An FPGA Prototype of Dual Link Algorithm for MIMO Interference Network

Mingda Zhou†, Xinming Huang†, Yuteng Zhou†, Xing Li‡, Youjian(Eugene) Liu‡
†Worcester Polytechnic Institute, Worcester, MA 01609, USA, {mzhou2, xhuang, ytchou}@wpi.edu
‡University of Colorado, Boulder, CO 80309, USA, {xing.li, youjian.liu}@colorado.edu

***Abstract*—This paper presents an FPGA-based prototype of the dual link algorithm that maximize the achievable weighted sum rate for MIMO interference network. The iterative algorithm is fast monotone convergent but it must be completed quickly through pilot signaling. Therefore we propose an FPGA-based implementation, targeting software defined radio (SDR) platforms, that is designed to rapidly process the received pilot signals, estimate local channel information, and compute the transmit signal covariance matrices. Compared to its implementation on a CPU, the FPGA implementation is 2 to 4 times faster using fixed-point or floating-point designs.**

***Index Terms*—software defined radio, MIMO, interference network, FPGA, rate maximization**

I. INTRODUCTION

The available radio spectrum for wireless communications is becoming more and more scarce and expensive. The popular method of increasing network capacity is to increase spatial reuse by reducing the cell size for spectrum reuse. However, increasing the density of access points or base stations cannot truly resolve the capacity crunch without proper interference management, meaning the transmit signals should be designed to strike an optimal balance between maximizing a link's own rate and reducing interference to other links. A simple example of MIMO interference networks is shown in Figure 1, where data links mutually interfere with each other.

Weighted minimum mean square error (WMMSE) algorithm [1] and polite water-filling (PWF) algorithm [2], [3] are two important algorithms that maximize the weighted sum rate of MIMO interference network. However, both the WMMSE and PWF algorithms have drawbacks. The PWF algorithm has a fast speed of convergence while it is not guaranteed to be monotone. The WMMSE algorithm converges to a stationary point but it converges slowly. Recently, we designed the dual link algorithm, in order to manage the interference and maximize the weighted sum rate of the network. The dual link algorithm is scalable and has the advantages of both WMMSE and PWF algorithms, i.e., fast monotone convergence. It jointly optimizes the covariance matrices and is ideally suited for distributed implementation that only requires local channel information in a time division duplex (TDD) system.

We aim to prototype our dual link algorithm using software defined radio (SDR). SDR was initially regarded as a promising solution to the demand of customized radio systems and has become a popular platform for prototyping and implementing wireless communication systems with special functionalities. An SDR system, in most cases, implement only basic digital signal processing operations such as filters on the FPGA while the rest of algorithms are usually running on the host CPU [4] [5]. When compared with a CPU, FPGA has the advantages of higher computing performance at lower power consumption for an application-specific task. Utilizing its rich resources of computational units, an FPGA can accelerate the implementation of complex algorithms even with strict real-time requirement [6].

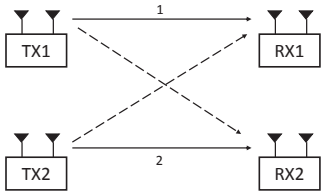

Fig. 1. A simple example of interference networks

In this paper, we present our dual link algorithm and the design of an FPGA prototype targeting SDR platforms. The main contributions of this paper are as follows: (a) The dual link algorithm has advantages of both PWF and WWMSE algorithms, which converges monotonically to a stationary point with very high convergence speed; (b) Compared with execution of the algorithm on CPU as most of the SDR implementations do, our FPGA based prototype requires less time on algorithm convergence, resulting in a speedup factor 2 or higher.

The rest of this paper is organized as follows. In Section II, our dual link algorithm is presented and compared with other rate-maximization algorithms. In Section III, we present the hardware architecture of a prototype which consists of a transmitter, a receiver and a covariance matrix calculation module. Two different structures are proposed: fixed-point structure (32 bits: 1 sign bit,15 integer bits and 16 fractional bits) with resource reuse and floating point structure (32 bits, single precision) without resource reuse. The results of FPGA implementations and their performance comparison between FPGA and CPU are analyzed in Section IV. Finally, conclusions are addressed in Section V.

