In contrast to the other complex operations that can be performed in a single clock cycle, the complex product needs two clocks (considering a full pipeline). In the first cycle, the inputs aI and bQ are passed to the multiplier `U2` and the inputs aQ and bI are routed to the multiplier `U3`. After rounding, the results of these two products are passed to the adder/subtracter `U14` where the quadrature part of the result, $aI \times bQ + aQ \times bI$ is calculated and stored in the register `U12`. In the next cycle, the inputs aI and bI are passed to the multiplier `U2` and the inputs aQ and bQ are passed to the multiplier `U3`. The calculated products are then rounded and passed to the adder/subtracter `U14` and the in-phase part of the result, $aI \times bI - aQ \times bQ$, is calculated. Moreover, the in-phase and the quadrature parts of the result are passed to the output through the multiplexers `U16`, `U15`, and `U17`.

In the datapath shown in Figure 4.27, the three RAMs `uRAM`, `wRAM`, and `vRAM` are used to keep the elemements of the $\mathbf{U}$, $\mathbf{W}$, and $\mathbf{V}$ matrices respectively. These memories are dual-port RAMs that can be accessed and programmed externally through the address

bus `xAddrBus` and the data bus `xDataBus` for the real-time configurablity. Moreover, the dual-port memory `tRAM` is used as a register bank for holding the intermediate results. The control unit can read the elements of $\mathbf{U}$, $\mathbf{W}$, and $\mathbf{V}$ matrices using the address bus `uwvAddr`. Further, to access `tRAM`, the control unit uses the address lines `tAddrA` and `tAddrB`.

In Figure 4.27 the module `Y5` is implemented to interface the matrix processor to the fading generator module in Figure 4.8. This module keeps two copies (`page0` and `page1`) of the difference samples from the previous stage. The quadrature input data is presented in 32-bit format, 16 bits of which are used to present the in-phase (real) part and the remaining 16 bits are used for presenting the quadrature (imaginary) part. Before being passed to the AU, the input samples are converted to 36-bit variables (18 bits for the in-phase part and 18 bits for the quadrature part).

When the Kronecker channel model is selected, the control unit reads the complex elements of the input matrix $\mathbf{D}$ form `Y5` through the multiplexer `Y6` and passes them to the AU. To perform the matrix operations needed for the Kronecker channel model, the matrix processor starts by calculating

$$\mathbf{T}_1 = \mathbf{UD}, \tag{4.68}$$

where the elements of $\mathbf{U}$ are read from `uRAM`, and the elements of the temporary matrix $\mathbf{T}_1$ are written to `tRAM`. To increase the system throughput and efficient use of the pipelined datapath, the control unit of the matrix processor performs the matrix multiplication (4.68) in N steps (assuming that $\mathbf{U}$ is $N \times N$ and $\mathbf{D}$ is $N \times M$). More specifically, the equation

(4.68) is calculated as

$$\begin{aligned}
\mathbf{T}_1 &= \begin{pmatrix} u_{11} & u_{12} & \cdots & u_{1N} \\ u_{21} & u_{22} & \cdots & u_{2N} \\ \vdots & \vdots & \ddots & \vdots \\ u_{N1} & u_{N2} & \cdots & u_{NN} \end{pmatrix} \begin{pmatrix} d_{11} & d_{12} & \cdots & d_{1M} \\ d_{21} & d_{22} & \cdots & d_{2M} \\ \vdots & \vdots & \ddots & \vdots \\ d_{N1} & d_{N2} & \cdots & d_{NM} \end{pmatrix} \\
&= \begin{pmatrix} u_{11}d_{11} & u_{11}d_{12} & \cdots & u_{11}d_{1M} \\ u_{21}d_{11} & u_{21}d_{12} & \cdots & u_{21}d_{1M} \\ \vdots & \vdots & \ddots & \vdots \\ u_{N1}d_{11} & u_{N1}d_{12} & \cdots & u_{N1}d_{1M} \end{pmatrix} + \\
&\quad \begin{pmatrix} u_{12}d_{21} & u_{12}d_{22} & \cdots & u_{12}d_{2M} \\ u_{22}d_{21} & u_{22}d_{22} & \cdots & u_{22}d_{2M} \\ \vdots & \vdots & \ddots & \vdots \\ u_{N2}d_{21} & u_{N2}d_{22} & \cdots & u_{N2}d_{2M} \end{pmatrix} + \\
&\quad \cdots \\
&\quad \begin{pmatrix} u_{1N}d_{N1} & u_{1N}d_{N2} & \cdots & u_{1N}d_{NM} \\ u_{2N}d_{N1} & u_{2N}d_{N2} & \cdots & u_{2N}d_{NM} \\ \vdots & \vdots & \ddots & \vdots \\ u_{NN}d_{N1} & u_{NN}d_{N2} & \cdots & u_{NN}d_{NM} \end{pmatrix}.
\end{aligned} \tag{4.69}$$

In the first step, the first column of $\mathbf{U}$ is multiplied by the first row of $\mathbf{D}$ and the results are stored in the temporary memory `tRAM`. In the second step, the second column of $\mathbf{U}$ is multiplied by the second row of $\mathbf{D}$ and so on, until the N^{th} column of $\mathbf{U}$ is multiplied by the N^{th} row of $\mathbf{D}$. Then the matrix processor accumulates the N sub-product matrices to generate the multiplication result. The reason for this out-of-order processing for complex matrix multiplication is that the AU has different latencies for complex addition and complex multiplication operations. By sorting the multiplication and addition operations we can increase the throughput of AU and avoid unnecessary bubbles in the pipeline.

Further, for the Kronecker model, the calculated temporary matrix $\mathbf{T}_1$ is used to generate the difference samples

$$\mathbf{E} = \mathbf{T}_1\mathbf{V}, \tag{4.70}$$

where the elements of $\mathbf{V}$ are read from `vRAM`, the elements of $\mathbf{T}_1$ are read from `tRAM`, and the elements of the output matrix $\mathbf{E}$ are written to the two page buffer `Y9` to be passed to the interpolators. This matrix multiplication is performed similar to the previous matrix multiplication. This operation is first broken into M multiplication operations between the columns of $\mathbf{T}_1$ and the rows of the matrix $\mathbf{V}$. Then the M sub-product matrices are

accumulated to generate the output matrix $\mathbf{E}$.

Simulating the Weichselberger model and the VCR model requires an additional Schur-Hadamard (or element-wise) product. More specifically, the simulator first calculates

$$\begin{aligned}
\mathbf{T}_2 &= \mathbf{W} \odot \mathbf{D} \\
&= \begin{pmatrix} w_{11} & w_{12} & \cdots & w_{1M} \\ w_{21} & w_{22} & \cdots & w_{2M} \\ \vdots & \vdots & \ddots & \vdots \\ w_{N1} & w_{N2} & \cdots & w_{NM} \end{pmatrix} \odot \begin{pmatrix} d_{11} & d_{12} & \cdots & d_{1M} \\ d_{21} & d_{22} & \cdots & d_{2M} \\ \vdots & \vdots & \ddots & \vdots \\ d_{N1} & d_{N2} & \cdots & d_{NM} \end{pmatrix} \\
&= \begin{pmatrix} w_{11}d_{12} & w_{12}d_{12} & \cdots & w_{1M}d_{1M} \\ w_{21}d_{21} & w_{22}d_{22} & \cdots & w_{2M}d_{2M} \\ \vdots & \vdots & \ddots & \vdots \\ w_{N1}d_{N1} & w_{N2}d_{N2} & \cdots & w_{NM}d_{1M} \end{pmatrix}.
\end{aligned} \tag{4.71}$$

Note that the elements of the matrix $\mathbf{W}$ are real-valued and therefore the AU can calculate each of the real-by-complex products in a single clock cycle (when the pipeline is full). After calculating $\mathbf{T}_2$, the matrix processor proceeds with calculating $\mathbf{T}_3 = \mathbf{U}\mathbf{T}_2$ and $\mathbf{E} = \mathbf{T}_3\mathbf{V}$. These matrix multiplications are also performed with the same matrix multiplication subroutine that breaks the calculations into N (or M) stages for effective use of the pipelined datapath.

4.6.2 Interpolation

For hardware simulation of the analytical MIMO channel models, we used a similar linear interpolator to the one presented in Section 4.4.2. In contrast to simulating physical models which only require superposition of zero-mean waves, simulating analytical MIMO channel models requires multiple fixed-point multiplication and addition operations.

The number of bits generated by a multiplier is the sum of the number of bits of the input operands. However, implementing a significantly wider datapath for the multiplication results was mainly avoided here to reduce the hardware complexity. Instead some of the output bits were trimmed. This, however, can increase the quantization noise that can affect the accuracy of the implemented fading simulator.

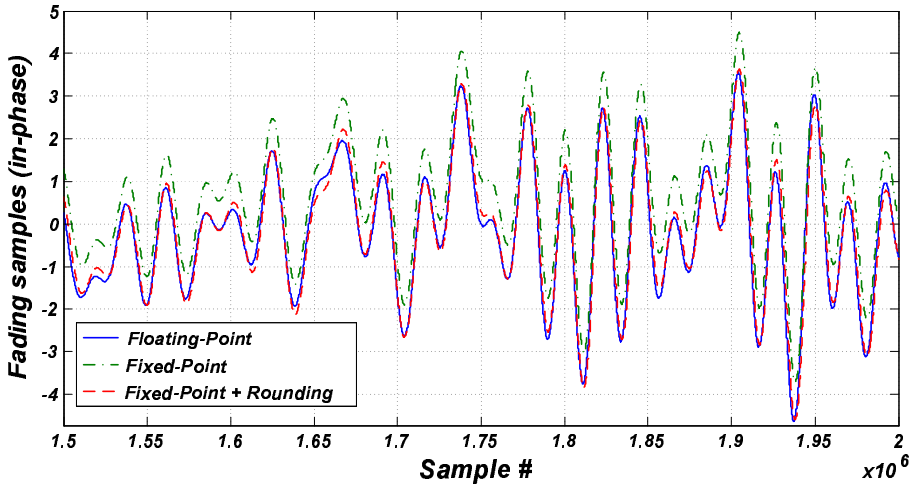


Figure 4.29: The effect of rounding on the interpolator output.

As an example, Figure 4.29 shows a drift in the interpolated fixed-point samples compared to the floating-point results for a Weichselberger channel model. This drift, due to quantization noise, has been accumulated over a large number of samples. Notice that the linear interpolator in equation (4.47) is implemented with a running summation which resembles a lowpass infinite impulse response (IIR) filter with the frequency response $H_i(e^{j2\pi f}) = 1/(1 - e^{-j2\pi f})$. This IIR filter accumulates the input samples (extremely large gain at zero frequency) and therefore any DC component in the input will add up over time.

The interpolator presented in Section 4.4.2 can be used effectively for interpolating samples generated by the fading simulator datapath in Figure 4.8, because the difference signal has no DC components. In other words, the impulse response of the difference block, $h_d[n] = \delta[n] - \delta[n-1]$ (see equation (4.48)), in frequency domain is $H_d(e^{j2\pi f}) = 1 - e^{-j2\pi f}$ which has no output at zero frequency, i.e., $H_d(e^{j2\pi 0}) = 0$. On the other hand, the quantization noise added by the matrix processor is not necessarily DC-free.

The rounding technique used in the matrix processor has a significant effect on the DC component of the added quantization noise. For example, rounding down to smaller fixed-point values always results in a positive residue (or quantization noise). Simply ignoring the extra least significant bits after a multiplication could be interpreted as rounding down to smaller fixed-point values.

An effective method is rounding the multiplication results to the nearest fixed-point samples. Rounding the multiplication results can significantly reduce the DC component of

the quantization noise, since the multiplication results are rounded up "hopefully" as many times as they are rounded down. Figure 4.29 also shows the improvement in removing the drift and the effectiveness of the rounding technique for the same Weichselberger fading channel model. As this figure shows, no clear DC bias (or drift) can be observed after two million samples when rounding the multiplier outputs.

We added the rounding functionality to the matrix processor and simulated different channel models and channel conditions. The fixed-point bit-true model was verified based on the computer simulation and we proceeded with implementing our fading simulator on a GVA-290 FPGA prototyping platform [161]. The fading simulator was set to generate 50 million samples per second. The generated fading samples seemed to have the required statistical properties (mean and variance). However, the mean and variance of the generated samples started to deviate from theoretical results after a few minutes. The speed and the direction of deviation for quadrature components of each fading sample were related to the simulated scenario and specific values of the $\mathbf{U}$, $\mathbf{W}$, and $\mathbf{V}$ matrices. In one case, a 12% change in the signal variance happened after five minutes of sample generation, or $5 \times 60 \times 50 \times 10^6 = 1.5 \times 10^{10}$ samples. This deviation was happening very slowly in time and could not be predicted with computer simulations due to the limited computer simulation speed.

We later found that the deviation is caused by accumulation of the DC component of quantization noise in the interpolators. More specifically, the assumption that samples are rounded up as many times as they are rounded down on average is not accurate. Even the slightest DC components in the quantization noise add up in the interpolator and can render it unstable after rather long periods of time.

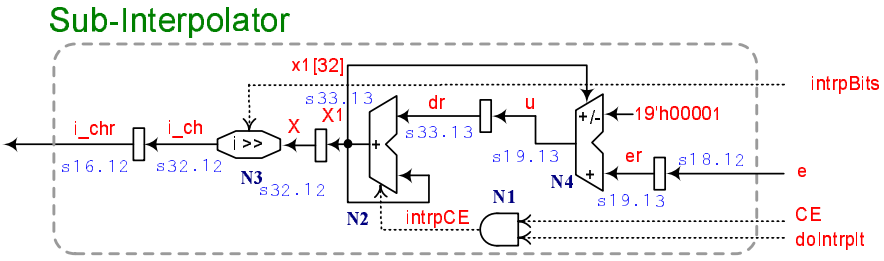


Figure 4.30: Datapath of the implemented sub-interpolator with DC cancellation.

To solve this problem, we modified the interpolator to remove the DC component from the output. The DC cancellation should not affect the overall low-frequency response of the

interpolator since the fading samples can have close-to-DC frequency components. However the exact DC components of the fading samples, e.g., contributions of line-of-sight path when the angle of arrival is exactly 90 degrees, would be affected.

Figure 4.30 shows the datapath of the modified interpolator. Only the in-phase branch is shown in this figure. Also, the two-page buffers are not implemented with the interpolators since they have been moved to the matrix processor. The main modification in the new interpolator is the addition of negative feedback to the accumulator `N2` (or integrator). In this feedback loop, the sign of the accumulator output (from the most significant bit) is fed back and subtracted from half of the least significant bit of the input. More specifically, 12-bits have been used to represent the fraction part of the difference input $e[n]$ as shown in Figure 4.30. The magnitude of the feedback signal is limited to half of the least significant bit of the input. Notice that the accumulator output is later divided by the interpolation factor in the barrel shifter `N3`. This would further reduce the relative amount of feedback to the output signal.

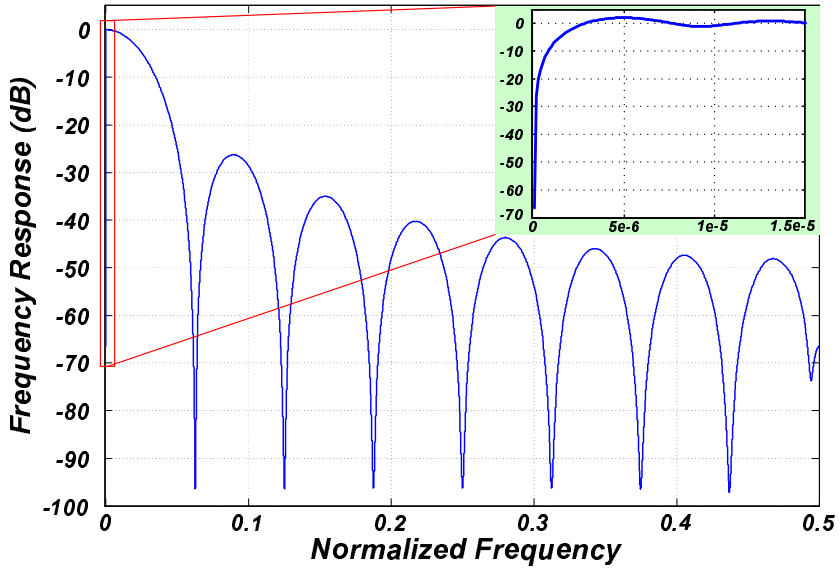


Figure 4.31: Frequency response of the implemented linear interpolator.

Figure 4.31 shows the frequency response of the modified linear interpolator which includes the modified integrator (`N2`, `N3`, and `N4`, in Figure 4.30) and the differentiator (`U18` and `U19` in Figure 4.8). In this figure, the interpolation factor is set to $I = 16$. As this figure shows, the addition of the negative feedback adds a sharp notch at the DC frequency. Particularly, the DC frequency has been attenuated more than 60 dB while the frequency

response goes back to 0 dB at 1.5×10^{-5} Hz. The other effect of the added feedback is reduced attenuation at 0.5 radians per sample. However, the attenuation is still sufficient to reduce the unwanted frequency components. Therefore the implemented circuit can be used effectively for interpolating the fading samples.

4.6.3 Simulation Results

To estimate the bit error rate (BER) performance of a communication system with the Monte Carlo (MC) simulation method, we have to measure the BER over a large number of independent problem instances [93]. Simulation of additive white Gaussian noise (AWGN) channels is straightforward as the system performance is averaged over a large number of independent instances of noise and data.

Simulation of time varying fading channels, on the other hand, requires significantly longer simulation times due to the dependence between the channel instances. To accurately estimate the BER performance of a communication system over a time-varying fading channel, the error performance needs to be averaged not only on independent instances of noise and data, but also on the fading channel samples. One solution is to estimate the error rate performance on a quasi-static channel in which fading samples are assumed to be constant over one frame of data. However, this assumption is not accurate particularly for fast fading scenarios and provides unrealistic results. Moreover, several communication blocks (such as channel codes, interleavers, channel estimation and equalization, timing estimation, frequency offset estimation and compensation, and automatic gain control) need to be verified on time-varying channels for different Doppler rates. Thus, an accurate performance estimation needs to be performed over a long period of time (compared to the channel coherence time $T_c \approx 0.423/f_D$ [20]), and a large number of independent instances of noise and data.

Due to the computational complexity of the fixed-point simulation, measuring the BER performance of a MIMO communication systems with our bit-true model on time-varying channels was very slow. We used our hardware simulation platform to measure the BER performance of a 2×2 MIMO system. The hardware simulation platform and the implemented MIMO system are discussed in Section 4.7. The transmitter under test utilizes an extended Golay channel code, a length-16383 interleaver, a 4-QAM modulator, and an ML decoder at the receiver. Also, perfect channel state information is assumed to be available to the receiver.

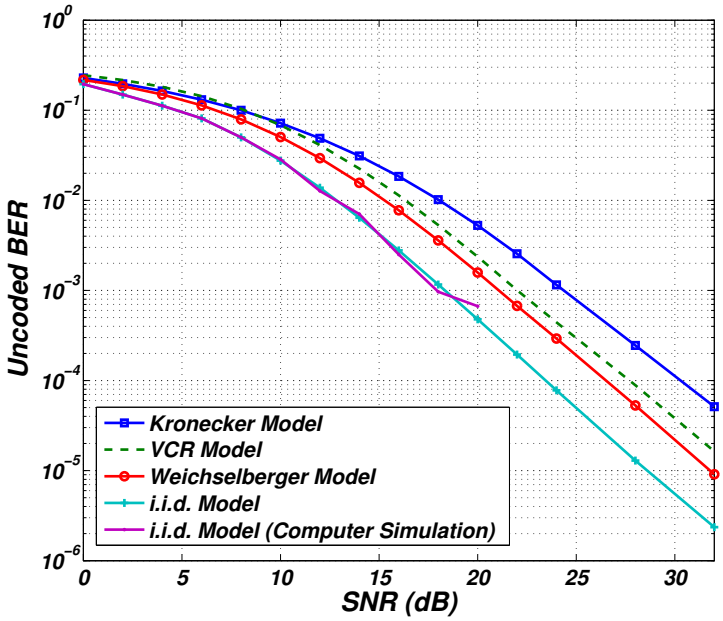


Figure 4.32: Uncoded bit error rate performance of a 2×2 MIMO system measured using the FPGA-based fading simulator for different channel models.

