A Real-Time Multi-Path Fading Channel Emulator Developed for LTE Testing

Elliot Briggs
Electrical and Computer Engineering
Texas Tech University
Lubbock, Texas 79409
Email: elliot.s.briggs@ttu.edu

Brian Nutter
Electrical and Computer Engineering
Texas Tech University
Lubbock, Texas 79409
Email: brian.nutter@ttu.edu

Dan McLane
Innovative Integration
2390-A Ward Ave.
Simi Valley, CA 93065
Email: dmclane@innovative-dsp.com

***Abstract*—When developing a communication system, it is useful to perform tests in channel environments in which the system is intended to operate. Many modern mobile communications systems must be tolerant to high mobility and fast-changing channel conditions, which make field testing difficult. In order to test specific channel conditions, or to repeat time-varying sequences that occur in the channel environment, it is useful to have a programmable system that can emulate the conditions found in actual time-varying radio channels in real-time. In the 3GPP long term evolution (LTE) standard, 3 channel models have been defined for conformance testing, each with varying mobility. An FPGA-based channel emulator has been developed that conforms to the LTE models yet remains flexible enough to allow users to define their own models. The channel emulator is built around flexibility, allowing the user to define both the power-delay profile as well as the maximum Doppler frequency with high resolution. The LTE specifications place many technical challenges on real-time channel emulation, including high sampling rate processing, fine delay resolution, and a requirement for 9 statistically independent channel paths to be processed simultaneously. A solution to these challenges that maintains user programmability is presented.**

I. Introduction

Performing bench-top tests, rather than field tests of a communications system saves considerable effort. In the development cycle of a communications system, a channel emulator is a valuable tool to perform top-level system performance tests without having to deploy the system into the field. By performing a wide range of tests that use a diverse set of channel models and conditions, a designer can better understand the performance of a system, where field testing in such a diverse condition set may not be feasible. Channel emulators also allow tests to be carefully defined and repeated, and they allow the same test to be performed on many devices in production testing. A programmable channel emulator allows the user to use predefined channel models or to use custom models that were created from actual-case channel measurements. With a real-time channel emulator, tests can be performed using the target hardware. Alternatively, the user can process a test signal in real-time to be stored and later used in simulation.

The conformance testing portion of the 3GPP long term evolution (LTE) specification defines 3 power-delay profiles (PDP) used throughout many test scenarios [1]. The three models, defined by the International Telecommunications Union (ITU), are the extended typical urban (ETU), extended vehicular A (EVA), and extended pedestrian A (EPA) models, each with varying Doppler frequencies specified in Hz (varying mobility conditions). For a designer to perform tests in accordance with the specification, the designer may either resort to software simulation or test using a hardware-based channel emulator. A channel emulator that is capable of supporting the number of modes required in the LTE specification must be programmable, or flexible enough to accommodate each PDP and maximum Doppler frequency.

The developed channel emulator allows the user to specify the PDP and Doppler frequency with high resolution, accommodating all of the specified LTE models. The channel emulator architecture has been developed for easy customization to support a wide variety of channel emulation requirements. In order to maximize compatibility, the design contains many programmable elements, each with a high level of resolution. The presented architecture allows the user to specify the Doppler frequency with resolution on the order of 10 mHz, the channel path delays with 10 ns resolution, and the complex channel gains with 16-bit precision for the real and imaginary components.

II. Jakes Processes

The Jakes process is a classical model of a densely scattered channel [2][3][4], which assumes an omni-directional antenna radiation pattern at both the transmitter and receiver. The Jakes process is useful for simulating realistic conditions in a time-varying mobile fading channel. In this model, each discrete path in the channel will be modeled by a Jakes process. The power spectral density of the Jakes process is defined by [2]

$$S(\nu) = \frac{1}{\pi f_d \sqrt{1-(\nu/f_d)^2}}\ , \tag{1}$$

where $|\nu| \leq f_d$ and $f_d = \frac{v}{\lambda}$. The carrier's wavelength, the mobile handset's velocity, and the maximum normalized Doppler frequency are indicated by λ, v and f_d respectively. To approximate the Jakes process, a finite-impulse response (FIR) filter shapes white Gaussian noise (WGN) to form the desired Jakes spectrum [3][4], defined in Eq. 1. To compute the coefficient set for the Jakes shaping filter, the following