II. ALGORITHM DESCRIPTION

A. The Dual Link Algorithm

We consider a general interference network with L interfering data links. Link l's physical transmitter is T_l, which has

L_{T_l} many antennas. Its physical receiver is R_l, which has L_{R_l} many antennas. The received signal at R_l is

$$\mathbf{y}_l \quad = \quad \sum_{k=1}^{L} \mathbf{H}_{l,k}\mathbf{x}_k + \mathbf{n}_l, \tag{1}$$

where $\mathbf{x}_k \in \mathbb{C}^{L_{T_k}\times 1}$ is the transmit signal of link k and is modeled as a circularly symmetric complex Gaussian vector; $\mathbf{H}_{l,k} \in \mathbb{C}^{L_{R_l}\times L_{T_k}}$ is the channel state information (CSI) matrix between T_k and R_l; and $\mathbf{n}_l \in \mathbb{C}^{L_{R_l}\times 1}$ is a circularly symmetric complex Gaussian noise vector with identity covariance matrix.

The optimization problem to be solved is the weighted sum-rate maximization under a total power constraint:

$$\begin{aligned} \textbf{WSRM\_TP}: \quad \max_{\mathbf{\Sigma}_{1:L}} \quad & \sum_{l=1}^{L} w_l \mathcal{I}_l\left(\mathbf{\Sigma}_{1:L}\right) \\ \text{s.t.} \quad & \mathbf{\Sigma}_l \succeq 0, \, \forall l, \\ & \sum_{l=1}^{L} \mathrm{Tr}\left(\mathbf{\Sigma}_l\right) \le P_{\mathrm{T}}, \end{aligned} \tag{2}$$

where $w_l > 0$ is the weight for link l, and we can set $w_l = 1$ such that each link are equally weighted. Assuming the channels are known at both the transmitters and receivers (CSITR), an achievable rate of link l is

$$\mathcal{I}_l\left(\mathbf{\Sigma}_{1:L}\right) = \log\left|\mathbf{I} + \mathbf{H}_{l,l}\mathbf{\Sigma}_l\mathbf{H}_{l,l}^{\dagger}\mathbf{\Omega}_l^{-1}\right| \tag{3}$$

where $\mathbf{\Sigma}_l$ is the covariance matrix of $\mathbf{x}_l$; and $\mathbf{\Omega}_l$ is the interference-plus-noise covariance matrix of the l^{th} link,

$$\mathbf{\Omega}_l = \mathbf{I} + \sum_{k=1,k\neq l}^{L} \mathbf{H}_{l,k}\mathbf{\Sigma}_k\mathbf{H}_{l,k}^{\dagger}. \tag{4}$$

The Dual Link algorithm for this optimization problem is given in Algorithm 1. It is an iterative algorithm with fast and monotone convergence [7]. Since the problem is non-convex, the algorithm converges to a local optimal point. Name the original channel forward link channel. The terms $\hat{\mathbf{\Sigma}}_l$ and $\hat{\mathbf{\Omega}}_l$ in the algorithm are corresponding terms in a reverse link channel, where the roles of the transmitters and receivers are exchanged and the channel $\mathbf{H}_{l,k}$ is replaced by $\mathbf{H}_{l,k}^{\dagger}$. The reverse links can be virtual links for the description of the algorithm. But in a TDD system, the reverse links exist physically, leading to a distributed implementation of the algorithm.

B. Distributed Algorithm and Local Channel Information Estimation

In a TDD system, the dual link algorithm can be readily implemented in a distributed and low complexity fashion. We take advantage of the physical reverse link for the distributed algorithm. For example, in Step 6 of Algorithm 1, to update the reverse link l's transmit signal covariance $\hat{\mathbf{\Sigma}}_l$, we only need to estimate local interference plus noise covariance $\mathbf{\Omega}_l$ and local total received signal covariance $\mathbf{\Omega}_l + \mathbf{H}_{l,l}\mathbf{\Sigma}_l\mathbf{H}_{l,l}^{\dagger}$ from the forward link received signal. This can be done with low