Figure 4.32 shows the uncoded BER performance of this 2×2 MIMO system for different channel models. In this platform, the sample rate is set to 3.125 million samples per second or 12.5 mbps (the maximum speed supported by the ML detector), and the Doppler frequency is $f_D = 350$ Hz. To estimate each BER point at each signal to noise ratio (SNR), we measured the performance over at least 1023 seconds of signal transmission on the hardware platform, and when at least 100 errors where collected from the Golay decoder output.

For the Kronecker channel model, the $\mathbf{U}$ and $\mathbf{V}$ matrices are set to

$$\mathbf{U}_K = \begin{pmatrix} -0.2281 - j0.6045 & 0.0659 + j0.6270 \\ -0.8782 + j0.6279 & 0.1516 - j0.0198 \end{pmatrix},$$

$$\mathbf{V}_K = \begin{pmatrix} 0.1914 - j0.3442 & -0.1091 - j0.0796 \\ 0.1021 + j1.2781 & 0.4248 + j0.0667 \end{pmatrix}.$$

Also, for the Weichselberger channel model, the values

$$\mathbf{U}_W = \begin{pmatrix} 0.0713 - j0.3103 & 0.2119 - j0.9238 \\ -0.9478 & 0.3184 \end{pmatrix},$$

$$\mathbf{W}_W = \begin{pmatrix} 0.3176 & 0.6355 \\ 0.9077 & 1.6340 \end{pmatrix},$$

$$\mathbf{V}_W = \begin{pmatrix} 0.8012 + j0.5202 & -0.2468 - j0.1602 \\ 0.2941 & 0.9556 \end{pmatrix},$$

are used in the simulator. Moreover, for the VCR channel models we set these matrices to

$$\mathbf{U}_V = \begin{pmatrix} 0.7679 + j0.0821 & -0.6316 - j0.0676 \\ 0.6352 & 0.7723 \end{pmatrix},$$
$$\mathbf{W}_V = \begin{pmatrix} 1.6356 & 0.4370 \\ 1.0452 & 0.2035 \end{pmatrix},$$
$$\mathbf{V}_V = \begin{pmatrix} 0.2519 + j0.7142 & 0.6530 \\ -0.2172 - j0.6158 & 0.7573 \end{pmatrix}.$$

The above values are chosen randomly to represent three different communication scenarios. Figure 4.32 also plots the BER performance of the 2×2 MIMO system under the i.i.d. channel model (independent fading). The floating-point computer simulation results for the i.i.d. channel model are also plotted in Figure 4.32. As this figure shows, the hardware simulation results follow the floating-point simulation results accurately which verifies the accuracy of our hardware fading simulation platform.

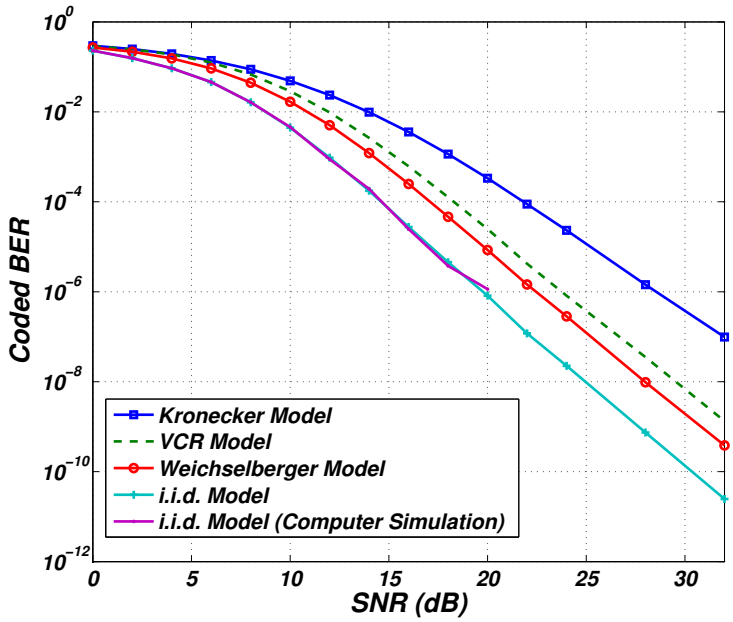


Figure 4.33: Coded bit error rate performance of a 2×2 MIMO system measured using the FPGA-based fading simulator for different channel models.

Figure 4.33 plots the coded BER performance of the above MIMO system. Also, floating-point computer simulation of the i.i.d. channel model is shown in this figure. Note that in Figure 4.33, the computer simulation results are given up to 20 dB SNR. This is due to the great computational complexity of the accurate BER performance measurement. In

Figure 4.33 we can verify that the hardware generated BER results accurately match the computer-generated BER performance.

4.6.4 Hardware Implementation Results

We implemented our matrix processor for simulating analytical MIMO channel models. Table 4.2 summarizes the synthesis results of 2×2 and 4×4 matrix processors on Xilinx Virtex-5 XCVLX110-3 and Altera Stratix III EP3SE50F780I4 FPGAs.

We did not use the on-chip block memories for implementing the 2×2 matrix processors because of the small memory sizes (we used the slice registers), while for the larger 4×4 matrix processor, we used the available 36-Kbit block memories to store $\mathbf{U}$, $\mathbf{W}$, $\mathbf{V}$, and $\mathbf{T}$ matrices. As Table 4.2 shows, when implemented on a XC5VLX110, the 4×4 matrix processor occupies only 1.8% of the configurable slices, two multipliers (DSP48E), and four 36-Kbit block memories and can operate at up to 234.1 MHz. When implemented on a EP3SE50, the 4×4 matrix processor occupies 1.8% of the adaptive look-up tables (ALUTs) and two multipliers and can operate at up to 185.8 MHz. Note that the higher efficiency in implementation on the Altera Stratix III device is due to the higher flexibility in block memory allocation.

Table 4.2 also summarizes the synthesis results of the implementation of the sub-interpolator on the above devices. As shown in this table, the sub-interpolator can operate at up to 448.7 MHz and 350 MHz when implemented on XC5VLX110 and EP3SE50 FPGAs respectively. Notice that the maximum speed of the sub-interpolator corresponds to the maximum sample generation rate of the fading simulator.

Table 4.2: Comparison of Implementation Results

Design	2×2 **MP**[a]	4×4 **MP**	4×4 **MP**	**SI**[b]	**SI**
Device	XC5VLX110	XC5VLX110	EP3SE50	XC5VLX110	EP3SE50
Max. Clock	234.9 MHz	234.1 MHz	185.8 MHz	448.7 MHz	350 MHz
Slices/ALUTs	2016 (2.9%)	1212 (1.8%)	711 (1.8%)	96 (0.1%)	111 (0.3%)
18 × 18 MULTs	2 (3.1%)	2 (3.1%)	2 (0.5%)	0	0
BRAM bits	0	147456 (3.1%)	6336 (0.1%)	0	0

[a] Matrix Processor
[b] Sub-Interpolator

4.7 Implemented MIMO Syetem

To verify our design, we implemented a fading simulation platform for 2×2 MIMO systems. The implemented platform could also simulate 4-path single-input single-output (SISO) channels.

The implemented MIMO fading channel simulator can simulate single- and double-scattering MIMO flat-fading channel models as well as i.i.d., Kronecker, Weichselberger, and VCR analytical channel models.

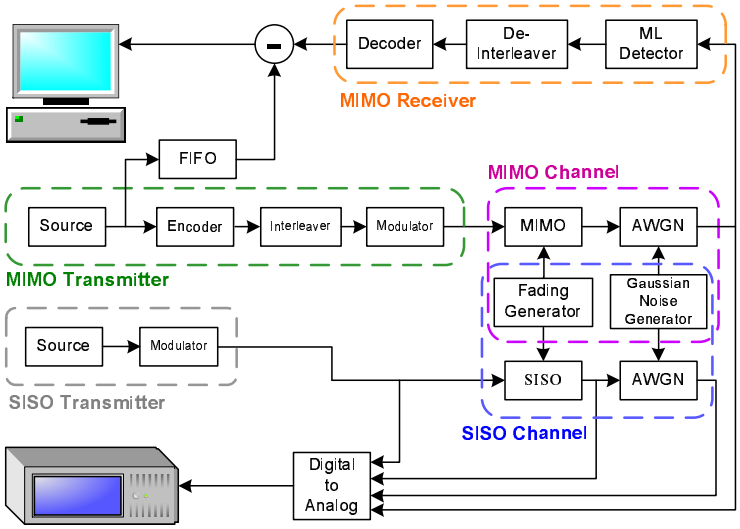


Figure 4.34: Block diagram of the implemented fading simulation platform.

Figure 4.34 shows the block diagram of the implemented fading simulation platform. On this platform, the fading simulator can be configured to simulate both single- and multiple-antenna systems. We implemented a 2×2 MIMO communication system for testing and verification of our fading simulator. In the implemented MIMO system, source bits are encoded using an extended Golay code and interleaved with a length-16383 interleaver. Then the interleaved bits are modulated to 4-QAM symbols and passed through the MIMO channel where they are affected by the multipath fading and corrupted with additive white Gaussian noise (AWGN). In the receiver, a maximum likelihood (ML) detector tries to estimate the transmitted bits. The ML detector can be configured in real-time to have access to complete and incomplete channel state information. After ML detection, the bit stream is de-interleaved, decoded, and compared to the transmitted bit stream.

To demonstrate the fading effects on the transmitted symbols, we also implemented a

single-antenna transmitter where the bits can be modulated using different schemes (BPSK, QPSK, 4-PAM, 4-QAM, 8-PSK, 16-PSK, 16-QAM, circular 8-QAM, and circular 16-QAM). As shown the in block diagram in Figure 4.34, the output of this transmitter can be passed to an oscilloscope through a digital-to-analog converter. Moreover, the faded samples (with and/or without noise) can be monitored on the oscilloscope as well. The output of the MIMO channel can also be shown on the oscilloscope screen.

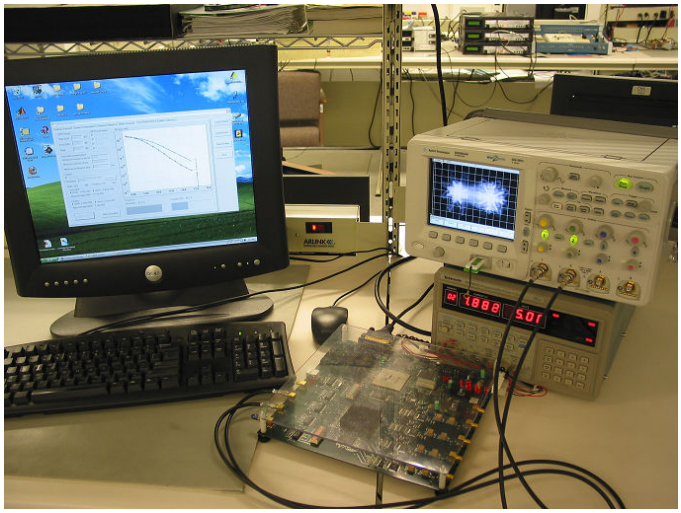


Figure 4.35: Fading simulation platform on a GVA-290 FPGA board.

We implemented our fading simulation platform on a GVA-290 FPGA board [161]. This board hosts two Xilinx Virtex-E XCV2000E FPGAs in addition to four digital-to-analog and four analog-to-digital converters. Figure 4.35 shows the picture of the implemented fading simulator on the GVA-290 board along with the power source, oscilloscope, and the control computer. The GVA-290 board is interfaced with the control computer through the parallel port.

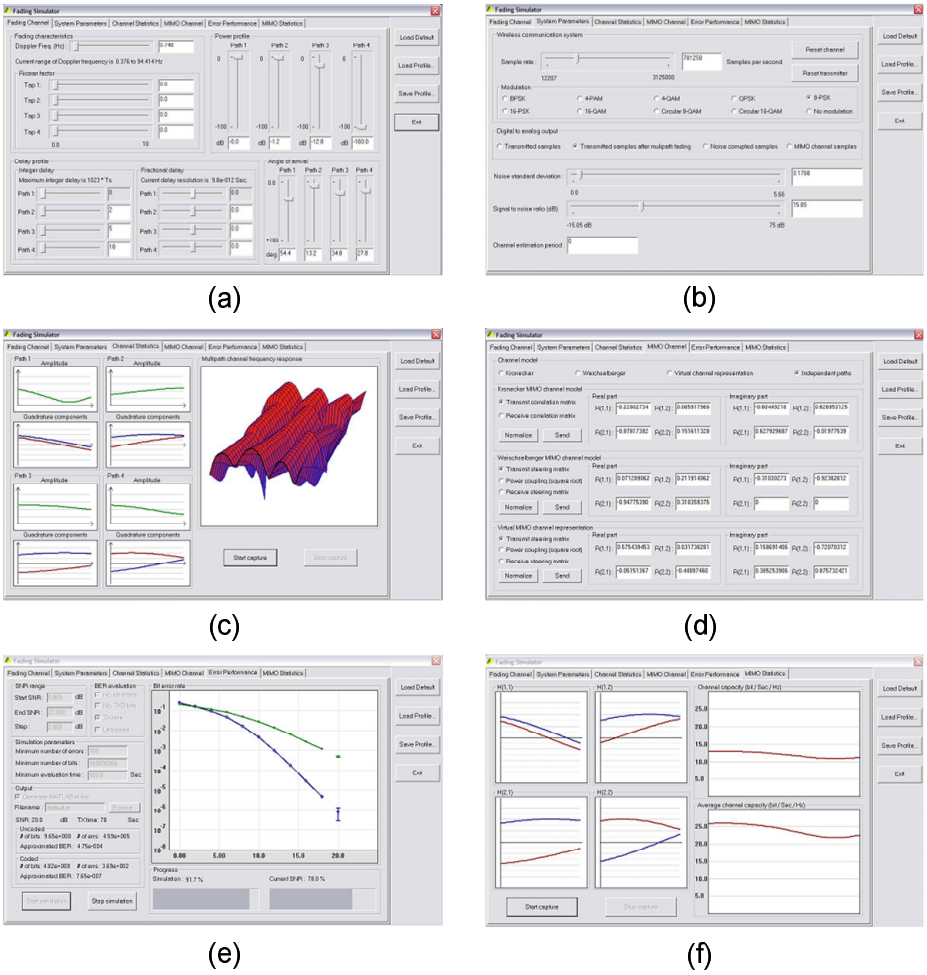


Figure 4.36: Graphical user interface of the implemented fading simulation platform.

We also developed a graphical user interface (GUI) through which we can configure our fading channel simulator in real-time. Figure 4.36 shows the six tabs of the fading simulator GUI. In the first tab in Figure 4.36 (a), we can set the Doppler frequency, SISO power profile, Rice factors, delay profile (integer and fractional), and the angles of arrival for different paths. In the second tab shown in Figure 4.36 (b), we can adjust the sample rate, change the sample modulation for the SISO system, select the digital-to-analog output, vary the signal-to-noise ratio, change the noise variance and also set the channel estimation pariod for the MIMO receiver. In the third tab shown in Figure 4.36 (c), we can monitor the complex fading samples of the SISO fading simulator along with the time-varying frequency response of the SISO fading channel. In the fourth tab shown in Figure 4.36 (d), the analytical MIMO channel model can be selected and the model parameters can be set and passed to the FPGA board. The fifth tab shown in Figure 4.36 (e) measures the bit error rate

performance of the implemented MIMO system using the Monte Carlo simulation method. The program can be configured to stop the simulation based on a combination of different criteria including the number of transmitted bits, number of errors, and transmission time. Moreover, the measured BER performance is also exported to MATLAB. Finally, in the sixth tab shown in Figure 4.36 (f), the complex MIMO fading channel samples are plotted. This tab also plots the instantaneous ergodic MIMO channel capacity and its time average for the selected signal to noise ratio.

Note that although the implemented fading simulator can simulate geometrical MIMO channel models, the capability of real-time configuration of the fading simulator for the geometrical models was not implemented in this version of the GUI software due to the time limitations. However, the fading simulator can be configured (using the synthesis tool) for the testing of the implemented MIMO system for different geometrical MIMO channel models. Moreover, the capability of real-time configuration for the geometrical models can be added to the next version of the GUI software.

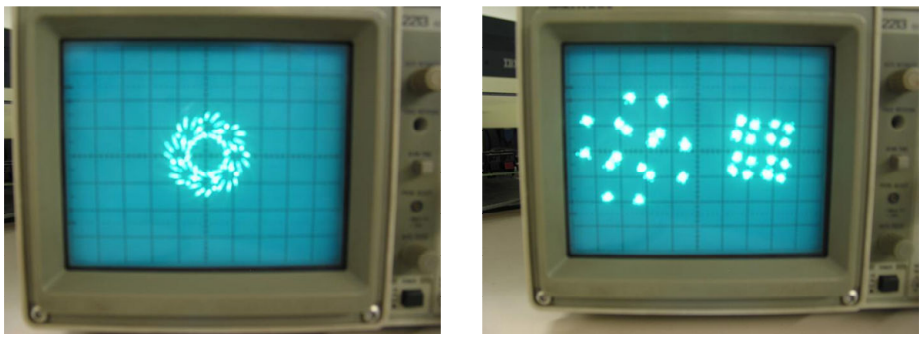


Figure 4.37: Pictures of the oscilloscope output for (a) the SISO and (b) the MIMO systems.

Figure 4.37 shows two outputs of the fading simulation platform on the oscilloscope screen. The Doppler frequency for these simulations was set to $f_D = 0.5$ Hz so that the changes in the scatter-plot could be easily followed. In Figure 4.37 (a), the oscilloscope screen shows the scatter plot of the noisy output of a SISO channel. In this picture, $8-$PSK modulated samples are passed though a two-path SISO fading channel and corrupted with AWGN. Figure 4.37 (b) shows the two noisy outputs of a the 2×2 MIMO channel where the transmitted bits are modulated with 4-QAM and the signal to noise ratio is 20 dB.

In the following parts we will briefly present some of the implemented blocks in the fading simulation platform.

4.7.1 4-QAM Modulator and MIMO Channel

In the implemented 2×2 MIMO system, 4-QAM modulated symbols are passed through the fading channel. The MIMO channel output samples can be expressed as

$$\begin{pmatrix} r_1 \\ r_2 \end{pmatrix} = \begin{pmatrix} h_{11} & h_{12} \\ h_{21} & h_{22} \end{pmatrix} \begin{pmatrix} s_1 \\ s_2 \end{pmatrix} + \begin{pmatrix} n_1 \\ n_2 \end{pmatrix}, \tag{4.72}$$

where h_{ij}, $i, j \in \{1, 2\}$, are the complex fading gains, s_i, $i \in \{1, 2\}$ are the transmitted 4-QAM symbols, and n_i, $i \in \{1, 2\}$ are the AWGN samples. Since the in-phase and the quadrature components of the 4-QAM symbols comprise only $+1$ and -1 values, the MIMO channel can be implemented without using multipliers. Decomposing the complex received samples to their in-phase and quadrature components, we can rewrite equation (4.72) as

$$\begin{pmatrix} r_1^I \\ r_2^I \\ r_1^Q \\ r_2^Q \end{pmatrix} = \begin{pmatrix} h_{11}^I & h_{12}^I & -h_{11}^Q & -h_{12}^Q \\ h_{21}^I & h_{22}^I & -h_{21}^Q & -h_{22}^Q \\ h_{11}^Q & h_{12}^Q & h_{11}^I & h_{12}^I \\ h_{21}^Q & h_{22}^Q & h_{21}^I & h_{22}^I \end{pmatrix} \begin{pmatrix} s_1^I \\ s_2^I \\ s_1^Q \\ s_2^Q \end{pmatrix} + \begin{pmatrix} n_1^I \\ n_2^I \\ n_1^Q \\ n_2^Q \end{pmatrix}, \tag{4.73}$$

where the superscripts $(\cdot)^I$ and $(\cdot)^Q$ denote the in-phase and quadrature components, respectively.

