FPGA Implementation of Isotropic and Nonisotropic Fading Channels

Amirhossein Alimohammad and Saeed Fouladi Fard

***Abstract*—A realistic fading channel simulator (FCS) is a key component for the development and faithful performance evaluation of wireless communication systems. This brief presents an efficient implementation of an FCS on a single field-programmable gate array. The FCS utilizes a configurable filter processor along with a multistage interpolator for a compact implementation. The FCS can be parameterized to accurately reproduce the statistical properties of both isotropic and nonisotropic scattering scenarios.**

***Index Terms*—Fading channel simulation, field-programmable gate array (FPGA), isotropic scattering, nonisotropic scattering.**

I. Introduction

WIRELESS communication systems must be designed to operate over radio channels in a wide variety of environments. To evaluate the performance of wireless communication systems, a fading channel simulator (FCS) can be used to replicate the behavior of different propagation environments and mobility conditions in a laboratory setting. It is generally easier and thus more common to design an FCS in software rather than in hardware. However, accurate simulation of radio channels is a computationally intensive process and, indeed, software simulation has become a serious bottleneck to timely design and verification. Hardware-based simulators have shown several orders of magnitude of speed-up over software-based simulators [1].

To simulate a fading channel, we need to generate a suitable sequence of fading gains. In an isotropic scattering fading channel with an omnidirectional antenna at the receiver, the incident directions of the received multipath signals, or angle of arrival (AoA), are equally distributed, and the fading (path) gains are modeled using a unit-variance zero-mean complex Gaussian process $c(t) = c_i(t) + jc_q(t)$ [2]. The real and imaginary parts of $c(t)$ are independent Gaussian processes with zero mean and equal variance [3]. Thus, the envelope $|c(t)| = \sqrt{c_i(t)^2 + c_q(t)^2}$ follows the Rayleigh distribution, and in the presence of a line-of-sight path, the amplitude statistics may follow the Rician distribution [4].

Fading samples are generally correlated, and to generate the in-phase and quadrature components of a fading process $c(t)$ with a particular temporal correlation between variates, a complex Gaussian random process $n(t) = n_i(t) + jn_q(t)$ with zero-mean, unit-variance, and uncorrelated samples is passed through a filter [5] with an appropriate frequency response $H(f)$. A linear filtering operation on the complex Gaussian samples with flat power spectral density (PSD) yields samples that also have a Gaussian distribution, with output spectrum $S_y(f) = |H(f)|^2$, where $|H(f)|^2$ is the squared magnitude response of the filter and $S_y(f)$ is the PSD of the output samples. This filter is often called the spectrum shaping filter (SSF) for it determines the power spectrum shape. Therefore, to generate temporally correlated fading samples, a stream of independent Gaussian samples is passed through an SSF with the magnitude response equal to the square root of the magnitude of the PSD of the desired fading process (i.e., $|H(f)| = |S_c(f)|^{1/2}$).

Assuming an isotropic scattering fading channel, the PSD associated with either the in-phase or quadrature component of a complex fading signal has the well-known Jakes' U-shaped form $S_{c_i}(f) = S_{c_q}(f) = (2\pi\sqrt{f_D^2 - f^2})^{-1}$ that is bandlimited to $\pm f_D$, where f_D is the maximum Doppler frequency [2]. Therefore, to generate a Rayleigh fading process, uncorrelated samples of a zero-mean complex Gaussian process pass through an SSF with a magnitude response equal to the square root of the magnitude of $S_c(f)$. The autocorrelation function (ACF) associated with either $c_i(t)$ or $c_q(t)$ is $R_{c_i,c_i}(\tau) = R_{c_q,c_q}(\tau) = \mathcal{J}_0(2\pi f_D\tau)$, where $\mathcal{J}_0(\cdot)$ is the zeroth-order Bessel function of the first kind [2]. For nonisotropic scattering scenarios, in which AoA is not uniformly distributed or when the antennas are not omnidirectional, the PSD of the fading process is asymmetric.

In this brief, we present a compact hardware implementation of the FCS on a single field-programmable gate array (FPGA). The FCS utilizes a configurable filter processor and a multistage structure with elastic buffers for interconnecting the elements of the FCS architecture. The FCS can be parameterized to accurately generate fading gains for both isotropic and nonisotropic scattering scenarios. The rest of this brief is organized as follows. Section II briefly explains important SSF design considerations. Section III presents FPGA implementation of isotropic and nonisotropic fading channels. Section IV summarizes the implementation and simulation results. Finally, Section V makes some concluding remarks.

