AN EFFICIENT FPGA BASED MIMO-MMSE DETECTOR

Hun Seok Kim[+], Weijun Zhu[#], Jatin Bhatia[#], Karim Mohammed[+], Anish Shah[+], and Babak Daneshrad[+]
[+]*Wireless Integrated Systems Research (WISR) Group, University of California, Los Angeles*
[#]*Silvus Communication Systems. Inc.*

ABSTRACT

This paper reports on a highly optimized 4x4 MMSE detector implementation. The work resulted in a real-time FPGA based implementation on a Xilinx Virtex-II 6000 part. It utilizes 8,513 logic slices, 64 multipliers, and 23 Block RAMs (less than 30% of the overall resources of this part). The design delivers over 420 Mbps sustained throughput, with a small 2.77 μs latency. Three main techniques are responsible for the improvements over other MIMO detectors reported in literature. They are: (a) the combination of a modified Gram-Schmidt QR decomposition algorithm with Square-Root linear MMSE detection; (b) a dynamic scaling algorithm that enhances numerical stability; and (c) an aggressive time-shared VLSI architecture. The above techniques are quite general and are readily applicable to any MIMO detector implementation.

1. INTRODUCTION

To date, thousands of papers have been published in the area of MIMO based information theory, algorithms, codes, MAC, etc [1]. By and large these works have been theoretical/simulation based and have focused on the algorithms and protocols that deliver superior BER for a given SNR. Little attention has been paid to the actual practical implementation of such algorithms in real-time systems that look to deliver 100's of Mbps sustained throughput to an end user or application. To better focus our effort and make our research results more relevant with main-stream MIMO systems, we decided to set the following specifications for our MIMO detector:

- **Throughput:** Ability to process a minimum of 14.4 *M* (*million*) 4x4 channel instances per second (equivalent to 345.6 *Mbps* when using 64 QAM)
- **Latency:** The entire detector latency should be below 4 μs. This is an important consideration in systems that require fast PHY turn around time in order to maintain overall system efficiency at the MAC (in CSMA/CA based MACs this latency directly translates into the speed with which ACK/NACK, CTS messages can be generated).
- **Hardware complexity:** The design should be such that it could easily fit onto a low end FPGA (i.e. Xilinx Virtex-II 3000) or occupy no more than 40% of the resources of a high end FPGA.

To put the above requirements into perspective, consider the needs of an 802.11n system [2][3]. The 4 μs of latency corresponds to 1/4 of the short inter-frame spacing (SIFS) time (16 μs for 802.11n [3]) which can be interpreted as the maximum latency allowed for the transmitter and receiver turnaround time. The throughput of 14.4*M channel instances per second* is required to complete the MIMO detection for 52 sub-carriers in a single OFDM symbol interval (3.6 μs with short guard interval).

Our studies have shown that a Square-Root MMSE detection algorithm which completely avoids matrix inversion is a feasible solution that meets our design requirements. At the conclusion of our work a real-time FPGA implementation of the MIMO detector was realized on a Xilinx Virtex-II FPGA, and was integrated into an end to end MIMO-OFDM testbed [2]. The resulting 4x4 MIMO detector uses 8,513 logic slices, 64 multipliers, and 23 Block RAMs. The design delivers over 420 Mbps sustained throughput, with a small 2.77 μs latency.

2. LINEAR MMSE MIMO DETECTION

We consider a linear MMSE MIMO detector as part of an entire 802.11n compliant 4x4 MIMO OFDM transceiver on a single FPGA [2]. We denote N_T and N_R as the number of transmit and receive antennas respectively. The MIMO detector is designed for the spatially multiplexed stream, where N_R is supposed to be greater than or equal to N_T. For each subcarrier, we denote the $N_R \times 1$ received vector $\mathbf{y}$ as the product of the $N_R \times N_T$ Rayleigh flat fading channel matrix $\mathbf{H}$ and the $N_T \times 1$ transmitted OFDM symbol vector $\mathbf{s}$ plus the $N_R \times 1$ additive white Gaussian noise vector $\mathbf{n}$ with covariance matrix $N_0 \cdot \mathbf{I}$. This relationship is shown in (1).

$$\mathbf{y} = \mathbf{H}\mathbf{s} + \mathbf{n} \tag{1}$$

Once the linear MMSE weight matrices $(\mathbf{H}^*\mathbf{H} + N_0 \cdot \mathbf{I})^{-1} \cdot \mathbf{H}^* = \mathbf{W}_{MMSE}$ for each OFDM sub-carrier is obtained, the MMSE solution $\hat{\mathbf{y}}$'s is computed by matrix-vector multiplication as shown in equation (2) [1], [14].

$$\hat{\mathbf{y}} = (\mathbf{H}^*\mathbf{H} + N_0 \cdot \mathbf{I})^{-1} \cdot \mathbf{H}^*\mathbf{y} = \mathbf{W}_{MMSE} \cdot \mathbf{y} \tag{2}$$

$(\cdot)^*$ is a conjugate-transpose operation.

