Electr Eng (2011) 93:83–90
DOI 10.1007/s00202-010-0190-2

ORIGINAL PAPER

Hardware realization of a low-complexity fading filter for multipath Rayleigh fading simulator

Serdar Özen · Ali Arsal · Kadir Atilla Toker

Received: 3 July 2009 / Accepted: 15 November 2010 / Published online: 12 December 2010

Abstract A low-complexity high performance Rayleigh fading simulator, and its Field Programmable Gate Array (FPGA) implementation are presented. This proposed method is a variant of the method of filtering of the white Gaussian noise where the filter design is accomplished in the analog domain and transferred into digital domain. The proposed model is compared with improved Jakes' model, auto-regressive (AR) filtering, existing auto-regressive moving-average (ARMA) filtering techniques, and inverse discrete Fourier transform (IDFT)-based techniques, in performance and computational complexity. The proposed method outperforms AR(20) filter and modified Jakes' generators in performance. Although IDFT method achieves the best performance, it brings a significant cost in storage. The proposed method achieves high performance with the lowest complexity, and its performance has been verified on commercially available FPGA platforms. Our fixed-point Rayleigh fading-channel emulator uses only 2% of the configurable slices, 1% of the Look-Up-Table (LUT) resources and 3% of the dedicated multipliers on the FPGA platform that has been used.

Keywords Multipath fading · Fading filter · Rayleigh fading simulator · ARMA filtering · Quantized filtering · FPGA · Emulator · Complexity

S. Özen (✉)
Department of Electrical and Electronics Engineering,
Izmir Institute of Technology, 35430 İzmir, Turkey
e-mail: serdarozen@iyte.edu.tr

A. Arsal
Faculty of Engineering and Natural Sciences,
Sabanci University, 34956 İstanbul, Turkey

K. A. Toker
Izmir University, 35290 İzmir, Turkey

1 Introduction

In this paper, a low-complexity prototype hardware architecture for modeling of Rayleigh fading process is developed targeting at the Xilinx [1] Virtex4-xc4vsx35 and Spartan3e - xc3s500e Field Programmable Gate Array (FPGA) development platforms. FPGAs offer a lot of flexibility, fixed-point arithmetic units with parameterized precisions, variable-length registers, numerous dedicated signal processing cores, speed, reliability, low turn-around time in the design phase, and very cheap alternative to the expensive emulators available on the market [2,3]. The FPGA realization of the proposed method has been implemented using constant coefficient multipliers and the theoretical performance of the proposed method has been verified using this FPGA implementation. Hardware-based emulators can greatly reduce the simulation time compared to software-based simulators [4]. Many laboratory channel simulation tools use hybrid DSP/FPGA solutions [5,6] or stand-alone FPGAs to generate wireless multipath channel models [7,8]. A much more cost-effective approach is to implement the entire emulator on a single FPGA chip [7]. The main contributions of our work are: (1) a novel low-complexity filter design scheme for fading-channel simulators with quantitative performance analysis; (2) demonstration of the hardware implementation of our filter-based simulator, that produces fading channel characteristics, realized on a portion of a commercially available FPGA platform.

The fading caused by multipath propagation in wireless communication systems is commonly modeled as a random process having Rayleigh distributed envelope, and is characterized by its power spectral density and its auto-correlation function. In the communications literature, Jakes' model [9] has been of great interest which is based on sum of sinusoids approach. Simulators based on white noise filtering methods [10,11] and on the Inverse Discrete Fourier Transform (IDFT) method [12,13] have also become popular. It was shown in [14] that the fading signals which are produced by classical Jakes' simulator are not wide-sense stationary (WSS). On the other hand simulators based on the IDFT method are of high-quality and efficient. A disadvantage of the IDFT method is that all samples are generated with a single fast Fourier transform (FFT), hence the storage requirements make it useless for the generation of very large number of samples and for sample-by-sample simulations. In this paper, we consider using a fading filter to filter white Gaussian noise using an Infinite Impulse Response (IIR) filter [10,15,16]. Unlike other filter structures [9,11–13,17], a different optimization and design criterion is used to set the filter parameters in the analog domain. This optimization yields the transfer function of the fading filter in the analog domain. Bilinear transform is then used to get the desired filter structure as an ARMA(γ, γ) filter, where γ is the filter order. Comparisons to other methods are made in terms of complexity and also in terms of performance using quantitative performance measures introduced in [18]. The quantitative performance measures [18] have not been investigated in the previous similar studies [6–8]. More recent work [19] proposes an ARMA(p,q) synthesis of the Rayleigh fading variates by first obtaining the AR(p) order filter similar to that of [11] via solving the Yule-Walker equations, followed by the determination of the ARMA(p,q) filter using the system identification procedures of [20]. Although the method of [19] is elegant and informative, the practical applicability is limited due to reported unstability problems with quantized filter coefficients. In [19] the quantitative performance measures of [18] have been studied with and without quantized filter coefficients, however, the ARMA(12,12) filter of [19] fails to yield the quantitative performance measures at and under 20 bits of quantization. In order for a proposed algorithm to have practical importance the proposed algorithm should be able to work at or below 16 bits of quantization levels. The ARMA(3,3) filter proposed in this study, yields acceptable performance levels even at 12 bits of quantization.