Using equation (4.73) designing a datapath for the MIMO fading channel is straightforward. As we will see in Section 4.7.2, the ML detector requires 16 clock cycles to detect the transmitted symbols. The ML detector is the bottleneck in the communication chain, limiting the transmission rate which limits the symbol rate to $F_{clk}/16$ symbols per second. Therefore we can use up to 16 clock cycles to calculate equation (4.73).

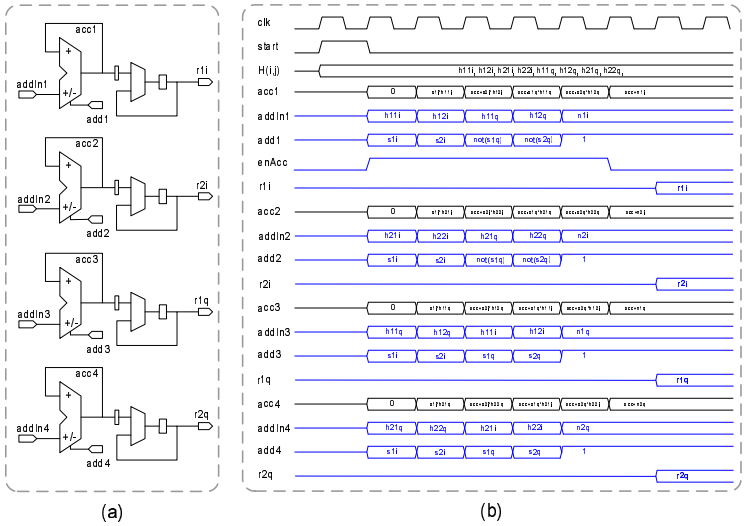


Figure 4.38: 4-QAM modulation and MIMO channel, (a) datapath and, (b) timing diagram.

Figure 4.38 shows the datapath and timing diagram of the implemented 4-QAM modulator and MIMO channel. As Figure 4.38 (a) shows, this block has been implemented with four accumulators with add/subtract inputs. The accumulator adds (subtracts) the input to (from) its current value if the `add` input is 1 (0). The quadrature components of the fading gains h_{ij}^I and h_{ij}^Q, the input samples s_i^I and s_i^Q, and the noise samples n_i^I and n_i^Q are assumed to be available and constant during each cycle. The fading gains are generated with our fading channel simulator. Also, to generate the AWGN samples we used the white Gaussian noise generator presented in [138].

The accumulators `acc1`, `acc2`, `acc3`, and `acc4` are reset at the beginning of each cycle (signaled by `start`). For the quadrature components of the input, the digital value 1 is assumed to represent arithmetic $+1$ and the digital value 0 represents arithmetic value -1. To calculate r_1^I, the accumulator `acc1` adds/subtracts the fading gains h_{11}^I, h_{12}^I, h_{11}^Q, h_{12}^Q according to s_1^I, s_2^I, $\texttt{not}(s_1^Q)$, $\texttt{not}(s_1^Q)$, respectively, (see equation (4.73)) and adds n_1^I. Finally, r_2^I, r_1^Q, and r_2^Q are calculated using a similar method as illustrated in Figure 4.38.

4.7.2 ML Detector

The ML detector performs an exhaustive search over all of the possible combinations of the transmitted signal to detect the received symbol vector [50]. Assuming that the transmitted symbols are modulated with 4-QAM scheme, for a 2×2 MIMO channel the ML detector can be expressed as

$$\hat{\mathbf{s}} = \min_{\mathbf{t} \in \{\pm 1 \pm 1j\}^2} ||\mathbf{r} - \mathbf{H}\mathbf{t}||^2, \tag{4.74}$$

where $\mathbf{r}$ denotes the received vector, $\mathbf{H}$ is the channel matrix, and $\mathbf{t}$ denotes the vector of tentative candidates. ML detection is a combinatorial optimization problem over all of the possible combinations of the transmitted signal. The cost function of this optimization

problem can be written as

$$
\begin{aligned}
c(\mathbf{t}) &= \|\mathbf{r}-\mathbf{H}\mathbf{t}\|^2, \\
&= \left\| \begin{pmatrix} r_1^I \\ r_2^I \\ r_1^Q \\ r_2^Q \end{pmatrix} - \begin{pmatrix} h_{11}^I & h_{12}^I & -h_{11}^Q & -h_{12}^Q \\ h_{21}^I & h_{22}^I & -h_{21}^Q & -h_{22}^Q \\ h_{11}^Q & h_{12}^Q & h_{11}^I & h_{12}^I \\ h_{21}^Q & h_{22}^Q & h_{21}^I & h_{22}^I \end{pmatrix} \begin{pmatrix} t_1^I \\ t_2^I \\ t_1^Q \\ t_2^Q \end{pmatrix} \right\|^2, \\
&= \left(r_1^I - t_1^I h_{11}^I - t_2^I h_{12}^I + t_1^Q h_{11}^Q + t_1^Q h_{12}^Q\right)^2 + \\
&\quad \left(r_2^I - t_1^I h_{21}^I - t_2^I h_{22}^I + t_1^Q h_{21}^Q + t_1^Q h_{22}^Q\right)^2 + \\
&\quad \left(r_1^Q - t_1^I h_{11}^Q - t_2^I h_{12}^Q - t_1^Q h_{11}^I - t_1^Q h_{12}^I\right)^2 + \\
&\quad \left(r_2^Q - t_1^I h_{21}^Q - t_2^I h_{22}^Q - t_1^Q h_{21}^I - t_1^Q h_{22}^I\right)^2. \qquad (4.75)
\end{aligned}
$$

Equation (4.75) can be used to implement the ML detector for the 2×2 MIMO system. Figure 4.39 shows the datapath of the implemented ML detector. In this figure, the section `Cost Function` calculates $c(\mathbf{t})$ according to equation (4.75). The quadrature components of the tentative samples, (i.e., t_1^I, t_1^Q, t_2^I, and t_2^Q) are modulated with the fading gains and subtracted from the input signal. For example, the first branch of the `Cost Function` section (including the adder/subtracters `U0`, `U4`, `U8`, `U12` and the multiplier `U16`) calculates $(r_1^I - t_1^I h_{11}^I - t_2^I h_{12}^I + t_1^Q h_{11}^Q + t_1^Q h_{12}^Q)^2$ (see equation (4.75)).

In Figure 4.39, the `FIFO` section delays each of the tentative symbols according to the latency of the `Cost Function` datapath so that the cost of each tentative symbol can be augmented with its corresponding symbol. Moreover, the section `Search`, finds the symbol with the minimum cost, which is the output of the ML detector.

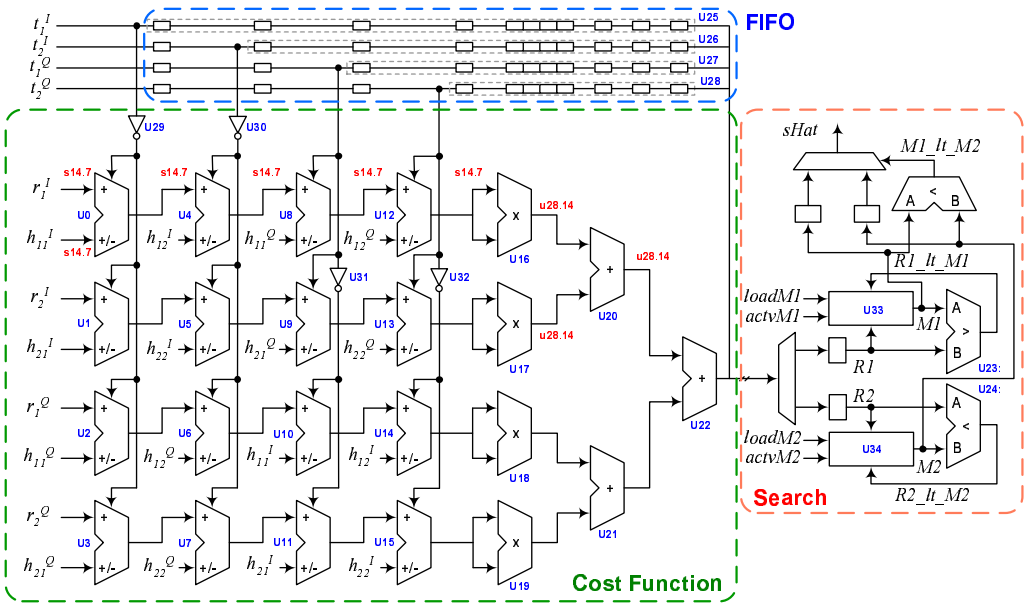


Figure 4.39: Datapath of the implemented exhaustive-search ML detector for the 2×2 MIMO system with 4-QAM modulated symbols.

We targeted Xilinx Virtex-E XCV2000E FPGAs for the implementation of our fading simulation platform since our available GVA-290 boards host this family of FPGAs. However, the Virtex-E family are relatively small FPGAs with no embedded multipliers. Hence we tried to optimize the size of our communication system with the minimum number of multipliers.

For a 2×2 MIMO system with 4-QAM modulated symbols, there are $4^2 = 16$ tentative symbols in the search space. According to equation (4.75), four multiplications are required for calculating the cost of each of the tentative symbols. Due to the size constraints, we used only four multipliers for the calculation of the cost function and shared the pipelined datapath for calculating the 16 costs. The 16 clock cycle latency of the ML detector is the bottleneck that limits the symbol rate of the MIMO communication system to $F_{clk}/16$.

Notice that three comparators are used in the `Search` section in Figure 4.39 for finding the tentative symbol with the minimum cost. It was due to the one clock cycle latency of the comparators that the sequence of the costs of tentative symbols was divided into two sub-streams. In Figure 4.39, the minimum costs of the two sub-streams (along with the tentative symbols corresponding to the minimum costs) are stored in the `M1` and `M2` registers. The final ML solution, `sHat`, is picked based on the minimum cost by comparing the final values of `M1` and `M2`.

4.7.3 Interleaver and De-interleaver

Errors in the wireless communication tend to happen in blocks when the signal experiences deep fades. However, a burst of errors can be overwhelming for the channel code (i.e., the error control code) that can only correct a certain number of errors in a block of data. This problem can be alleviated by randomizing the errors in a block of data using interleavers [275, 276]. Interleaving spreads the transmitted data over time, which is called time diversity, without adding any overhead.

Interleavers can be classified into block, convolutional, and pseudo-random interleavers. In block interleavers, coded bits are first written in row format in a matrix and then read in column format. Convolutional interleavers break the stream of coded bits into several sub-streams and delay each sub-stream for a certain amount of time. A pseudo-random interleaver is a variation of a block interleaver where coded bits are written linearly into a memory and read out randomly based on a pseudo-random sequence. For our MIMO communication system, we implemented a pseudo-random interleaver of length 16383.

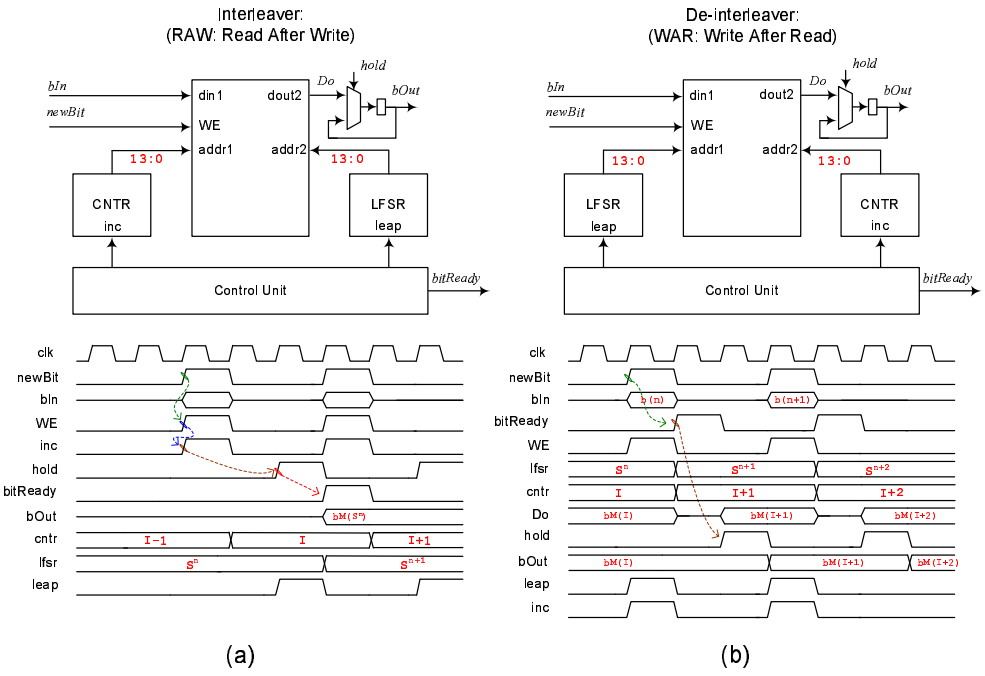


Figure 4.40: Datapath of the implemented (a) interleaver, and (b) de-interleaver with the corresponding timing diagrams.

Figure 4.40 shows the datapath and the timing diagram of the implemented pseudo-random interleaver and de-interleaver. In the interleaver, a 14-bit counter is used to write

the coded input bits into a dual-port 16384×1 memory. This counter counts linearly from 1 to 16383 and goes back to 1. At the output, a 14-bit linear feedback shift register (LFSR) is used to read out the coded bits randomly from the memory. Notice that the counter does not generate 0. It is because 0 is not among the values that are generated by a LFSR, hence the interleaver length is $16383 = 16384 - 1$. The reverse operation happens in the de-interleaver where the received bits are written randomly into a dual-port memory using the same pseudo-random sequence and later read out using a counter that counts from 1 to 16383.

When a new bit is passed to the interleaver (indicated by the `newBit` signal), it is written into the memory location addressed by the counter `cntr` and the counter is incremented for the next cycle. Then an output bit is read out from the location determined by the current state of the LFSR. The LFSR is then updated for the next cycle. Note that the input bits must be written into the memory before reading the output bits to maintain the integrity of the data sequence.

The inverse operation happens in the de-interleaver. When a new bit is ready to be written into the memory, the de-interleaver first reads one bit from `cntr` location of the memory and informs the next stage using the `bitReady` signal. The counter `cntr` is also updated for the next cycle. Then the de-interleaver writes the input bit into memory and updates the LFSR for the next cycle. Here, the data is read out of the memory and stored in the `bOut` register before writing the input bit. Moreover, the extracted LFSR bits are shuffled to decrease the correlation between the generated values.

4.7.4 Extended Golay Code

Channel codes or "error control codes" are used in communication systems to detect and possibly correct the errors that happen during data transmission. This is accomplished by adding redundant data to the transmitted message. The binary Golay code is one of the most important types of linear binary block codes. The extended binary Golay code has been used in many real-world applications including the Voyager spacecraft program during the early 1980s [277].

We used the extended binary Golay code in our 2×2 MIMO communication system for the detection and possible correction of occurring errors. This code can be generated by the 12×24 generator matrix $\mathbf{G} = [\mathbf{I}, \mathbf{B}]$ where $\mathbf{I}$ is the 12×12 identity matrix and $\mathbf{B}$ is

given by

$$\mathbf{B} = \begin{bmatrix} 1 & 1 & 0 & 1 & 1 & 1 & 0 & 0 & 0 & 1 & 0 & 1 \\ 1 & 0 & 1 & 1 & 1 & 0 & 0 & 0 & 1 & 0 & 1 & 1 \\ 0 & 1 & 1 & 1 & 0 & 0 & 0 & 1 & 0 & 1 & 1 & 1 \\ 1 & 1 & 1 & 0 & 0 & 0 & 1 & 0 & 1 & 1 & 0 & 1 \\ 1 & 1 & 0 & 0 & 0 & 1 & 0 & 1 & 1 & 0 & 1 & 1 \\ 1 & 0 & 0 & 0 & 1 & 0 & 1 & 1 & 0 & 1 & 1 & 1 \\ 0 & 0 & 0 & 1 & 0 & 1 & 1 & 0 & 1 & 1 & 1 & 1 \\ 0 & 0 & 1 & 0 & 1 & 1 & 0 & 1 & 1 & 1 & 0 & 1 \\ 0 & 1 & 0 & 1 & 1 & 0 & 1 & 1 & 1 & 0 & 0 & 1 \\ 1 & 0 & 1 & 1 & 0 & 1 & 1 & 1 & 0 & 0 & 0 & 1 \\ 0 & 1 & 1 & 0 & 1 & 1 & 1 & 0 & 0 & 0 & 1 & 1 \\ 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 0 \end{bmatrix}$$

The code rate for the $(24, 12)$ extended Golay code is $R = 1/2$. This code has minimum distance $d_{min} = 8$ and can correct up to three errors.

Encoding the data bits using the Golay code is straightforward. Assuming $\mathbf{u} = [u_{11}, u_{10}, ..., u_0]$ to be the vector of source bits, the coded bits can be calculated as $\mathbf{v} = [\mathbf{u}, \mathbf{p}] = \mathbf{uG}$ in GF(2), i.e., the Galois field of two elements [278], where $\mathbf{p}$ is the length 12 row vector of parity bits.

To decode the extended binary Golay code, we used the imperfect maximum likelihood decoding (IMLD) algorithm from [277]. This algorithm tries to find all of the error pattens $\mathbf{e}$ of weight at most 3. The error pattern $\mathbf{e}$ is denoted as $\mathbf{e} = [\mathbf{e}_1, \mathbf{e}_0]$ where $\mathbf{e}_0$ and $\mathbf{e}_1$ are the lower and upper parts of $\mathbf{e}$ each with 12 bits.

Assume that $\mathbf{w} = [w_{23}, w_{22}, ..., w_0]$ represents the received vector and let $\mathbf{o}_i$ to be a row vector of length 12 with a 1 in the i^{th} position and zeros elsewhere. Also, let us denote the i^{th} row or $\mathbf{B}$ as $\mathbf{b}_i$. The IMLD algorithm tries to find the error pattern of the received vector by computing the syndrome $\mathbf{s} = \mathbf{wH}$ where $\mathbf{H} = \mathbf{G}^T$ is the parity check matrix. This algorithm is represented from [277] in Algorithm 3.

Algorithm 3 IMLD decoding for extended Golay code

1: Compute the syndrome $\mathbf{s} = \mathbf{wH}$.
2: if $(\text{weight}(\mathbf{s}) \leq 3)$ then set $\mathbf{e} = [\mathbf{s}, \mathbf{0}]$, and goto 8.
3: if $(\text{weight}(\mathbf{s} + \mathbf{b}_i) \leq 2)$ for some row $\mathbf{b}_i$ of $\mathbf{B}$ then set $\mathbf{e} = [\mathbf{s} + \mathbf{b}_i, \mathbf{o}_i]$, and goto 8.
4: Compute the second syndrome $\mathbf{sB}$.
5: if $(\text{weight}(\mathbf{sB}) \leq 3)$ then set $\mathbf{e} = [\mathbf{0}, \mathbf{sB}]$, and goto 8.
6: if $(\text{weight}(\mathbf{sB} + \mathbf{b}_i) \leq 2)$ for some row $\mathbf{b}_i$ of $\mathbf{B}$ then set $\mathbf{e} = [\mathbf{o}_i, \mathbf{sB} + \mathbf{b}_i]$, and goto 8.
7: The error pattern cannot be determined. Exit.
8: The decoded vector is $\hat{\mathbf{v}} = \mathbf{w} + \mathbf{e}$. Exit.

