In this paper, we implement a real-time hardware triply selective Rayleigh fading simulator. This simulator incorporates the inter-tap and spatial correlation matrices into multiple uncorrelated frequency-flat Rayleigh fading waveforms (including temporal correlation) to simulate a multiple-input multiple-output (MIMO) triply selective Rayleigh fading channel. In the correlation incorporation procedure, this simulator uses a Kronecker product method to save a large amount of hardware memories. Occupying 34% hardware resources of one Stratix III FPGA chip, this simulator can simulate $4 \times 4$ MIMO fading channels with 10 correlated delay-taps per subchannel in real-time for a symbol rate of $3.69 \mu s$. Accuracy of this simulator is proved by comparing the statistical properties of its outputs to corresponding theoretical values, and they match perfectly.

**Index Terms**— FPGA hardware implementation, MIMO triply selective Rayleigh fading channel, real-time simulation.

### 1. INTRODUCTION

Wireless fading channel modeling and simulation provide a fast and low-cost method for testing and verifying algorithm design, transceiver products, and channel capacity analysis. One significant statistical property for wireless fading channel models is the correlation of fading channels waveforms. The subchannels of a MIMO Rayleigh fading channel are time-selective (described by temporal correlation), frequency-selective (exhibiting inter-tap correlation), and space-selective (associated with spatial correlation of transmitters and receivers). This is referred to as the triply selective fading channel containing three types of correlations.

A discrete-time MIMO triply selective Rayleigh fading channel model and software simulation are proposed by [1]. But hardware implementation of MIMO triply selective simulators presents some challenges in accurately computing and incorporating three types of correlations into the discrete-time model. Current reported hardware MIMO simulators do not implement all three types of correlations and may result in inaccurate channel characteristics. For example, the simulator in [2] outputs multiple uncorrelated frequency-flat Rayleigh fading waveforms as MIMO subchannels; while another simulator in [3] attempts to incorporate the inter-tap and spatial correlation matrices into multiple frequency-flat Rayleigh fading waveforms.

In this paper, we propose a hardware implementation method for the discrete-time MIMO triply selective fading simulator on a Stratix III FPGA DSP development kit. This simulator implements all three types of correlations of triply selective channels. The frequency-flat Rayleigh fading waveforms with temporal correlation or Doppler spectrum are generated using a Sum-of-Sinusoids (SOS) method. The inter-tap correlation matrix associated with multipath delay spread is computed according to a channel power delay profile (PDP) and transmit/receive filters. The spatial correlation matrices, including the transmit correlation and receive correlation matrices, are pre-defined inputs associated with antenna arrangements. The matrix square roots of correlation matrices are calculated using an eigenvalue decomposition (EVD) method. Then they are combined with multiple uncorrelated frequency-flat Rayleigh fading waveforms using the Kronecker product and vector multiplicity. The results of the Kronecker product are computed in real-time for saving hardware memory. Statistical properties of simulator outputs are analyzed and compared to corresponding theoretical ones for performance evaluation.

### 2. DISCRETE-TIME MIMO TRIPLY SELECTIVE RAYLEIGH FADING MODEL

With accurate statistical properties and computational efficiency for hardware implementation, the discrete-time MIMO triply selective fading model in [1] is chosen as the basis of our hardware implementation. In [1], the MIMO channel matrix at time instant $k$ and delay tap $q$ can be represented as an $(OPQ) \times 1$ coefficient vector $h_{vec}(k)$, which is defined as

$$h_{vec}(k) = [h_{1,1}(k), \ldots, h_{1,P}(k) \mid \ldots \mid h_{O,1}(k), \ldots, h_{O,P}(k)]^T \quad (1)$$
where $P$ and $O$ are the numbers of transmit and receive antennas, respectively (Note we assume the sampling interval being $T_s$). The vector $h_{o,p}(k)$ is the $(o,p)$-th subchannel FIR coefficient vector at time instant $k$, which is given by

$$h_{o,p}(k) = [h_{o,p}(−Q_1,k),...,h_{o,p}(q,k),...,h_{o,p}(Q_2,k)]$$

where $Q_1$ and $Q_2$ are nonnegative integers representing the range of $q$, and $Q = Q_1 + Q_2 + 1$.