All the MMSE detector implementations reported in the literature [8] – [11] use explicit matrix inversion of $(\mathbf{H}^*\mathbf{H} + N_0 \cdot \mathbf{I})^{-1}$. Of these, [8][10][11] use QR decomposition to solve the matrix inversion problem, while [9] uses the matrix inversion lemma to arrive at the solution. It is worth noting that equation (2) requires an inversion of a squared matrix ($\mathbf{H}^*\mathbf{H}$). In fixed point hardware implementation, a matrix inversion followed by matrix square operations is generally not desirable because of numerical stability issues [14].

3. MMSE MIMO DETECTION USING SQUARE-ROOT ALGORITHM

Fortunately, the inverse matrix operation as well as the matrix squaring operation can be avoided in the linear MMSE detection problem by using the equivalent Square-Root reformulation [6][13] steps (3) - (5). The Square-Root MMSE formulation exploits the structure of the $(N_R+N_T)\times N_T$ compound matrix $\mathbf{A}_{SQ}^{1/2}$ in order to eliminate the need for the matrix inversion operation and reduces the precision requirements of the system. The Square-Root MMSE formulation was first introduced in [6] where it was used in the implementation of V-BLAST type detectors [12][13]. Its application to the linear MMSE MIMO detector has hitherto been unexplored and is one of the contributions of the present work.

$$\mathbf{A}_{SQ}^{1/2} = \begin{bmatrix} \mathbf{H} \\ \sqrt{N_0}\cdot\mathbf{I} \end{bmatrix} = \mathbf{Q}_{SQ}\mathbf{R}_{SQ} = \begin{bmatrix} \mathbf{Q}_1 \\ \mathbf{Q}_2 \end{bmatrix}\mathbf{R}_{SQ} \tag{3}$$

$$\sqrt{N_0}\cdot\mathbf{I} = \mathbf{Q}_2\mathbf{R}_{SQ}, \quad \mathbf{R}_{SQ}^{-1} = \frac{\mathbf{Q}_2}{\sqrt{N_0}} \tag{4}$$

$$\hat{\mathbf{y}} = \frac{\mathbf{Q}_2}{\sqrt{N_0}}\mathbf{Q}_1^{*}\mathbf{y} = \mathbf{W}_{MMSE}\cdot\mathbf{y} \tag{5}$$

One interesting fact is that not only $\mathbf{R}_{SQ}$ but even $\mathbf{Q}_2$ in (3) is an upper triangular matrix. This helps us to save multipliers in the hardware implementation. Once the QR decomposition of $\mathbf{A}_{SQ}^{1/2}$ is complete, we can obtain the MMSE weight matrix $\mathbf{W}_{MMSE}$ through matrix-matrix multiplication (5), completely avoiding matrix inversion due to the useful property in (4).

3.1 Algorithms for QR Decomposition

The most computationally intensive part of the Square-Root MMSE algorithm is the QR decomposition on the $(N_R+N_T)\times N_T$ matrix $\mathbf{A}_{SQ}^{1/2}$. We examine two types of low complexity QR decomposition techniques for FPGA implementation – the first is the modified Gram-Schmidt QR decomposition and the second is the Givens rotation based QR decomposition [14].

It is well known that unitary transformation based Givens rotation method is more numerically stable than the Gram-Schmidt method [14]. However, when one considers an FPGA based implementation, multiplication-intensive methods such as the modified Gram-Schmidt QR decomposition are usually more desirable than a CORDIC-intensive Givens rotation QR algorithm since a number of dedicated multipliers are available in FPGAs without extra cost whereas a CORDIC operator consumes significant amount of FPGA slices. When $N_R = N_T = 4$, Givens rotation QR method requires 152 CORDIC operations to compute $\mathbf{W}_{MMSE}$. Even a fully pipelined, highly time-shared architecture where one hardware CORDIC unit is shared for 10 operations will cost 15 hardware CORDIC units, which correspond to more than 14,000 slices (with 16 bit precision assumption) in Xilinx Virtex-II FPGAs.