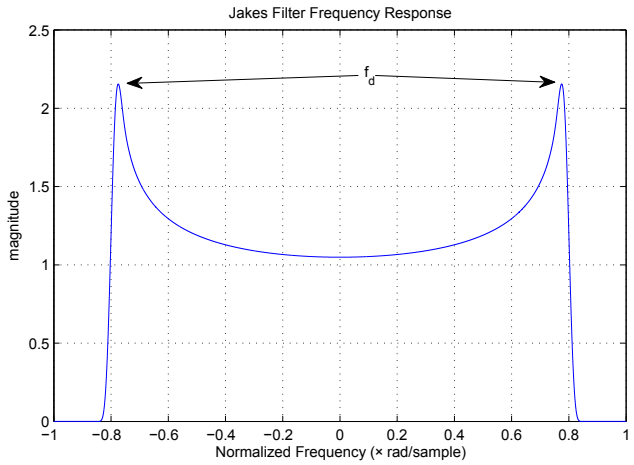

Fig. 1. Frequency Response of the Jakes FIR Filter (Linear)

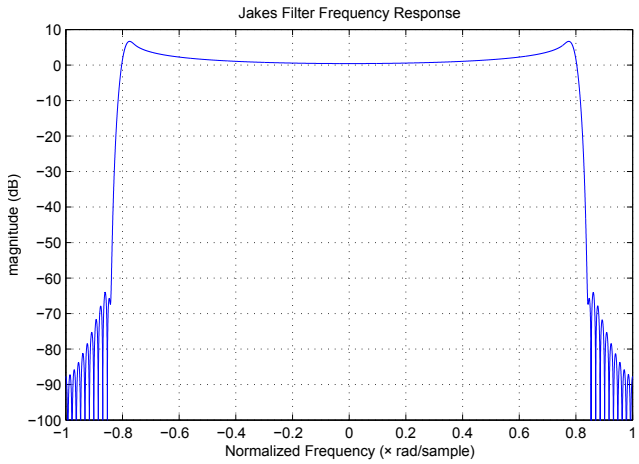

Fig. 2. Impulse Response of the Jakes FIR Filter (dB)

equation is used [3].

$$h_J[t] = 2.583A^{1/2} f_d x^{1/4} J_{1/4}(x) \ , \tag{2}$$

where $J_{1/4}(\cdot)$ is the fractional Bessel function of the $1/4$ kind, $x = 2\pi f_d|t|$, and $A^{1/2}$ is a constant chosen so that $h_J(t)$ has unity power gain. The filter order is chosen so that the filter is as close to critically sampled as possible, while allowing adequate roll-off for later upsampling stages. In this architecture, the Doppler frequency is variable, so the peak of the "bathtub" shape indicates the normalized value of f_d. After choosing an adequate sample spacing and number of samples for t in Eq. 2, the coefficient set $h_J[t]$ must be windowed to guide the tails of the impulse response to zero; the Bessel function in Eq. 2 has infinitely long tails. A Blackman window is applied to $h_J[t]$ to form the final coefficient set.

The frequency and impulse response of the FIR Jakes filter is shown in Fig. 1-3. The log-magnitude frequency response is presented in Fig. 2 to illustrate the roll-off characteristics of the filter produced by the windowing operation. Fig. 3 shows the contribution of the Bessel function, the windowing

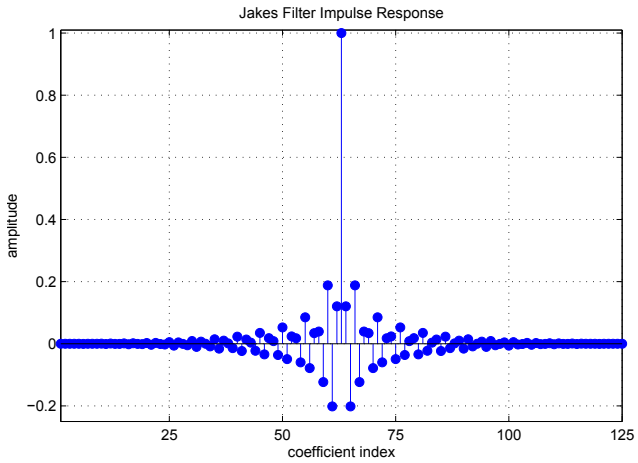

Fig. 3. Impulse Response of the Jakes Filter (After Windowing)