In this study, we also aim to demonstrate the implementation of our low-complexity fading simulator on standard commercially available hardware FPGA development platforms, where our implementation takes up only a small portion of the available hardware resources, without sacrificing from the quality of the fading process. With the advancement of the wireless communication systems and standards, it is becoming more important that the new systems are able to operate under severe multipath environments. Hence it may be desirable for system developers to be able to simulate, in hardware, a dense multipath environment with several tens of independent fading variates, where implementation of such a severe multipath scenario on a single FPGA development board is preferred over using several platforms with hybrid solutions [5,6]. Although the specifics of particular wireless systems and standards are out of the scope of this study, as an example, an ultrawideband (UWB) channel with 300 independent multipath components has been considered in [21]. In order to be able to simulate such a severe multipath environment on a single commercially available hardware platform, each individual multipath component simulator block must occupy only a small portion of the available resources. Hence, the study here demonstrates a compact fading process emulator which yields accurate performance under 12, 14 and 16 bits of quantization levels while achieving the lowest complexity to be able to fit in small portion of commercially available FPGA platforms.

The organization of this paper is as follows: After a brief overview of Rayleigh fading statistics of the wireless channels, Sect. 2 provides the derivation of the proposed fading filter, followed by performance and complexity comparisons. Our proposed hardware implementation of the filters is provided in Sect. 3. Finally Sect. 4 provides concluding remarks.

2 Derivation of the fading filter

2.1 Rayleigh fading statistics

Rayleigh fading process is characterized by the Gaussian WSS uncorrelated scattering fading model [22]. This statistic generally depends on the propagation geometry, the velocity of the mobile, and the antenna characteristics. A common assumption is that the propagation path consists of two-dimensional isotropic scattering with a vertical monopole antenna at the receiver [9]. In this case the in-phase or quadrature part of the received signal envelope must be independent and each must have zero mean for Rayleigh fading, and theoretical spectral density of in-phase, or the quadrature, part of the received faded signal envelope is

$$S(f) = \begin{cases} \frac{\sigma^2}{2\pi f_d \sqrt{1-(f/f_d)^2}}, & \text{if } |f| \leq f_d, \\ 0, & \text{else}, \end{cases} \tag{1}$$

where σ^2 is the rms value of the envelope of the waveform. f_d is the Doppler frequency which is defined as the ratio of the vehicle speed, V, to the wavelength, λ, $f_d = V/\lambda$, and $\lambda = c/f_c$ where $c = 3 \times 10^8 m/s$ is the speed of the light, and f_c is the carrier frequency. The corresponding normalized, unit-variance, continuous time autocorrelation function of the fading gain of the wireless channel under these conditions is $R(\tau) = J_0(2\pi f_d|\tau|)$, where $J_0(\cdot)$ is the zeroth-order Bessel function of the first kind. For the discrete-time simulation of this model, ideally generated in-phase and quadrature Gaussian processes should each have the autocorrelation sequence $R[n] = J_0(2\pi f_m|n|)$, where $f_m = f_d T$ is the Doppler frequency normalized by the sampling rate $1/T$.

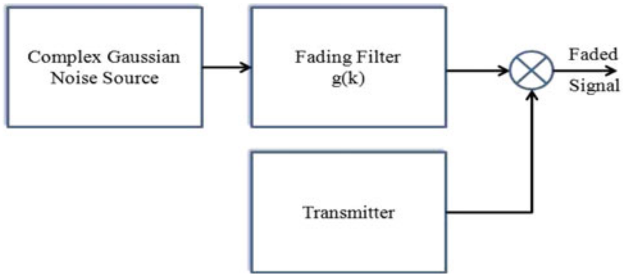

Fig. 1 Faded signal generator that uses low-pass filtered white complex Gaussian noise