II. SSF Design Considerations

For a typical wireless communication scenario, the Doppler frequency f_D is significantly smaller than the signal sample rate F_s. Thus, SSF would have an extremely narrow bandwidth and a very sharp cutoff. We can reduce the complexity of SSF

Manuscript received December 27, 2012; revised May 21, 2013; accepted August 21, 2013. This brief was recommended by Associate Editor Y. Miyanaga.

A. Alimohammad is with the Department of Electrical and Computer Engineering, San Diego State University, San Diego, CA 92182 USA (e-mail: aalimohammad@mail.sdsu.edu).

S. F. Fard is with PMC-Sierra, Calgary, AB T2L 2K7, Canada.

Color versions of one or more of the figures in this brief are available online at http://ieeexplore.ieee.org.

Digital Object Identifier 10.1109/TCSII.2013.2281768

by designing it at a much lower sampling frequency $F_1 \ll F_s$, thereby increasing the computational efficiency and also improving the stability and the accuracy of the designed SSF [6]. The resulting low-rate signal can then be interpolated to reach the target sampling frequency $F_s = F_1 \times I$. Interpolation of samples generated by the SSF at a low sampling frequency can be performed by inserting $I-1$ zeros between each pair of successive samples of filtered Gaussian variates (i.e., fading variates) and then passing the resulting stream through a low-pass filter with cutoff frequency π/I (in radians per second). Note that, for a practical mobile system, the factor I can be relatively large and, thus, the complexity of the real-time interpolation filter can be significant. To reduce the computational complexity of the low-pass filter, interpolation is accomplished using a multistage interpolator design, where $F_s = F_1 \times \prod_{j=0}^{T} I_j$, $T+1$ is the number of interpolation stages and I_j is the integer upsampling factor at the jth stage.

An important point is that, if the stopband attenuation of the shaping filter is not sufficiently large, then the out-of-band noise that passes through the filter will degrade the accuracy of the statistics of the generated fading variates. Nevertheless, designing a narrow-band filter with a sharp cutoff and a large attenuation inevitably leads to a high-order filter, which directly impacts the hardware complexity of the FCS. Therefore, to obtain the closest approximation to the desired frequency response with a relatively small filter order, the SSF and the first interpolation stage are suggested to be designed "together" [7]. In this approach, the shaping filter is designed to accurately approximate the desired PSD in the passband region; however, it might not provide enough attenuation in the out-of-band region. The first-stage interpolation filter is then designed not only to eliminate the image signals caused by upsampling but also to provide additional attenuation in the out-of-band region and over the transition region to ensure a sharp cutoff. This approach minimizes the approximation error in the passband $|f| < f_D$ while maximizing the attenuation in the stopband and, hence, it results in a much more accurate statistical properties of the generated fading samples.

Given the desired PSD $S_c(f) = (\pi f_D \sqrt{1-(f/f_D)^2})^{-1}$ of an isotropic fading channel, the low-pass SSF has a symmetric frequency response and, hence, its poles and zeros appear in complex conjugate pairs; therefore, this filter can be realized using cascaded filter structures. The magnitude response of the SSF can be efficiently approximated with an infinite-duration impulse response (IIR) filter of order Γ as

$$H(e^{jw}) \prod_{k=1}^{\Gamma/2} \left\{ g_k \times \frac{1 + b_1^k e^{-j\omega} + b_2^k e^{-j2\omega}}{1 + a_1^k e^{-j\omega} + a_2^k e^{-j2\omega}} \right\} \qquad (1)$$

which is equivalent to the magnitude response of $\Gamma/2$ cascaded canonic second-order sections (SOSs) [7]. In (1), b_1^k, b_2^k, a_1^k, and a_2^k denote the coefficients, and g_k denotes the real-valued scaling factor of the kth SOS. Since the filter input has a Gaussian distribution, scaling factor g_k is used in each stage to maintain the signal magnitude within the representable range. The SSF filter coefficients and scaling factors can be obtained using the MATLAB filter design fdatool standard tool and iirlpnorm function [8]. Since only the amplitude response affects the correlation properties and no restrictions are imposed on the phase response, we also use an elliptic IIR low-pass filter (EILPF) in the first interpolation stage.