In simulation, the vector $h_{vec}(k)$ can be simulated by

$$h_{vec}(k) = C^\frac{1}{2}(0) \cdot \Phi(k) = (\Psi^\frac{1}{2}_{Rs} \otimes \Psi^\frac{1}{2}_{Ts} \otimes C^\frac{1}{2}_{ISI}) \cdot \Phi(k)$$

where $\otimes$ denotes the Kronecker product; $X^\dagger$ is the square root of matrix $X=\mathbf{X}^* \cdot (\mathbf{X}^*)^H$; the matrices $\Psi_{Rs}$ and the matrix $\Psi_{Ts}$ are the spatial correlation matrices determined by the transmit and receive antennas, respectively; $C_{ISI}$ is the inter-tap correlation matrix; the vector $\Phi(k)$ is an $(OPQ) \times 1$ vector and $\Phi(k)=[Z_1(k),Z_2(k),...,Z_{OPQ}(k)]^T$, where $Z_i(k)=Z_{c_i}(k)+jZ_{s_i}(k)$ is one of multiple uncorrelated frequency-flat Rayleigh fading waveforms. Each frequency-flat Rayleigh fading waveform $Z_i(k)$ can be efficiently simulated by the SOS method proposed in [4].

For the proposed simulator, the square roots of spatial correlation matrices $\Psi_{Rs}$ and $\Psi_{Ts}$ are specified by users. The inter-tap correlation matrix $C_{ISI}$ is denoted as

$$C_{ISI} = \begin{pmatrix}
 c(−Q_1,−Q_1) & \cdots & c(−Q_1,Q_2) \\
 \vdots & \ddots & \vdots \\
 c(Q_2,−Q_1) & \cdots & c(Q_2,Q_2)
\end{pmatrix}$$

where its coefficients $c(q_1,q_2)$ can be calculated by

$$c(q_1,q_2) = \sum_{n=1}^{N} \sigma_n R_{P_1P_2}(q_1T_s−\tau_n)R_{P_2P_R}(q_2T_s−\tau_n)$$

where $R_{P_1P_2}(ξ)$ is the convolution function of the transmit filter and receiver filter, $N$ is the number of total resolvable paths in PDPs; $*$ denotes the conjugate operator. Parameters $σ_n$ and $τ_n$ are determined by the discrete-time PDPs, $G(τ)=\sum_{n=1}^{N} σ_n^2 δ(τ−τ_n)$, which are often specified by communication standards like [5].

### 3. HARDWARE IMPLEMENTATION METHOD

Our proposed hardware simulator can output $h_{vec}(k)$ in real-time. For the convenience of description, we give new indices elements of $h_{vec}(k)$,

$$H(l,k) = h_{o,p}(q,k)$$

where $l = Q \cdot [(o - 1) \cdot P + (p - 1)] + (q + Q_1 + 1)$. Therefore, $h_{vec}(k)$ can be described as

$$h_{vec}(k) = [H(1,k), H(2,k),...,H((OPQ),k)]^T$$

where $H(l,k)$ is a complex fading coefficient and $H(l,k)=Hc(l,k)+jHs(l,k)$.

The proposed hardware simulator consists of four major modules, as shown in Fig. 1. The flat Rayleigh fading generator (FRFG) module generates multiple uncorrelated frequency-flat Rayleigh fading waveforms $Z_i(Rk)$, which have a decimation rate $R$. The $C^\frac{1}{2}_{ISI}$ generator module computes the inter-tap correlation matrix and its square root. The correlation multiplier (CM) module implements the Kronecker product and vector multiplicity in hardware to perform (3). The interpolator module linearly interpolates samples with an interpolation rate $R$ to increase the output speed for meeting real-time requirement.

Among the four modules, the $C^\frac{1}{2}_{ISI}$ generator module and CM module are novel hardware implementations proposed by our paper. The FRFG and interpolator module are similar to the SOS simulator in [3] and will not be described here.

The $C^\frac{1}{2}_{ISI}$ generator module consists of two submodules: the $C_{ISI}$ generator module that computes the coefficients of $C_{ISI}$ according to (4) and (5), and the matrix square root (MSR) module that calculates the square root of $C_{ISI}$. The datapath of the $C^\frac{1}{2}_{ISI}$ generator module is shown in Fig. 2. The Counters $q_1$ and $q_2$ are up counters with same output ranges from $−Q_1$ to $Q_2$. The Counter $q_1$ increases by one in every $(NQ)$ basic clock periods (BCP); while the Counter $q_2$ increases by one in every $N$ BCPs. Two buffers store $σ_n^2$ and $τ_n$, respectively, and sequentially output them. The results of $R_{P_1P_2}(ξ)$ and $R_{P_2P_R}(ξ)$ are computed using a lookup table (LUT) scheme. We note the result of $R_{P_1P_2}(ξ)$ is always a real value, which causes $R_{P_1P_2}(ξ) = R_{P_1P_2}(ξ)$. Besides, $R_{P_2P_R}(ξ)$ is an even function, so the size of the LUT can be reduced to half by only storing the result of $R_{P_2P_R}(ξ)$ where $ξ$ is a nonnegative value. In our implementation, the LUT $R_1$ is a $D_1$-entry LUT and store the
results of $R_{p}P_{p}(\xi)$ where $\xi=(0 : 4T_s)$. The LUT $R_2$ is a copy of $R_1$. If the results of $[p_1T_s - \tau_n]$ and $[q_2T_s - \tau_n]$ are larger than $4T_s$, the $R_1$ and $R_2$ output zeros. If not, they are converted into the proper read addresses of $R_1$ and $R_2$. The outputs of $R_1$ and $R_2$, and corresponding $\sigma_n$ are multiplied together. In every $N$ BCPs, the accumulator sums $N$ previous inputs to obtain one coefficient of $C_{ISI}$, $c(q_1, q_2)$. The $C_{ISI}$ generator module sequentially outputs the coefficients of $C_{ISI}$ in a row-wise order.