2.2 Novel filter design

A straightforward method to simulate a faded signal is to amplitude modulate the carrier signal with a low-pass filtered Gaussian noise source as shown in Fig. 1. To obtain time varying frequency selective fading channel we must have a bank of these fading filters where each filter generates the corresponding fading channel tap. A fading filter with impulse response $g(k)$ can be designed so that its output spectral density is an approximation to theoretical spectral density of the complex envelope of the faded signal $S(f)$ of (1). We will use filter structures that were proposed in [10]. Consider the elementary first-order filter transfer function $G_1(s)$, and the second-order filter transfer function $G_2(s)$ where

$$G_1(s) = \frac{w_x}{s + w_x}, \quad \text{and} \quad G_2(s) = \frac{w_x^2}{s^2 + \frac{w_x s}{Q} + w_x^2}, \tag{2}$$

in which the parameter Q controls the gain of the peak of the frequency response of the filter where the peak occurs at $w = w_x$ rad/s; for example for the third-order filter if $Q = \sqrt{10}$ then the magnitude of $G(\cdot)$ has a gain of 7 dB at $w = w_x$, in which 10 dB gain results from the second-order filter and -3 dB results from the first-order part making the overall gain of 7 dB. Then we can have fading filter continuous time transfer functions with higher orders of γ, $G_\gamma(s)$, that are given by

$$G_\gamma(s) = \begin{cases} G_2^{\gamma/2}(s), & \text{if } \gamma \text{ even}, \\ G_1(s)G_2^{(\gamma-1)/2}(s), & \text{if } \gamma \text{ odd}, \end{cases} \tag{3}$$

where $G_1(s)$ and $G_2(s)$ are as given in (2). To find the parameters of the fading filter transfer function, $G_\gamma(s)$, we will first set the filter order γ and Q. Then defining $S(f;\epsilon)$, as an approximation to the theoretical spectral density of (1), by

$$S(f;\epsilon) = \begin{cases} \frac{\sigma^2}{2\pi f_d\sqrt{1-(f/f_d)^2}}, & \text{if } |f| \le f_d - \epsilon, \\ 0, & \text{else}, \end{cases} \tag{4}$$

where $\epsilon \in \mathbf{R}^+$ is a small positive real number, which can be taken as multiples of the smallest positive number the computing platform that can handle. Then we solved the numerical optimization problem, for fixed γ, f_d and Q,

$$w_x = \arg\min \| S(f;\epsilon) - |G_\gamma(j2\pi f)|^2 \|. \tag{5}$$

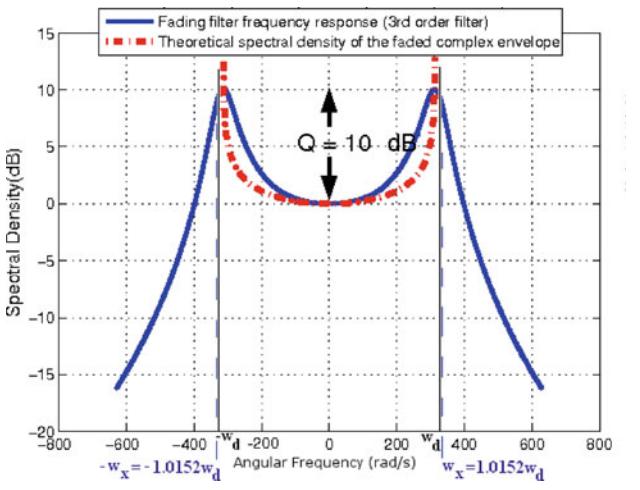

Fig. 2 Theoretical and approximate spectral density (for the filter $G_3(s)$)

Table 1 Ratio of w_x/w_d tabulated with respect to various filter orders and desired peak in dB at $w = w_x$

Filter order	Desired peak in dB at $w = w_x$		
γ	10	15	20
2	1.0200	1.0055	1.0025
3	1.0152	1.0060	1.0017
4	1.0668	1.0401	1.0247
5	1.0668	1.0413	1.0228

The result of this numerical optimization (5) gives the frequency, w_x, that minimizes the norm of the distance between the modified theoretical spectral density and the theoretical fading filter spectrum. Theoretical and approximate spectral density, where the approximate spectral density is for the output of the filter $G_3(s)$, are provided in Fig. 2. For given Doppler angular frequency $w_d = 2\pi f_d$, the ratios of w_x to w_d are provided in Table 1 with respect to different filter orders γ and desired peak values of $|G_\gamma(w)|^2$ at $w = w_x$. For the transfer functions provided in the s-domain, we can use the *bilinear transform* to get $G_\gamma(z)$ with an ARMA(γ,γ) model, or *impulse invariance method* to get a $G_\gamma(z)$ with an AR(γ) model, which is an all pole filter, where

$$G_\gamma(z) = \frac{\sum_{k=0}^{\gamma} g_k^{MA} z^{-k}}{1 - \sum_{k=1}^{\gamma} g_k^{AR} z^{-k}} \tag{6}$$

with $\{g_k^{AR}\}_{k=1}^{\gamma}$, $\{g_k^{MA}\}_{k=0}^{\gamma}$ are the auto-regressive, and the moving-average filter taps, of the ARMA(γ,γ) model, respectively. The generated Rayleigh fading process has an autocorrelation function, $R_{xx}[k]$, which can be found by directly using Wiener–Khinchine theorem [23]. That is,

$$R_{xx}[k] = \sigma^2 g[k] * g[-k] \quad (7)$$

where σ^2 is the variance of the complex zero-mean white Gaussian noise, and $g[k] = \mathcal{Z}^{-1}(G_\gamma(z))$ is the discrete time filter impulse response and as given as the inverse $\mathcal{Z}$-transform of the transfer function $G_\gamma(z)$, and $*$ denotes the discrete-time convolution.

