Digital Signal Processing in a Real-Time Propagation Simulator

Mark Kahrs, *Member, IEEE*, and Christopher Zimmer

***Abstract*—The hardware and software interface to a programmable real-time RF propagation simulator is described. First, fading models are reviewed; and next, the digital filter design of the fade spectra filtering is detailed along with the implementation on a floating point digital signal processor (DSP). The hardware system design is given, followed by the application of this system to radio link propagation.**

***Index Terms*—Digital signal processor (DSP) implementation, filter design, propagation simulation.**

I. Introduction

PROPAGATION of radio waves directly affects the performance of a radio receiver. When evaluating new protocols for mobile networks, it is advantageous to perform laboratory experiments under varying propagation conditions. Such experiments are best done in real time; and therefore, the simulator must vary the propagation conditions in real time. This paper describes the real-time signal processing algorithms and architecture for a mobile-radio simulator.

A. Propagation Model

Given a transmitted signal $T(t) = Re\{s(t)e^{j\omega_0 t}\}$ where $s(t)$ is a complex-valued information signal and ω_0 is the carrier frequency, the simulator's output must be $R(t) = Re\{\rho(t)e^{j\omega_0 t}\}$ where $\rho(t)$ is the sum of the attenuated paths as defined in (1).

$$\rho(t) = \sum_{k=0}^{N} \alpha_k s_k(t - \tau_k) e^{j\theta_k} + n(t). \tag{1}$$

Each of the N paths (k) from a transmitter to a receiver is described by its fading-function amplitude (α_k), its carrier phase (θ_k), and its path delay (τ_k). Each of these parameters is time varying according to the propagation model. N is chosen according to the propagation environment; in a reflective environment, N is larger. $n(t)$ is a time-varying noise added to the received signal. Each path can have a different noise function as well as a general noise function, such as a noise source close to the receiver. The complex information signal for a given path can be written in its quadrature parts as

$$s_k(t - \tau_k) = I_k(t - \tau_k) + jQ_k(t - \tau_k). \tag{2}$$

Manuscript received November 1, 2003; revised October 14, 2005.

M. Kahrs is with the Department of Electrical and Computer Engineering, University of Pittsburgh, Pittsburgh, PA 15261 USA (e-mail: kahrs@ee.pitt.edu).

C. Zimmer is with the GRT-MARS, Mountainside, NJ 07092 USA.

Digital Object Identifier 10.1109/TIM.2005.861491

The model requires the fading function to be complex valued. It is possible to replace the fading function with the aggregate noise function of the model if the τ_k is set to an appropriate delay and α_k models the noise. The noise in each path can also be included in each α_k function. The model is illustrated in Fig. 1.

B. Fading Model

The fading functions of a mobile radio channel are Rayleigh distributed [1]. Among the methods proposed for simulation of the radio channel, the most popular method is quadrature amplitude modulation (QAM) by white Gaussian noise sources.

Such a process can be simulated by choosing two numbers, one for I and one for Q, with mutually independent Gaussian statistics that have a zero mean and an equal unit variance. The random-number generator (RNG) takes uniformly distributed and independent white noise random numbers and converts them into Gaussian distributed random numbers. A pair of these Gaussian numbers are taken to be the real and complex part of the fading sequence. However, these numbers alone do not lead to the power spectra for the process that agree with measured data. Therefore, the transfer function $H(z)$ must transform the power spectra into one that corresponds to the environment, including the type of antenna in use and the speed of the vehicle. Clarke [2] and Parsons and Bajwa [3] have shown that the received power spectrum at an antenna can be represented by one of the three power spectra shown in Fig. 2.

The first case is the typical spectra that a vertical monopole antenna sees from a line of sight transmission or from a large amount of local scattering. Its power density spectrum $\Phi(\omega)$ is approximated by

$$\Phi(\omega) = \begin{cases} \frac{a_v}{\sqrt{1-\left(\frac{\omega}{\omega_D}\right)^2}} & : |\omega| < \omega_D \\ 0 & : |\omega| > \omega_D \end{cases} \tag{3}$$

where ω_D is the Doppler frequency at velocity v, and a_v is its amplitude.

The second case in Fig. 2 occurs when signals come from directly ahead of or behind a car, e.g., when a mobile is driving down a city street. The power spectra is approximated by a Gaussian shape $\Phi(\omega) = e^{-x^2/2}$ that is shifted by the Doppler frequency ω_D. Note that the spectra can be nonsymmetric about $\omega = 0$. Therefore, to simulate this power spectra, one must specify the two Doppler frequencies ω_{D-} and ω_{D+}, the bandwidth of each sideband $\Delta\omega_-$ and $\Delta\omega_+$, and the amplitudes of each sideband a_+ and a_-.

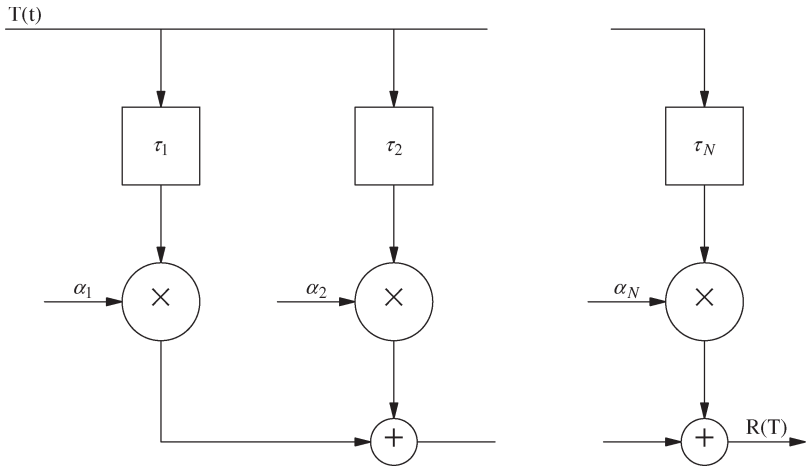


Fig. 1. Simulation of multipath fading.