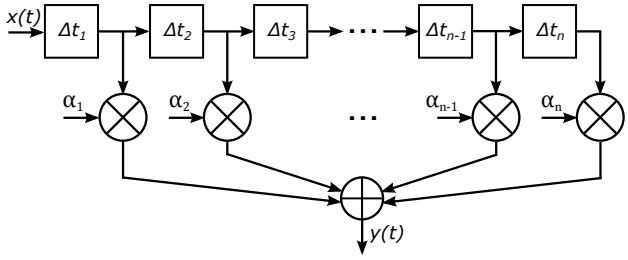

Fig. 4. Channel Filter with Variable Delay Elements and Time-Varying Stochastic Complex Coefficients

operation, and the sample spacing used to generate $h_J[t]$. Two real-valued Jakes filters separately operate on the real and imaginary part of the complex WGN to form the positive and negative frequency ranges found in Figs. 1 and 2.

III. Channel Filter Architecture

The PDP defines both the expected power and the absolute time delay of each path in the channel. The transmitted signal is convolved with the channel as it travels to the receiver. To model this process, a discrete convolution is performed between the transmitted signal and the emulated channel. The time-varying impulse response of the channel is defined by

$$h(\tau, t) = \sum_i \alpha_i(t)\, \delta(\tau - \tau_i) \ . \tag{3}$$

In Eq. 3, τ_i defines the absolute delay of each i^{th} path in the channel, and α_i contains the complex channel attenuation factors for each i^{th} path at time t. Each element in α_i is an independent, time-varying, stochastic Jakes process. The structure shown in Fig. 4 performs the discrete convolution between the transmitted signal $x(t)$ and the channel's time-varying complex path gains $\alpha_i(t)$ to form the received signal $y(t)$. In Fig. 4, the absolute delay values specified by τ_i have been transformed to differential values between each channel path, given by Δt_i.

To allow flexibility in hardware implementation, the design shown in Fig. 4 will be decomposed into basic elements

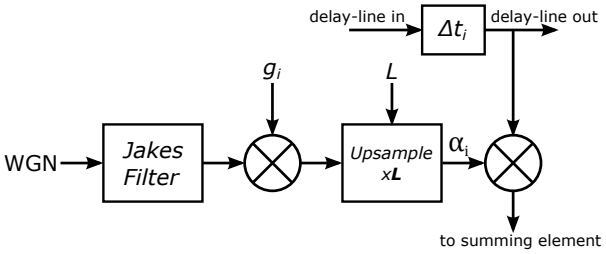

Fig. 5. Unit Cell of the Channel Filter

TABLE I
LTE SPECIFIED ITU CHANNEL MODELS

path index	ETU		EVA		EPA	
	delay (ns)	power (dB)	delay (ns)	power (dB)	delay (ns)	power (dB)
1	0	-1	0	0	0	0
2	50	-1	30	-1.5	30	-1
3	120	-1	150	-1.4	70	-2
4	200	0	310	-3.6	80	-3
5	230	0	370	-0.6	110	-8
6	500	0	710	-9.1	190	-17.2
7	1600	-3	1090	-7.0	410	-20.8
8	2300	-5	1730	-12.0	-	-
9	5000	-7	2510	-16.9	-	-

required to emulate each individual path. Each element can be duplicated as needed, depending on the requirements of the particular channel models to be used. The WGN input to the Jakes filter component is taken from a single unit-variance WGN generator, which is shared among each instantiation of the structure in Fig. 5. The WGN source is designed so that each generated sample is exclusive to the path generator that uses it. The Jakes filter shapes the WGN, generating the Jakes process for the channel path. The programmable gain g_i sets the expected power level for the channel path coefficient, which is then upsampled by a factor of L. The upsampling factor determines the final Doppler frequency. If the path coefficient α_i is generated at the constant sampling rate of the channel filter, the rate of the Jakes filter is increased with decreasing L ($f_{max} \propto 1/L$), thus widening the Doppler spectrum, increasing f_{max}. The programmable delay element in Fig. 5 sets the differential delay between each adjacent path in the overall structure. The delay value is defined by Δt_i.