2.3 Performance and complexity evaluation

The tested simulation methods in comparison to our proposed method are as follows.

2.3.1 Our proposed filter design method

Our filter design was carried out in the analog domain and then transferred into the digital domain and implemented via ARMA model or AR model by bilinear transform using the MATLAB function `bilinear`, or impulse invariance method by MATLAB function `impinvar`, respectively. After the filter coefficients were calculated, the Rayleigh fading sequence was generated by filtering white Gaussian noise sequence by the filter that was calculated using the MATLAB function `filter`.

2.3.2 IDFT method

The simulator used was implemented as described in [13]. The MATLAB function `ifft` was used for IDFT computation.

2.3.3 AR method

The method of [11] was implemented via MATLAB function `filtic`, to generate first p stationary Rayleigh fading samples, where p is the model order, and then to generate the other samples we generated a white random sequence as an input to the filter.

2.3.4 WSS-improved Jakes' model

The method used was based on the sum of sinusoids technique of [17]. The normalized low-pass discrete fading process is generated by finite number of sinusoids, therefore this WSS simulator is not autocorrelation ergodic; hence, theoretical calculations of quality measures cannot be done for this method.

2.3.5 ARMA(12,12) model

The method of [19] has been implemented in a way similar to that of the AR Method of [11].

Table 2 Quality measures for the IDFT, our proposed filter design, AR filtering, sum of sinusoids and ARMA(12,12) methods of generating Rayleigh random sequences for covariance sequence of length 200

		$\mathcal{G}_{mean}$(dB)	$\mathcal{G}_{max}$(dB)
IDFT method	(T)	0.00076	0.00081
	(E)	0.0035	0.0037
Proposed filter design	ARMA(2,2) (T)	2.5066	2.5505
	ARMA(2,2) (E)	2.5068	2.5514
	AR(2) (T)	2.6707	2.7247
	AR(2) (E)	2.6768	2.7313
	ARMA(3,3) (T)	1.9777	1.9962
	ARMA(3,3) (E)	1.9775	1.9979
	AR(3) (T)	2.0924	2.1173
	AR(3) (E)	2.1447	2.1727
AR filtering	AR(20) (T)	2.7	2.9
	AR(20) (E)	2.6	2.9
	AR(50) (T)	0.29	0.43
	AR(50) (E)	0.26	0.40
	AR(100) (T)	0.13	0.28
	AR(100) (E)	0.11	0.26
Sum of sinusoids	8 sinusoids (E)	36.223	37.730
	16 sinusoids (E)	4.0264	6.4140
	64 sinusoids (E)	0.0211	0.0370
	128 sinusoids (E)	0.0027	0.0049
ARMA filtering	ARMA(12,12) (E)	0.56	0.68

2.4 Performance quality measures

The quality measures that were first introduced in [18], are the *mean basis power margin*

$$\mathcal{G}_{mean} = \frac{1}{\sigma_X^2 L} trace\{C_X C_{\hat{X}}^{-1} C_X\}, \quad (8)$$

and, the *maximum basis power margin*

$$\mathcal{G}_{max} = \frac{1}{\sigma_X^2} max\{diag\{C_X C_{\hat{X}}^{-1} C_X\}\}. \quad (9)$$

In (8) and (9), σ_X^2 is the variance of the reference distribution, $C_{\hat{X}}$ is the $L \times L$ covariance matrix of any length-L subset of adjacent samples produced by the stationary random sequence generator, and C_X represents the desired covariance matrix of L ideally distributed samples.

2.5 Performance and complexity comparisons

The quality measure comparison results according to Table 2, that have been obtained using the measures provided in (8) and (9), compare the quality of the unquantized simulator outputs. The real and imaginary parts of the of the fading

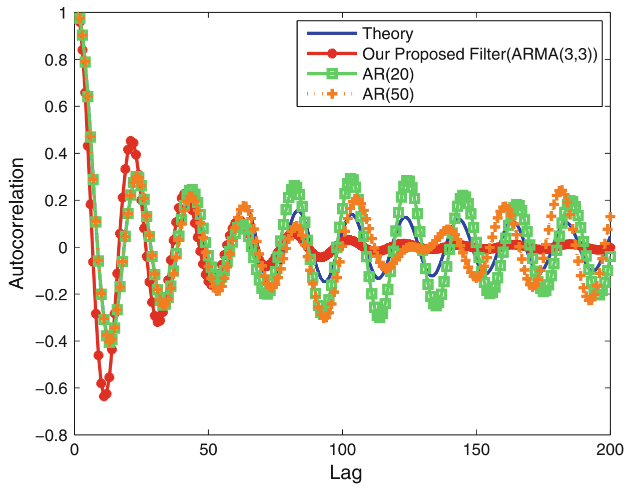

Fig. 3 The empirical autocorrelations for AR(20) model, AR(50) model and our proposed ARMA(3,3) generator

process are ideally uncorrelated, and the results reported here are only for the real parts of the simulator output; similar results are achieved for the imaginary sequences and these are omitted for brevity. Perfect Rayleigh fading sequence generation method corresponds to 0 dB for both measures.