We implemented pipelined datapaths for encoding and decoding of the extended Golay code. The implemented decoder can correct all of the error patterns with one, two, and three

errors. More error patterns can also be detected and reported for requesting retransmission.

4.7.5 Fractional Delay

The fractional delay interpolator is an important component of a fading channel simulator. Assuming uniform sampling, fractional delay refers to a delay that is a non-integer multiple of the sample interval T_s. Several methods have been used for approximating the fractional delay including the use of lowpass, and allpass filters and polynomial-based interpolation [279–283]. One of the most attractive implementations for fractional delay interpolators is the Farrow structure [284]. The Farrow interpolator approximates the delayed signal with a piecewise polynomial curve in time as

$$\hat{x}((n-\mu)T_s) = \sum_{l=0}^{L} b_l(n)\mu^l, \tag{4.76}$$

where $|\mu| \leq 0.5$ (μT_s is the fractional delay), and L is the polynomial degree. The polynomial coefficients $\{b_l(n)\}_{l=0}^{L}$ in equation (4.76) are calculated as

$$b_l(n) = \sum_{k=0}^{K_F} c_l^k x((n-k)T_s), \tag{4.77}$$

where K_F denotes the filter order and $\{c_l^k\}$, $l = 0, ..., L$, $k = 0, ..., K_F$, denote the constant filter coefficients [284]. Notice that the coefficients $\{b_l(n)\}_{l=0}^{L}$ are independent of μ, hence the Farrow structure can be used to generate time-variable fractional delay by only varying the delay parameter μ. Figure 4.41 (a) shows the Farrow structure for time-varying fractional delay.

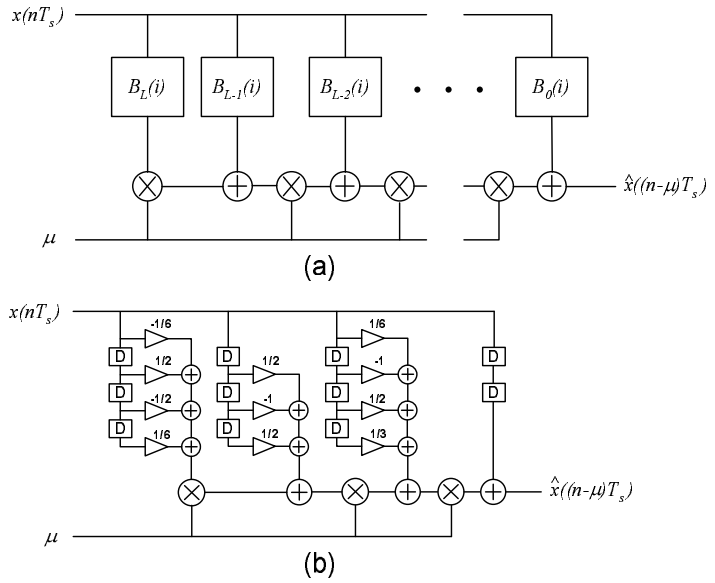


Figure 4.41: (a) Farrow structure for time-varying fractional delay. (b) Farrow structure for a cubic interpolator.

In our hardware simulation platform, we implemented the the Farrow structure for a cubic polynomial interpolation (see Figure 4.41 (b)). In this interpolator, the polynomial degree $L = 3$ and the coefficients are calculated with $K_F = 3$ FIR filters. The polynomial coefficients for this structure are given by

$$\begin{aligned} b_3(n) &= x((n-3)T_s)/6 - x((n-2)T_s)/2 + x((n-1)T_s)/2 - x(nT_s)/6, \\ b_2(n) &= x((n-3)T_s)/2 - x((n-2)T_s) + x((n-1)T_s)/2, \\ b_1(n) &= x((n-3)T_s)/3 + x((n-2)T_s)/2 - x((n-1)T_s) + x(nT_s)/6, \\ b_0(n) &= x((n-2)T_s). \end{aligned} \tag{4.78}$$

Figure 4.42 shows the datapath of the implemented delay module. Notice that in equation (4.78) the filter coefficients for the cubic Farrow interpolators are limited to $\{\pm 1, \pm 1/2, 1/3, \pm 1/6\}$. Therefore we can generate the polynomial coefficients $\{b_l(n)\}_{l=0}^{3}$, with shifting, negation, and summation operations in addition to one constant multiplication as shown in Figure 4.42 (a).

Figure 4.42 (b) shows the datapath of the Farrow polynomial for fractional delay. Figure 4.42 (b) also shows the datapath for integer delay that can be used to extend the range of fractional delay. Notice that since the Farrow polynomial coefficients in equation (4.77) are independent of the fractional delay, the same polynomial coefficients can be used for generating various fractional delays. This is particularly helpful for simulating multipath delay where multiple fractional delays are required.

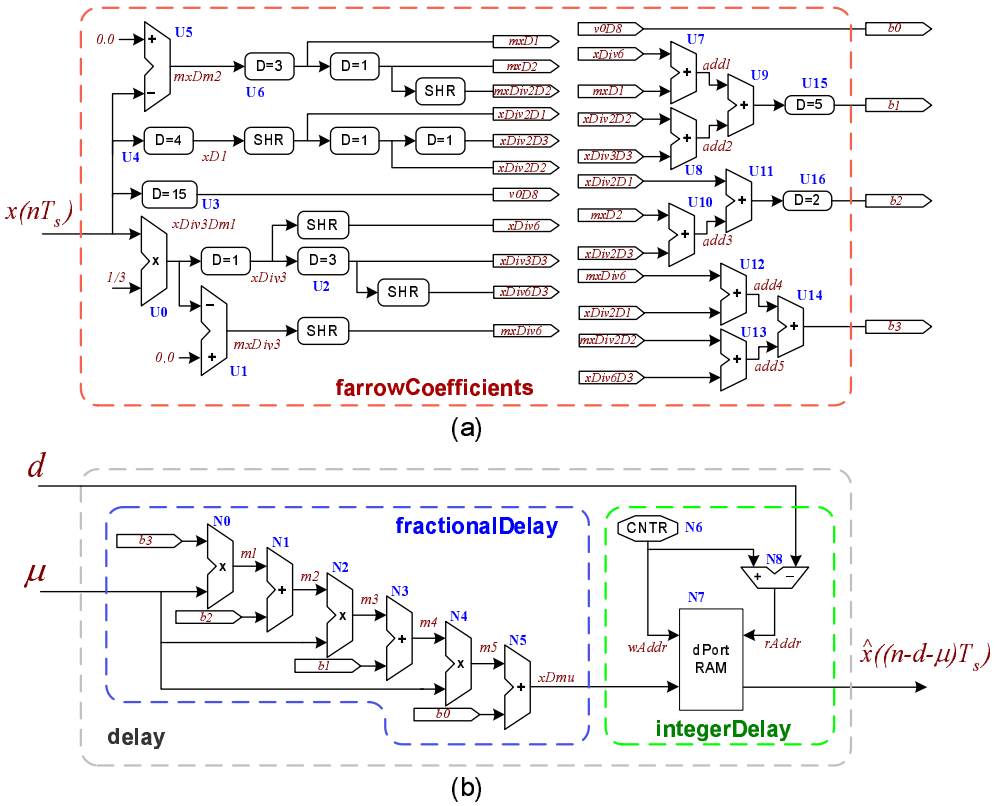


Figure 4.42: Datapath of the cubic Farrow interpolator for fractional delay. (a) Farrow filter coefficient generator, and (b) Farrow polynomial and integer delay.

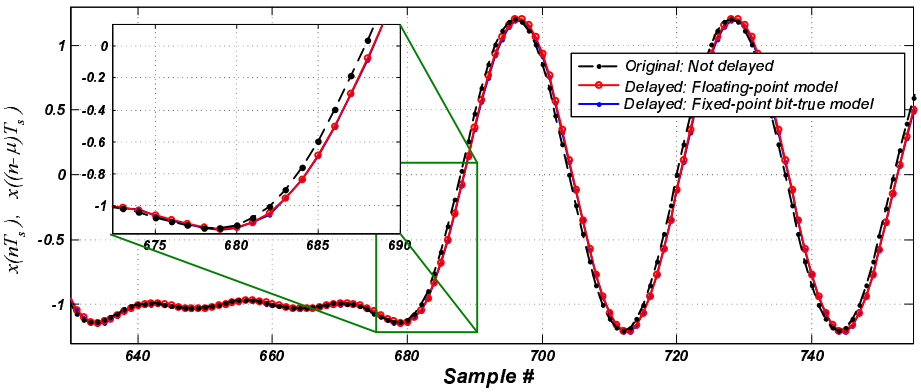


Figure 4.43: Output samples for delayed BPSK signal using the fixed-point bit-true model.

Figure 4.43 shows the output of our fixed-point bit-true model for the cubic Farrow interpolator. In this figure, a BPSK signal is delayed for half of the signal period (i.e., $\mu T_s = 0.5T_s$). The original signal and the response of the floating-point model are plotted as well. As this figure shows, our fixed-point implementation result accurately matches that

of the floating-point model, which verifies the accuracy of our hardware implementation.

4.7.6 Hardware Implementation Results

As mentioned before, we implemented our fading simulation platform for 2×2 MIMO and SISO systems on a GVA-290 (Revision-B) FPGA development platform [161]. This board hosts two Xilinx Virtex-E XCV200E FPGAs and two Xilinx Spartan-II FPGAs. This board also includes four 100 MS/s analog-to-digital and four 100 MS/s digital-to-analog converters.

Table 4.3 summarizes the FPGA implementation results of different components. The synthesis results are provided for the Xilinx Virtex-E XCV2000EBG560-6 FPGA and for the Xilinx Virtex-4 XC4VSX55FF1148-12 FPGA. For this implementation, the fading simulation cores were configured to use the resources available on the Virtex-E FPGA family.

From the results presented in this table, we can conclude that the implemented MIMO communication system (source, encoder, interleaver, detector, de-interleaver, and decoder) utilizes less than 9% of the available configurable slices on a Virtex-E XCV2000E FPGA while the rest of the system (fading simulator, BER performance measurement, initialization, and interfacing modules) consume a much larger portion of the available resources (more than 60%).

The implemented fading simulation platform and the BER performance measurement cores along with the analog and digital access to different parts of the system on a GVA-290 board can be used for testing and verification of more complex wireless communication systems. More specifically, with one Virtex-E XCV2000E FPGA dedicated to fading simulation and interfacing, the other on-board FPGAs can be used for the rapid prototyping of wireless communication systems. In addition, the implemented fading simulation and BER performance measurement platform can be easily adapted to faster and more recent FPGA boards for rapid prototyping of wireless communication system in baseband and intermediate frequency.

Overall, to develop the above fading simulation platform more than 55,000 lines of code were written in the five languages MATLAB, MEX, Visual C++, 80386-Assembly, and Verilog hardware description language (HDL). Fixed-point arithmetic was required for the development of the bit-true models. However, due to the slow execution of the MATLAB fixed-point library, we developed our own fixed-point library in MEX (C programming for MATLAB) in which we used 32-bit machine language sections frequently to speed up the

fixed-point simulation. The main skeleton of most of the designed parts were developed in MATLAB. However we often used MEX programming to speed-up the computer simulation process. Moreover, we used Verilog HDL for the hardware implementation of different parts, and finally, the GUI for the fading simulation platform was developed in Visual C++.

Table 4.3: Fading Simulation Platform Implementation Results

Design	**Device**	**Max. Clock**	**Slices**	**BRAMs**
Source	XCV2000E	299.9 MHz	37 (0.2%)	0
Source	XC4VSX55	778.0 MHz	37 (0.15%)	0
Encoder	XCV2000E	126.6 MHz	50 (0.3%)	0
Encoder	XC4VSX55	421.5 MHz	52 (0.2%)	0
Interleaver	XCV2000E	117.1 MHz	49 (0.3%)	4 (2.5%)
Interleaver	XC4VSX55	336.7 MHz	49 (0.2%)	4 (1.25%)
MIMO channel[a]	XCV2000E	99.5 MHz	998 (5.2%)	8 (5.0%)
MIMO channel	XC4VSX55	303.8 MHz	979 (4.0%)	5 (1.6%)
ML detector	XCV2000E	76.3 MHz	947 (4.9%)	0
ML detector	XC4VSX55	188.3 MHz	939 (3.8%)	0
De-interleaver	XCV2000E	115.7 MHz	50 (0.3%)	4 (2.5%)
De-interleaver	XC4VSX55	332.2 MHz	51 (0.2%)	4 (1.25%)
Decoder	XCV2000E	80.8 MHz	438 (2.3%)	0
Decoder	XC4VSX55	191.2 MHz	407 (1.7%)	0
Fading generator	XCV2000E	70.9 MHz	3601 (18.8%)	7 (4.4%)
Fading generator	XC4VSX55	185.2 MHz	3522 (14.3%)	7 (2.2%)
Noise generator	XCV2000E	99.5 MHz	634 (3.3%)	8 (5.0%)
Noise generator	XC4VSX55	303.8 MHz	632 (2.6%)	5 (1.6%)
4-Path delay[b]	XCV2000E	52.6 MHz	2247 (11.7%)	12 (7.5%)
4−Path delay	XC4VSX55	174.2 MHz	2193 (8.9%)	12 (3.75%)
Entire system[c]	XCV2000E	52.6 MHz	13436 (70.0%)	47 (29.4%)
Entire system	XC4VSX55	174.2 MHz	13247 (53.9%)	44 (13.75%)

[a] Includes 4-QAM Modulator, MIMO channel, and Gaussian noise generator.

[b] Includes Farrow coefficient generator for in-phase and quadrature (I/Q) paths, three I/Q Farrow interpolators for fractional delay with 16-bit resolution, and three I/Q delay modules for up to 1023 tap delay.

[c] Includes the MIMO transmitter and receiver, SISO transmitter, MIMO and SISO fading channel simulator, BER measurement units, digital-to-analog interface modules, initialization logic, and FPGA-PC interface modules.

4.8 Summary and Conclusions

In this chapter briefly reviewed different MIMO channel models. In general, MIMO channel models can be categorized, based on the modeling approach, into analytical and physical channel models. Analytical channel models characterize the MIMO channel response in a mathematical/analytical fashion, while physical models describe the MIMO channel based on the physical characteristics of wave propagation.

We also discussed a simulation model for the efficient simulation of geometric MIMO channel models in hardware. We showed the effectiveness of our hardware geometric MIMO fading channel simulator design by simulating the geometric elliptical model, one-ring, and two-ring models using fixed-point bit-true simulation and actual hardware results.

We also proposed a compact and flexible hardware fading simulator for the simulation of analytical MIMO channel models including the Kronecker model, the Weichselberger model, and the virtual channel representation model. The accuracy of our implementation and hardware generated results were also verified with the floating-point computer simulations.

We implemented our MIMO fading simulator on a GVA-290 FPGA board for prototyping and verifying SISO and MIMO wireless systems. The implemented MIMO fading channel simulator can simulate several different geometric and analytical MIMO channel models. The implemented fading simulator can also simulate multipath propagation scenarios in single antenna fading channels with Rayleigh and Rician fading models. Moreover, the multipath propagation delay can be accurately simulated on the fading simulator platform with the implemented fractional-delay circuit.

Moreover, we implemented a 2×2 MIMO communication system and tested its performance using the implemented fading simulation platform. The implemented MIMO system includes a length-16383 random interleaver and de-inerleaver, an extended Golay code decoder and encoder, and a ML detector. We also implemented a bit error rate performance measurement platform and the required software and hardware components. Using this tester and the fading simulator, we can evaluate the bit error rate performance of a wide variety of wireless communication systems under various channel models and propagation conditions.

Chapter 5

Conclusions and Future Work

5.1 Main Contributions

In this thesis we proposed three new channel models for the accurate simulation of Rayleigh and Rician fading. We also proposed compact hardware architectures for the fast and efficient on-chip simulation of wireless fading channels. We also proposed a new especially compact hardware implementation of an accurate fading simulator in which fading samples are generated differentially and interpolated with a compact architecture. More than one thousand paths can be fit on a conventional FPGA with this method, with each path generating more than 300 million samples per second. Compared to one of our early designs, the proposed fading simulator is not only 18 times smaller and 50% faster, it can generate significantly more accurate fading samples.

We proposed two architectures for the homogeneous FPGA implementation of filter-based fading simulators. We also proposed a multi-stage filter design technique for the efficient hardware simulation of Rayleigh fading channels. Moreover, we presented an elastic design for the robust implementation of a multipath fading simulator that can absorb the clock frequency mismatches between hardware modules. A fixed-point implementation of this four-path fading simulator on a Xilinx Virtex-II Pro XC2VP100-6 FPGA utilizes only 13.9% of the configurable slices and 2.7% of the on-chip 18×18 multipliers and can generate up to 4×73 million samples per second. We also proposed the first hardware simulator for non-isotropic Rayleigh fading channels. Our fixed-point implementation of this simulator on a Xilinx Virtex-II Pro XC2VP100-6 FPGA utilizes only 6.8% of the configurable slices and can generate up to 300 million samples per second.

We presented a new transformation-based fading simulator for the compact and efficient implementation of Nakagami-m and Weibull fading channels. This fading simulator

converts the Rayleigh fading samples to Nakagami-m or Weibull fading samples. A new method for the approximation of the transfer function was presented which was based on hybrid logarithmic-linear segmentation with semi-floating-point curve fitting. Compared to the simple table look-up approach for approximating the transfer function, the proposed method provided significant savings in the storage requirements.

We also presented a technique for the design of stable IIR filters with fixed-point complex and real coefficients. Filter design with this technique results in very compact hardware implementations. It was shown that the new filter design technique could be used for the simulation of a wide range of fading channel conditions including non-isotropic Rayleigh fading channels and the TGn channel model for the IEEE 802.11n MIMO wireless LAN standard. Using the proposed filter design technique should result in significant savings in the hardware implementation and substantial increases in the system throughput. In one example, compared to a previous design, the new filter design technique resulted in a nine times reduction in the number of configurable slices and more than 22 times higher throughput.

We proposed a compact and efficient FPGA fading simulator that could support the simulation of the i.i.d., Kronecker, Weichselberger, and VCR MIMO fading channel models. A new stable interpolator structure was proposed for this fading simulator. When implemented on a Xilinx Virtex-5 XC5VLX110-3 FPGA, the matrix processor of this design for a 4×4 MIMO system utilizes only 1.8% of the configurable slices, two multipliers, and four block memories, and can operate at up to 234.1 MHz. This MIMO fading simulator was implemented on our bit error rate testing platform. Moreover, we presented a compact and efficient simulator for geometric MIMO channel models. The proposed fading simulator could simulate a wide variety of single- and double-bounce geometric MIMO channel models. We verified the accuracy of our fading simulator by comparing the fixed-point bit-true results with theoretical references. Three common geometric MIMO fading models, namely the one-ring and two-ring models and the geometrical elliptic MIMO fading channel model, were simulated with our fading simulator.

We implemented a fading simulation and bit error rate testing platform for the verification of our fading simulators, and for testing MIMO systems. This platform was implemented on a GVA-290 FPGA development board which hosts two Xilinx Virtex-E XCV2000E FPGAs. The implemented platform can be used for the verification of single- and multiple-antenna wireless systems. It supports the simulation of various fading chan-

nel models including the single antenna-models AWGN, Rayleigh, Rician, and the MIMO models AWGN, Rayleigh, Rician, one-ring, two-ring, geometric elliptic, i.i.d., Weichselberger, Kronecker, and VCR channel models. The implemented platform also supports multipath integer delays (up to 1023 taps) and fractional delays.

We tested a sample 2×2 MIMO system with our channel simulation and system verification platform. The MIMO system under test included an extended Golay code encoder and decoder, length-16383 random interleaver and de-interleaver, and a maximum likelihood detector.

5.2 Future Work

The future work could be followed in a number of directions.