The sampling rate of the filter is defined by the minimum spacing of the channel taps in the desired set of channel models, or the inverse of the minimum differential tap-delay between each path in the intended channel PDPs. For the LTE models listed in Table I, the sample rate is determined to be $f_s = 1/10$ ns $= 100$ MHz. Therefore, to provide an updated filter coefficient set at the sampling rate of the channel filter, the upsampling stage in Fig. 5 must be programmed with a rate of

$$L = \text{round}\left(\frac{f_s}{f_{max} f_d}\right) \quad . \tag{4}$$

For this application, $f_d \simeq .778$ (see Fig. 1), so to use the EVA70 model [1], which has a maximum Doppler frequency of $f_{max} = 70$ Hz, L is set to $1,836,210$, resulting in an actual Doppler frequency of ~ 70.0000023 Hz. The sampling rate of the filter can be easily adapted to other models with differing minimum tap-delay spacing; the only changes are to f_s and L.

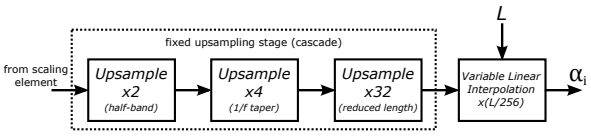

Fig. 6. Upsampling Filter with Variable and Fixed Stages

IV. UPSAMPLING FILTER DESIGN

With the channel filter operating at a high sampling rate, the upsampling stage within each channel path coefficient generator must have a high upsampling factor. The slow sampling rate of the Jakes process, which operates at a slightly higher rate than the Doppler frequency, must be upsampled to the rate of the channel filter. Using some multi-rate design tricks [5], this process becomes realistically realizable in hardware.

Each channel path generator must be designed with careful consideration of resource usage; each path generator must be duplicated many times, depending on the channel model. To minimize complexity, a cascade of upsampling filters has been created, each utilizing a very high level of resource sharing. With an upsampling factor on the order of 1 million, an FPGA implementation that uses a high clock frequency allows the low-rate filters to have a high level of resource sharing. While resource sharing may reduce the number of required multiply-accumulate (MAC) operations, the amount of memory needed to store the coefficient sets must also be considered. Again, using some design tricks [5], the memory requirements of the upsampling operation can be dramatically reduced as well.

A. Fixed-Rate Upsampling Filter Design

Some a priori knowledge allows the upsampling filter design to be optimized. In all realistic situations, the upsampling factor should not be reduced so that the Doppler frequency becomes unrealistically high ($f_{max} \propto 1/L$). The relationship between carrier wavelength, mobile handset velocity, and the Doppler frequency (Eq. 1) suggests that assumptions can be made to define an operating range of Doppler frequencies. The LTE models each use a Doppler frequency of 5, 70, and 300 Hz [1]. To allow flexibility outside of the LTE context, an upper limint of 1,000 Hz is arbitrarily chosen.

1) Upsampling Filter Partitioning: By creating an operating range of Doppler frequencies, the upsampling component can be partitioned into a cascade of fixed and variable-rate sections (Fig. 6). By partitioning the workload of the upsampling filter, the higher complexity fixed-rate portion can be positioned in the system so that it operates at the lowest possible rate, allowing the highest level of resource sharing.