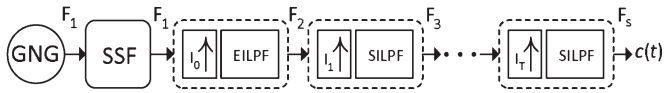

Fig. 1. Block diagram of the FCS.

After the SSF output is upsampled I_0 times and passed to the EILPF generating $F_2 = I_0 \times F_1$ samples per second, the samples need to be oversampled and passed through low-pass filters to obtain the target output sample rate F_s. We note that utilizing conventional FIR or IIR filters at this stage 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 zero-order hold filters with impulse response $d_{I_j}(n) = [I_j^{-1}, I_j^{-1}, \ldots, I_j^{-1}]_{1\times I_j}$, where I_j is the oversampling rate [9]. Such fading-specific interpolation low-pass filters (SILPFs) can be easily implemented without multiplication. Fig. 1 shows the structure of the designed FCS.

For isotropic scattering, the PSD of fading samples is symmetric and, hence, poles and zeros of (1) appear in complex conjugate pairs. Therefore, the SSF has real-valued coefficients and can be implemented using $\Gamma/2$ SOSs. In contrast to isotropic fading, the PSD of fading samples in nonisotropic scattering is not symmetric in general, and hence, the SSF coefficients are potentially complex [10]. We will refer to filters with real coefficients as "real filters" and filters with complex coefficients as "complex filters." When the filter coefficients are complex, the poles and zeros of the filter do not appear in complex conjugate pairs. In [7], we used the MATLAB filter design fdatool standard tool and iirlpnorm function to obtain the real-valued IIR filter coefficients for isotropic fading channels. However, this approach is not appropriate in designing the complex filters required for shaping the spectrum of nonisotropic channels.

In [11] and [12], we proposed a novel ellipsoid iterative optimization algorithm for designing stable complex IIR filters with quantized coefficients. Note that the precision of the fixed-point filter coefficients plays an important role in the stability and accuracy of IIR filters. To make sure that the filters are stable under quantization effects and to reduce quantization noise, the algorithm utilizes a least-squares cost function augmented with a parameterizable barrier function to control the location of the poles at any desired safe distance from the unit circle. Moreover, techniques such as pole-zero ordering and augmenting auxiliary poles and zeros were applied in minimizing the wordlength of the variables and the quantization effects. It was shown in [11] and [12] that, for nonisotropic fading channels, the SSF with a prescribed amplitude response $|H(e^{j\omega})|$ can be designed as a product of first-order sections (FOSs) as

$$H(e^{j\omega}) = \prod_{k=1}^{\Gamma} g_k \times \frac{1 - b_1^k e^{-j\omega}}{1 - a_1^k e^{-j\omega}} \qquad (2)$$

where g_k is the positive scaling factor, Γ is the filter order, and b_1^k and a_1^k are the kth complex zero and pole, respectively.

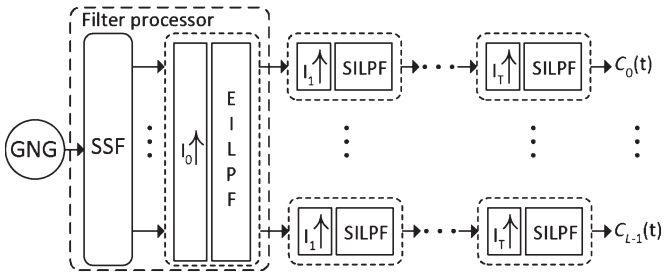

Fig. 2. Block diagram of the multipath FCS.

III. Architecture of the Channel Simulator

Fig. 2 shows the block diagram of the designed multipath FCS. It can generate L independent fading processes (paths) with different correlation properties. The fading processes can be used to model, for example, frequency-selective channels or fading channels in multiple-input–multiple-output systems. For an efficient hardware implementation of the FCS, we note that the shaping and first-stage interpolation low-pass filters operate at relatively low sampling rates and, hence, it is advantageous to reuse hardware to implement both filters. In the proposed FCS architecture, the Gaussian samples generated by the Gaussian noise generator (GNG) are passed to the "filter processor," which runs the operations of the designed SSFs for L independent paths (or threads). Each thread of data is then upsampled I_0 times and passed through an EILPF implemented using the same filter processor. After the EILPF, the data streams are passed to L independent series of T configurable SILPFs.