- The proposed fading simulators are all targeted for the baseband verification of wireless channels. However, the addition of a *radio frequency* (RF) stage could broaden the range of applications. Such RF stages could be designed to work with either simplex or duplex wireless communication systems.
- In this thesis, we targeted the main elements of fading simulation. For example, we designed the hardware parts required for generating fading samples for the IEEE 802.11n fading channel, and we presented an efficient hardware for the simulation of fractional delay. However, we did not implement a functional 802.11n fading channel simulator. Various standardized fading channel models (including the TGn channel model for the IEEE 802.11n standard) could be simulated efficiently with the parts designed in this thesis.
- Some aspects of fading channels were not included in our work. For example, our work can be extended to vector fading channel models and time-varying channel models. Also, our models can be extended to take into account the antenna polarization, and waveguide effects in the the fading channels.

In addition to the above subjects, I will continue working on three other problems. Particularly, I will work on compact and efficient implementation of stable IIR filter processors, and extension of our fading model to include other distributions for the *angle-of-arrival* (AOA) and the phase of the fading samples. Also, I plan to pursue the simulation of multi-node wireless networks. In the following sections I will briefly discuss these subjects.

5.2.1 Pipelined IIR Filter Processor

Digital IIR filters have a wide range of applications in electrical engineering. In this thesis we used digital IIR filters for the accurate simulation of fading channels. However, due to the inherent feedback in the IIR filters, these structures are naturally susceptible to instability. Efficient implementation of IIR filters with fixed-point arithmetic can also contribute to numerical error due to fixed word-length and can render these filters unstable.

In Chapter 3 we proposed a filter design technique for the compact and stable implementation of fixed-point filters with real and complex coefficients. The designed filters can be used for the compact and stable implementation of IIR filters for different applications.

A multi-rate and multi-channel programmable IIR filter processor can be designed based on our proposed filter design technique. This IIR filter processor needs to be pipelined for maximum efficiency. Also, it needs to be flexible so that it can perform the operations of IIR filters of various orders with real and complex coefficients. Moreover, this filter processor needs to be able to operate on several IIR filters with different sample rates. This capability is particularly important when the filter processor is operating on data from multiple sources of different rates, potentially from multiple clock domains. In addition, this filter processor needs to be able to perform interpolation, decimation, zero-padding, and concatenation of various filters. Moreover, this filter processor needs to be able to add (or inject) potentially complex tones to the different stages of each IIR filter. This could be used for a variety of applications including built-in self-test functions or the simulation of Rician fading samples.

However, designing such a filter processor would not be trivial. A pipelined IIR filter processor needs to be implemented with out-of-order processing (because of the feedback in IIR filters and the latency of the pipelined arithmetic unit). This can be challenging if the filter structure (i.e., real/complex coefficients, first-, or second-order section processing, and filter order) is programmable. Also, the filter processor needs to "execute" the filter sections in reverse order to avoid overwriting the intermediate samples. Finally, the execution of each filter needs to be tied to a pair of input and output flags indicating activation or deactivation of each filter (for rate control).

During the course of this research project, we realized the need for such a filter processor and we designed the potential datapath and structure. However, due to time limitations this processor design was not implemented. Figure 5.1 (a) shows the designed datapath for this processor. In this datapath, the filter coefficients are stored in `RAM R4` and `RAM R5`

while the intermediate signals are stored in RAM R2 and RAM R3. Moreover, the dual-port memory RAM R1 keeps the output samples. Figure 5.1 (b) shows the datapath of the arithmetic unit of this processor which was implemented for performing the matrix operations in the Section 4.6.1 of this thesis. This arithmetic unit can perform complex additions, complex-by-complex products and real-by-complex products.

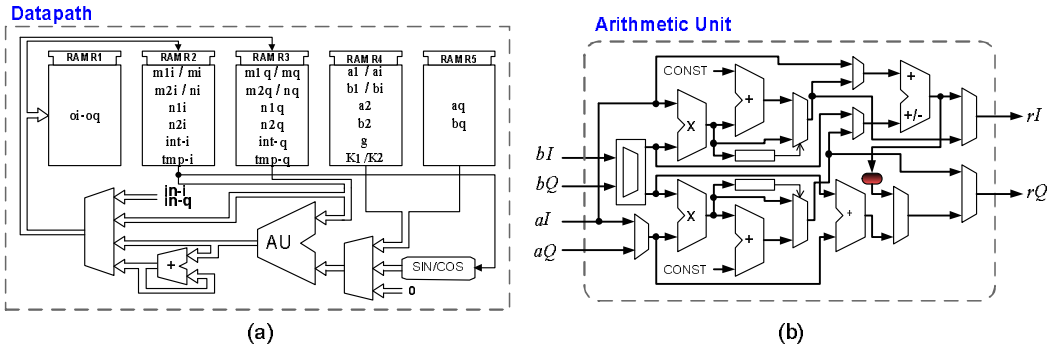


Figure 5.1: (a) Datapath of the proposed pipelined IIR filter processor, (b) Datapath of the arithmetic unit.

Figure 5.2 shows the control unit of the proposed pipelined IIR filter processor. In this control unit, the instructions are the stored in the memory RAM R6. A section of this memory is allocated to the *interrupt service routine* (ISR) that is mainly used for the construction and initialization of the IIR filters. The memory RAM R7 will store the micro-instructions, and RAM R8 is the stack that is mainly used for the function calls.

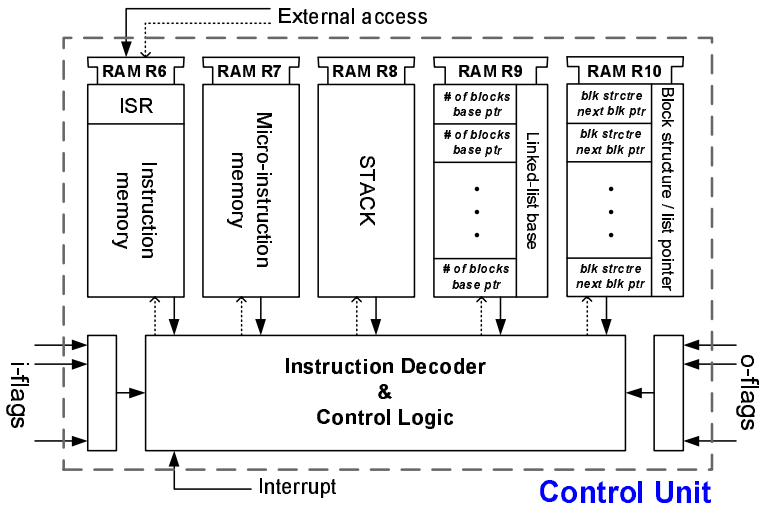


Figure 5.2: Control unit of the proposed pipelined IIR filter processor.

Instruction	Description	Instruction	Description
NOP	No Operation Parameters: none	RUNFA	Run filter if active Parameters: input, output, FILTER
WAIT	Wait for N cycles, Parameters: N	RUNFIA	Run filter if input active Parameters: input, output, FILTER
JMP	Jump to ADR, Parameters: ADR	RUNFOA	Run filter if output active Parameters: input, output, FILTER
JMPI	Jump to ADR if inactive, Parameters: ADR, FILTER	MOVE	Move a value to a biquad register Parameters: regID, bqdNo, val
REP	Repeated Procedure call Parameters: ADR, N	ALLOC	Allocate a biquad to a given filter Parameter: FILTER
CALL	Procedure call Parameters: ADR	RMV	Remove last biquad Parameters: FILTER
RET	Return Parameters: none	RST	Reset all internal memories
RUNF	Run filter Parameters: input,output, FILTER		

Figure 5.3: Basic instruction set for the proposed IIR filter processor.

In this processor, we suggest that the filter structure be stored as a linked-list so that addition and extraction of first- and second-order sections can be performed easily. This way the filter order can be adjusted dynamically and several filters of various orders can be implemented. Moreover, the input/output activation flags are allocated to filters, not to individual sections of each filter. Using this linked-list structure enables the filter processor to control the rate of execution for individual sections of each filter. In the control unit in Figure 5.2, `RAM R9` stores structures that record the number of blocks of first- or second-order sections of each filter. These structures also store the number of blocks in each filter. The linked-list that keeps the address of the filter blocks and the structure of each filter block, which is whether this block belongs to a first-order section with complex coefficients or it belongs to a second-order section with real coefficients, is stored in `RAM R10`.

Figure 5.3 summarizes the basic instruction set of this filter processor. The basic instructions for performing one cycle of the operations of an IIR filter are `RUNF`, `RUNFA`, `RUNFIA`, and `RUNFOA` that read one sample from the specified input (constant 0, processor input, frequency tone, etc.) and writes one sample to the specified output. The filter structures are built using the `ALLOC`, `RMV`, and `MOVE` instructions. Finally, some instructions for flow control and function call are assigned, including some instructions (e.g., `REP`) that are necessary for interpolation or decimation operations.

5.2.2 Extensions on AOA and Phase Distribution

In this thesis we presented several architectures for the efficient simulation of different single-antenna channel models. More specifically, we discussed *sum-of-sinusoids* (SOS) based and filter-based simulation of Rayleigh and Rician fading channels. Finally, we discussed the hardware simulation of Nakagami-m and Weibull fading channels.

The scope of our single-antenna fading channel simulation, however, was limited to the most common models in which the AOA was either uniformly distributed or followed the von Mises distribution [91]. Other than these two distributions, several other models including the geometrically-based PDFs [128, 129], Gaussian PDF [130], quadratic PDF [131], Laplace PDF [120], and cosine PDF [132], have also been proposed to model the distribution of AOA.

In Chapter 3, we proposed an architecture for the efficient conversion of Rayleigh fading samples to Nakagami-m and Weibull fading samples. Beaulieu assumed that in a Nakagami-m fading channel, the phase of the complex Nakagami-m fading samples is uniformly distributed [42, 43]. However, a study by Yacoub [285] suggests that the phase of Nakagami-m samples is not uniformly distributed when $m \neq 1$. This work could be continued by researching new ways to incorporate other proposed PDFs for the AOA in the fading models. Finally, new ways need to be found to ensure that the generated Nakagami-m fading samples have the appropriate phase distribution.

5.2.3 Radio-Frequency Multi-Node Fading Channel Simulator

Another possible extension of the work in this thesis is to develop models and platforms for the simulation of wave propagation between more than two wireless systems. Such a fading simulator could be used not only for the testing and development of wireless devices, but also for testing interoperability and inter-compatibility between different devices and also for the development of higher-level networking protocols.

Essentially, the number of wireless links grows with the square of number of nodes in a network of wireless devices. Hence, the computational intensity of simulating wireless networks grows quickly with the number of nodes in the network. Fortunately, with the compact and efficient designs provided in this thesis, it is possible to simulate a large number of fading channels on a single FPGA device. However, additional storage capacity might be necessary if long delays need to be implemented for networks that are distributed over larger areas.

We have successfully developed fading channel simulators for single and multiple-antenna transceivers. To extend the current fading channel simulators to a multi-node fading simulator, we would need to add the capability of incorporating more detailed fading channel models to the current fading channel simulator. Moreover, the baseband simulation of large networks with distributed fading simulation that are connected with digital links can be challenging due to digital timing and clock distribution issues. We suggest that such a multi-node fading simulator should be implemented in baseband, with distributed fading simulators that are interconnected using *intermediate frequency* (IF) links. Also, *radio frequency* (RF) sections need to be added to the current fading simulator for interfacing the system to general wireless devices.

Figure 5.4: Diagram of a multi-node fading channel simulator.

Figure 5.4 shows a diagram of the proposed fading simulator. This fading simulator could be used not only for the testing of individual transceivers, but also for testing inter-operability and inter-compatibility of a relatively large number of nodes, for *media access control* (MAC), and *physical layer* (PHY) protocol design. In the diagram shown in Fig-

ure 5.4, the fading effects on the transmitted signal from `node A` to the rest of the nodes are simulated in the local `Multipath-Multinode Fading Emulator`. Then each faded stream is routed to the target device through the `Signal Switch`. This process is controlled by the central `Emulation Controller` that realizes different fading scenarios corresponding to different physical environments.

Bibliography

[1] Y. R. Zheng and C. Xiao. Improved models for the generation of multiple uncorrelated Rayleigh fading waveforms. *IEEE Commun. Lett.*, 6:256–258, 2002.

[2] A. Alimohammad, S. Fouladi Fard, B. F. Cockburn, and C. Schlegel. An improved SOS-based fading channel emulator. In *Proceedings of the IEEE 66th Vehicular Technology Conference (VTC'07)*, pages 931–935, 2007.

[3] A. Alimohammad, S. Fouladi Fard, B. F. Cockburn, and C. Schlegel. An accurate and compact Rayleigh and Rician fading channel simulator. In *Proceedings of the IEEE Vehicular Technology Conference*, pages 409–413, 2008.

[4] S. Fouladi Fard, A. Alimohammad, B. F. Cockburn, and C. Schlegel. High path-count multirate Rayleigh fading channel simulator with time-multiplexed datapath. In *To Appear in the IEEE International System on Chip Conference (SOCC'09)*, September 2009.

[5] S. Fouladi Fard, A. Alimohammad, B. F. Cockburn, and C. Schlegel. Ultra-compact channel simulator for high path count Rayleigh and Rician fading simulation. *Submitted to IET Communications*, 2009.

[6] A. Alimohammad and B. F. Cockburn. Compact implementation of a sum-of-sinusoids Rayleigh fading channel simulator. In *Proceedings of the IEEE ISSPIT*, pages 253–257, 2006.

[7] A. Alimohammad and B. F. Cockburn. A reconfigurable SOS-based Rayleigh fading channel simulator. In *IEEE Intl. Workshop on Signal Processing Systems, Design and Implementation*, pages 39–44, 2006.

[8] A. Alimohammad, S. Fouladi Fard, B. F. Cockburn, and C. Schlegel. A novel technique for efficient hardware simulation of spatiotemporally correlated MIMO fading channels. In *Proceedings of the IEEE Intl. Conference on Communications*, pages 718–724, 2008.

[9] A. Alimohammad, S. Fouladi Fard, B. F. Cockburn, and C. Schlegel. A compact fading channel simulator using timing-driven resource sharing. In *Proceedings of the IEEE International Conference on Application-specific Systems, Architectures and Processors (ASAP'07)*, pages 154–159, April 2007.

[10] A. Alimohammad, S. Fouladi Fard, B. F. Cockburn, and C. Schlegel. A compact single-FPGA fading channel simulator. *IEEE Trans. Circuits Syst. II*, 55(1):84–88, January 2008.

[11] A. Alimohammad, S. Fouladi Fard, B. F. Cockburn, and C. Schlegel. A flexible filter processor for fading channel emulation. In *IEEE Symposium on Field-Programmable Custom Computing Machines (FCCM'07)*, pages 339–342, April 2007.

[12] S. Fouladi Fard, A. Alimohammad, M. Khorasani, C. Schlegel, and B. F. Cockburn. A compact and accurate FPGA based nonisotropic fading channel simulator. In *Proceedings of the Canadian Conference on Electrical and Computer Engineering, 2007 (CCECE'07)*, pages 1239–1242, April 2007.

[13] S. Fouladi Fard, A. Alimohammad, B. F. Cockburn, and C. Schlegel. A single-FPGA filter-based multipath fading emulator. In *To Appear in the Proceedings of the IEEE Global Communications Conference (GLOBECOM'09)*, December 2009.

[14] S. Fouladi Fard, A. Alimohammad, B. F. Cockburn, and C. Schlegel. Filter design and hardware implementation of nonisotropic fading channels with arbitrary temporal correlation. *Submitted to IET Signal Processing.*, 2009.

[15] P. Bello. Characterization of randomly time-variant linear channels. *IEEE Trans. Commun.*, 11(4):360–393, 1963.

[16] R. H. Clarke. A statistical theory of mobile-radio reception. *Bell System Technical Journal*, 47:957–1000, 1968.

[17] W. C. Jakes. *Microwave Mobile Communications*. Piscataway, NJ: Wiley-IEEE Press, 1974.

[18] J. I. Smith. A computer generated multipath fading simulation for mobile radio. *IEEE Trans. Veh. Technol.*, 24(3):39–40, August 1975.

[19] G. L. Stüber. *Principles of Mobile Communication*. Kluwer Academic Publishers, New York, 2 edition, 2001.

[20] T. S. Rappaport. *Wireless Communications: Principles and Practice*. Prentice Hall, 2002.

[21] M. A. Wickert and J. Papenfuss. Implementation of a real-time frequency-selective RF channel simulator using a hybrid DSP-FPGA architecture. *IEEE Trans. Microw. Theory Tech.*, 49(8):1390 – 1397, 2001.

[22] D. Derrien and E. Boutillon. Quality measurement of a colored Gaussian noise generator hardware implementation based on statistical properties. In *Proceedings of the IEEE Intl. Symposium on Signal Processing and Information Technology (ISSPIT'02)*, 2002.

[23] NoiseCom, Paramus, NJ. *NoiseCom MP-2500 Multipath Fading Emulator, Technical Manual*, 1996.

[24] Product Manual, NJZ-1600B, Japan Radio Co. *Multipath Fading Simulator*, 2005.

[25] Spirent Communications. *SR5500 Wireless Channel Emulator*, 2006.

[26] Technical Overview, N5115A, Agilent Technologies Inc. *Baseband Studio for Fading*, 2005.

[27] Rohde & Schwarz. *Baseband Fading Simulator ABFS, Reduced costs through baseband simulation, No. 163*, 1999.

[28] Technical Manual, SIMSTAR-1, Ascom. *Full Featured Channel Simulator*, 2002.

[29] A. Alimohammad, B. F. Cockburn, and C. Schlegel. Area-efficient parallel white Gaussian noise generator. In *Proceedings of the IEEE Canadian Conference on Electrical and Computer Engineering*, pages 1855–1858, 2005.

[30] M. Pätzold. *Mobile Fading Channels*. West Sussex, U.K.:Wiley, 2002.

[31] P. M. Shankar. *Introduction to Wireless Systems*. John Wiley & Sons, Inc., 2001.

[32] W. H. Tranter, K. Sam Shanmugan, T. S. Rappaport, and K. L. Kosbar. *Principles of Communication Systems Simulation with Wireless Applications*. Prentice Hall, 2003.

[33] H. Hashemi. The indoor radio propagation channel. *Proceedings of the IEEE*, 81(7):943–968, July 1993.

[34] D. C. Cox. 910 MHz urban mobile radio propagation: multipath characteristics in New York city. *IEEE Trans. Veh. Technol.*, 22(4):104–109, 1973.

[35] G. L. Turin *et al*. A statistical model of urban multipath propagation. *IEEE Trans. Veh. Technol.*, 21(1):1–11, 1972.

[36] M. J. Gans. A power spectral theory of propagation in the mobile radio environment. *IEEE Trans. Veh. Technol.*, 21(1):27–38, 1972.

[37] W. C. Y. Lee. *Mobile Communications Engineering*. New York: McGraw Hill, 1982.

[38] M. Pätzold, U. Killat, F. Laue, and Y. Li. A new and optimal method for the derivation of deterministic simulation models for mobile radio channels. In *IEEE Veh. Tech. Conf.*, pages 1423–1427, 1996.

[39] M. Nakagami. The m-distribution: A general formula of intensity distribution of rapid fading. In W. C. Hoffman, editor, *Statistical Methods in Radio Wave Propagation*, pages 3–36, Oxford, U.K.: Pergamon, 1960.

[40] W. Braun and U. Dersch. Physical mobile radio channel model. *IEEE Trans. Veh. Technol.*, 40:472–482, March 1991.

[41] Q. T. Zhang. A generic correlated Nakagami fading model for wireless communications. *IEEE Trans. Commun.*, 51(11):1745–1748, November 2003.

[42] N. C. Beaulieu and C. Cheng. Efficient Nakagami-m fading channel simulation. *IEEE Trans. Veh. Technol.*, 54:413–424, 2005.