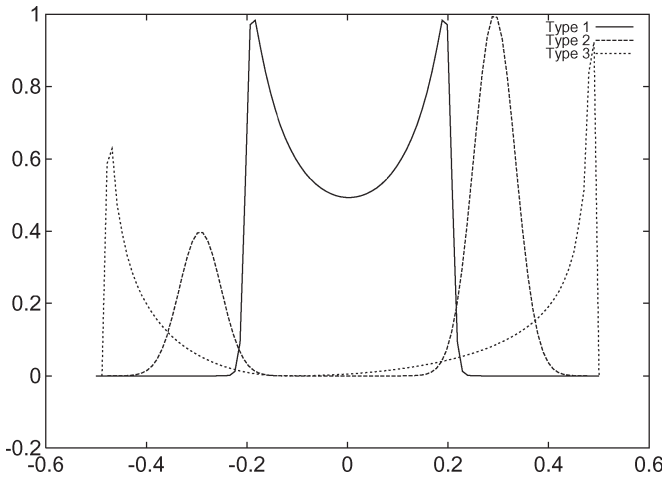


Fig. 2. Fading power spectra.

Finally, the third spectrum in Fig. 2 occurs when echoes also undergo scattering. It also represents the spectrum seen by a vertical loop antenna in the plane of vehicle motion. It is characterized by

$$\Phi(\omega) = \frac{a\left(\frac{\omega}{\omega_D}\right)^2}{\sqrt{1-\left(\frac{\omega}{\omega_D}\right)^2}}. \tag{4}$$

Note that it too can be asymmetric about $\omega = 0$, so two amplitudes a_+ and a_- and two Doppler frequencies ω_D+ and ω_D- must be specified.

With regard to (3), it is possible to implement the power-spectra shaping as either low-pass or bandpass digital filters. A higher carrier frequency would have a higher Doppler shift for a car traveling at some velocity. Then, in modeling an environment at near maximum shift conditions, that is, a 960-MHz carrier with a mobile velocity of 80 mi/h (128.7 km/h) and all waves coming from the front of or behind the mobile $(\cos(\alpha) = 1)$, the Doppler shift would be only 411.8 Hz (2.589 krad).

II. Filter Design

The next three sections discuss the filter design for each of the fading spectra discussed in the previous section. The last section discusses the use of the Hilbert transformer in creating more complex spectra.

A. Type-1 Filters

The frequency response of the type-1 filter is shown in Fig. 2 and (3). Equation (3) can be approximated by a Butterworth filter [4]. This is converted to the digital (z) domain via the bilinear transform [5].

There are two critical frequencies, $\omega = 0$ and $\omega = \omega_D$, which are transformed to the z-domain with the relation $z = \mathrm{e}^{j\omega}$ when evaluated on the z-domain unit circle. These two points shall be called z_0 and z_ω, respectively. A vector can be drawn from each of the poles, p_1 and p_2, to each of these points on the z-domain unit circle. The value of the polynomial at each endpoint frequency on the z unit circle is then the product of the length of the two vectors, and M is the magnitude of the polynomial at that frequency. Allowing the pole p_1 to be written as $a + jb$, then

$$M = z^2 - 2az + (a^2 + b^2). \tag{5}$$

By solving the simultaneous equations for a and b, the pole positions are found

$$a = -\frac{1}{2}\frac{M_0 - M_\omega - \left(z_0^2 - z_\omega^2\right)}{z_0 - z_\omega} \tag{6}$$

$$b = \sqrt{M_0 - z_0^2 + 2a\, z_0 - a^2}. \tag{7}$$

This results in two sets of roots: $a \pm jb$. The root $a - jb$ is chosen so that it lies on the left half of the Butterworth circle, a condition of stability.

Since the cutoff frequency depends on the sampling frequency (as this is a normalized design) and this filter is only one section, it is impossible to design a filter that has the amplitude response desired at both z_0 and z_ω. There is a tradeoff in magnitude response to be made at one frequency if the

magnitude is matched at the other frequency. Since this is an all-pole filter, it has an amplitude response resembling a parabola. The slope of the parabola changes as the poles are moved with respect to the cutoff frequency. However, since it is an all-pole filter, the amplitude response tends toward infinity at the Nyquist frequency $(f_s/2)$, that is, it did not cut off at the desired frequency. A filter that had both the correct magnitude and frequency response could be designed by windowing the response of the special section with the appropriate magnitude response (parabolic slope) with a Butterworth low-pass filter that has the proper frequency cutoff.

B. Type-2 Filters

Recall that the shape of the bandpass filter is Gaussian and can be asymmetric about $\omega = 0$. The asymmetry is brought about by the phase-splitter network to be discussed in Section II-D. The phase splitter takes the output from two symmetric bandpass filters (about $\omega = 0$), each with different bandwidths and amplitude responses, and uses one of the filters to construct the output for $\omega < 0$ and uses the other filter to construct the output for $\omega > 0$. Therefore, only the symmetric bandpass filter design will be discussed later.

Humpherys [6] provides a table of pole locations for a normalized Gaussian magnitude approximation for analog (s-domain) low-pass filters. The table is already in the form of products of second-order sections. These coefficients are used to create sections that are transformed into the digital (z) domain. It should be noted here that only one or two sections may be created for reasons to be discussed shortly. The low-pass filter must be frequency transformed to a bandpass filter to meet the filter requirements. The method chosen to do this does a frequency transform of the digital-domain filter [5]. The filter can also be frequency transformed in the analog domain [4], [7], [8] and then transformed into the digital domain with the bilinear transform.

The frequency transform causes some nonlinearity in the resultant frequency response. Langenthal [9] proposed a method for symmetric low pass to bandpass frequency translations by multiplying the low-pass z-domain polynomials by a cosine function. Although the symmetry of the shifted function is correct, the method can only shift the response by 90°. That is, the low-pass filter is shifted from being symmetric by about $\omega = 0$ to $\omega = f_s/4$. Therefore, this method makes the choices of sampling frequency very limited. However, the spectral distortion introduced by the frequency-transform method of the last paragraph is acceptable in this application.