Algorithm 1 The Dual Link Algorithm

1. Initialize $\mathbf{\Sigma}_l$'s, s.t. $\sum_{l=1}^{L}\mathrm{Tr}\left(\mathbf{\Sigma}_l\right) = P_{\mathrm{T}}$
2. $R \Leftarrow \sum_{l=1}^{L} w_l\mathcal{I}_l\left(\mathbf{\Sigma}_{1:L}\right)$
3. Repeat
4. $R^{'} \Leftarrow R$
5. $\mathbf{\Omega}_l \Leftarrow \mathbf{I} + \sum_{k\neq l}\mathbf{H}_{l,k}\mathbf{\Sigma}_k\mathbf{H}_{l,k}^{\dagger}$
6. $\hat{\mathbf{\Sigma}}_l \Leftarrow \frac{P_{\mathrm{T}}w_l\left(\mathbf{\Omega}_l^{-1} - \left(\mathbf{\Omega}_l + \mathbf{H}_{l,l}\mathbf{\Sigma}_l\mathbf{H}_{l,l}^{\dagger}\right)^{-1}\right)}{\sum_{k=1}^{L} w_k\mathrm{tr}\left(\mathbf{\Omega}_k^{-1} - \left(\mathbf{\Omega}_k + \mathbf{H}_{k,k}\mathbf{\Sigma}_k\mathbf{H}_{k,k}^{\dagger}\right)^{-1}\right)}$
7. $\hat{\mathbf{\Omega}}_l \Leftarrow \mathbf{I} + \sum_{k\neq l}\mathbf{H}_{k,l}^{\dagger}\hat{\mathbf{\Sigma}}_k\mathbf{H}_{k,l}$
8. $\mathbf{\Sigma}_l \Leftarrow \frac{P_{\mathrm{T}}w_l\left(\hat{\mathbf{\Omega}}_l^{-1} - \left(\hat{\mathbf{\Omega}}_l + \mathbf{H}_{l,l}^{\dagger}\hat{\mathbf{\Sigma}}_l\mathbf{H}_{l,l}\right)^{-1}\right)}{\sum_{k=1}^{L} w_k\mathrm{tr}\left(\hat{\mathbf{\Omega}}_k^{-1} - \left(\hat{\mathbf{\Omega}}_k + \mathbf{H}_{k,k}^{\dagger}\hat{\mathbf{\Sigma}}_k\mathbf{H}_{k,k}\right)^{-1}\right)}$
9. $R \Leftarrow \sum_{l=1}^{L} w_l\mathcal{I}_l\left(\mathbf{\Sigma}_{1:L}\right)$
10. until $\left|R - R^{'}\right| \le \epsilon$ or a fixed number of iterations are reached.

complexity because the channel has summed the interference for us for free. There is no need to estimate $\mathbf{H}_{k,l}$ for all k and l. The reverse link calculation in Step 8 can be done similarly using the physical reverse link received signal.

The distributed algorithm and local channel information estimation for Step 6 is as follows. Assume forward link l uses precoding matrix $\mathbf{V}_l$ to transmit orthogonal pilot signal $\mathbf{P}_l \in \mathbb{C}^{L_{T_l}\times n}$ with n channel uses, where $\mathbf{V}_l\mathbf{V}_l^{\dagger} = \mathbf{\Sigma}_l$ and $\mathbf{P}_l\mathbf{P}_l^{\dagger} = n\mathbf{I}_{L_{T_l}\times L_{T_l}}$. For example, Hadamard matrices may be used for the pilots. In practice, pilots of different users can be near orthogonal. The received signal of forward link l is

$$\mathbf{Y}_l \quad = \quad \sum_{k=1}^{L}\mathbf{H}_{l,k}\mathbf{V}_k\mathbf{P}_k + \mathbf{N}_l \in \mathbb{C}^{L_{R_l}\times n}.$$

The least square based estimation of link l's own signal covariance is

$$\mathbf{H}_{l,l}\mathbf{\Sigma}_l\mathbf{H}_{l,l}^{\dagger} \doteq \mathbf{A}_l \quad = \quad \left(\frac{\mathbf{Y}_l\mathbf{P}_l^{\dagger}}{n}\right)\left(\frac{\mathbf{Y}_l\mathbf{P}_l^{\dagger}}{n}\right)^{\dagger}. \tag{5}$$

The estimated total received signal covariance of link l is

$$\mathbf{\Omega}_l + \mathbf{H}_{l,l}\mathbf{\Sigma}_l\mathbf{H}_{l,l}^{\dagger} \doteq \mathbf{B}_l \quad = \quad \frac{\mathbf{Y}_l\mathbf{Y}_l^{\dagger}}{n}. \tag{6}$$