In all cases, the reference autocorrelation function is $R[n] = J_0(2\pi f_m|n|)$ with a normalized Doppler frequency of $f_m = 0.05$. An autocorrelation sequence length of 200 was considered for evaluation of all theoretical (T) results. When counting the length of the sequence, only the non-negative delays were taken into account. For the empirical (E) results, time average correlations were calculated based on 2^{20} generated samples. The computed quality measures were then averaged over 50 independent simulation trials. Plots of the empirical autocorrelation functions of the AR(20) model and our proposed Rayleigh fading generator via ARMA(3,3) are shown in Fig. 3. The AR model of [11] provides a more precise match to the desired autocorrelation function as the order of the model used increases. But our proposed filter design method provides similar accuracy with much lower order models.

The quality measure comparison results, which are presented in Table 3, compare the quality of the *quantized* simulator outputs. Table 4 provides the number of real multiplications required to generate 2^{20} complex Rayleigh variate samples.

As an example, our ARMA(3,3) fading sequence generator has a significant computational and performance advantage over AR(20) and AR(50) generators of [11], and ARMA(12,12) generator of [19]: Proposed ARMA(3,3) model requires one-third of the multiplications while achieving about 0.7 dB lower *mean basis power margin* and about 1 dB lower *maximum basis power margin*. Proposed ARMA(3,3) fading generator outperforms modified Jakes' generator with 8 and 16 sinusoids by 32 and 2 dB, respectively in performance, while requiring less than one-tenth of the multiplications required by the Jakes' generators with 8 and 16 sinusoids. The main advantage of the method provided herein is that the samples of the fading sequence can be generated as they are required, on a sample-by-sample basis, while achieving the lowest complexity of all the Rayleigh fading generators mentioned. The computational efficiency of the IDFT method brings a cost in storage requirements as all samples are generated using a single IFFT. Our proposed fading generator and other generators do not have such a limitation.

Table 3 Performance analysis of quantized proposed filter design ARMA(3,3) method, AR method of [11], IDFT [13], sum of sinusoids (SOS) method of [17], ARMA of [19]

	Quantization (bits)	$\mathcal{G}_{mean}$(dB)	$\mathcal{G}_{max}$(dB)
IDFT method	28	0.0074	0.0076
of [13]	16	3.63	3.68
ARMA(3,3)	16	4.4501	4.5823
presented	14	5.7078	5.9139
in this study	12	5.7975	6.0118
AR(50)	22	0.65	0.68
of [11]	20	Fails	Fails
SOS(64)	16	6.6854	6.8993
of [17]	14	6.6357	6.8355
	12	7.2277	7.5505
ARMA(12,12)	22	10.23	10.35
of [19]	20	Fails	Fails

Table 4 Computational complexity (number of real multiplications required) comparison to generate 2^{20} complex samples

		Number of real multiplications
IDFT method		44×10^6
Proposed filter	ARMA(2,2)	8×10^6
design	AR(2)	5×10^6
	ARMA(3,3)	12×10^6
	AR(3)	6×10^6
AR filtering	AR(20)	36×10^6
	AR(50)	87×10^6
	AR(100)	173×10^6
ARMA filtering	ARMA(12,12)	42×10^6
Sum of sinusoids	8 sinusoids	178×10^6
	16 sinusoids	356×10^6
	64 sinusoids	$1{,}424 \times 10^6$
	128 sinusoids	$2{,}848 \times 10^6$

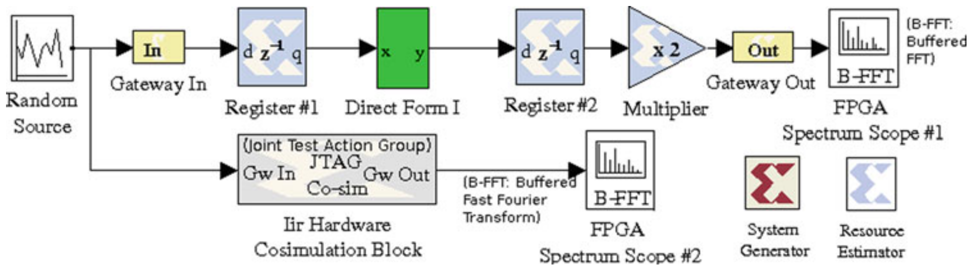

Fig. 4 The Simulink, Xilinx System Generator co-simulations of the third-order filter for multipath Rayleigh fading simulation on the development board