Note that the frequency transformation of the low-pass filter to the bandpass filter transforms the second-order sections to fourth-order sections.

C. Type-3 Filters

The frequency response of the type-3 filter is shown in Fig. 2 and (4). Techniques used in both type-1 and type-2 filter designs were reused and extended to design this filter. The frequency-response method from the type-1 filter was extended to get a zero magnitude at $\omega = 0$ and controllable magnitude and shape in the passband. This is accomplished by placing a zero at $\omega = 0$ and controlling the shape of the passband with pole placement. By placing a double pole on the negative real axis within the unit z circle, the magnitude function of the second-order section is minimized around $\omega = 0$. If the poles are placed closer to zero, the magnitude of the function stays close to zero further into the passband. Moving the poles closer to -1 causes the magnitude of the function to rise faster, putting more energy into the passband at lower frequencies. If the poles are not on the negative real axis (that is, the poles have no imaginary part), then the magnitude of the function can never be zero. These poles and zeros form one second-order section.

The passband of this special section is the entire frequency range up to the Nyquist frequency. Therefore, as discussed for a type-1 filter, the response of the special section must be filtered with a Butterworth filter to give it the desired response. As with the type-1 filter, it is possible to select different pole positions for the special section to get the desired magnitude response in the passband. The Butterworth filter will further restrict the response of the special section so that the entire filter will have the desired cutoff frequency.

As with the type-2 filter, it may be desirable for the response of the filter system to be asymmetric about $\omega = 0$. Again, this is accomplished by designing two symmetric filters and passing their output through the phase-splitting network. The phase-splitting network takes the response of one filter for $\omega < 0$ and the response of the other filter for $\omega > 0$ and adds them together to give the desired response.

D. Phase Splitter

As discussed above, the phase splitter takes as its input the output from two symmetric filters with zero magnitude at $\omega = 0$ (such as the type-2 and type-3 filters above) and combines them. At the output of the phase splitter, the magnitude response is that of one filter for $\omega < 0$ and that of the other filter for $\omega > 0$. This filter function makes asymmetric filters possible.

A Hilbert transformer (also known as a phase splitter) can be used to perform this function [10]. Oppenheim and Schafer [5] broadly define the Hilbert transform as the relation between the magnitude and the phase of the Fourier transform. An ideal Hilbert transformer for signal processing use can be implemented as an all-pass filter with a constant 90° phase shift. In reality, although the constant 90° phase shift is possible, the passband of the filter has some ripple, and frequencies close to zero and close to the Nyquist frequency are attenuated.

In a classic paper by Gold, Oppenheim, and Rader [11], the implementation of a Hilbert transformer as a phase-splitting network is discussed. The phase splitter is implemented as two all-pass filters whose phase differs by exactly 90°. The same signal is fed to the inputs of the two networks and comes out on the other sides of the networks exactly 90° out of phase. The phase characteristics of each network varies over the frequency range. What is important, though, is that the phase difference is always 90°.

A method to design these filter networks is given by Gold and Rader [12]. They use elliptic filters in the design of the two networks that have a phase difference of $(\pi/2) \pm \epsilon$ (where ϵ is

the phase error) over the frequency range θ_a to θ_b. The order of the filter increases with smaller ϵ's and the closer θ_a and θ_b are to their limits (zero and the Nyquist frequency, respectively). The resulting coefficients from their design algorithm are the poles of each of the branches of the filter. However, they note that any network is all pass, provided that the poles and the zeros can be put in reciprocal pairs.

An efficient block processing algorithm is provided by Gold and Rader that was implemented for use in the simulator. Their design process was also used. Routines to evaluate the elliptic integrals were found by Press *et al.* in [13].

III. Random-Number Generation

Gaussian random-number generation is computationally expensive. In the implementation, we explored two different methods using 1) the Central Limit Theorem and 2) Knuth's statistical method.

A. Central-Limit-Theorem Method

The computation of Gaussian distributed random-number generators follow a method using the Central Limit Theorem [14]. Twelve uniform random numbers in the range of $0 \leq n \leq 1$ are summed and made zero mean. This gives a Gaussian distributed random number with a unit variance. It provides a method to shift the mean and change the variance. Uniform random numbers are generated with a linear congruential generator (LCG) routine. This routine is the most commonly used routine in computing uniform random numbers and is also discussed in Knuth [15]. The output of the LCG routine is periodic, and the period is dependent on the register width of a given machine. To make the numbers more random, the output of the LCG is "shuffled." An array of length M is filled with random numbers from the LCG. When the caller to the shuffling routine requests a random number, the LCG is called by the shuffling routine to supply a random number. This random number is used to index one of the random numbers in the array of random numbers, which is returned to the caller. The empty slot in the array is filled with the random number that was used to index the array. This procedure does not change the set of numbers ultimately received by the caller, but it does change the order in which they are received.

B. Knuth's Statistical Method

Knuth's statistical method [15] is as follows.

1) Generate two uniform random variables U_1 and U_2. Set $V_1 = 2U_1 - 1$ and $V_2 = 2U_2 - 1$ (V_1 and V_2 are now uniformly distributed between -1 and $+1$).
2) Set $S = {V_1}^2 + {V_2}^2$.
3) If $S \geq 1$, go to step 1 [Is the point $(V1, V2)$ inside the unit circle?].
4) Compute two Gaussian distributed random samples as follows:

$$X_1 = V_1\sqrt{\frac{-2\log S}{S}}, \quad X_2 = V_2\sqrt{\frac{-2\log S}{S}}. \qquad (8)$$

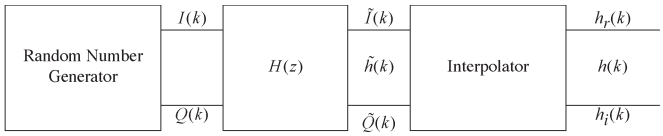


Fig. 3. Fading DSP block diagram.