Then, instead of using Step 5, which requires global channel knowledge, the interference plus noise covariance $\mathbf{\Omega}_l$ can be estimated as

$$\mathbf{\Omega}_l \quad \doteq \quad \mathbf{B}_l - \mathbf{A}_l. \tag{7}$$

Using (6) and (7), Step 6 can be calculated. Note that the normalization in Step 6 is to satisfy the total power constraint and can be implemented by adjusting in small steps and sharing one scalar constant

$$\mu = \frac{1}{P_{\mathrm{T}}}\sum_{l=1}^{L} w_l\mathrm{tr}\left(\mathbf{\Omega}_l^{-1} - \left(\mathbf{\Omega}_l + \mathbf{H}_{l,l}\mathbf{\Sigma}_l\mathbf{H}_{l,l}^{\dagger}\right)^{-1}\right), \tag{8}$$

similar to power control in CDMA networks. Step 8 of Algorithm 1 can be similarly calculated using reverse link

pilot and received signals. Once the covariance matrices $\mathbf{\Sigma}_l$ and $\hat{\mathbf{\Sigma}}_l$ are calculated, the precoding or beamforming matrices $\mathbf{V}_l$ and $\hat{\mathbf{V}}_l$ can be calculated using Cholesky decomposition using FPGA [8].

As seen from the above, the distributed algorithm is scalable and only needs local information, except for the sharing of the normalization constant μ.

C. Algorithm Simulation

To evaluate the performance, simulations of our rate maximization algorithm, PWF algorithm and WMMSE algorithm are conducted and compared in MATLAB. In the simulations, a cluster of 10 users each with 2-by-2 MIMO and 4000 bits pilot are generated and channels between every pair of users do not change during simulations. The relationship between sum rate of the network and the number of iterations is illustrated in Figure 2. It can be observed that the dual link sum rate maximization algorithm converges to its final result rapidly and smoothly. Meanwhile, the sum rate of the network approaches closely to its final value after about 10 iterations.

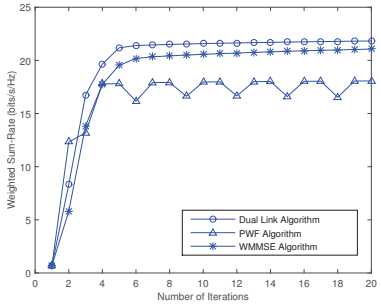

Fig. 2. Simulation results of rate maximization algorithm

III. SYSTEM PROTOTYPING USING FPGA

A typical MIMO system usually consists of two parts: receiver and transmitter [9]. Resembling basic radio structures, in our prototype design, we introduce an additional computational module called covariance matrix calculator that computes $\mathbf{\Omega}$ and $\mathbf{\Sigma}$ in every iteration. The top-level system diagram of an individual user is illustrated in Figure 3. Note that Y_I, Y_Q, X_I and X_Q represent the real and imaginary part of received signal and transmitted signal, respectively.

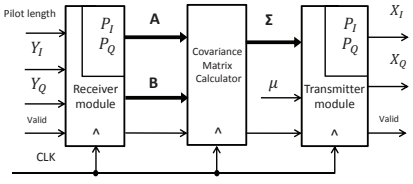

Fig. 3. Overall system diagram of an FPGA prototype

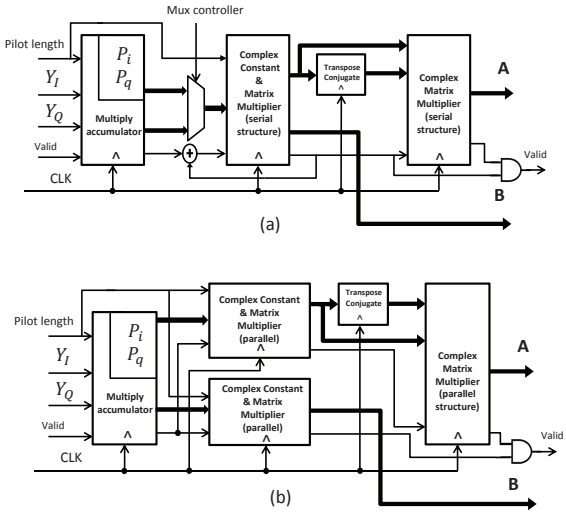

Fig. 4. (a): Fixed-point structure of the receiver module; (b) Floating-point structure of the receiver module

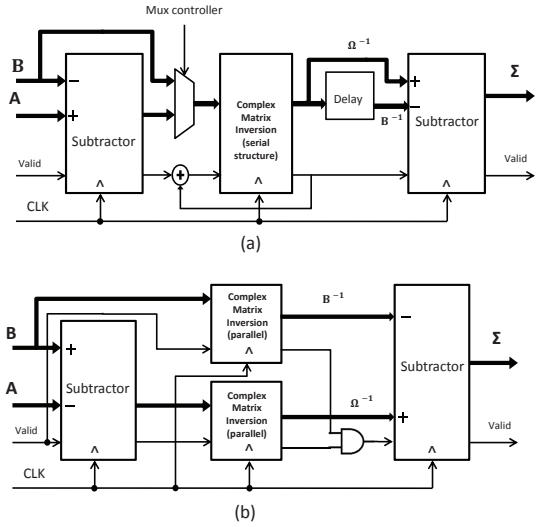

Fig. 5. (a): Covariance calculation in fixed-point structure; (b) Covariance calculation in floating-point structure