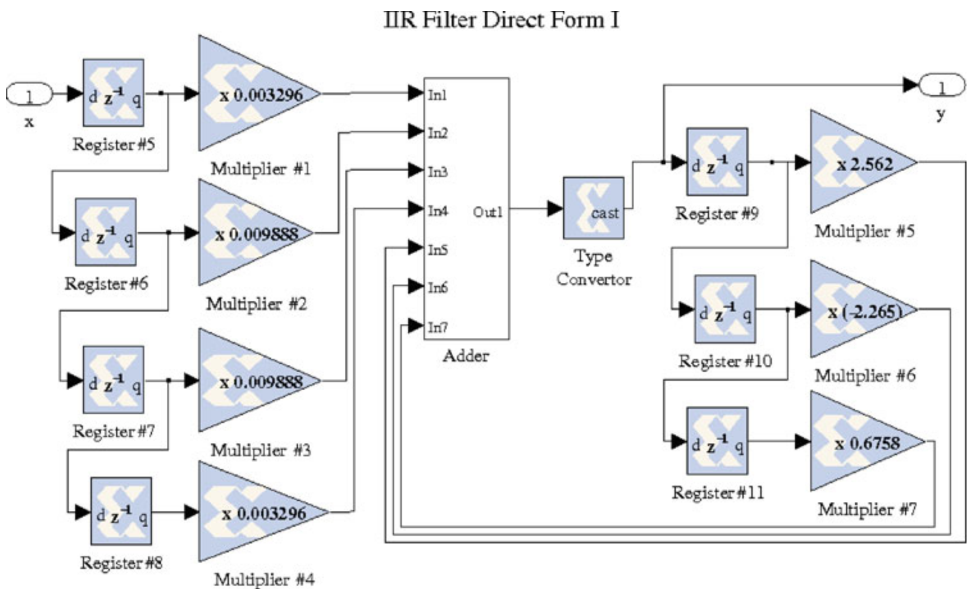

Fig. 5 Direct form IIR digital Filter FPGA implementations using System Generator for DSP

As can be seen in Table 2, although our proposed unquantized ARMA(3,3) fading generator has poorer performance than unquantized Jakes' generator with 64 sinusoids and unquantized ARMA(12,12) generator of [19], as shown in Table 3, our proposed ARMA(3,3) fading generator, which is quantized by 16 bits, outperforms modified Jakes' generator with 64 sinusoids quantized by 16 bits and ARMA(12,12) generator of [19] quantized by 22 bits by 2 and 6 dB, respectively. Also AR(50) generator of [11] quantized by 20 bits and ARMA(12,12) generator of [19] quantized by 20 bits both fail because of the stability problems; hence AR(50) generator of [11] and ARMA(12,12) generator of [19] both have limited practical use since they both diverge at and under 20 bits of quantization.

3 Hardware implementation of the proposed algorithm

The FPGA implementation of our proposed algorithm, performed using Xilinx's System Generator, is shown in Fig. 4. System Generator Tool produces a design that is targeted towards Xilinx Virtex4 ML402 development kit, consisting of a Xilinx xc4vsx35 FPGA chip. The primary simulation and debugging tool used for this paper was MATLAB [24].

The integrity of the third-order IIR Filter has been confirmed with bit and cycle modeling within the Matlab/Simulink environment. The communication between development board and Matlab/Simulink on the host Personal Computer (PC) is via the Universal Serial Bus (USB) interface. Simulink version 7 and Xilinx's co-simulation tools were also used for debugging [1].

IIR filters with general purpose multipliers can be used as building blocks for cascade or parallel realizations of higher order IIR filters. Figure 5 shows the the detailed implementation of the third-order direct form-I filter, with seven multipliers, using Xilinx System Generator for DSP.

System Generator translates the Simulink model into a hardware realization by mapping Xilinx block set elements and converts Simulink hierarchy into hierarchical VHDL netlist. The used multiplier block is an Intellectual Property (IP) core from Xilinx that implement a multiplier.

The delay element z^{-1} in an IIR filter signifies a full word delay. The delay is used to align data words and to propagate control signals that must also be properly synchronized.