C. Comparison

The Central-Limit-Theorem algorithm took approximately 1500 clock cycles (375 instructions) to generate one Gaussian distributed sample when implemented in digital signal processor (DSP)-32 code using optimized machine code algorithms. Removing the shuffling and reducing the summed samples from twelve to six cut the execution time to 450 clock cycles ($\approx$ 113 instructions) per Gaussian random sample. Recall that two such samples must be generated, doubling the execution time. This amount of time is unacceptable and needed to be reduced. Also, recall that there is a limited amount of time with which to generate two Gaussian random numbers and filter them.

The statistical method averages approximately 650 clock cycles to generate two Gaussian samples. Knuth notes that the algorithm executes the loop of steps 1–3 1.27 times on the average with a standard deviation of 0.587. Note that the quantity inside the square-root symbol is identical for both equations and needs to be computed only once. Five standard deviations bring the total times that the loop is executed to under three times; this makes Knuth's routine slightly faster in generating Gaussian samples. The square root of (8) and the functions inside of it take 400 clocks (100 instructions) while generating two random numbers take 140 clocks (35 instructions), so that the computed worse case time for the generation of two Gaussian distributed samples is 860 clock cycles. However, when the algorithm was tested on the DSP-32 simulator, the worst case execution time was found to be over 1300 clock cycles.

The first method (Central Limit Theorem) was again evaluated using the sum of six numbers instead of twelve. The Gaussian statistics of the output samples were found to be unchanged. Execution time is a guaranteed 350 clock cycles for two Gaussian samples. Therefore, the modified Central-Limit-Theorem method was chosen for the simulator.

Since the Gaussian number generation is so costly, the clock that measures the computed value of I and Q to the latch that feeds the multiplier had to be slowed down from 1/256th the data rate to 1/1024th the clock rate. This further emphasizes the need for a fast Gaussian sample generator.

IV. Fading Calculation

Then, we show how to use the filter design in Section II and the random-number generation discussed in Section III to calculate fading models. As shown in Fig. 3, the Gaussian RNG is filtered by $H(z)$ (in quadrature) and then interpolated to the sampling rate of the multiplier.

The Gaussian distributed noise generator supplies numbers that have identical power spectra to the input of each channel. However, it is more complicated to generate the asymmetrical

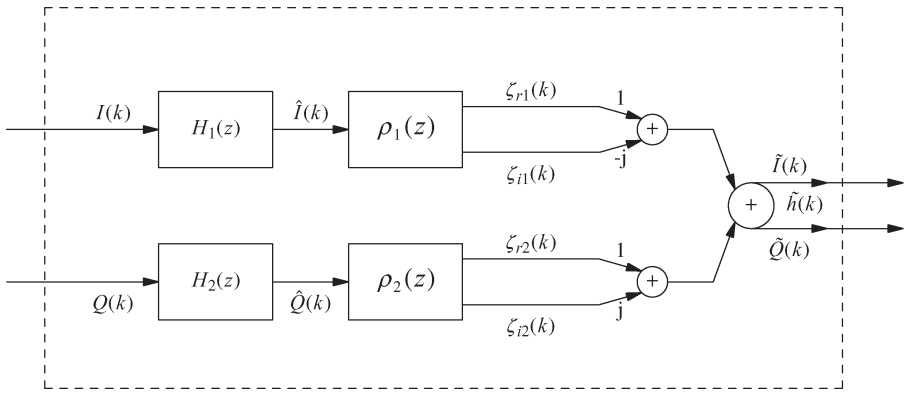


Fig. 4. Quadrature DSP.

spectra of the second and third functions of Fig. 2. The nature of low-pass filters (and bandpass filters that are transformed from low-pass prototypes) is that they are symmetric about $\omega = 0$. Schüssler *et al.* [10] describe the necessary operation as a Hilbert transformer implemented as two all-pass filters, each of which shapes either the upper half or the lower half of the filter spectrum. What is actually implemented is a complex modulation of the filtered real and imaginary parts to remove unwanted sidebands and join the two desired spectra together.

In Fig. 4, the filter functions $H_1(z)$ and $H_2(z)$ are symmetric low-pass or bandpass filters. Their output can be labeled $\tilde{I}(k)$ and $\tilde{Q}(k)$ for the first filter model of Fig. 2. However, for the other two filter models, the rest of the network of Fig. 4 is required. ρ_1 and ρ_2 are all-pass phase-splitting networks. The gain for each branch in the network is unity but the outputs, $\zeta_{r_i}(k)$ and $\zeta_{i_i}(k)$, are exactly 90° out of phase. By adding the outputs of ρ_1 (ζ_{r1} and ζ_{i1}) in the manner shown ($\zeta_{r1} - j\zeta_{i1}$), the spectrum for $\omega > 0$ is obtained and the spectrum for $\omega < 0$ is zero. Also, by adding the outputs of ρ_2, the spectrum for $\omega < 0$ is obtained while the spectrum for $\omega > 0$ is zero. Then, the desired output spectrum $\tilde{h}(k)$ (for Fig. 2, types 2 and 3) is obtained by adding the upper and the lower frequency parts together. Then, note that $H_1(z)$ shapes the frequency response for $\omega > 0$, and $H_2(z)$ shapes the frequency response for $\omega < 0$. Also, note that the frequency response of the functions in Fig. 2 (types 2 and 3) have zero magnitude at $\omega = 0$.

Each block $\rho_i(z)$ contains the two branch phase splitter described above. The output of each block is treated as a complex number, with the imaginary output of $\rho_1(z)$ also being multiplied by -1. The spectrum for $\omega > 0$ is obtained by doing this, while the outputs of $\rho_2(z)$ form the spectrum for $\omega < 0$. Adding the two phase-splitter outputs together as complex quantities sums the two half spectra together, forming the necessary I and Q inputs to the fading channel system. The $H(z)$ block of Fig. 3 can then be realized by Fig. 4.