[43] N. C. Beaulieu and C. Cheng. An efficient procedure for Nakagami-m fading simulation. In *Proceedings of the IEEE Global Telecommunications Conference (GLOBECOM'01)*, volume 6, pages 3336–3342, 2001.

[44] A. Alimohammad, S. Fouladi Fard, B. F. Cockburn, and C. Schlegel. Signal filtering and filter design techniques. In *International Patent Application, Reference No 2006045 (PCT)*, May 2008.

[45] A. Alimohammad, S. Fouladi Fard, B. F. Cockburn, and C. Schlegel. A single-FPGA multipath MIMO fading channel simulator. In *Proceedings of the IEEE International Symposium on Circuits and Systems*, pages 308–311. Seattle, Washington, USA, May 2008.

[46] K. Ohtani, K. Daikoku, and H. Omori. Burst error performance encountered in digital land mobile radio channel. *IEEE Trans. Veh. Technol.*, 30(4):156–160, November 1981.

[47] J. M. Morris. Burst error statistics of simulated Viterbi decoded BPSK on fading and scintillating channels. *IEEE Trans. Commun.*, 40:34–41, 1992.

[48] R. Wang and D. C. Cox. Understanding cellular ad hoc networks: How much an accurate physical layer model matters. In *Proceedings of the IEEE Asilomar Conference*, pages 1753–1757, 2003.

[49] L. Lindbom, A. Ahlen, M. Sternad and M. Falkenstrom. Tracking of Time-Varying

Mobile Radio Channels-Part II: A Case Study. *IEEE Trans. Commun.*, 50:156–167, January 2002.

[50] J. G. Proakis. *Digital Communications*. McGraw-Hill, 2001.

[51] S. O. Rice. Mathematical analysis of random noise. *Bell System Technical Journal*, 23:282–332, 1944.

[52] E. N. Gilbert. Energy reception for mobile radio. *Bell System Technical Journal*, pages 1779 – 1803, October 1965.

[53] M. Pätzold and R. Garcia and F. Laue. Design of high-speed simulation models for mobile fading channels by using table look-up techniques. *IEEE Trans. Veh. Technol.*, 49(4):1178–1190, July 2000.

[54] M. F. Pop and N. C. Beaulieu. Limitations of sum-of-sinusoids fading channel simulators. *IEEE Trans. Commun.*, 49:699–708, 2001.

[55] P. Hoeher. A statistical discrete-time model for the WSSUS multipath channel. *IEEE Trans. Veh. Technol.*, 41:461–468, 1992.

[56] Y. Li and X. Huang. The simulation of independent Rayleigh faders. *IEEE Trans. Commun.*, 50(9):1503–1514, 2002.

[57] W. Cheng-Xiang and M. Pätzold. Efficient simulation of multiple cross-correlated Rayleigh fading channels. In *IEEE PIMRC*, pages 1526 – 1530, 2003.

[58] R. Brown, R. Beaudin, and H. Hamel. Frequency selective RF channel simulator. In *IEEE MILCOM*, pages 617–621, 2002.

[59] M. Cui, H. Murata, and K. Araki. Real-time MIMO received signal generator for spatial multiplexing systems. In *Proceedings of the IEEE Intl. Vehicular Technology Conference (VTC'04)*, pages 4345 – 4348, 2004.

[60] T. P. Wang, C. H. Liao, and T. D. Chiueh. A real-time digital baseband MIMO channel emulation system. In *Proceedings of the IEEE International Symposium on Circuits and Systems (ISCAS'07)*, pages 2606 – 2609, 2007.

[61] C. H. Liao, T. P. Wang, and T. D. Chiueh. A novel low-complexity Rayleigh fader for real-time channel modeling. In *Proceedings of the IEEE International Symposium on Circuits and Systems (ISCAS'07)*, pages 2602 – 2605, 2007.

[62] W. C. Jakes. *Microwave Mobile Communications*. Wiley-IEEE Press, Piscataway, NJ, 1994.

[63] I. S. Gradshteyn and I. M. Ryzhik. *Table of Integrals, Series, and Products*. San Diego: Academic Press, 2000.

[64] R. H. Clarke and W. L. Khoo. 3-D mobile radio channel statistics. *IEEE Trans. Veh. Technol.*, 46:798–799, August 1997.

[65] K.-W. Yip and T.-S. Ng. Discrete-time model for digital communications over a frequency-selective Rician fading WSSUS channel. *IEE Proc. Commun.*, 143(1):37–42, 1996.

[66] M. Pätzold, U. Killat, F. Laue, and Y. Li. On the statistical properties of deterministic simulation models for mobile fading channels. *IEEE Trans. Veh. Technol.*, 47:254–269, 1998.

[67] C. Xiao, Y. R. Zheng, and N. Beaulieu. Statistical simulation models for Rayleigh and Rician fading. In *Proceedings of the IEEE Intl. Conference on Communications*, pages 3524–3529, 2003.

[68] C. Xiao, Y. R. Zheng, and N. C. Beaulieu. Novel sum-of-sinusoids simulation models for Rayleigh and Rician fading channels. *IEEE Trans. Wireless Commun.*, 5(12):3667 – 3679, 2006.

[69] M. K. Simon and M. S. Alouini. *Digital communication over fading channels*. John Wiley & Sons, New York, NY, 2nd edition, 2005.

[70] P. Dent, G. E. Bottomley, and T. Croft. Jakes fading model revisited. *Electronics Letters*, 29(13):1162–1163, June 1993.

[71] C. S. Patel, G. L. Stüber, and T. G. Pratt. Comparative analysis of statistical models for the simulation of Rayleigh faded cellular channels. *IEEE Trans. Commun.*, 53:1017–1026, 2005.

[72] T. M. Wu and S.-Y. Tzeng. Sum-of-sinusoids-based simulator for Nakagami-m fading channels. In *In proceedings of the IEEE Vehicular Technology Concerence (VTC'03)*, pages 158 – 162, 2003.

[73] D. D. Patil. On the simulation of Nakagami-m fading channels using sum-of-sinusoids method. Master's thesis, University of Missouri-Columbia, Dec 2006.

[74] I. Crohn and E. Bonek. Modeling of intersymbol-interference in a Rayleigh fast fading channel with typical delay power profiles. *IEEE Trans. Veh. Technol.*, 41(4):438–447, November 1992.

[75] C. X. Wand and M. Pätzold. A novel generative model for burst error characteriza-

tion in Rayleigh fading channels. In *Proceedings of the IEEE Personal, Indoor and Mobile Radio Communications Conference (PIMRC'03)*, volume 1, pages 960–964, September 2003.

[76] A. Papoulis and S. U. Pillai. *Probability, Random Variables and Stochastic Processes*. McGraw-Hill, New York, NY, 4 edition, 2002.

[77] C-X. Wang and M. Pätzold. Methods of generating multiple uncorrelated Rayleigh fading processes. In *Proceedings of the IEEE Vehicular Technology Conference (VTC'03)*, pages 510–514, 2003.

[78] M. Pätzold and B. O. Hogstad. Classes of sum-of-sinusoids Rayleigh fading channel simulators and their stationary and ergodic properties - Part I. *WSEAS Transactions on Mathematics*, 5(2):222 – 230, Feb 2006.

[79] M. Cui, H. Murata, and K. Araki. FPGA implementation of 4×4 MIMO test-bed for spatial multiplexing systems. In *Proceedings of the IEEE International Symposium on Personal, Indoor and Mobile Radio Communications (PIMRC'04)*, pages 3045–3048, 2004.

[80] A. M. M. Donald and J. C. Olivier. A comparative study of deterministic and stochastic sum-of-sinusoids models of Rayleigh-fading wireless channels. In *Proceedings of the IEEE Wireless Communications and Networking Conference (WCNC'07)*, pages 2027–2031, 2007.

[81] Y. R. Zheng and C. Xiao. Simulation models with correct statistical properties for Rayleigh fading channels. *IEEE Trans. Commun.*, 51:920–928, 2003.

[82] M. Pätzold and B. O. Hogstad. Classes of sum-of-sinusoids Rayleigh fading channel simulators and their stationary and ergodic properties - Part II. *WSEAS Transactions on Mathematics*, 4(4):441 – 449, Oct 2005.

[83] P. Hoeher and A. Steingaß. Modeling and emulation of multipath fading channels using controlled randomness. In *Proceedings ITG-Fachtagung "Wellenausbreitung bei Funksystemen und Mikrowellensystemen"*, pages 209–220, 1998.

[84] A. Zajić and G. L. Stüber. Efficient simulation of Rayleigh fading with enhanced de-correlation properties. *IEEE Trans. Wireless Commun.*, 5(7):1866–1875, 2006.

[85] P. Pampaloni and S. Paloscia. *Microwave Radiometry and Remote Sensing of the Earth's Surface and Atmosphere*. VNU Science Press, 2000.

[86] A. Alimohammad, S. Fouladi Fard, B. F. Cockburn, and C. Schlegel. A compact Rayleigh and Rician fading simulator based on random walk processes. *to appear in*

IET Communications, 2009.

[87] P. L'Ecuyer. Tables of maximally equidistributed combined LFSR generators. *Math. of Comp.*, 68(225):261–269, 1999.

[88] C. Komninakis. A fast and accurate Rayleigh fading simulator. In *Proceedings of the IEEE Global Telecommunications Conference*, pages 3306–3310, 2003.

[89] D. J. Young and N. C. Beaulieu. The generation of correlated Rayleigh random variates by inverse Fourier transform. *IEEE Trans. Commun.*, 48:1114–1127, 2000.

[90] K. E. Baddour and N. C. Beaulieu. Autoregressive models for fading channel simulation. In *Proceedings of the IEEE Global Telecommunications Conference*, pages 1187–1192, 2001.

[91] A. Abdi, J. Barger, and M. Kaveh. A parametric model for the distribution of the angle of arrival and the associated correlation function and power spectrum at the mobile station. *IEEE Trans. Veh. Technol.*, 51(1):425–434, May 2002.

[92] V. Erceg et al. *IEEE 802.11 document 03/940r4 (TGn Channel Models)*. Garden Grove, California, May 2004.

[93] M. C. Jeruchim, P. Balaban and K. S. Shanmugan. *Simulation of Communication Systems : Modeling, Methodology, and Techniques*. New York: Kluwer Academic Publishers, 2000.

[94] A. Verschoor, A. Kegel, and J. C. Arnbak. Hardware fading simulator for a number of narrowband channels with controllable mutual correlation. *Electronics Letters*, 24(22):1367–1369, 1988.

[95] R. A. Goubran, H. M. Hafez, and A. U. H. Sheikh. Implementation of a real-time mobile channel simulator using a DSP chip. *IEEE Trans. Instrum. Meas.*, 40(4):709–714, 1991.

[96] M. S. Lim and H. K. Park. The implementation of the mobile channel simulator in the baseband and its application to the quadrature type GMSK modem design. In *IEEE Veh. Tech. Conf.*, pages 469–500, 1990.

[97] P. M. Crespo and J. Jimenez. Computer simulation of radio channels using a harmonic decomposition technique. *IEEE Trans. Veh. Technol.*, 44(3):414–419, August 1995.

[98] K. W. Yip and T. S. Ng. Efficient Simulation of Digital Transmission over WSSUS Channels. *IEEE Trans. Commun.*, 43:2907–2913, December 1995.

[99] A. Stéphenne and B. Champagne. On the simulation of multi-path vectorial channels for the evaluation of antenna array systems. In *Proceedings of the IEEE Canadian Conference on Electrical and Computer Engineering (CCECE'07)*, volume 1, pages 347–350, May 1997.

[100] A. Stéphenne and B. Champagne. A new multi-path vector channel simulator for the performance evaluation of antenna array systems. In *Proceedings of the 8th IEEE International Symposium on Personal, Indoor and Mobile Radio Communications (PIMRC'97)*, volume 3, pages 1125–1129, September 1997.

[101] A. Stéphenne and B. Champagne. Effective multi-path vector channel simulator for antenna array systems. *IEEE Trans. Veh. Technol.*, 49(6):2370 – 2381, 2000.

[102] Z. Luo and W. Zhang. Simulation for correlated Rayleigh fading channels by FIR pulse-shaping filtering. In *Proceedings of the International Conference on Wireless Communications, Networking and Mobile Computing (WiCom'07)*, pages 1091–1094, September 2007.

[103] G. J. R. Povey, P. M. Grant, and R. D. Pringle. A decision-directed spread-spectrum RAKE receiver for fast-fading mobile channels. *IEEE Trans. Veh. Technol.*, 45(3):491–502, November 1996.

[104] M. Lecours and F. Marceau. Design and implementation of a channel simulator for wideband mobile radiotransmission. In *Proceedings if the IEEE Vehicular Technology Confecence (VTC'89)*, pages 652–655, 1989.

[105] O. A. Dobre and D. Budimir. The modeling of a flat Rayleigh fading mobile radio channel. In *Proceedings of the 10th Mediterranean Electrotechnical Conference (MELECON'00)*, volume 1, pages 416–419, 2000.

[106] F. Sattar, and M. Mufti. VLSI architecture of Rayleigh fading simulator based on IIR filter and polyphase interpolator. In *Proceedings of the 16th International Conference on Microelectronics (ICM'04)*, pages 291–294, December 2004.

[107] J. K. Hwang, J. D. Li, R. L. Chung, C. Y. Chen. Efficient structure for FPGA implementation of a configurable multipath fading channel emulator. In *Proceedings of International Symposium on Intelligent Signal Processig and Communication Systems (ISPACS'06)*, pages 481–484, 2006.

[108] A. Alimohammad and B. F. Cockburn. Modeling and hardware implementation aspects of fading channel simulators. *IEEE Trans. Veh. Technol.*, 2008.

[109] K. E. Baddour and N. C. Beaulieu. Autoregressive modeling for fading channel

simulation. *IEEE Trans. Wireless Commun.*, 4(4):1650–1662, July 2005.

[110] L. B. Jackson. *Digital Filters and Signal Processing*. Kluwer Academic Publishers, 1989.

[111] The Mathworks. *Filter Design Toolbox For Use with Matlab, User's Guide*, 2005.

[112] J. S. Sadowsky, and v. Kafedziski. On the correlation and scattering functions of the WSSUS channel for mobile communications. *IEEE Trans. Veh. Technol.*, 47(1):270–282, February 1998.

[113] M. Pätzold, Y. Li, and F. Laue. A study of a land mobile satellite channel model with asymmetrical Doppler power spectrum and lognormally distributed line-of-sight component. *IEEE Trans. Veh. Technol.*, 47:297–310, February 1998.

[114] M. Pätzold, U. Killat, Y. Li, and F. Laue. Modeling, analysis, and simulation of non frequency-selective mobile radio channels with asymmetrical Doppler power spectral density shapes. *IEEE Trans. Veh. Technol.*, 46:494–507, May 1997.

[115] F. P. Fontan, M. A. V. Castro, J. Kunisch, J. Pamp, E. Zollinger, S. Buonomo, P. Baptista, and B. Arbesser. A versatile framework for a narrow- and wide-band statistical propagation model for the LMS channel. *IEEE Trans. Broadcast.*, 43:431–458, 1997.

[116] W. R. Braun and U. Dersch. A physical mobile radio channel model. *IEEE Trans. Veh. Technol.*, 40:472–482, May 1991.

[117] A. Kuchar, E. A. Aparicio, J. P. Rossi, and E. Bonek. Azimuth, elevation, and delay of signals at mobile station site. *Wireless personal communications: emerging technologies for enhanced communications*, pages 99–110, 1999.

[118] J. P. Rossi, J. P. Barbot, and A. J. Levy. Theory and measurement of the angle of arrival and time delay of UHF radio waves using a ring array. *IEEE Trans. Antennas Propag.*, 45:876–884, 1997.

[119] J. Fuhl, J. P. Rossi, and E. Bonek. High-resolution 3-D direction-of-arrival determination for urban mobile radio. *IEEE Trans. Antennas Propag.*, 45:672–682, 1997.

[120] Q. Spencer, M. Rice, B. Jeffs, and M. Jensen. A statistical model for angle of arrival in indoor multipath propagation. In *Proceedings of the IEEE Vehicular Technology Conference (VTC'97)*, pages 1415–1419, Phoenix, AZ, 1997.

[121] P. C. Fannin and A. Molina. Analysis of mobile radio channel sounding measurements in inner city Dublin at 1.808 GHz. *IEE Proceed. Commun.*, 143:311–316, 1996.

[122] J. G. Wang, A. S. Mohan, and T. A. Aubrey. Angles-of-arrival of multipath signals in indoor environments. In *Proc. IEEE Vehicular Technology Conf.*, pages 155–159, Atlanta, GA, 1996.

[123] S. Guerin. Indoor wideband and narrowband propagation measurements around 60.5 GHz in an empty and furnished room. In *Proceedings of the IEEE Vehicular Technology Conference (VTC96)*, pages 160–164, Atlanta, GA, 1996.

[124] T. Lo and J. Litva. Angles of arrival of indoor multipath. *Electronics Lett.*, 28:1687–1689, August 1992.

[125] D. O. Reudink. Large-scale variations of the average signal. In W. C. Jakes, editor, *Microwave Mobile Communications*, chapter 2, pages 79–131. Wiley-IEEE Press, New York, May 1994.

[126] W. C. Y. Lee. Finding the approximate angular probability density function of wave arrival by using a directional antenna. *IEEE Trans. Antennas Propag.*, 21:328–334, 1973.

[127] A. Krantzik and D. Wolf. Distribution of the fading intervals of modified Suzuki processes. *Signal Processing V: Theories and Applications*, pages 361–364, 1990.

[128] P. Petrus, J. H. Reed, and T. S. Rappaport. Effects of directional antennas at the base station on the Doppler spectrum. *IEEE Commun. Lett.*, 1:40–42, 1997.

[129] J. C. Liberti and T. S. Rappaport. A geometrically based model for line-of-sight multipath radio channels. In *Proceedings of the IEEE Intl. Conference on Vehicular Technology (VTC'96)*, pages 844–848, Atlanta, GA, 1996.

[130] J. Salz and J. H. Winters. Effect of fading correllation on adaptive arrays in digital mobile radio. *IEEE Trans. Veh. Technol.*, 43:1049–1057, November 1994.

[131] K. Anim-Appiah. Complex envelope correlations for nonisotropic scattering. *Electronics Lett.*, 34:918–919, 1998.

[132] M. D. Austin and G. L. Stüber. Velocity adaptive handoff algorithms for microcellular systems. *IEEE Trans. Veh. Technol.*, 43:549–561, 1994.

[133] K. V. Mardia. *Statistics of Directional Data*. Academic, New York, 1972.

[134] V. I. Tikhonov. The effects of noise on phase-lock oscillation operation. *Automatika I Telemekhanika*, 22(9), 1959.

[135] A. Abdi, K. Wills, H. Barger, S. Alouini, and M. Kaveh. Comparison of level crossing rate and average fade duration of Rayleigh, Rice, and Nakagami fading models

with mobile channel data. In *Proccedings of the IEEE Vehic Tech. Conf (VTC'00).*, pages 1850–1857, Boston, MA, 2000.

[136] K. Steiglitz. Computer-aided design of recursive digital filters. *IEEE Trans. Audio and Electroacoustics*, 18(2):123–129, June 1970.

[137] A. V. Oppenheim, R. W. Schafer, and J. R. Buck. *Discrete-time signal processing*. Prentice Hall, 2 edition, 1999.

[138] A. Alimohammad, S. Fouladi Fard, B. F. Cockburn, and C. Schlegel. A compact and accurate Gaussian variate generator. *IEEE Trans. Very Large Scale Integr. (VLSI) Syst.*, 16(5):517–527, May 2008.

[139] Xilinx. *Spartan-3 starter kit board user guide*, May 2005.

[140] A. Antoniou. *Digital Filters: Analysis, Design, and Applications*. McGraw-Hill, 2 edition, May 2000.

[141] A. G. Deczkey. Synthesis of recursive digital filters using the minimum p-error criterion. *IEEE Trans. Audio Electroacoust.*, 20:257–263, 1972.