A. Receiver Module Design

Figure 4 is the diagram of receiver module of a single user. As the pilot $\mathbf{Y}$ is received, multiply accumulator computes the signal covariance and accumulator outputs are then divided by the length to produce the intermediate matrices $\mathbf{B}$ and $\mathbf{A}$ as in (5) and (6). Owing to the independent Gaussian noise and mutual orthogonality of pilots, it is not necessary to find the summation of interference and noise by adding them up. We can simply exploit received signal and foregone pilots to compute intermediate matrices and process them later in the covariance matrix calculator. Part (a) of Figure 4 depicts the receiver module of fixed point structure and part (b) depicts that of floating point structure.

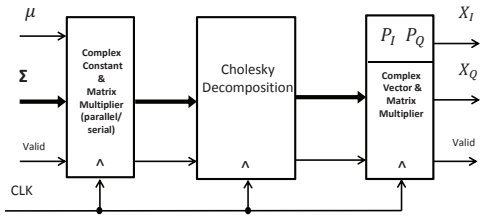

Fig. 6. System diagram of the transmitter module

B. Covariance Calculation Module Design

Figure 5 illustrates the hardware processing steps leading to signal covariance matrix $\mathbf{\Sigma}$. After computation of $\mathbf{A}$ and $\mathbf{B}$, we subtract $\mathbf{A}$ from $\mathbf{B}$ to find the interference-plus-noise matrix $\mathbf{\Omega}$ as in (7). Applying matrix inversion and another subtraction, the signal covariance matrix $\mathbf{\Sigma}$ can be obtained. In terms of the matrix inversion, adjoint matrix method [10] is employed which requires to calculate the reciprocal of the determinant of a complex matrix. Since the determinant of a complex matrix is a complex number under most conditions, we compute the reciprocal of the complex number during matrix inversion module. Assume we have a complex number $a_I + a_Q j$, then its reciprocal is $b_I + b_Q j$, where $b_I = \frac{a_I}{a_I^2 + a_Q^2}$ and $b_Q = \frac{-a_Q}{a_I^2 + a_Q^2}$.

C. Transmitter Module Design

The transmitter module shown in Figure 6 is simply a serial structure. With intermediate input $\mathbf{\Sigma}$ and the power adjustment coefficient μ (as in (7) and (13)) which is given by a central controller in the network, it is easily to obtain adjusted signal covariance matrix $\hat{\mathbf{\Sigma}}$. From the derivation in Section II, it is obvious that both $\mathbf{\Sigma}$ and $\hat{\mathbf{\Sigma}}$ are positive semi-definite matrix. Although strictly speaking Cholesky decomposition is only applicable to positive definite matrix, we can still apply Cholesky decomposition to $\mathbf{\Sigma}$ and obtain the decomposed lower triangular matrix $\mathbf{V}$ [11]. We then transmit the product of $\mathbf{V}$ and pilot $\mathbf{P}_l$.

IV. RESULTS AND ANALYSIS

To the authors' best of knowledge, there is currently no similar FPGA based prototype for our dual link algorithm. Therefore the performance comparison is made between two different structures proposed in Sections III. Our prototype designs, simulation and synthesis for FPGAs are targeted on Xilinx ZC706 which is one of the designate FPGA boards for AD-FMCOMMSx-EBZ series SDR platforms.

A. Synthesis and Resource Utilization

Table I illustrate the resource utilization of both fixed-point and floating-point designs on ZC706, respectively. It is obvious that the resource usage of floating-point structure is much higher than that of fixed-point structure, except for the block memory which stores pilot signals. On the other hand, the achievable clock rate of floating-point structure is 337.84 MHz while fixed point structure only achieves 181.82 MHz clock rate, only around half of 337.84 MHz. This is because the fixed-point structure with resource reuse needs strict timing control where the paths of controlling signal becomes the critical path, hence the maximum clock rate of fixed-point structures drops down. This controller design will be improved in our future work.