Instead of an interpolator, as was designed into Schüssler's [10] system, a zero-order hold was provided. A zero-order hold has a frequency response given by [5]

$$H_0(j\Omega) = \frac{2\sin\left(\frac{\Omega T}{2}\right)}{\Omega} e^{-j\frac{\Omega T}{2}} \tag{9}$$

whereas an interpolator has a flat frequency response. This effect can be compensated for by predistorting the output of the fading sequence generator by

$$\tilde{H}_r(j\Omega) = \begin{cases} \frac{\frac{\Omega T}{2}}{\sin\left(\frac{\Omega T}{2}\right)} e^{\frac{j\Omega T}{2}} & : |\Omega| < \frac{\pi}{T} \\ 0 & : |\Omega| > \frac{\pi}{T} \end{cases}. \tag{10}$$

Fitzgerald and Anderson [16] analyze the distortion introduced by a zero-order hold and supply a model for the frequency content that the hold introduces by using a Fourier-series approach. It should also be noted that if the low-pass filters used in the propagation models have their cutoff within the main lobe of (9), spectral distortion is minimized and no corrective action is necessary if one is willing to allow the minimal distortion.

Note that for any of the symmetric filters, the phase splitter is not necessary since the two filters are identical and have equal energy about $\omega = 0$. In this case, the outputs of the filters directly feed the I and Q inputs of the fading model cards. Also in this case, the noise shaping filter coefficients are identical; there are just two different noise samples that need to be filtered.

V. Real-Time Simulator System

Before discussing the hardware of the real-time simulator system, a brief introduction to past real-time simulators will be presented. Then, the overall architecture of the simulator will be discussed along with the software.

A. Brief History of Hardware Simulators

Hardware simulators can be divided into two categories: the "predigital" era of delays and then the digital era, where the signal is digitized and then stored in a queue.

1) Analog Delay Simulators: Table I provides a history of earlier hardware simulators that implement both path delays and fading, which is reproduced in the following table.[1] The table only represents systems that did not implement the path delays digitally. As noted in the table, a Zener diode and filter were used by some authors to generate the Gaussian noise that produces Rayleigh fading.

[1]Note that Fitting's paper lacked enough detail to determine the exact methods of delay and noise generation but coax delay lines, and Zener diodes are reasonable assumptions for the period. He also notes the trend toward all digital methods, using DSPs for fading model generation.

TABLE I
ANALOG DELAY SIMULATORS

Reference	Delay	Fade
Fitting[17]	Coax?	Diode?
Arredondo et al.[18]		Phase shifter → modulator
Caples[19]	SAW delay line	Diode → CCD → LPF → mixer
Arnold[20]	SAW delay line	PRN → DAC → mixer
Berthoumieux[21]	SAW delay line	computer → DAC → mixer
Lecours[22]	SAW delay line	DSP → DAC → mixer
BrTelecom[23]	Optical fiber	DSP → DAC → mixer

TABLE II
DIGITAL DELAY SIMULATORS

Reference	Fading implementation
Casas, Leung[24]	Sum of nine weighted sinusoids of different frequencies.
Hewlett-Packard 11759[25]	microcontroller → DAC → mixer
Schüssler, et al.[10][26]	DSP
Cullen, et al.[27]	IMS A110 + TMS320 → table
Olmos, et al.[28]	hardware multiplier + TMS320C
Salkintzis [29]	TMS320C
Wickert & Papenfuss[30]	TMS320 + FPGA interpolator
Jämsä, et al.[31], [32]	Complex FIR ASICs

2) Digital Delay Simulators: With the development of fast analog-to-digital (A/D) and digital-to-analog (D/A) converters and digital memory, it became possible to implement the delays totally in digital form. Table II shows systems with digital delays and different implementations of the fading functions.

It should be noted that as semiconductor technology has improved, so has simulator implementation. For example, a 100-MHz clock rate [29] for the delay section is now possible, as well as the use of DSP microprocessors [30] instead of hard-wired multipliers.

B. Hardware

Fig. 5 is a diagram of the simulator system. As shown, the input RF signal is down-converted before sampling. A 10-MHz sampling frequency yields a 5-MHz baseband signal, and the down modulation is performed to center the down-modulated signal around the baseband. The 100-ns period of the clock results in 30 m of path delay per delay step.

The RF oscillator is under the control of the workstation and is set to the proper frequency, depending on the transmitter carrier, to give the desired baseband into the A/D converters that are part of the RF frontend.

Schüssler *et al.* [26] described an analog complex-modulation scheme to derive the in-phase and quadrature (I and Q) components. Each of these are low-pass filtered and passed to separate A/D converters. This was also used in our system. Another idea is to use a completely digital complex mixer. The transmitter signal is down modulated to the baseband IF by an analog mixer. It is then sampled by a single A/D converter. The quadrature sampler then provides the I and Q inputs to the simulator at the 10-MHz rate.

The fading channel simulator was made up of twelve circuit cards, each of which simulates one of the possible paths from transmitter to receiver as shown in Fig. 6.

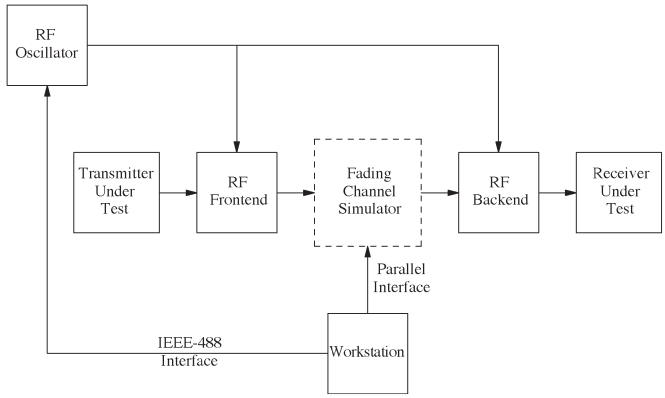


Fig. 5. Block diagram.