[142] A. G. Deczkey. Equiripple and minimax (Chebyshev) approximations for recursive digitalfilters. *IEEE Trans. Acoust., Speech, Signal Process.*, 22:98–111, 1974.

[143] H. G. Martinez and T. W. Parks. Design of recursive digital filters with optimum magnitude and attenuation poles on the unit circle. *IEEE Trans. Acoust., Speech, Signal Process.*, 26:150–157, 1978.

[144] X. Chen and T. W. Parks. Design of IIR filters in the complex domain. *IEEE Trans. Acoust., Speech, Signal Process.*, 38(6):910–920, June 1990.

[145] W. S. Lu. Design of stable minimax IIR digital filters using semidefinite programming. In *The 2000 IEEE International Symposium on Circuits and Systems*, pages 355–358, Geneva, Switzerland, May 2000.

[146] Y. C. Lim, J. H. Lee, C. K. Chen, and R. H. Yang. A weighted least squares algorithm for quasi-equiripple FIR and IIR digital filter design. *IEEE Trans. Signal Process.*, 40:551–558, 1992.

[147] W. S. Lu, S. C. Soo-Chang Pei, and C. C. Tseng. A weighted least-squares method for the design of stable 1-D and 2-D IIR digital filters. *IEEE Trans. Signal Process.*, 46:1–10, January 1998.

[148] M. C. Lang. Least-squares design of IIR filters with prescribed magnitude and phase responses and a pole radius constraint. *IEEE Trans. Signal Process.*, 48(11):3109–

3121, November 2000.

[149] W. S. Lu. Design of stable IIR digital filters with equiripple passbands and peak-constrained least-squares stopbands. *IEEE Trans. Circuits Syst. II*, 46:1421–1426, November 1998.

[150] T. Matsunaga, M. Yoshida, and M. Ikehara. Design of IIR digital filters in the complex domain by transforming the desired response. *IEEE Trans. Signal Process.*, 52(7):1975–1982, July 2004.

[151] L. B. Jackson. Frequency-domain Steiglitz-McBride method for least-squares IIR filter design, ARMA modeling, and periodogram smoothing. *IEEE Signal Process. Lett.*, 15:49–52, 2008.

[152] T. Kobayashi and S. Imai. Design of IIR digital filters with arbitrary log magnitude function by WLS techniques. *IEEE Trans. Acoust., Speech, Signal Process.*, 38:247–252, February 1990.

[153] R. Vuerinckx, Y. Rolain, j. Schoukens, and R. Pintelon. Design of stable IIR filters in the complex domain by automatic delay selection. *IEEE Trans. Signal Process.*, 44(9):2339–2344, September 1996.

[154] A. Tarczynski and G. D. Cain. A new algorithm for designing near-optimal Chebyshev IIR and FIR filters. In *Proceedings of the 38th Midwest Symposium on Cicuits and Systems*, pages 584–587, Rio de Janeiro, Brazil, August 1995.

[155] G. Cortelazzo and M. R. Lightner. Simultaneous design in both magnitude and group delay of IIR and FIR filters based on multiple criterion optimization. *IEEE Trans. Acoust., Speech, Signal Process.*, 32:949–967, October 1984.

[156] A. Tarczynski, G. D. Cain, E. Hermanowicz, and M. Rojewski. A WISE method for designing IIR filters. *IEEE Trans. Signal Process.*, 49(7):1421–1432, July 2001.

[157] N. Z. Shor. The development of numerical methods for nonsmooth optimization in the USSR. *History of Mathematical Programming. A Collection of Personal Reminiscences*, pages 135–139, 1991.

[158] R. G. Bland, D. Goldfarb, and M. J. Todd. The ellipsoid method: A survey. *Operations Research*, 29(6):1039–1091, 1981.

[159] A. K. Aboagye. *Overflow Avoidance Techniques in Cascaded IIR Filter Implementations on the TMS320 DSPs*. Texas Instruments application report, 1999.

[160] Xilinx. *Virtex-II Pro^{TM} Platform FPGAs: Functional Description*, January 2003.

[161] GV Associates. *GVA-290 Xilinx Virtex-E Hardware Accelerator*, August 2009.

[162] A. Mehrnia and A. N. Willson. On optimal IFIR filter design. In *Proceedings of the IEEE International Symposium on Circuits and Systems (ISCAS'04)*, volume 3, pages 133–136, May 2004.

[163] T. Aulin. Characteristics of a digital mobile radio channel. *IEEE Trans. Veh. Technol.*, 30(2):45–53, May 1981.

[164] D. Laurenson, A. U. Sheikh, and S. McLaughlin. Characterization of the indoor mobile radio channel using a ray tracing technique. In *Proceedings of International Conference on Personal, Cordless and Mobile Communications*, pages 65–68, Vancouver, 1992.

[165] A. Mehrnia and H. Hashemi. Mobile satellite propagation channel. part 1-a comparative evaluation of current models. In *Proceedings of the Vehicular Technology Conference, 1999 (VTC'99)*, volume 5, pages 2775–2779, September 1999.

[166] A. M. O. Ribeiro and E. Conforti. Field measurements of the level crossing rate in the Nakagami mobile channel environment. In *Proceedings of the IEEE International Conference on Microwave and Optoelectronics*, pages 517–520, July 2005.

[167] A. Lakhzouri, E. S. Lohan, I. Saastamoinen, and M. Renfors. Measurement and characterization of satellite-to-indoor radio wave propagation channel. In *Proceedings of the European Navigation Conference (ENC-GNSS'05)*, Munich, Germany, July 2005.

[168] A. Lakhzouri, E. S. Lohan, I. Saastamoinen, and M. Renfors. Interference and indoor channel propagation modeling based on GPS satellite signal measurements. In *Proceedings of ION GPS*, pages 896–901, September 2005.

[169] J. C. S. S. Filho, M. D. Yacoub, and G. Fraidenraich. A simple accurate method for generating autocorrelated Nakagami-m envelope sequences. *IEEE Commun. Lett.*, 11(3):231–233, March 2007.

[170] K. W. Yip and T. S. Ng. A simulation model for Nakagami-m fading channels, $m < 1$. *IEEE Trans. Commun.*, 48(2):214–221, February 2000.

[171] Q¿ T. Zhang. A decomposition technique for efficient generation of correlated Nakagami fading channels. *IEEE J. Sel. Areas Commun.*, 18:2385–2392, November 2000.

[172] Z. Song, K. Zhang, and Y. L. Guan. Generating correlated Nakagami fading signals with arbitrary correlation and fading parameters. In *Proceedings of the IEEE In-*

ternational Conference on Communications (ICC'02), volume 3, pages 1363–1367, 2002.

[173] K. Zhang, Z. Song, and Y. L. Guan. Simulation of Nakagami fading channels with arbitrary cross-correlation and fading parameters. *IEEE Trans. Wireless Commun.*, 3(5):1463–1468, September 2004.

[174] C. D. Iskander and P. T. Mathiopoulos. Finite-state Markov modeling of diversity Nakagami channels. In *Proceeding of the 7th Canadian Workshop on Information Theory*, Vancouver, BC, 2001.

[175] C. D. Iskander and P. T. Mathiopoulos. Fast simulation of diversity Nakagami fading channels using finite-state Markov models. *IEEE Trans. Broadcast.*, 49(3):269–277, September 2003.

[176] E. Pajala, T. Isotalo, A. Lakhzouri, E. S. Lohan, and M. Renfors. An improved simulation model for Nakagami-m fading channels for satellite positioning applications. In *Proceedings of the 3rd Workshop on Positioning, Navigation, and Communication (WPNC'06)*, pages 81–89, Hannover, Germany, March 2006.

[177] C. H. Sim. Generation of Poisson and Gamma random vectors with given marginal and covariance matrix. *Journal of statistical Computation and simulation*, 47(1-2):1–10, 1993.

[178] F. Babich and G. Lombardi. Statistical analysis and characterization of the indoor propagation channel. *IEEE Trans. Commun.*, 48(3):455–464, March 2000.

[179] M. A. Taneda, J. Takada, and K. Araki. A new approach to fading: Weibull model. In *Proceedings of the IEEE International Symposium on Personal Indoor Mobile Radio Communictions*, pages 711–715, September 1999.

[180] N. C. Sagias and G. K. Karagiannidis. Gaussian class multivariate Weibull distributions: Theory and applications in fading channels. *IEEE Trans. Inform. Theory*, 51(10):3608–3619, October 2005.

[181] J. G. Proakis. *Digital communications*. McGraw-Hill, New York, 4th edition edition, 2001.

[182] C. D. Iskander and P. T. Mathiopoulos. Analytical envelope correlation and spectrum of maximal-ratio combined fading signals. In *IEEE Pacific Rim Conference on Communications, Computers and signal Processing, PACRIM'03*, volume 1, pages 446–449, August 2003.

[183] I. S. Grandshteyn and I. M. Ryzhik. *Table of integrals, series, and products*. Aca-

demic press, 6th edition edition, 2000.

[184] G. J. Foschini and M. J. Gans. On limits of wireless communications in a fading environment when using multiple antennas. In *Proceedings of Wireless Personal Communications*, volume 6, pages 311–335, March 1998.

[185] E. Telatar. Capacity of multi-antenna Gaussian channels. *Euro. Trans. Telecom.*, 10(6):585–595, 1999.

[186] Broadcom Corporation, 5300 California Ave., Irvine, CA. *BCM4705 Gigabit MIMO Processor*.

[187] Qualcomm, 5775 Morehouse Drive, San Diego, CA. *Qualcomm N-Stream MIMO Wireless WCN1320 Chip*.

[188] Beceem Communications, 3960 Freedom Circle, Santa Clara, CA. *Beceem BCS5200 WiMAX CPE MIMO solution*.

[189] Belkin International, 501 West Walnut Street, Compton, CA. *Belkin N1 Wireless Router*.

[190] A. F. Molisch. *Wireless Communications*. Wiley-IEEE Press, New York, NY, USA, 2005.

[191] A. F. Molisch, H. Asplund, R. Heddergott, M. Steinbauer, and T. Zwick. The COST 259 directional channel model Part-I: overview and methodology. *IEEE Trans. Wireless Commun.*, 5(12):3421–3433, 2006.

[192] G. Tsoulos, editor. *MIMO system technology for wireless communications*. CRC Press, 2006.

[193] J. P. Kermoal, L. Schumacher, P. E. Mogensen, and K. I. Pedersen. Experimental investigation of correlation properties of MIMO radio channels for indoor picocell scenarios. In *Proceedings of the IEEE Vehicular Technology Conference, IEEE VTC Fall*, volume 1, pages 14–21, 2000.

[194] K. I. Pedersen, J. B. Andersen, J. P. Kermoal, and P. Mogensen. A stochastic multiple-input-multiple-output radio channel model for evaluation of space-time coding algorithms. In *Proceedings of the IEEE Vehicular Technology Conference (VTC'00)*, volume 2, pages 893–897, 2000.

[195] K. Yu, M. Bengtsson, B. Ottersten, D. McNamara, P. Karlsson, and M. Beach. A wideband statistical model for NLOS indoor MIMO channels. In *Proceedingsof the IEEE Vehicular Technology Conference. IEEE VTC Spring*, volume 1, pages 370–

374, 2002.

[196] K. Yu, M. Bengtsson, B. Ottersten, D. McNamara, P. Karlsson, and M. Beach. A 20 MHz HiperLAN/2 MIMO channel model in NLOS indoor scenarios. In *Proceedings of the Konferensen RadioVetenskap och Kommunikation (RVK02)*, pages 311–315, 2002.

[197] M. Stege, J. Jelitto, M. Brozel, and G. Fettweis. A multiple input-multiple output channel model for simulation of Tx- and Rx-diversity wireless systems. In *Proceedings of the IEEE Vehicular Technology Conference (VTC'00)*, volume 2, pages 833–839, 2000.

[198] L. M. Correia. *Wireless flexible personalised communications*. John Wiley, 2001.

[199] T. Svantesson. A physical MIMO radio channel model for multi-element multi-polarized antenna systems. In *Proceedings of the IEEE Vehicular Technology Conference (VTC'01)*, volume 2, pages 1083–1087, 2001.

[200] D-S. Shiu, G. J. Foschini, M. J. Gans, and J. M. Kahn. Fading correlation and its effect on the capacity of multielement antenna systems. *IEEE Trans. Commun.*, 48(3):502 – 513, Mar 2000.

[201] D. Gesbert, H. Bölcskei, D. Gore, and A. Paulraj. MIMO wireless channels: capacity and performance prediction. In *Proceedings of the IEEE Global Telecommunications Conference*, volume 2, pages 1083–1088, November 2000.

[202] K. Yu, M. Bengtsson, B. Ottersten, P. Karlsson, D. McNamara, and M. Beach. Measurement analysis of NLOS indoor MIMO channels. In *Proceedings of the IST Mobile Communications Summit*, pages 277–282, September 2001.

[203] K. Yu, M. Bengtsson, B. Ottersten, D. McNamara, P. Karlsson, and M. Beach. Second order statistics of NLOS indoor MIMO channels based on 5.2 GHz measurements. In *Proceedings of the IEEE Global Telecommunications Conference*, volume 1, pages 156–160, November 2001.

[204] J. W. Wallace and M. A. Jensen. Statistical characteristics of measured MIMO wireless channel data and comparison to conventional models. In *Proceedings IEEE Vehicular Technology Conference. IEEE VTC Fall*, volume 2, pages 1078–1082, 2001.

[205] A. Abdi and M. Kaveh. Space-time correlation modeling of multielement antenna systems in mobile fading channels. In *Proceedings International Conference on Acoustics, Speech, and Signal Processing (ICASSP'01)*, volume 4, pages 2505–2508, May 2001.

[206] A. Abdi and M. Kaveh. A space-time correlation model for multielement antenna systems in mobile fading channels. *IEEE J. Sel. Areas Commun.*, 20(3):550–560, April 2002.

[207] A. Sayeed. Modeling and capacity of realistic spatial MIMO channels. In *Proceedings of the IEEE International Conference on Acoustics, Speech, and Signal Processing (ICASSP'01)*, volume 4, pages 2489–2492, 2001.

[208] P. Almers et al. Survey of channel and radio propagation models for wireless MIMO systems. *EURASIP Journal on Wireless Communications and Networking*, 2007:1–19, 2007.

[209] H. Özcelik. *Indoor MIMO channel models*. PhD thesis, Vienna University of Technology,, Vienna, Austria, 2004.

[210] A. G. Burr. Capacity bounds and estimates for the finite scatterers MIMO wireless channel. *IEEE J. Sel. Areas Commun.*, 21(5):812–818, June 2003.

[211] B. Clerckx and C. Oestges. Space-time code design for correlated Ricean MIMO channels at finite SNR. *IEEE Trans. Signal Process.*, 56(9):4365–4376, September 2008.

[212] C N. Chuah, D. N. C. Tse, J. M. Kahn, and R. A. Valenzuela. Capacity scaling in MIMO wireless systems under correlated fading. *IEEE Trans. Inform. Theory*, 48(3):637–650, March 2002.

[213] D. Chizhik, F. Rashid-Farrokhi, J. Ling, and A. Lozano. Effect of antenna separation on the capacity of BLAST of correlated channels. *IEEE Commun. Lett.*, 4(11):337–339, November 2000.

[214] B. Holter and G. E. Oien. On the amount of fading in MIMO diversity systems. *IEEE Trans. Wireless Commun.*, 4(5):2498–2507, September 2005.

[215] J. W. Wallace and M. A. Jensen. Modeling the indoor MIMO wireless channel. *IEEE Trans. Antennas Propag.*, 50(5):591–599, 2002.

[216] A. Molisch. A generic model for MIMO wireless propagation channels in macro- and microcells. *IEEE Trans. Signal Process.*, 52(1):61–71, January 2004.

[217] M. Zhang, P. J. Smith, and M. Shafi. An extended one-ring MIMO channel model. *IEEE Trans. Wireless Commun.*, 6(6):2759–2764, August 2007.

[218] M. Pätzold and B. O. Hogstad. A space-time channel simulator for MIMO channels based on the geometrical one-ring scattering model. *Wireless Communications and*

Mobile Computing Special Issue on Multiple-Input Multiple-Output (MIMO) Communications, 4:727–737, November 2004.

[219] G. J. Byers and F. Takawira. Spatially and temporally correlated MIMO channels: Modeling and capacity analysis. *IEEE Trans. Veh. Technol.*, 53(3):634–643, May 2004.

[220] S. Wang, A. Abdi, J. Salo, H. M. El-Sallabi, J. W. Wallace, P. Vainikainen, and M. A. Jensen. Time-varying MIMO channels: Parametric statistical modeling and experimental results. *IEEE Trans. Veh. Technol.*, 56(4):1949–1963, July 2007.

[221] M. Pätzold and B. O. Hogstad. A wideband MIMO channel model derived from the geometric elliptical scattering model. In *Proceedings of the International Symposium on Wireless Communication Systems*, pages 138–143, 2006.

[222] S. Loredo, A. Rodríguez-Alonso, and R. P. Torres. Indoor MIMO channel modeling by rigorous GO/UTD-based ray tracing. *IEEE Trans. Veh. Technol.*, 57(2):680–692, March 2008.

[223] J. P. Kermoal, L. Schumacher, P. E. Mogensen, and K. I. Pedersen. A stochastic MIMO radio channel model with experimental validation. *IEEE J. Sel. Areas Commun.*, 20(6):1211–1226, 2002.

[224] K. I. Pedersen, P. E. Mogensen, and B. H. Fleury. A stochastic model of the temporal and azimuthal dispersion seen at the base station in outdoor propagation environments. *IEEE Trans. Veh. Technol.*, 49:437–477, 2000.

[225] J. Fuhl, A. Molisch, and E. Bonek. Unified channel model for mobile radio systems with smart antennas. In *IEE Proceedings of Radar, Sonar and Navigation,*, volume 145, pages 32–41, 1998.

[226] W. Weichselberger, M. Herdin, H. Özcelik, and E. Bonek. A stochastic MIMO channel model with joint correlation of both link ends. *IEEE Trans. Wireless Commun.*, 5(1):90–99, 2006.

[227] A. M. Sayeed. Deconstructing multiantenna fading channels. *IEEE Trans. Signal Process.*, 50(10):2563–2579, 2002.

[228] Z. Hong, K. Liu, R. W. Heath, and A. M. Sayeed. Spatial multiplexing in correlated fading via the virtual channel representation. *IEEE J. Sel. Areas Commun.*, 21(5):856–866, June 2003.

[229] M. Debbah and R. R. Müller. MIMO channel modeling and the principle of maximum entropy. *IEEE Trans. Inform. Theory*, 51(5):1667–1690, 2005.

[230] M. Guillaud, M. Debbah, and A. L. Moustakas. A maximum entropy characterization of spatially correlated MIMO wireless channels. In *Proceedings of the Wireless Communications and Networking Conference*, pages 1039–1044, March 2007.

[231] J. Ling, D. Chizhik, and R. A. Valenzuela. Predicting multielement receive & transmit array capacity outdoors with ray tracing. In *Proceedings of the IEEE Vehicular Technology Conference (VTC'01)*, volume 1, pages 392–394, Rhodes, Greece, May 2001.

[232] C. Cheon, G. Liang, and H. L. Bertoni. Simulating radio channel statistics for different building environments. *IEEE J. Sel. Areas Commun.*, 19(11):2191–220, 2001.

[233] K. Kuroda, K. Sakaguchi, J. C. Takada, and K. Araki. FDM based MIMO spatiotemporal channel sounder. In *Proceedings of the 5th International Symposium on Wireless Personal Multimedia Communications*, volume 2, pages 27–30, October 2002.

[234] P. Goud, R. Hang, D. Truhachev, and C. Schlegel. A portable MIMO testbed and selected channel measurements. *EURASIP Journal on Applied Signal Processing*, 2006:1–10, 2006.