The MSR module employs the EVD method to find the matrix square root of $C_{ISI}$ [6]. We note $C_{ISI}$ is a symmetric positive definite matrix, whose eigenvalues are always positive. The coefficients of $C_{ISI}$ are stored in a buffer and sent to the EVD module which performs EVD. We employ the Jacobi rotation algorithm to perform EVD, since it is a well-known and accurate method for hardware implementation, the details of which are introduced in [7]. The EVD module outputs three matrices $V_{C_{ISI}}$, $D_{C_{ISI}}$, and $V_{C_{ISI}}^{-1}$, where $C_{ISI} = V_{C_{ISI}}D_{C_{ISI}}V_{C_{ISI}}^{-1}$. The coefficients of $D_{C_{ISI}}$ sequentially pass through a square root calculator. Their results are the eigenvalues of the matrix $D_{C_{ISI}}^{{1}/2}$. The coefficients of $C_{ISI}^{{1}/2} = V_{C_{ISI}}D_{C_{ISI}}^{{1}/2}V_{C_{ISI}}^{-1}$ are computed using two matrix multiplier modules. Eventually, the MSR module sequentially outputs them in the row-wise order.

The CM module incorporates the inter-tap and spatial correlation matrices into multiple uncorrelated frequency-flat Rayleigh fading waveforms. It consists of two submodules: the Kronecker product (KP) module, which computes the Kronecker product of $\Psi_{Rx}^{{1}/2}$, and $C_{ISI}^{{1}/2}$ to obtain $C_{R_1}^{{1}/2}(0)$, and the vector multiplier (VM) module, which implements $C_{R_1}^{{1}/2}(0) \cdot \Phi(k)$. Our proposed simulator does not need to store the large size matrix $C_{R_1}^{{1}/2}(0)$, but employs the KP module to compute it in real-time. The datapath of the CM module is shown in Fig. 3. The RAM A, RAM B, and RAM C store the coefficients of $\Psi_{R_1}^{{1}/2}(O \times O)$, $\Psi_{R_1}^{{1}/2}(P \times P)$, and $C_{ISI}^{{1}/2}(Q \times Q)$ in the row-wise order. Several counters, multipliers, and adders work together to generate the proper read addresses for three RAMs. The clock periods of Counters 1 – 6 are measured by integer BCPs, while their module is related to $O$, $P$, and $Q$. Two multipliers are employed to multiply outputs of three RAMs together. Their results are the coefficients of the matrix $C_{R_1}^{{1}/2}(0)$ in the row-wise order.

The VM module takes multiple uncorrelated Rayleigh fading waveforms $Z_{c_1}(Rk)$ and $Z_{s_1}(Rk)$ from the FRFG module, and rearranges their order by using two buffers. Taking the buffer storing $Z_{c_1}(Rk)$ for example, the buffer stores the sequence: $Z_{c_1}(Rk)$, $Z_{c_2}(Rk)$, ..., $Z_{c_{OPQ}}(R_k)$, repeatedly outputs it $(OPQ)$ times, and then do the same to the next sequence: $Z_{c_1}(R(k+1))$, ..., $Z_{c_{OPQ}}(R(k+1))$. The outputs of two buffers are separately multiplied by the coefficients of $C_{ISI}^{{1}/2}(0)$. In every $(OPQ)$ BCPs, the accumulator sums the $(OPQ)$ previous inputs to obtain one single $H_c(l, Rk)$ or $H_s(l, Rk)$. Therefore, it takes $(OPQ)$ BCPs to generate one single $H_c(l, Rk)$ or $H_s(l, Rk)$, and $(OPQ)^2$ BCPs to generate all $H_c(l, Rk)$ or $H_s(l, Rk)$ where $l$ ranges from 1 to $(OPQ)$ and $k$ is fixed.