If we allow the digitized signal input to the simulator to be described by $u(k) = I(k) + jQ(k)$ and the fading function generated for each path to be given by $h_v(k)$, then Fig. 7 schematically shows the operation of one of the simulation cards.

The propagation delay paths on each card are implemented with a 2048-word first-in first-out (FIFO). At a 10-MHz clock rate, this provides 200-μs maximum delay in 100-ns steps.

The fading sequence generator provides a complex value every 1/256 of the system clock rate. The digital interpolator then provides an input to the multiplier on every clock cycle. There are two hardware multipliers, one for I and one for Q. Similarly, there are two hardware adders to process the data in real time.

The fading-channel-simulator block of Fig. 5 can then be given as Fig. 6, where each block v performs the function of Fig. 2.

There are four parts to the program downloaded to a simulator card:

1) the Gaussian distributed random-number generator;
2) the filter routine;
3) the phase splitter;
4) the filter coefficients.

Each of the fading channel cards contains an AT&T DSP-32C DSP chip and 16 kB of static random access memory (SRAM). Therefore, DSP programs were developed to run on the DSP-32C signal processor and use less than 16 kB of memory. Given the small memory, the solution was to design infinite-impulse-response (IIR) filters that provide the required noise shaping responses and use the standard DSP library functions. Also, a Gaussian RNG was developed (as discussed in Section III) as well as the phase-splitting network. IIR filtering was chosen, since it provides better filtering performance than finite-impulse-response (FIR) filtering in the short amount of real time that can be devoted to the filter algorithm. An output sample must be provided every 1/256 of the clock rate (10 MHz/256 $\approx$ 39 KHz). As previously discussed in Section III-C, the Gaussian number generation is very costly, leaving little time for filtering.

VI. USING THE SIMULATOR

It has been shown in [33], [34] that Rayleigh fading can be modeled by two independent Gaussian distributed noise

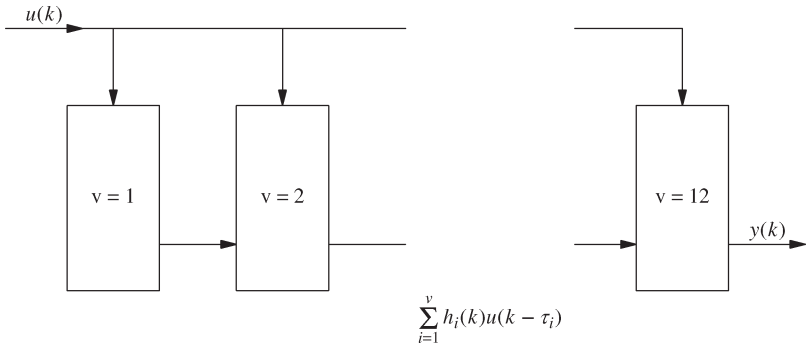


Fig. 6. System flow.

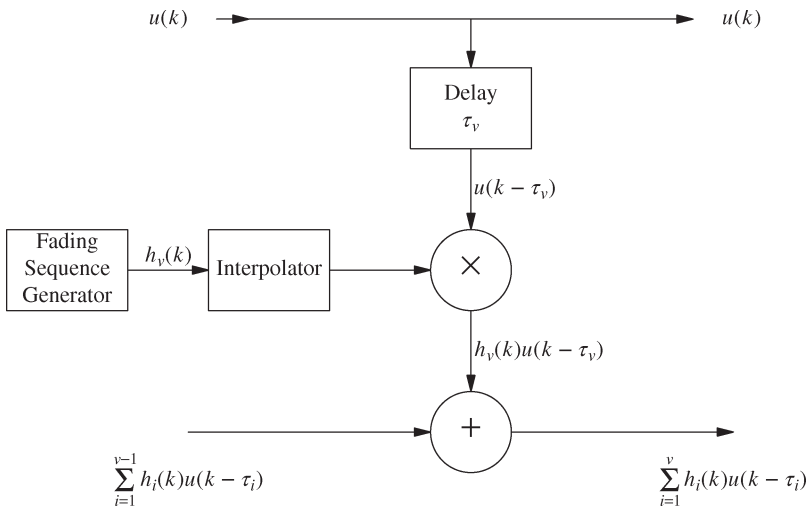


Fig. 7. Delay-path block diagram.

processes that have their frequency content limited to lie within a range of frequencies. Since the transmitted signal is a complex quantity, each Gaussian process models one component of the complex signal. This process produces a Rayleigh-fading envelope. The limit of the frequency content is the Doppler frequency, as discussed in Section I-B. Rayleigh fading is the only type of fading that has so far been implemented on the simulator; one only needs to design the Doppler aspects of the environment with the filter-design programs in Section II. Other types of fading, such as Rician, and impulsive noise sources could be easily incorporated in the existing structure.

A. Choosing a Filter Model

As discussed in Section I-A, there are three types of filters that will model most environments, and their responses are given in the three parts of Fig. 2. Their uses can be summarized, respectively, as

1) symmetrical spectra modeling homogeneous wave propagation;
2) asymmetric spectra modeling receiving conditions that depend on specific angles of the wave's arrivals;
3) asymmetric spectra modeling a combination of types 1 and 2.

One needs to determine the terrain surrounding the receiver and select filter models accordingly. A cutoff frequency must be defined for each filter that is the Doppler frequency for the arriving wave path. Since data on measured Doppler environments exist over different types of terrain, help is available in selecting filter models. Each increment of time delay represents different length paths from the transmitter to the receiver. If two paths result in a similar time delay, it is modeled by a type-3 filter from the above list.

The user supplies the velocity of the vehicle, the frequency of the carrier, and the angle of arrival of the wave in a particular path. The angle of arrival of the wave is a necessary parameter, since we are only modeling twelve paths to the receiver.