To generate each Gaussian variate, we simply added 12 uniformly distributed random variables $u_i \in [-0.5, 0.5)$. Since the fading samples are strongly correlated, generating the uniform samples with a simple linear feedback shift register structure does not change the correlation properties significantly. Also, from the central limit theorem, filtering the input process with the SSF and EILPF will help in improving the Gaussian distribution of the fading samples. For a greater accuracy, we can always utilize a high-quality GNG from [13].

For simulating isotropic fading channels, the SSF and EILPF can be implemented efficiently using cascaded SOSs. These filters are implemented using cascaded FOSs for simulating nonisotropic scenarios. The operations of the cascaded FOS and SOS are performed using the filter processor datapath shown in Fig. 3. The main element of this architecture is a multiplier–accumulator unit that multiplies the in-phase and quadrature signal components by real-valued coefficients when the channel is isotropic and by complex-valued coefficients when the channel is nonisotropic. To make our design more resilient against overflows and the resulting instability, the adder saturates its output in case of overflow. Note that the filtering operations of a real filter with cascaded SOSs require the same number of multiplications and additions as a complex filter with cascaded FOSs.

For isotropic scattering fading channels, the coefficients of the SOSs are stored in $\mathsf{RAMa}_{1,2}$ and $\mathsf{RAMb}_{1,2}$, and the scaling factors are stored in RAMg. When implementing nonisotropic fading channels, the coefficients of FOSs are stored in $\mathsf{RAMa}_{i,q}$ and $\mathsf{RAMb}_{i,q}$. $\mathsf{RAMm}_{1,2}/\mathsf{m1}_{i,q}$ and $\mathsf{RAMm}_{3,4}/\mathsf{m2}_{i,q}$ store the real-valued contents of the four memories in every SOS or

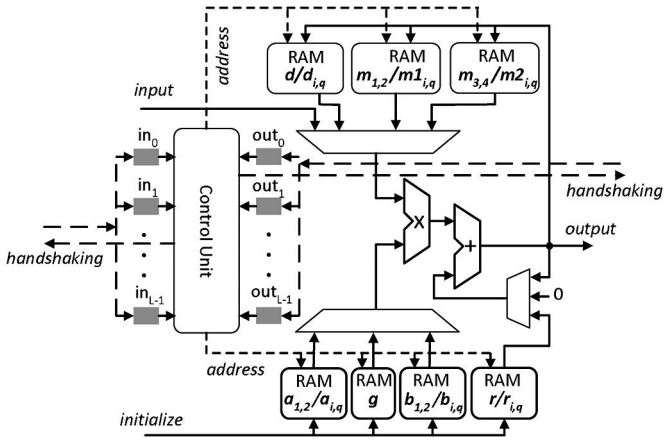

Fig. 3. Datapath of the filter processor.

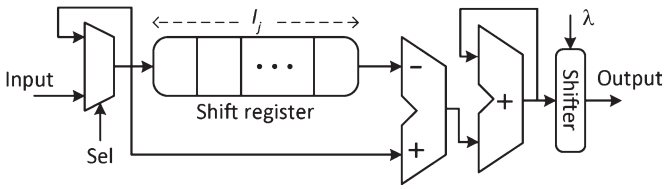

Fig. 4. Datapath of the proposed SILPF.

the complex-valued contents of the two memories in every FOS. When simulating Rician fading channels, a bias can be added to the output value from $\mathsf{RAMr}/\mathsf{r}_{i,q}$. The input Gaussian samples enter this datapath via the multiplexer, and the output of each filter section is written into $\mathsf{RAMd}/\mathsf{d}_{i,q}$, which holds the intermediate results. This datapath can also be used in performing zero-padding for interpolation, when required.

The control sequence for running the cascaded filter sections is straightforward. However, when emulating multiple paths, where each path could have different filter specifications, flexible implementation of the control unit becomes challenging. To improve the robustness of the filter processor, two flags in_ℓ and out_ℓ, where $\ell = 1, \ldots, L$, are assigned to every thread of cascaded filter sections (i.e., for each individual path). These flags govern the data flow through each thread. For example, for the ℓth thread (path), if in_ℓ is high, then the input data are ready to be read, and if out_ℓ is high, then the data can be written to the next stage. For each thread, the filter processor keeps executing the filter sections unless either of the input or output flags are deasserted. To prevent the overwriting of unprocessed data, the filter sections in each thread are scheduled to be executed from the last section to the first.