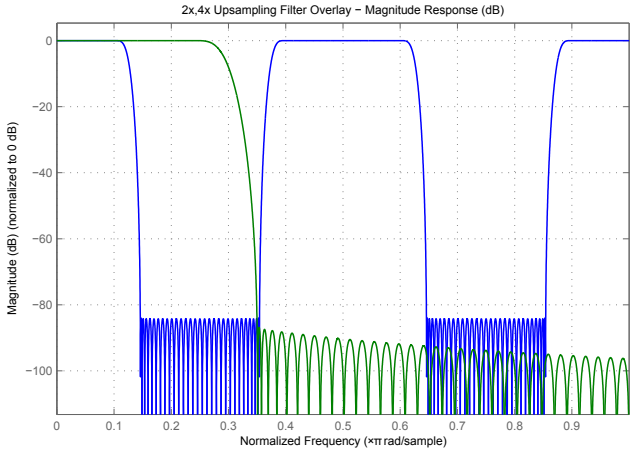

Fig. 7. Upsampling Filters: 2X Overlaid with 4X Frequency Response

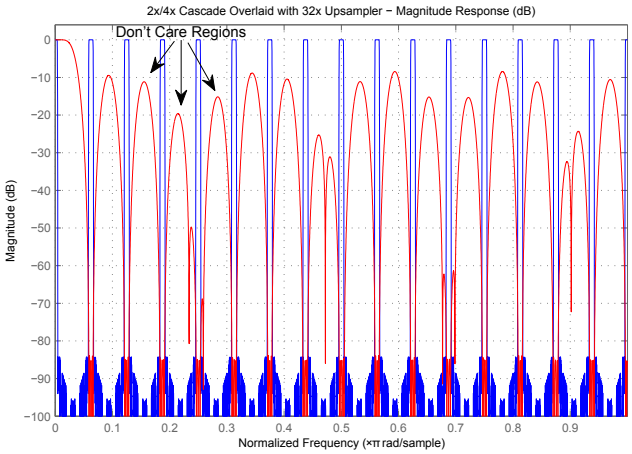

Fig. 8. Upsampling Filters: 2X/4X Cascade Overlaid with 32X Frequency Response

In a cascaded design, the Doppler frequency resolution is inversely proportional to the factor of the fixed-rate component. Also, as the factor of the fixed-rate stage increases, the hardware resource sharing factor decreases. Assuming a maximum Doppler frequency of 1,000 Hz, a final sampling frequency of 100 MHz, and a normalized Doppler frequency $f_d = .778$, the minimum value of L is approximately 260,000. A cascade of fixed-rate upsampling stages with a total upsampling factor of 256X has been found to minimize resource consumption while balancing the Doppler resolution and the workload of the variable-rate component. A factor of 256X defines the minimum variable-rate factor of $\sim 1,000$X, and a Doppler frequency resolution of $\sim .00976$ Hz.

2) Fixed-Rate Cascaded Upsampler Components: Shown in Fig. 6, the first upsampling stage is a 2X half-band upsampling filter. The half-band property exploits zero-valued coefficients to eliminate many of the required MAC operations in the filter [5]. In addition, the half-band filter's symmetry allows the filter to be folded, further reducing the number of coefficients to be stored in memory as well as the overall workload of the filter. With fewer MAC operations and a smaller memory footprint, the symmetric half-band filter can offer a sharper transition band and a higher level of attenuation in the stop-band than a non-half-band design with the same number of coefficients. The frequency response of this filter is the most critical, because it largely determines the pass-band and roll-off behavior in the final response.

Placed after the 2X upsampling stage, a 4X upsampler selects the desired spectral copy with its pass-band. This filter is designed with a $1/f$ stop-band taper, which helps reduce the effects of stop-band quantization error [5]. The transition band is placed at the highest frequency possible so that the number of coefficients in the filter is minimized. The 4x upsampling stage is symmetric, so the filter structure can be folded, requiring only half of the coefficients to be stored in memory. The overlaid responses of both 2X and 4X upsampling stages are shown in Fig. 7.

Finally, a 32X upsampling filter is placed after the 2X/4X

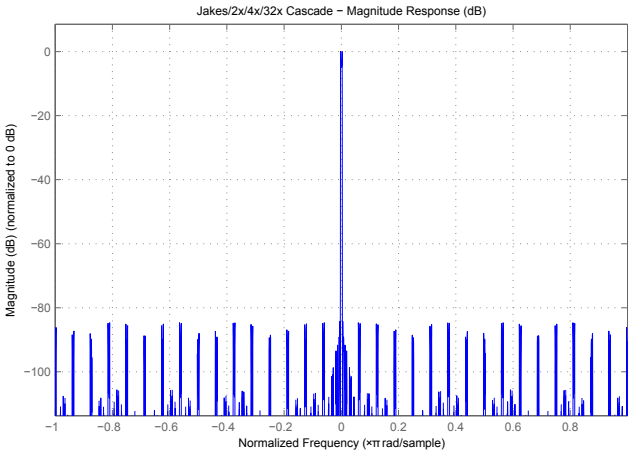

Fig. 9. Full Cascaded Fixed Upsampler Response Including Jakes (Full Frequency Range)