A second method to determine the filter's Doppler cutoff frequency would be to assume that the arriving waves are homogeneous and then test under maximum Doppler shift. The worst case Doppler shift is given as $f_m = (vf_c/c)$; where f_c is the carrier frequency, and c is the speed of light. Here, the angle of arrival [α in (1)] is set to zero or to π, setting $\cos\alpha$ equal to ± 1. One would then have to assume that if the test worked under maximum Doppler shift, then it would also work under less severe shift conditions. Type-1 filter is selected.

A third way to define the filter cutoff frequency is to determine the Doppler frequencies that should be tested [35]. These can be less than or equal to the worst case Doppler shift f_m. Then, the angle of arrival can be determined from $\alpha = \cos^{-1}(f_D/f_m)$. Similarly, a range of angular arrivals (α_1, α_2) can be selected and divided over the number of simulator

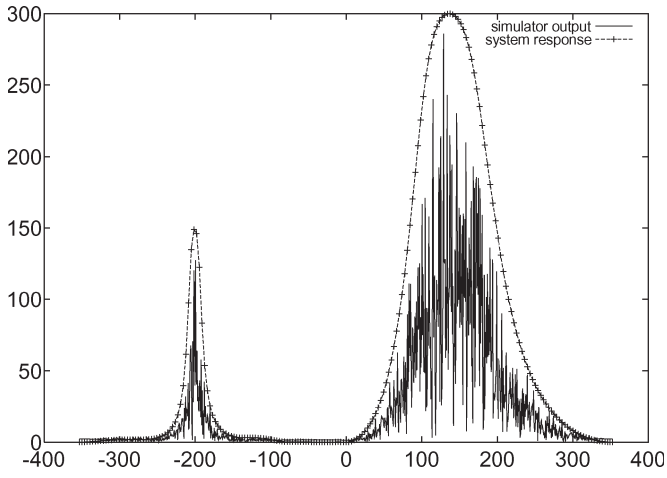


Fig. 8. Simulator output.

cards ($N = 12$ for the simulator discussed here) to give the angular resolution as $(\alpha_2 - \alpha_1)/N$ ([35], pp. 209–214). Again, assuming homogeneous propagation, type-1 filter is selected.

B. Test Results

A filter system was designed and downloaded to the zeroth card. The simulator was started and the data output from the adder was recorded. By looking at the Fourier transform of the recorded data, one can see that it is a noise that is multiplied by the filter response function. The results are shown graphically in Fig. 8.

The solid line shows the magnitude-squared plot of the frequency response of the system. For $\omega < 0$, the response is -200 ± 17 Hz with a magnitude squared at the center frequency of 4. For $\omega > 0$, the response is 141 ± 55 Hz with a magnitude squared at the center frequency of 2. The dotted line is the output of the simulator. It is the Fourier transform of filtered Gaussian distributed noise and has energy only in the passband of the filter.

VII. Conclusion

Digital signal processing of propagation models enables the testing of various radio transmission and receiving algorithms. We have shown how careful design and measurement of filtering and random-noise generation algorithms is necessary to create a real-time hardware simulator for use in studying the effect of propagation on transmitter and receiver designs.

Acknowledgment

The authors would like to thank M. Choey, W. S. Johnson, and J. Kilpatrick, who spent many hours constructing the circuits, and J. F. Kaiser for putting the authors on the right track in the filter design.

References

[1] *Microwave Mobile Communications*, W. C. Jakes, Ed. New York: Wiley, 1974.

[2] R. H. Clarke, "A statistical theory of mobile-radio reception," *Bell Syst. Tech. J.*, vol. 47, no. 6, pp. 957–1000, Jul./Aug. 1968.

[3] J. D. Parsons and A. S. Bajwa, "Wideband characterization of fading mobile radio channels," *Proc. Inst. Elect. Eng.—Part F*, vol. 129, no. 2, pp. 95–101, Apr. 1982.

[4] M. T. Jong, *Methods of Discrete Signal and System Analysis*. New York: McGraw-Hill, 1982.

[5] A. V. Oppenheim and R. W. Schafer, *Discrete Time Signal Processing*. Englewood Cliffs, NJ: Prentice-Hall, 1989.

[6] D. S. Humpherys, *The Analysis, Design and Synthesis of Electrical Filters*. Englewood Cliffs, NJ: Prentice-Hall, 1970.

[7] A. G. Constantinides, "Frequency transformations for digital filters," *Electron. Lett.*, vol. 3, no. 11, pp. 487–489, Nov. 1967.

[8] ——, "Frequency transformations for digital filters," *Electron. Lett.*, vol. 4, no. 7, pp. 115–116, Apr. 1968.

[9] I. M. Langenthal, "The synthesis of symmetrical bandpass digital filters," in *Proc. Symp. Computer Processing Communications*, J. Fox, Ed. Brooklyn, NY: Polytechnic Press (Wiley), 1969, pp. 279–294.

[10] H. W. Schüssler, J. Thielecke, K. Preuss, W. Edler, and M. Gerken, "A digital frequency-selective fading simulator," *Frequenz*, vol. 43, no. 2, pp. 47–55, Feb. 1989.

[11] B. Gold, A. V. Oppenheim, and C. M. Rader, "Theory and implementation of the discrete Hilbert transform," in *Proc. Symp. Computer Processing Communications*, J. Fox, Ed. Brooklyn, NY: Polytechnic Press (Wiley), 1969, pp. 235–250.

[12] B. Gold and C. M. Rader, *Digital Processing of Signals*. New York: McGraw-Hill, 1969.

[13] W. H. Press, B. P. Flannery, S. A. Teukolsky, and W. T. Vetterling, *Numerical Recipes in C, The Art of Scientific Computing*. Cambridge, U.K.: Cambridge Univ. Press, 1988.

[14] S. J. Orfanidis, *Optimum Signal Processing*. New York: Macmillan, 1988.

[15] D. E. Knuth, *The Art of Computer Programming: Volume 2 / Seminumerical Algorithms*. Reading, MA: Addison-Wesley, 1969.