Fig. 4 shows the datapath of the parametric SILPF that upsamples the input signal and uses a multiplication-free filter for interpolation [9]. The interpolation factor I_j is passed to this circuit as a parameter to specify the length of the shift register. After interpolation, the signal amplitude can be multiplied by $2^{-\lambda}$ (shifted to the right) to adjust the signal power, where λ is another input parameter passed to this circuit.

To interconnect the elements of the FCS datapath shown in Fig. 2 and also to interface the FCS with external (off-the-chip) hardware, we utilize elastic buffers. The elastic buffer is basically a random access memory (RAM) with two (potentially) asynchronous ports as shown in Fig. 5. The elastic buffer allows

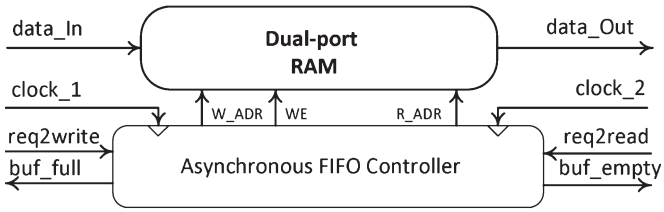

Fig. 5. Block diagram of the elastic buffer.

the input data to be written into the memory using an input clock and then read out according to an output clock. Therefore, consecutive blocks do not have to be strictly synchronized with respect to an instantaneous data throughput. It is important to note that, for correct operation of the elastic buffer, the read cycle must occur after the write cycle and not before or concurrently with the write cycle. The elastic buffer is designed to operate in two modes: handShaking and continuous. In the handShaking mode, the receiver block requests to read data samples (req2read) 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 arrive. Likewise, the transmitter requests to write into the elastic buffer (req2write), and it is informed by asserting buf˙full when the buffer is full. The continuous mode is used in interfacing external clock domains without handshaking capability. 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 overwrites some of the 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 rereading the last data word. The consequences of repeating read data depend on the application. In practical wireless systems, the sample rate F_s is much higher than the maximum Doppler frequency f_D and, therefore, immediate fading samples have almost equal values. Now, we show that the elastic buffer in the continuous mode does not have a significant effect on the statistics of the generated fading samples. Let us 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, $y_k(t)$ 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

$$R_{yy}(0) = E\{y(t)y(t)^*\} = \int_{-\infty}^{+\infty} S_c(f)\left|1 - e^{-j2\pi f k T_s}\right|^2 df$$

$$= \mathcal{J}_o(0) - \mathcal{J}_o(2\pi k f_D T_s) \approx (k\pi f_D T_s)^2 \qquad (3)$$

where in (3) we used the expansion of the zeroth-order Bessel function $\mathcal{J}_o(x) = \sum_{m=0}^{\infty}((-1)^m x^{2m})/(2^{2m}(m!)^2)$. Equation (3) shows that, when $F_s \gg f_D$, the power of the difference signal goes to zero and, hence, 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 elastic buffers, the clock mismatch between the transmitter and the receiver has a similar effect as a slight (probably negligible in most cases) change in the maximum Doppler frequency.

TABLE I
IMPLEMENTATION RESULTS FOR DIFFERENT FILTER DESIGNS

Design	**Wordlength**	**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

IV. FPGA IMPLEMENTATION AND SIMULATION RESULTS

We propose to design the SSF at a sampling frequency F_1, where $4f_D < F_1 \leq 8f_D$ [12], and exploit power-of-2 interpolation factors to further reduce the hardware complexity. The generated samples from the SSF are upsampled $I_0 = 16$ times and passed through the EILPF. Then, the samples are passed through T successive SILPFs, in which the ith SILPF interpolates the signal 2^{k_i} times. The target output sampling rate is thus $F_s = F_1 \times 16 \times \prod_{i=1}^{T} 2^{k_i}$. Table I shows the implementation results of an order $\Gamma = 6$ elliptic low-pass filter with a sample rate of 4800 Hz, pass frequency of 1200 Hz, stop frequency of 1500 Hz, passband ripple of 1 dB, and stopband attenuation of 50 dB on a Xilinx Virtex-II Pro FPGA. It can be seen that the cascaded direct-form-II (DF-II) structure requires the largest number of configurable slices and also runs the slowest. The direct-form-I (DF-I) implementation of the same filter with the pole-zero ordering (DF - I + ord.) reduces the required number of configurable slices significantly and also increases the performance. Augmenting a zero at dc (DF - I + ord. + aug.) can further reduce the required number of configurable slices while providing the highest performance. As this table shows, due to the reduced wordlength, the hardware complexity has been reduced by more than 55%, and the maximum operation speed has been increased by more than 34%. The bit-true fixed-point simulation results in [12] also showed that the cascaded DF-I structure provides superior accuracy and reduced range of variables compared to the other candidate structures. Therefore, we designed and implemented the SSF and EILPF using the cascaded DF-I structure.