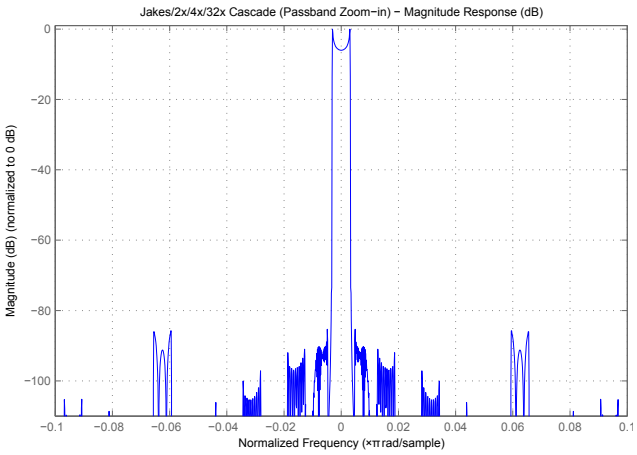

Fig. 10. Full Cascaded Fixed Upsampler Response Including Jakes (Magnified Along the Frequency Axis)

TABLE II
FILTER LENGTHS

Filter	Filter Length	Optimized Length
Jakes	125	63
2x half-band upsampler	59	16
4x 1/f taper upsampler	90	45
32x reduced length upsampler	139	70
total:	413	194

cascade, attenuating only the portions of spectrum that contain spectral copies after the rate transition (Fig. 8). Using the Remez algorithm, a filter is generated that fits this profile [5]. The main-pass band is allowed to be very wide, which slowly tapers into the first stop-band. The width of the pass-band retains spectral flatness throughout the range of frequencies occupied by the selected spectral copy sourced from the 2X/4X cascade. The bands between each spectral copy are considered "don't care" bands, which have been labeled in Fig. 8. The spectral content in these bands has already been attenuated by the 2X/4X cascade; no further attenuation is necessary. By using "don't care" bands, the coefficient length is dramatically reduced, minimizing the workload and storage requirements of the filter. Fig. 9 shows the response of the full cascade of filters, including the Jakes filter. Fig. 10 shows the pass-band region with higher detail, magnified 10X along the frequency axis. The spectral copies attenuated by the 32X stage push the stop-band well below -80 dB, the territory of a practical D/A noise-floor.

The final filter coefficient lengths for each of the fixed-rate upsampling filters are included in Table II. The coefficient vector lengths are given before and after optimization.

B. Variable-Rate Linear Interpolator Design

Linear interpolation relies on only two points to compute the interpolated values. The equation below is used to compute each value between two points generated by the fixed-rate upsampling filter cascade.

$$s\left[n\right] = \frac{1}{N}\left(x\left[m\right]n + x\left[m-1\right]\left(1-n\right)\right) \ , \qquad (5)$$

where $n = [0, 1, \ldots, N-1]$, $x[m]$ is the vector of samples produced by the 256X upsampling filter set, and $N = \text{round}\left(L/256\right)$. In FPGA hardware, the n and $(1-n)$ terms can be generated by an up-counter and a down-counter, respectively. The $1/N$ term can be generated by a hardware division element, which need not be optimized for speed because L is not intended to be dynamically varied. This term can be shared among all paths, so that only a single division component is required because each path uses the same value of L. The linear interpolator requires modest resources and can be easily be scaled to support very large values of N. As seen in Fig. 11, the error added by the linear interpolation is below the target noise floor set by the fixed upsampling stages. Fig. 12 more clearly illustrates the pass-band of the filter with a magnified view along the frequency axis (1000X magnification of Fig. 11). Note that Figs. 11 and 12 illustrate