A random noise signal generator, using Gaussian-Ziggurat method [24,25], is used to generate a discrete time Gaussian white noise signal, and is passed through our ARMA(3,3) filter. Co-simulation has been used to verify our filter frequency response of the real hardware as shown in Fig. 6. The

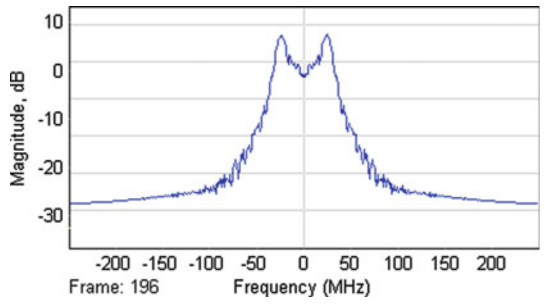

Fig. 6 The frequency response of the proposed filter implemented on Virtex4 hardware platform

Table 5 Design statistics of the simulator on two different FPGAs

	Spartan3e xc3s500e	Virtex4 xc4vsx35
Number of slices (percentage)	277 out of 4,656 (5%)	365 out of 15,360 (2%)
Number of input LUTs (percentage)	473 out of 9,312 (5%)	574 out of 30,720 (1%)
Number of IOBs (percentage)	39 out of 232 (16%)	39 out of 448 (8%)
Number of GCLKs (percentage)	1 out of 24 (4%)	1 out of 32 (3%)
Number of multipliers (percentage)	7 out of 20 (35%)	7 out of 192 (4%)

frequency response of output of the co-simulation block has been captured in a frame-by-frame basis where each frame consists of 1,024 samples, and the frequency response corresponding to the 196th frame has been given in Fig. 6.

3.1 Resource consumption

Our filter has been implemented on Xilinx's Spartan3e-xc3s500e and Virtex4-xc4vsx35 development platforms, where both platforms have 14 bits of quantization. Design statistics and resource consumption of our filter, such as the number of slices, number of Input-Output Blocks (IOB), number of Global Clocks (GCLK) and number of multipliers are shown in Table 5 for both platforms. As a comparison our proposed design uses 277 slices on the Spartan3e-xc3s500e platform, and 365 slices on the Virtex4-xc4vsx35 platform, whereas the reported slice usage in [26] is 6,178 on Virtex-xc2v1500-5 platform. Similarly the Look-Up Tables (LUT) used in our design are 473 and 574 in Spartan and Virtex4 platforms, respectively, while the reported LUT usage in [26] is 6,232 on Virtex-xc2v1500-5 platform.

It is possible to further reduce the hardware cost using multiplierless implementation of the IIR filter on the FPGA platforms [27,28]. Further details of the multiplierless implementation of IIR filters is out of the scope of this study.

4 Conclusion

A low-complexity and high-performance implementation of a Rayleigh fading channel emulator was presented. The hardware realization of our channel simulator was implemented using FPGAs, and verified with Xilinx co-simulation. The FPGAs have been adapted well to the design of the fading filter design for multipath Rayleigh due to the use of IP cores from Xilinx.

Our proposed ARMA(3,3) filter has been compared with improved Jakes' model of [17], AR fading filter approximation of [11], ARMA(12,12) model of [19], and the IDFT technique of [13], in terms of performance measures with and without quantization, and in terms of computational complexity. Our ARMA(3,3) Rayleigh fading generator, outperforms AR(20) generator of [11], by about 1 dB in both performance measures provided, while requiring one-third of the multiplications required by the AR(20) generator. Similarly, our ARMA(3,3) fading generator outperforms modified Jakes' generator with 8 and 16 sinusoids by 32 and 2 dB, respectively, while requiring less than one-tenth of the multiplications required by the Jakes' generators with 8 and 16 sinusoids. While the IDFT method of [13] achieves the best performance in terms of the quality measures, it brings a significant cost in storage requirements as all samples are generated using a single IFFT. Thus, the IDFT method is undesirable from simulation point of view when the Rayleigh fading samples are generated as they are required.

Our proposed ARMA(3,3) fading generator quantized by 16 bits, outperforms both ARMA(12,12) generator of [19] quantized by 22 bits and modified Jakes' generator quantized by 16 bits. ARMA(12,12) generator fails when using quantization levels lower than 22 bits [19]. Although unquantized AR(50) has better performance than our proposed simulator, it requires more than 7×10^6 multiplications required by our generator and also it fails under 20 bits quantization levels. Our proposed ARMA(3,3) fading generator works even at 12 bits quantization levels.

The main advantage of our ARMA(3,3) Rayleigh fading generator is that it provides accurate performance under all quantization levels for practical usage, such as 12, 14 and, 16 bits of quantization levels, while achieving the lowest complexity of all the Rayleigh fading generators mentioned. Moreover one can fit several ARMA(3,3) generators on a commercially available FPGA board to develop a laboratory tool to implement and realize severe wireless multipath channels similar to those depicted in [21]. If one prefers to use any other fading simulator mentioned in this study, it would not be possible to fit in on a commercially available

FPGA platform as many independent fading generators as the ARMA(3,3) generators would fit in, due to difference in complexity and hardware resource consumption.

Using Xilinx ISE 9.2i, the fading filter presented in Sect. 2 was described in Simulink/VHDL synthesized and tested on Xilinx Spartan3e and Virtex4 platforms. The coefficients were generated using Matlab 7.2 [24].