Using our filter design approach presented in [11] and [12], we designed a complex SSF for simulating a nonisotropic fading channel with normalized Doppler frequency of -0.15625, directivity $\kappa = 5$, and AoA $\tilde{\psi} = \pi/5$ using 12 cascaded FOSs. We used 12 bits for representing the coefficients of the FOSs. As Fig. 6(a) shows, the frequency response of the designed fixed-point filter with stopband attenuation $\varepsilon = 0.001$ closely matches the desired response. In another example, we designed and implemented the SSF for an isotropic fading channel with a Doppler frequency of 9 Hz and $f_D/F_1 = 0.125$, using ten cascaded FOSs. We used 16 bits for representing the coefficients of the FOSs. Fig. 6(b) shows that the designed fixed-point filter accurately produces the desired response within the passband and provides more than 55-dB attenuation in the stopband. The aforementioned results were generated using the fixed-point bit-true model of the FCS. The hardware FCS is designed and

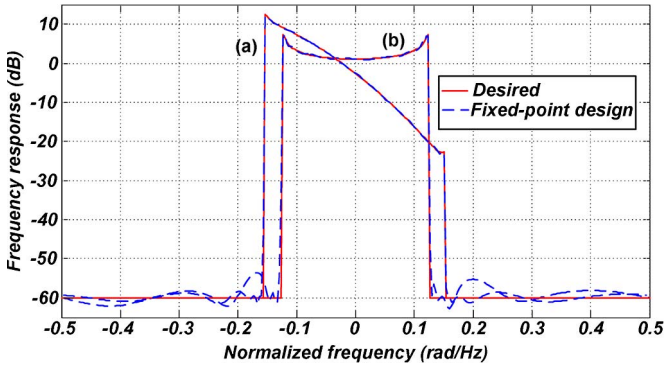

Fig. 6. Frequency response of the designed fixed-point SSFs and the desired responses.

TABLE II
CHARACTERISTICS OF THE FILTER PROCESSORS

Design	[14]	[15]	This work
Scattering	Nonisotropic	Isotropic	Nonisotropic
Fading	Rayleigh	Rician	Rician
Filter coefficients	Complex	Real	Complex
Coefficient wordlength	32-bits	32-bits	16-bits
Datapath wordlength	40-bits	36-bits	18-bits
Configurable slices	10328	1348	1164
Resource utilization	11.6%	1.5%	1.3%
Max. clock freq. (MHz)	128	87	234
Output rate (KSamp/s)	26	435	585

implemented to generate fixed-point results that are identical to those produced by our bit-true software simulator.

Table II summarizes the implementation results of three filter processors 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 16. The design in [14] was reconfigured to meet the aforementioned 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., look-up table-based memories), and no dedicated multipliers are used. Note that the design from [15] processes filters with real coefficients and therefore requires half the memory to store the coefficients and also preforms half the arithmetic operations of its complex-valued counterpart. However, the proposed filter architecture results in a more compact and efficient design, as shown in Table II. Compared to the design in [14], the new filter processor utilizes almost nine times fewer configurable slices and can process 22.5 times more samples per second. Fig. 7 shows the desired filter response and the measured output power spectrum of an implemented SSF in 12-bit fixed-point format on a GVA-290 FPGA development board [16], which verifies the correct operation of the FPGA-based FCS.

V. CONCLUSION

This brief has presented a compact hardware implementation of an FCS on a single FPGA. Fading samples are generated at a low rate and are then interpolated to match the desired sample rate. The simulator accurately and efficiently implements both isotropic and nonisotropic scattering scenarios. The accuracy and performance of the proposed FCS were verified with bit-true fixed-point simulations of different fading channel scenarios. The implemented FCS requires fewer hardware resources while providing a greater fading sample generation rate compared to the previously published designs. The proposed FCS is well suited for use in an FPGA-based error rate performance verification system.