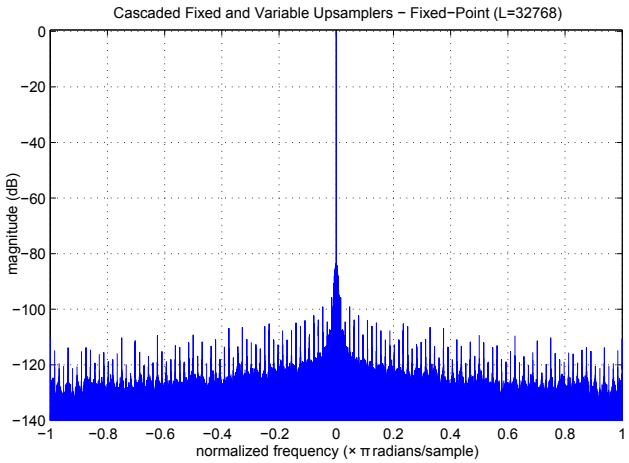

Fig. 11. Full Cascaded Upsampler Response

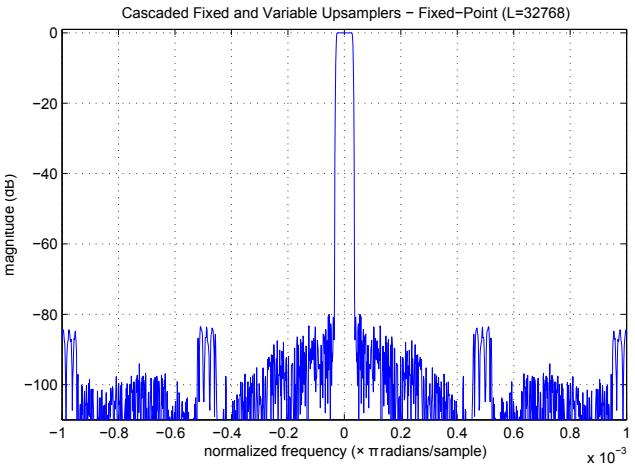

Fig. 12. Full Cascaded Upsampler Response (Magnified Along the Frequency Axis)

fixed-point FPGA simulations, which show the true response of the implemented design. In these simulations L has been set to 32,768.

V. VARIABLE DELAY ELEMENTS

The delay lines in Figs. 4 and 5 require variable delay elements for each value of Δt_i. The dual-port BRAM elements

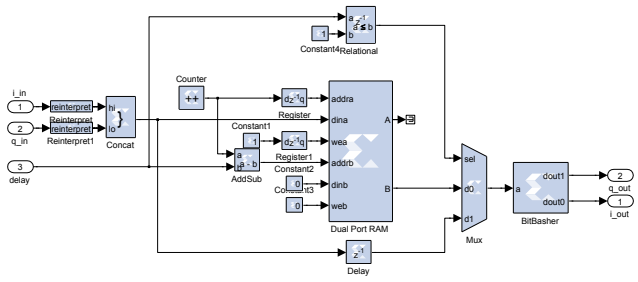

Fig. 13. Variable Delay Element Using a Xilinx Dual-Port BRAM

TABLE III
UNIT CELL FPGA RESOURCE CONSUMPTION (VIRTEX5 SX95T)

	Elements Used/Availbale	Ratio
Occupied Slices	857/14,720	5%
BRAM	6/244	2%
DSP48E	21/640	3%

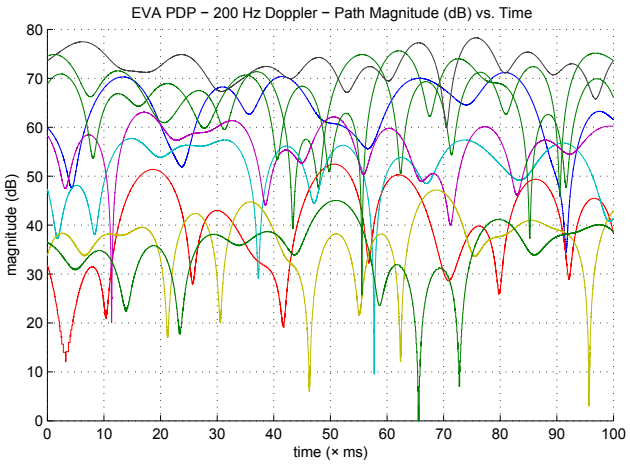

Fig. 14. Magnitude Path Gains using EVA200 Model

in the Xilinx Virtex5/6 FPGAs can be used to efficiently implement a long, variable delay element. A single BRAM can be used to implement a programmable delay element capable of delaying 32-bit data (concatenated 16-bit I and Q) between 2 and 1023 cycles. The RAM writes the incoming data into the "dina" port. The write address is incremented on each clock cycle by a free-running counter. The delay value is subtracted from the value of this counter to determine the read address; therefore the read address always lags the write address by a fixed interval, thus creating the delay. If a delay of 1 is required, the BRAM is bypassed with a MUX selecting a single delay element (Fig. 13). The user programs each value of Δt_i according to the PDP in the desired channel model, e.g. Table I. With $f_s = 100$ MHz, each added delay increases the variable delay element by 10 ns.