### 4. EXAMPLES AND PERFORMANCE EVALUATION

The discrete-time MIMO triply selective fading simulator was implemented on an Altera Stratix III EP3SL150F1152C2N FPGA DSP development kit. We used Quartus II version 8.0, DSP Builder version 5.0, and Matlab Simulink for this development.

Memory usage evaluation for the KP module was performed. The proposed KP module computes $C_{R_1}^{{1}/2}(0)$ in real-time without storing. An alternative method is the pre-compute and store method where the matrix $C_{R_1}^{{1}/2}(0)$ is pre-computed by software and stored in

---

**Fig. 3.** The datapath of the CM module

**Fig. 4.** The memory usage comparison of the pre-compute and store method and the proposed KP method
The performance of the simulator was evaluated through a hardware implementation example with specified parameters. The Matlab simulation and theoretical results with identical parameters have been reported by [1]. We analyzed the statistical properties of hardware outputs and compared them to the theoretical ones. The size of $\Psi_{Tx}$, $\Psi_{Rx}$, and $C_{ISI}$, were $O=P=2$, and $Q=4$, respectively. The matrices $\Psi_{Tx}$ and $\Psi_{Rx}$ were given as follows:

$$
\Psi_{Tx} = \begin{pmatrix}
1.0000 & 0.2154 \\
0.2154 & 1.0000
\end{pmatrix}
$$

$$
\Psi_{Rx} = \begin{pmatrix}
1.0000 & -0.3042 \\
-0.3042 & 1.0000
\end{pmatrix}
$$

The PDP was an exponential function for $0 \leq \tau_n \leq 5 \mu s$. The transmit filter was a linearized Gaussian filter with a time-bandwidth product 0.3, and the receive filter was an SRC filter with a roll-off factor 0.3. Other implementation parameters were: $f_{\text{clock}}=50\,\text{MHz}$, $T_s=3.69\,\mu s$, $f_c T_s=0.001$, and the interpolation rate $R=140$.

Based on the hardware outputs, the auto/cross-correlation between several triply selective channels was computed and depicted in Fig. 5. The matrix $C_{ISI}$ with $q_1 = -1, 0, 1, 2$ and $q_2 = -1, 0, 1, 2$ was computed and shown as

$$
C_{ISI} = \begin{pmatrix}
0.0011 & 0.0426 & 0.0178 & -0.0016 \\
0.0426 & 0.3664 & 0.3407 & 0.0367 \\
0.0178 & 0.3407 & 0.5583 & 0.1414 \\
-0.0016 & 0.0367 & 0.1414 & 0.0602
\end{pmatrix}
$$

Therefore, three theoretical curves were 0.5583, 0.3407 and -0.1036 multiplying by $J_0[2\pi f_c(k_1 - k_2)T_s]$, respectively. As can be seen, the correlation curves of hardware outputs matched them very well.

We evaluated hardware resource usage using a hardware implementation example with $O=P=4$ and $Q=10$. Hardware usage is summarized in Table I, where ALUT denotes adaptive look-up table, DLR is dedicated logic register, BM denotes block memory, and DSP means the DSP blocks (high-speed 18-bit multipliers). The percentage uses of total hardware resources were roughly one third for ALUT, DLR, and BM of one Stratix III FPGA chip and slightly more than a half of the DSP multipliers were utilized.

<table>
<thead>
<tr>
<th></th>
<th>ALUT</th>
<th>DLR</th>
<th>BM bits</th>
<th>DSP</th>
</tr>
</thead>
<tbody>
<tr>
<td>$C_{ISI}$ Generator</td>
<td>22636</td>
<td>36586</td>
<td>1194944</td>
<td>143</td>
</tr>
<tr>
<td>FRFG</td>
<td>11988</td>
<td>231</td>
<td>618743</td>
<td>16</td>
</tr>
<tr>
<td>CM</td>
<td>648</td>
<td>1111</td>
<td>10304</td>
<td>10</td>
</tr>
<tr>
<td>Other</td>
<td>120</td>
<td>429</td>
<td>96098</td>
<td>25</td>
</tr>
<tr>
<td>Total</td>
<td>35392</td>
<td>38357</td>
<td>1920089</td>
<td>194</td>
</tr>
</tbody>
</table>

(31%) (34%) (34%) (51%)

5. CONCLUSIONS

A hardware discrete-time MIMO triply selective Rayleigh fading simulator has been implemented on an Altera Stratix III FPGA DSP development kit. This simulator is capable of simulating MIMO triply selective fading channels with all three types of correlations in real-time. The outputs of the simulator are evaluated and proved to contain accurate statistical properties as expected.

6. REFERENCES