[235] M. Steinbauer et al. Array measurement of the double-directional mobile radio channel. In *Proceedings of rht IEEE Vehicular Technology Conference (VTC'00)*, volume 3, pages 1656–1662, Tokyo, Japan, 2000.

[236] T. Zwick, C. Fischer, and W. Wiesbeck. A stochastic multipath channel model including path directions for indoor environments. *IEEE J. Sel. Areas Commun.*, 20(6):1178–1192, 2002.

[237] A. Saleh and R. Valenzuela. A statistical model for indoor multipath propagation. *IEEE J. Sel. Areas Commun.*, 5(2):128–137, February 1987.

[238] C. C. Chong, C. M. Tan, D. Laurenson, S. McLaughlin, M. A. Beach, and A. R. Nix. A new statistical wideband spatiotemporal channel model for 5-GHz band WLAN systems. *IEEE J. Sel. Areas Commun.*, 21(2):139–150, 2003.

[239] M. Steinbauer. *The radio propagation channel - a non-directional, directional, and double-directional point-of-view*. PhD thesis, Vienna University of Technology, Vienna, Austria, 2001.

[240] M. Steinbauer, A. Molisch, and E. Bonek. The double-directional radio channel. *IEEE Antennas Propag. Mag.*, 43(4):51–63, 2001.

[241] A. F. Molisch, A. Kuchar, J. Laurila, K. Hugl, and R. Schmalenberger. Geometry-

based directional model for mobile radio channels - principles and implementation. *Euro. Trans. Telecom.*, 14(4):351–359, 2003.

[242] W. C. Y. Lee. Effects on correlation between two mobile radio base-station antennas. *IEEE Trans. Commun.*, 21:1214–1224, 1973.

[243] J. J. Blanz and P. Jung. A flexibly configurable spatial model for mobile radio channels. *IEEE Trans. Commun.*, 47(3):367–371, March 1998.

[244] O. Norklit, and J. B. Andersen. Diffuse channel model and experimental results for array antennas in mobile environments. *IEEE Trans. Antennas Propag.*, 46(6):834–840, June 1998.

[245] P. Petrus, J. H. Reed, and T. S. Rappaport. Geometrical-based statistical macrocell channel model for mobile environments. *IEEE Trans. Commun.*, 50(3):495–502, March 2002.

[246] Y. Z. Mohasseb and M. P. Fitz. A 3-D spatio-temporal simulation model for wireless channels. *IEEE J. Sel. Areas Commun.*, 20(6):1193– 1203, August 2002.

[247] C. Oestges, V. Erceg, and A. J. Paulraj. A physical scattering model for MIMO macrocellular broadband wireless channels. *IEEE J. Sel. Areas Commun.*, 21(5):721–729, June 2003.

[248] J. Laurila, A. F. Molisch, and E. Bonek. Influence of the scatterer distribution on power delay profiles and azimuthal power spectra of mobile radio channels. In *Proceedings of the 5th International Symposium on Spread Spectrum Techniques and Applications (ISSSTA'98)*, volume 1, pages 267–271, September 1998.

[249] H. Suzuki. A statistical model for urban radio propagation. *IEEE Trans. Commun.*, 25(7):673–680, July 1977.

[250] C. Bergljung and P. Karlsson. Propagation characteristics for indoor broadband radio access networks in the 5 GHz band. In *Proceedings of the 9th IEEE International Symposium on Personal, Indoor and Mobile Radio Communications (PIMRC'98)*, volume 2, pages 612–616, September 1998.

[251] A. Kuchar, J. P. Rossi, and E. Bonek. Directional macro-cell channel characterization from urban measurements. *IEEE Trans. Antennas Propag.*, 48(2):137–146, 2000.

[252] F. Adachi, M. T. Feeney, A. G. Williamson, and J. D. Parsons. Cross-correlation between the envelopes of 900 MHz signals received at a mobile radio base station site. *IEE Proc. Commun., Radar, Signal Processing*, 133:506–512, 1986.

[253] T. S. Chu and L. J. Greenstein. A semi-empirical representation of antenna diversity gain at cellular and PCS base stations. *IEEE Trans. Commun.*, 45:644–646, 1997.

[254] U. Martin. Spatio-temporal radio channel characteristics in urban macrocells. *IEE Proc. Radar, Sonar, Navig.*, 145:42–49, 1998.

[255] P. Pajusco. Experimental characterization of D.O.A at the base station in rural and urban area. In *Proceedings of the IEEE Vehicular Technology Conference*, pages 993–997, 1998.

[256] Y. C. Ko, A. Abdi, M. S. Alouini, and M. Kaveh. Average outage duration of diversity systems over generalized fading channels. In *Proceedings of the IEEE Wireless Communications and Networking Conference*, pages 216–221, 2000.

[257] C. Tepedelenlioglu, A. Abdi, G. Giannakis, and M. Kaveh. Estimation of Doppler spread and signal strength in mobile communications with applications to handoff and adaptive transmission. *Wireless Commun. Mob. Comput.*, 1:221–242, 2001.

[258] G. J. Byers and F. Takawira. The influence of spatial and temporal correlation on the capacity of MIMO channels. In *Proceedings of Wireless Communications and Networking 2003, (WCNC'03)* , pages 359–364, March 2003.

[259] A. Abdi and M. Kaveh. A versatile spatio-temporal correlation function for mobile fading channels with non-isotropic scattering. In *Proceedings of the IEEE Workshop on Statistical Signal Array Processing*, pages 58–62, 2000.

[260] M. Pätzold and N. Youssef. Modelling and simulation of directionselective and frequency-selective mobile radio channels. *International Journal of Electronics and Communications*, 55(6):433–442, November 2001.

[261] C. H. Lee. A novel space-time MIMO channel model. In *Proceedings of the IEEE International Conference on Hybrid Information Technology (ICHIT'06)*, volume 1, pages 599–605, November 2006.

[262] D. Chizhik, G. J. Foschini, M. J. Ganz, and R. A. Valenzuela. Keyholes, correlations, and capacities of multielement transmit and receive antennas. *IEEE Trans. Commun.*, 1(4):361–368, April 2002.

[263] D. Gesbert, H. Bölcskei, D. Gore, and A. Paulraj. Outdoor MIMO wireless channels: models and performance prediction. *IEEE Trans. Commun.*, 50(12):1926–1934, December 2002.

[264] P. Almers, F. Tufvesson, and A. F. Molisch. Keyhole effect in MIMO wireless channels: Measurements and theory. *IEEE Trans. Commun.*, 5(12):3596–3604, Decem-

ber 2006.

[265] M. Pätzold, B. 0. Hogstad, N. Youssef, and D. Kim. A MIMO mobile-to-mobile channel model: part i - the reference model. In *Proceedings of the 16th IEEE International Symposium on Personal, Indoor and Mobile Radio Communications, (PIMRC'05)*, volume 1, pages 573–578, September 2005.

[266] B. O. Hogstad, M. Pätzold, N. Youssef, and D. Kim. A MIMO mobile-to-mobile channel model: Part II - The simulation model. In *Proceedings of the 16th IEEE International Symposium on Personal, Indoor and Mobile Radio Communications, (PIMRC'05)*, volume 1, pages 562–567, September 2005.

[267] K. Yu and B. Ottersten. Models for MIMO propagation channels, a review. *Wireless Commun. Mobile Comput.*, 2(7):653–666, 2002.

[268] D. Chizhik, J. Ling, P. W. Wolniansky, R. A. Valenzuela, N. Costa, and K. Huber. Multiple-input-multiple-output measurements and modeling in Manhattan. *IEEE J. Sel. Areas Commun.*, 21(4):321–331, April 2003.

[269] D. P. McNamara, M. A. Beach, and P. N. Fletcher. Spatial correlation in indoor MIMO channels. In *Proceeding of the IEEE International Symposium of Personal, Indoor and Mobile Radio Communications (PIMRC'02)*, volume 1, pages 290–294, 2002.

[270] D. P. McNamara, M. A. Beach, P. N. Fletcher, and P. Karlsson. Initial investigation of multiple-input multiple-output channels in indoor environments. In *Proceedings of the IEEE Benelux Chapter Symposium on Communications and Vehicular Technology*, pages 139–143, October 2000.

[271] W. Weichselberger, H. Özcelik, M. Herdin, and E. Bonek. A novel stochastic MIMO channel model and its physical interpretation. In *Proceedings of the International Symposium on Wireless Personal Multimedia Communications (WPMC'03)*, October 2003.

[272] H. Özcelik, N. Czink, and E. Bonek. What makes a good MIMO channel model? In *Proceedings of the IEEE Vehicular Technology Conference (VTC'05)*, volume 1, pages 156–160, June 2005.

[273] L. Wood and W. S. Hodgkiss. Understanding the Weichselberger model: A detailed investigation. In *Proceedings of the IEEE Military Communications Conference (MILCOM'08)*, pages 1–7, November 2008.

[274] S. Wyne, A. F. Molisch, P. Almers, G. Eriksson, J. Karedal, and F. Tufvesson.

Outdoor-to-indoor office MIMO measurements and analysis at 5.2 GHz. *IEEE Trans. Veh. Technol.*, 57(3):1374–1386, May 2008.

[275] C. Heegard and S. Wicker. *Turbo Coding*. Kluwer, Massachusetts, 1999.

[276] B. Sklar. *Digital Communications, Fundamentals and Applications*. Prentice Hall, Englewood Cliffs, NJ, 1993.

[277] D. G. Hoffman, D. A. Leonard, C. C. Lindner, K. T. Phelps, C. A. Rodger, and J. R. Wall, editor. *Coding Theory, The Essentials*. Marcel Dekker, Inc., New York, NY, 1991.

[278] S. Lin and D. J. Costello. *Error Control Coding*. Prentice Hall, Upper Saddle River, NJ, 2 edition, 2004.

[279] V. Välimäki and T. I. Laakso. Principles of fractional delay filters. In *Proceedings of the IEEE International Conference on Acoustics, Speech, and Signal Processing (ICASSP'00)*, volume 6, pages 3870–3873, 2000.

[280] J. Vesma and T. Saramaki. Optimization and efficient implementation of FIR filters with adjustable fractional delay. In *Proceedings of IEEE International Symposium on Circuits and Systems (ISCAS'97)*, volume 4, pages 2256–2259, 1997.

[281] J. S. Park, B. K. Kim, J. G. Chung, and K. K. Parhi. High-speed tunable fractional-delay allpass filter structure. In *Proceedings of the International Symposium on Circuits and Systems (ISCAS'03)*, volume 4, pages 165–168, 2003.

[282] T. I. Laakso, V. Välimäki, M. Karjalainen, and U. K. Laine. Splitting the unit delay. *IEEE Signal Process. Mag.*, 13(1):30–60, January 1996.

[283] L. Erup, F. M. Gardner, and R. A. Harris. Interpolation in digital modems - part II: Implementation and performance. *IEEE Trans. Commun.*, 41(6):998–1008, June 1993.

[284] C. W. Farrow. A continuously variable digital delay element. In *Proceedings of the IEEE Interanational Symposium on Circuits and Systems (ISCAS'88)*, pages 2641–2645, June 1988.

[285] G. F. M. D. Yacoub and J. S. Filho. Nakagami-m phase-envelope joint distribution. *IEE Electronics Letters*, 41:259–261, March 2005.

Appendix A

List of Publications

Co-Authorship Statement

The research on fading channel simulation at the HCDC and VLSI labs was first started by Dr. Amirhossein Alimohammad and Dr. Bruce Cockburn. Their original research was focused on hardware simulation of isotropic single antenna fading channels using the sum-of-sinusoids method and the filter-based approach.

In this thesis, the original work was extended in several directions. We proposed especially compact fading channel simulators that went beyond Dr. Alimohammad's work. We proposed faster, more compact (more than 95%), and more accurate fading simulators for isotropic and non-isotropic scattering. We also proposed elastic designs, multiplication-free designs, differential designs, and multiple-antenna fading simulators. We also covered fractional delay simulation and proposed a new design procedure for fixed-point filter design.

The publications that originated from this work incorporate materials that result from joint work. Dr. Bruce Cockburn, my supervisor, provided us with supervision, guidance, new ideas, and overall verification of the publications. Dr. Amirhossein Alimohammad helped us with new ideas, simulation, structure, and verification of our publications. Finally, Dr. Christian Schlegel was my co-supervisor and provided us with help and guidance throughout our work.

(1) Patent

[1] A. Alimohammad, S. Fouladi Fard, B. F. Cockburn, and C. Schlegel, "Signal filtering and filter design techniques," *International Patent Application,* Reference No. 2006045 (PCT), May 2008.

(2) Fading simulation

[1] S. Fouladi Fard, A. Alimohammad, B. F. Cockburn, and C. Schlegel, "A flexible FPGA-based MIMO geometric fading channel simulator for rapid prototyping," in *Proceedings of the International Conference on Field-Programmable Technology (to appear)*, 2009. (**Conference**)

[2] S. Fouladi Fard, A. Alimohammad, B. F. Cockburn, and C. Schlegel, "A versatile fading simulator for on-chip verification of MIMO communication systems," in *Proceedings of the IEEE International System on Chip Conference (to appear)*, 2009. (**Conference**)

[3] S. Fouladi Fard, A. Alimohammad, B. F. Cockburn, and C. Schlegel, "High path-count multirate Rayleigh fading channel simulator with time-multiplexed datapath," in *Proceedings of the IEEE International System on Chip Conference (to appear)*, 2009. (**Conference**)

[4] S. Fouladi Fard, A. Alimohammad, B. F. Cockburn, and C. Schlegel, "A single FPGA filter-based multipath emulator," in *Proceedings of the IEEE Global Communications Conference (to appear)*, 2009. (**Conference**)

[5] S. Fouladi Fard, A. Alimohammad, B. F. Cockburn, and C. Schlegel, "A compact hardware simulator for geometric MIMO fading channel models," submitted to *IET Circuits, Devices and Systems,* 2009. (**Journal**)

[6] S. Fouladi Fard, A. Alimohammad, B. F. Cockburn, and C. Schlegel, "Filter design for non-isotropic fading channels with arbitrary temporal correlation," submitted to *IET Signal Processing,* 2009. (**Journal**)

[7] S. Fouladi Fard, A. Alimohammad, B. F. Cockburn, and C. Schlegel, "Ultra-compact FPGA simulator for high path-count Rayleigh and Rician fading," submitted to *IET Communications,* 2009. (**Journal**)

[8] S. Fouladi Fard, A. Alimohammad, B. F. Cockburn, and C. Schlegel, "Efficient hardware simulation of Nakagami-m and Weibull fading channels," submitted to *IET Communications,* 2009. (**Journal**)

[9] S. Fouladi Fard, A. Alimohammad, B. F. Cockburn, and C. Schlegel, "Iterative barrier-constrained complex IIR filter design for compact and stable hardware implementation," submitted to the *IEEE Transactions on Circuits and Systems I,* 2009. (**Journal**)

[10] S. Fouladi Fard, A. Alimohammad, M. Khorasani, C. Schlegel, and B. F. Cockburn. "A compact and accurate FPGA based nonisotropic fading channel simulator," In *Proceedings of the Canadian Conference on Electrical and Computer Engineering, 2007 (CCECE'07)*, pp. 1239-1242, April 2007. (**Conference**)

[11] A. Alimohammad, S. Fouladi Fard, B. F. Cockburn, and C. Schlegel, "A compact fading channel simulator using timing-driven resource sharing," In *Proceedings of the IEEE International Conference on Application-specific Systems, Architectures and Processors (ASAP'07),* pp. 154-159, April 2007. (**Conference**)

[12] A. Alimohammad, S. Fouladi Fard, B. F. Cockburn, and C. Schlegel, "A compact single-FPGA fading channel simulator," *IEEE Transactions on Circuits and Systems II*, Vol. 55, No. 1, pp. 84-88, January 2008. (**Journal**)

[13] A. Alimohammad, S. Fouladi Fard, B. F. Cockburn, and C. Schlegel, "A flexible filter processor for fading channel emulation," In *IEEE Symposium on Field-Programmable Custom Computing Machines (FCCM'07)*, pp. 339-342, April 2007. (**Conference**)

[14] A. Alimohammad, S. Fouladi Fard, B. F. Cockburn, and C. Schlegel, "An improved SOS-based fading channel emulator," In *Proceedings of the IEEE 66th Vehicular Technology Conference (VTC'07),* pp. 931-935, 2007. (**Conference**)

[15] A. Alimohammad, S. Fouladi Fard, B. F. Cockburn, and C. Schlegel, "A Compact Rayleigh and Rician fading simulator based on random-walk processes," in *IET Electronic Letters (to appear),* 2009. (**Journal**)

[16] A. Alimohammad, S. Fouladi Fard, B. F. Cockburn, and C. Schlegel, "An accurate and compact Rayleigh and Rician fading channel simulator," In *Proceedings of the IEEE Vehicular Technology*

Conference, pp. 409-413, 2008. (**Conference**)

[17] A. Alimohammad, S. Fouladi Fard, B. F. Cockburn, and C. Schlegel, "A novel technique for efficient hardware simulation of spatiotemporally correlated MIMO fading channels," In *Proceedings of the IEEE International Conference on Communications,* pp. 718-724, 2008. (**Conference**)

[18] A. Alimohammad, S. Fouladi Fard, B. F. Cockburn, and C. Schlegel, "A single-FPGA multipath MIMO fading channel simulator," In *Proceedings of the IEEE International Symposium on Circuits and Systems,* pp. 308-311, May 2008. (**Conference**)

(3) Random number generation

[1] A. Alimohammad, S. Fouladi Fard, B. F. Cockburn, and C. Schlegel, "A compact and accurate Gaussian variate generator," *IEEE Transactions on Very Large Scale Integration (VLSI) Systems*, Vol. 16, No. 5, pp. 517527, May 2008. (**Journal**)

[2] A. Alimohammad, S. Fouladi Fard, B. F. Cockburn, and C. Schlegel, "Compound uniform random number generators with on-chip correlation and distribution measurements," In *Proceedings of the IEEE International Conference on Field-Programmable Technology (FPT'07),* pp. 377-380, December 2007. (**Conference**)

[3] A. Alimohammad, S. Fouladi Fard, B. F. Cockburn, and C. Schlegel, "On the efficiency and accuracy of hybrid pseudo-random number generators for FPGA-based simulations," In *Proceedings of the IEEE International Parallel and Distributed Processing Symposium (IPDPS'08),* pp. 1-8, 2008. (**Conference**)

(4) Testing communication systems

[1] A. Alimohammad, S. Fouladi Fard, B. F. Cockburn, "FPGA-based accelerator for the verification of leading-edge wireless systems," in *Proceedings of the IEEE Design Automation Conference,* pp. 1-4, 2009. (**Conference**)

[2] A. Alimohammad, S. Fouladi Fard, B. F. Cockburn, "A flexible layered architecture for accurate digital baseband algorithm development and verification," in *Proceedings of the Design, Automation and Test in Europe (DATE'09),* pp. 1-6, 2009. (**Conference**)

[3] A. Alimohammad, S. Fouladi Fard, B. F. Cockburn, "FPGA-accelerated baseband design and verification of broadband wireless systems," in *Proceedings of the Workshop on Test of Wireless Circuits and Systems (to appear),* 2009. (**Conference**)

[4] A. Alimohammad, S. Fouladi Fard, B. F. Cockburn, "FPGA-accelerated baseband design and verification of broadband MIMO wireless systems," in *Proceedings of the IEEE International Conference on Advances in System Testing and Validation (to appear),* 2009. (**Conference**)

[5] A. Alimohammad, S. Fouladi Fard, B. F. Cockburn, "Hardware-based error rate testing of digital baseband communication systems," in *Proceedings of the IEEE International Test Conference,* pp. 1-10, 2008. (**Conference**)

(5) MIMO systems

[1] A. Alimohammad, S. Fouladi Fard, B. F. Cockburn, and C. Schlegel, "An improved layered MIMO detection algorithm with near-optimal performance," in *IET Electronic Letters (to appear),* 2009. (**Journal**)