We report here the hardware co-simulation of the proposed channel emulator uses only 2% of configurable slices, and 4% of the dedicated multipliers and 1% of the available LUTs on the Virtex4 xc4vsx35 platform.

Acknowledgments This research has been supported by EU-FP6 Marie Curie International Reintegration Fellowship Grant, Contract number MIRG-029157.

References

1. Xilinx. http://www.xilinx.com
2. Agilent Technologies Inc (2005) Baseband studio for fading. Santa Clara, CA
3. Rohde, Schwarz (1999) Baseband fading simulator ABFS, reduced costs through baseband simulation
4. Jämsä T, Poutanen T, Meinila J (2001) Implementation techniques of broadband radio channel simulators. In: Proceedings of the IEEE Vehicular Technology Conference, pp 433–437
5. Salkintzis AK (1999) Implementation of a digital wideband mobile channel simulator. IEEE Trans Broadcasting 45:122–128
6. Wickert MA, Papenfuss J (2001) Implementation of a real-time frequency-selective RF channel simulator using a hybrid DSP-FPGA architecture. IEEE Trans Microw Theory Tech 49:1390–1397
7. Alimohammad A, Fard SF, Cockburn BF, Schlegel C (2008) A compact single-FPGA fading-channel simulator. IEEE Trans Circuit Syst II 55:84–88
8. Kahrs M, Zimmer C (2006) Digital signal processing in a real-time propagation simulator. IEEE Trans Instrum Meas 55:197–205
9. Jakes WC (1994) Microwave mobile communications. 2nd edn. IEEE Press, New York
10. Özen S, Zoltowski MD (2001) A fading filter approximation to enable state-space modeling and joint data channel estimation of (time-varying) frequency selective channels with antenna arrays. IEEE Circuits and Systems Society Notre Dame Workshop on Wireless Communication and Networking, South Bend, IN
11. Baddour KE, Beaulieu NC (2005) Autoregressive modeling for fading channel simulation. IEEE Trans Wirel Commun 4:1650–1662
12. Smith JI (1975) A computer generated multipath fading simulation for mobile radio. IEEE Trans Veh Technol VT-24:39–40
13. Young DJ, Beaulieu NC (2000) The generation of correlated Rayleigh random variates by inverse discrete Fourier transform. IEEE Trans Commun 48:1114–1127
14. Young DJ, Beaulieu NC (2001) Limitations of sum-of-sinusoids fading channel simulators. IEEE Trans Commun 49:699–708
15. Arsal A (2008) A study on wireless channel models: simulation of fading, shadowing and further applications, M.S. Thesis, Department of Electrical and Electronics Engineering, Izmir Institute of Technology, Izmir, Turkey
16. Arsal A, Özen S (2008) A fading filter design for multipath Rayleigh fading simulation and comparisons to other simulators. IEEE 19th International Symposium on Personal, Indoor and Mobile Radio Communications (PIMRC 2008)
17. Zheng YR, Xiao C (2002) Improved models for the generation of multiple uncorrelated Rayleigh fading waveforms. IEEE Commun Lett 6(6):256–258
18. Young DJ, Beaulieu NC (2003) Power margin quality measures for correlated random variates derived from the normal distribution. IEEE Trans Inf Theory 49:241–252
19. Mehrpouyan H, Blostein SD (2008) ARMA synthesis of fading channels. IEEE Trans Wirel Commun 7(8):2846–2850
20. Manolakis DG, Ingle VK, Kogon SM (2000) Statistical and adaptive signal processing. McGraw-Hill Series, USA
21. Homier EA, Scholtz RA (2002) Rapid acquisition of ultra-wideband signals in the dense multipath channel. IEEE Conference on Ultra Wideband Systems and Technologies, pp 105–109
22. Bello PA (1963) Characterization of randomly time-variant linear channels. IEEE Trans Commun Syst CS-11:360–393
23. Proakis JG, Manolakis DK (2007) Digital signal processing: principles, algorithms and applications, 4th edn. Prentice Hall, New Jersey
24. Mathworks. http://www.mathworks.com
25. Moler CB (2004) Numerical computing with MATLAB, 1st edn. SIAM, Philadelphia
26. Surendra B, Srinivas B, Rambabu C, Gogoi A (2003) A prototype architecture for the accurate modeling of Rayleigh fading waveforms. In: Proceedings of the international conference on information, communination and signal processing, vol 2, pp 1101–1105
27. Püschel M, Zelinski AC, Hoe JC (2004) Custom-optimized multiplierless implementations of DSP algorithms. In: Proceedings of the international conference on computer-aided design (ICCAD), pp 175–182
28. Voronenko Y, Püschel M (2007) Multiplierless multiple constant multiplication. ACM Trans Algorithms 3(2)