TABLE I
FPGA RESOURCE UTILIZATION OF AN INDIVIDUAL USER ON ZC706

Structure	Resource	Utilization	Available	Utilization %	Max Clock Rate
Fixed Point	LUTs	6654	218600	3.04	181.82 MHz
	Registers	9923	437200	2.27	
	DSP	160	900	17.78	
	Block RAM	4	545	0.73	
Floating Point	LUTs	30933	218600	14.15	337.84 MHz
	Registers	54466	437200	12.46	
	DSP	408	900	45.33	
	Block RAM	4	545	0.73	

B. Performance Evaluation

Simulations are performed to evaluate the processing latency and accuracy of two structures. We also conduct simulations of single user single iteration $1,000,000$ times on an Intel i5 quad-core CPU with 8GB memories as performance guideline. Table II shows a single iteration processing latency of a single user which includes all process time after receiving the signal and before transmitting the signal out.

TABLE II
COMPARISON BETWEEN FPGA AND CPU

Platform	FPGA		CPU
Device	ZC706	ZC706	i5 quad core
Structure	fixed point	floating point	floating point
Error range	$\approx 3 \times 10^{-3}$	$10^{-4} - 10^{-3}$	$\leq 10^{-4}$
Single user single iteration processing latency	$1579ns$	$880ns$	$3380ns$
Speedup factor	2.14	3.84	1

From Table II, when comparing with floating point structure, the fixed point structure has both advantages and disadvantages. The fixed point structure with many serial structured computational units utilizes less FPGA resources but it produces a larger range of errors and longer latency. Both proposed structures on an FPGA, however, can accommodate a shorter processing latency than that performed by CPU, with a speed up factor of 2.14 and 3.84, respectively.

V. CONCLUSIONS

In this paper, we propose an FPGA based prototype of the dual link rate maximization algorithm, targeting SDR platforms. By designing and implementing the prototype on an FPGA, the algorithm is executed more than 2 times than that on a CPU platform. We implement both fixed-point structure with serial units and floating-points structure with parallel units. The floating-point structure has a shorter processing time, higher accuracy while the fixed-point structure has much higher resource efficiency.

REFERENCES

[1] Q. Shi, M. Razaviyayn, Z.-Q. Luo, and C. He, "An iteratively weighted mmse approach to distributed sum-utility maximization for a mimo interfering broadcast channel," *IEEE Trans. Signal Processing*, vol. 59, no. 9, pp. 4331 –4340, Sep. 2011.

[2] A. Liu, Y. Liu, H. Xiang, and W. Luo, "Duality, polite water-filling, and optimization for MIMO B-MAC interference networks and iTree networks," *IEEE Trans. Info. Theory*, in revision, submitted Apr. 2010.

[3] ——, "Polite water-filling for weighted sum-rate maximization in B-MAC networks under multiple linear constraints," *IEEE Trans. Signal Processing*, vol. 60, no. 2, pp. 834 –847, Feb. 2012.

[4] H. Arslan, *Cognitive radio, software defined radio, and adaptive wireless systems*. Springer, 2007, vol. 10.

[5] M. Dillinger, K. Madani, and N. Alonistioti, *Software defined radio: Architectures, systems and functions*. John Wiley & Sons, 2005.

[6] S. Che, J. Li, J. W. Sheaffer, K. Skadron, and J. Lach, "Accelerating compute-intensive applications with GPUs and FPGAs," in *IEEE Symposium on Application Specific Processors*, 2008, pp. 101–107.

[7] X. Li, S. You, L. Chen, A. Liu, and Y. E. Liu, "A new algorithm for the weighted sum rate maximization in MIMO interference networks," in *Proc. IEEE Wireless Communications and Networking Conference (WCNC)*, Mar. 2015.

[8] J. Luo, Q. Huang, S. Chang, X. Song, and Y. Shang, "High throughput cholesky decomposition based on fpga," in *6th IEEE International Congress on Image and Signal Processing (CISP)*, vol. 3, 2013, pp. 1649–1653.

[9] P. Rapajic and B. Vucetic, "Linear adaptive transmitter-receiver structures for asynchronous CDMA systems," in *IEEE Third International Symposium on Spread Spectrum Techniques and Applications*, 1994, pp. 181–185.

[10] J. Eilert, D. Wu, and D. Liu, "Efficient complex matrix inversion for MIMO software defined radio," in *IEEE International Symposium on Circuits and Systems*, 2007, pp. 2610–2613.

[11] Z. Dostál, T. Kozubek, A. Markopoulos, and M. Menšík, "Cholesky decomposition of a positive semidefinite matrix with known kernel," *Applied Mathematics and Computation*, vol. 217, no. 13, pp. 6067–6077, 2011.