VI. RESULTS

The unit cell (Fig. 5) contains all of the necessary elements for each channel path, other than the summation block shown in Fig. 4 and the WGN generator. The WGN generator can be shared among all unit cells in the channel filter, so it is not included in the final resource consumption report. The summation block can be implemented as an adder tree with a depth dependent on the number of implemented unit cells; the effect on the final resource consumption is minimal with the number of paths specified in the LTE channel models. Table III shows the logic consumption for one unit cell of the channel filter, which includes a dual-channel Jakes filter (one filter for each I and Q), path-gain multipliers, dual variable-rate upsamplers (both fixed and variable rate components), complex channel coefficient multipliers, and a variable delay

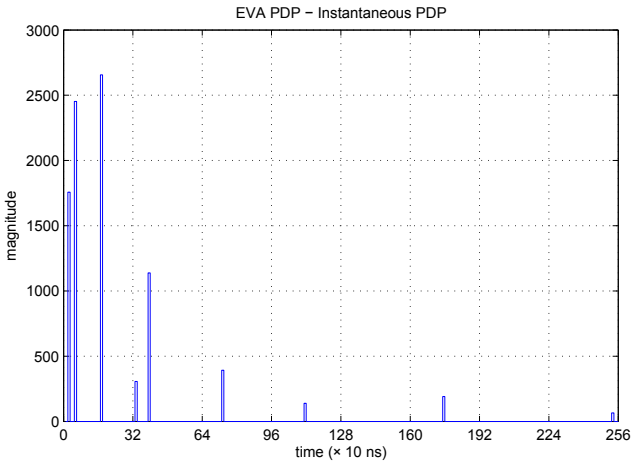

Fig. 15. Instantaneous PDP using the EVA PDP

element. The consumption percentages are generated using the MAP process in the Xilinx design flow for the Xilinx Virtex5 SX95T FPGA using ISE version 13.2.

Fig. 14 shows the time progression of the complex path-gain magnitudes using the EVA200 model. The path-gains are generated using the Innovative Integration X5-400M hardware, equipped with a Xilinx Virtex5 SX9T FPGA, and are logged to the host PC via PCI-express. Each path gain has been downsampled by a factor of 128X before logging. Each path has been scaled using the expected path magnitudes for the EVA model (Table I). The width of the lobes in Fig. 14 are in the range of 5-20 ms, which are expected for a 200 Hz maximum Doppler frequency.

Fig. 15 shows the instantaneous PDP captured from an EVA model emulation; the gain scaling as well as the variable delay elements are appearent here. The EVA model has a total of 9 paths placed at the delays specified in Table I. Note the specified attenuations in Table I are the expected values, which differ from the magnitudes show in Fig. 15.

VII. CONCLUSION

A flexible architecture that can accommodate the LTE channel models specified for conformance testing have been implemented in hardware. The presented architecture utilizes many programmable parameters, allowing the user to utilize a wide variety of channel PDPs and Doppler conditions. The Jakes process generator was presented, showing a method to implement a Jakes shaping filter using a specially designed FIR filter. A unit cell path generator was developed, which can be duplicated in hardware, depending on the number of paths required by the PDP. In addition, a novel multi-rate signal processing technique was presented, allowing upsampling factors on the order of 1 million to be created that requires very few hardware resources.

ACKNOWLEDGMENT

The authors would like to thank Innovative Integration for funding and equipment used for this research.

REFERENCES

[1] 3GPP TS 36.141 V8.9.0: “Base Station (BS) conformance testing”, December 2009.

[2] W.C. Jakes, *Microwave Mobile Communications*, Wiley, New York, 1974

[3] M. Jeruchim, P. Balaban, K. Shanmugan, *Simulation of Communication Systems: Modeling, Methodologies, and Techniques*, Kluwer, New York, 2000

[4] M. Patzold, *Mobile Fading Channels*, Wiley, West Sussex, England, 2002

[5] F. Harris. “Resampling Filters”, in *Multirate Signal Processing for Communications Systems*, Upper Saddle River, NJ: Prentice Hall PTR, 2004, ch. 7, sec. 6