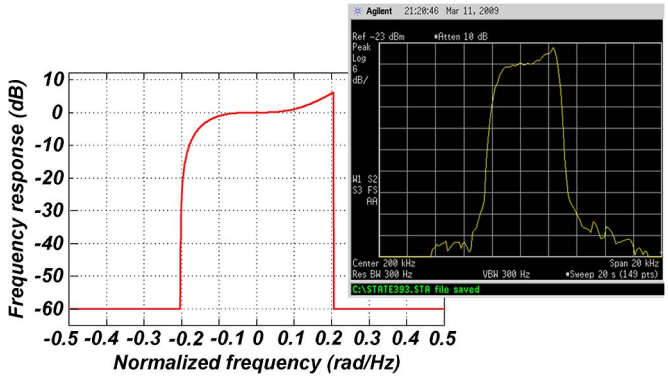

Fig. 7. Desired frequency response and measured power spectrum of the filtered noise.

REFERENCES

[1] A. Alimohammad, S. F. Fard, and B. F. Cockburn, "FPGA-based accelerator for the verification of leading-edge wireless systems," in *Proc. IEEE Intl. Design Autom. Conf.*, 2009, pp. 844–847.
[2] P. Bello, "Characterization of randomly time-variant linear channels," *IEEE Trans. Commun.*, vol. COM-11, no. 4, pp. 360–393, Dec. 1963.
[3] R. N. Kolte, S. C. Kwatra, and G. H. Stevens, "Computer controlled hardware simulation of fading channel models," in *Proc. IEEE Intl. Conf. Commun.*, 1998, pp. 1646–1650.
[4] G. L. Stüber, *Principles of Mobile Communication*. Norwell, MA, USA: Kluwer, 2001.
[5] J. I. Smith, "A computer generated multipath fading simulation for mobile radio," *IEEE Trans. Veh. Technol.*, vol. VT-24, no. 3, pp. 39–40, Aug. 1975.
[6] A. Alimohammad and B. F. Cockburn, "Modeling and hardware implementation aspects of fading channel simulators," *IEEE Trans. Veh. Technol.*, vol. 57, no. 4, pp. 2055–2069, Jul. 2008.
[7] A. Alimohammad, S. F. Fard, B. F. Cockburn, and C. Schlegel, "A compact single-FPGA fading channel simulator," *IEEE Trans. Circuits Syst. II*, vol. 55, no. 1, pp. 84–88, Jan. 2008.
[8] *MATLAB Filter Design Toolbox, User's Guide*, The Mathworks, Natick, MA, USA, 2005.
[9] T. Saramaki and J. Yli-Kaakinen, "A novel systematic approach for synthesizing multiplication-free highly-selective FIR half-band decimators and interpolators," in *Proc. IEEE Asia Pacific Conf. Circuits Syst.*, 2006, pp. 920–923.
[10] A. V. Oppenheim, R. W. Schafer, and J. R. Buck, *Discrete Time Signal Processing*. Englewood Cliffs, NJ, USA: Prentice-Hall, 1999.
[11] A. Alimohammad, S. F. Fard, and B. F. Cockburn, "Accurate simulation of nonisotropic fading channels with arbitrary temporal correlation," *IET Commun.*, vol. 6, no. 5, pp. 557–564, Mar. 2012.
[12] A. Alimohammad, S. F. Fard, and B. F. Cockburn, "Filter-based fading channel modeling," *Model. Simul. Mobile Radio Channels*, vol. 2012, pp. 705078-1–705078-10, 2012, Special Theme Issue of Modeling and Simulation in Engineering.
[13] A. Alimohammad, S. F. Fard, B. F. Cockburn, and C. Schlegel, "A compact and accurate Gaussian variate generator," *IEEE Trans. Very Large Scale Integr. Syst.*, vol. 16, no. 5, pp. 517–527, May 2008.
[14] S. F. Fard, A. Alimohammad, M. Khorasani, C. Schlegel, and B. F. Cockburn, "A compact and accurate FPGA based nonisotropic fading channel simulator," in *Proc. IEEE Proc. Canadian Conf. Elect. Comput. Eng.*, 2007, pp. 1239–1242.
[15] S. F. Fard, A. Alimohammad, B. F. Cockburn, and C. Schlegel, "A single FPGA filter-based multipath fading emulator," in *Proc. IEEE Proc. GLOBECOM*, 2009, pp. 1–5.
[16] *GVA-290 Xilinx VirtexE Hardware Accelerator*, GV Associates, Ramona, CA, USA, Aug. 2009.