[16] R. Fitzgerald and W. Anderson, "Spectral distortion in sampling rate conversion by zero-order polynomial interpolation," *IEEE Trans. Signal Process.*, vol. 40, no. 6, pp. 1576–1579, Jun. 1992.

[17] R. C. Fitting, "Wideband troposcatter radio channel simulator," *IEEE Trans. Commun. Technol.*, vol. COMM-15, no. 4, pp. 565–570, Aug. 1975.

[18] G. A. Arredondo, W. H. Chriss, and E. H. Walker, "A multipath fading simulator for mobile radio," *IEEE Trans. Veh. Technol.*, vol. VT-22, no. 4, pp. 241–244, Nov. 1973.

[19] E. L. Caples, K. E. Massad, and T. R. Minor, "A UHF channel simulator for digital mobile radio," *IEEE Trans. Veh. Technol.*, vol. VT-29, no. 2, pp. 281–289, May 1980.

[20] H. W. Arnold and W. F. Bodtmann, "A hybrid multi-channel hardward simulator for frequency-selective mobile radio paths," *IEEE Trans. Commun.*, vol. COMM-31, no. 3, pp. 370–377, Mar. 1983.

[21] D. Berthoumieux and J. M. Pertoldi, "Hardware propagation simulator of the frequency selective fading channel at 90 MHz," in *Proc. 2nd Nordic Seminar on Digital Land Mobile Radio Communication*, Stockholm, Sweden, Sep. 1986, pp. 331–336.

[22] M. Lecours and F. Marceau, "Design and implementation of channel simulator for wideband mobile radio transmission," in *Proc. Vehicular Technology Conf. (VTC)*, San Francisco, CA, 1989, pp. 652–655.

[23] M. R. L. Hodges, S. A. Jensen, and P. R. Tattersall, "Laboratory testing of digital cellular radio systems," *Br. Telecom Technol. J.*, vol. 8, no. 1, pp. 57–66, Jan. 1990.

[24] E. F. Casas and C. Leung, "A simple digital fading simulator for mobile radio," *IEEE Trans. Veh. Technol.*, vol. 39, no. 3, pp. 205–212, 1988.

[25] A. Kovalick and P. Titchener, "Bridge the gap between simulation and hardware prototyping," *Electron. Des.*, vol. 39, no. 11, pp. 77–88, Jun. 13, 1991.

[26] H. W. Schüssler, J. Thielecke, K. Preussand, H. Brehm, and M. Werner, "A digital frequency-selective fading simulator—an intermediate report about the state of the development," Lehrstuhl für Nachrichtentechnik, Universitat Erlangen-Nürnberg, Erlangen, Germany, Dec. 1985. Tech. Rep.

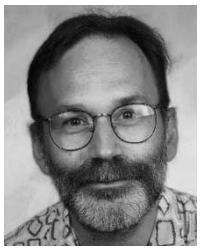
[27] P. J. Cullen, P. C. Fannin, and A. Garvey, "Real-time simulation of randomly time-variant linear systems the mobile radio channel," *IEEE Trans. Instrum. Meas.*, vol. 43, no. 4, pp. 583–591, Aug. 1994.

[28] J. J. Olmos, A. Gelonch, F. Casadevall, and G. Femenias, "Design and implementation of a wide-band real-time mobile channel emulator," *IEEE Trans. Veh. Technol.*, vol. 48, no. 3, pp. 746–764, May 1999.

[29] A. K. Salkintzis, "Implementation of a digital wide-band mobile channel simulator," *IEEE Trans. Broadcast.*, vol. 45, no. 1, pp. 122–128, Mar. 1999.

[30] M. A. Wickert and J. Papenfuss, "Implementation of a real-time

frequency- selective rf channel simulator using a hybrid DSP-FPGA architecture," *IEEE Trans. Microw. Theory Tech.*, vol. 49, no. 8, pp. 1390–1397, Aug. 2001.
[31] T. Jämsä, T. Poutanen, and J. Meinilä, "Implementation techniques of broadband radio channel simulators," in *Proc. Vehicular Technology Conf.*, Rhodes, Greece, 2001, vol. 1, pp. 433–437.
[32] T. Poutanen and J. Kolu, "Correlation control in the multichannel fading simulators," in *Proc. Vehicular Technology Conf.*, Rhodes, Greece, 2001, vol. 1, pp. 318–322.
[33] S. O. Rice, "Mathematical analysis of random noise," *Bell Syst. Tech. J.*, vol. 23, no. 3, pp. 282–332, Jul. 1944.
[34] ——, "Mathematical analysis of random noise," *Bell Syst. Tech. J.*, vol. 24, no. 1, pp. 46–156, Jan. 1945.
[35] W. C. Y. Lee, *Mobile Communications Engineering*. New York: McGraw-Hill, 1982.



Mark Kahrs (S'78–M'82) received the A.B. degree in applied physics and information science (with high honors) from Revelle College, University of California, San Diego, in 1974 and the Ph.D. degree in computer science from the University of Rochester, Rochester, NY, in 1984.

He has held positions at Stanford University, Stanford, CA, Xerox PARC, Institute de Research Acoustique Musique, Paris, France, Bell Laboratories, and Rutgers University, New Brunswick, NJ. In the spring of 2001, he was a Fulbright Scholar at the Acoustics Laboratory, Helsinki University of Technology. He is currently a Visiting Associate Professor in the Department of Electrical Engineering, University of Pittsburgh, Pittsburgh, PA. His microwave and RF interests include hardware description languages for microwave and RF design, applications of digital signal processing to RF instrumentation, and high-speed sampling circuits and methodology.

Christopher Zimmer received the B.S. degree from the New Jersey Institute of Technology, Newark, in 1983 and the M.S. degree from Rutgers, the State University of New Jersey, New Brunswick, in 1993, all in electrical engineering.

He was a member of Technical Staff (MTS) at Bell Laboratories and is currently the Head of the Electrical Engineering Department at GRT-MARS, Mountainside, NJ.