3.2 Algorithm Enhancement for Gram-Schmidt based QR

The main drawback of CORDIC operation-free Gram-Schmidt QR method is its relative numerical instability compared to Givens rotation method. However, we have found that the modified Gram-Schmidt based QR in Square-Root MMSE detection can be made more numerically stable by exploiting the fact that the $\mathbf{R}_{SQ}$ matrix from the QR decomposition does not contribute to the MMSE solutions. Essentially, we can apply any preprocessing to $\mathbf{A}_{SQ}^{1/2}$ as long as $\mathbf{Q}_{SQ}$ remains the same, even if it does not preserve $\mathbf{R}_{SQ}$. As we can see from equation (6), scaling of the i_{th} column $\mathbf{v}_i$ with an arbitrary constant c_i has the property of preserving the $\mathbf{Q}_{SQ}$ matrix but not necessarily $\mathbf{R}_{SQ}$.

$$\begin{aligned} &\mathbf{A}_{SQ}^{1/2} = [\mathbf{v}_1 \quad \ldots \quad \mathbf{v}_{N_T}] = \mathbf{Q}_{SQ}\cdot\mathbf{R}_{SQ} \\ &\Rightarrow \tilde{\mathbf{A}}_{SQ}^{1/2} = [c_1\mathbf{v}_1 \quad \ldots \quad c_{N_T}\mathbf{v}_{N_T}] = \mathbf{Q}_{SQ}\cdot\tilde{\mathbf{R}}_{SQ} \end{aligned} \tag{6}$$

The modified and scaled Gram-Schmidt QR decomposition for Square-Root MMSE is shown in Table 1. The scaling steps correspond to steps (d) - (g) in Table 1.

Table 1 - Modified Scaled Gram-Schmidt

(a) $\mathbf{A}_{SQ}^{1/2} = \begin{bmatrix} \mathbf{H} \\ \sqrt{N_0}\cdot\mathbf{I} \end{bmatrix} = \begin{bmatrix} \mathbf{v}_1 & \mathbf{v}_2 & \ldots & \mathbf{v}_{N_T} \end{bmatrix}$

(b) *for* $i = 1$ *to* N_T

(c) *for* $j = i$ *to* N_T

(d) while$\left(\max\left\{\left|\Re(v_{1,j})\right|,\left|\Im(v_{1,j})\right|,\ldots\left|\Im(v_{N_R+N_T,j})\right|\right\} < 2^L\right)$

(e) $\mathbf{v}_j := 2\mathbf{v}_j$

(f) while$\left(\max\left\{\left|\Re(v_{1,j})\right|,\left|\Im(v_{1,j})\right|,\ldots\left|\Im(v_{N_R+N_T,j})\right|\right\} > 2^U\right)$

(g) $\mathbf{v}_j := \mathbf{v}_j / 2$

(h) *end*

(i) $r_{ii} := \|\mathbf{v}_i\|, \quad \mathbf{u}_i := \dfrac{\mathbf{v}_i}{\|\mathbf{v}_i\|}$

(j) *for* $j = i+1$ *to* N_T

(k) $r_{ij} := \mathbf{u}_i^{*}\cdot\mathbf{v}_j$

(l) $\mathbf{v}_j := \mathbf{v}_j - r_{ij}\cdot\mathbf{u}_i$

(m) *end*

(n) *end*

where $\mathbf{v}_i = [v_{1,i} \quad \cdots \quad v_{N_R+N_T,i}]^T$, $\Re(\cdot)$ and $\Im(\cdot)$ stand for real and imaginary part of a complex number respectively, $\mathbf{Q}_{SQ} = \begin{bmatrix} \mathbf{u}_1 & \ldots & \mathbf{u}_{N_T} \end{bmatrix}$, L and U are predefined lower and upper bound.

In order to identify the impact of the scaling, fixed point simulations were run using the IEEE 802.11n channel model D [15]. Our simulation setup includes a complete IEEE802.11n reference system including all the transmitter and receiver elements. The 802.11n

convolutional encoder along with a soft-decision Viterbi decoder were also included in the simulation. The number of antennas at the transmitter or receiver side was set to 4 with 1000 bytes of payload data. 64QAM constellation with a coding rate of 2/3 was selected for the simulation. This particular configuration will require higher SNR than other modulation and coding schemes and as such will be much more sensitive to quantization noise and stability issues that plague finite precision systems.

The required bit precisions for the QR decomposition are presented in Table 2. These precisions will ensure that the performance loss of the finite precision system is within 0.5 dB of the floating point system for a packet error rate (PER) of 1%. The impact of scaling on the modified Gram-Schmidt QR based Square-Root MIMO detection algorithm is shown in Figure 1. The proposed scaling technique delivers a savings of approximately 5-bit for the Gram-Schmidt QR algorithm. Note that a similar scaling technique can be applied to Givens rotation based QR decomposition in Square-Root MMSE detection. However, the impact of the scaling on a Givens rotation QR based Square-Root algorithm is not as significant (Table 2) due to the already well behaved numerical properties of that algorithm. The significance of the scaling technique on Gram-Schmidt QR decomposition comes from the fact that each column orthogonalization makes the magnitude of the remaining columns become smaller as the orthogonalization step continues. As Table 2 shows, the proposed scaling technique in Square-Root MMSE detection enhances the numerical stability of the modified Gram-Schmidt QR decomposition to a level that is comparable to the unitary transform based QR.

Table 2 - Bit Precision Requirement

	Square Root MMSE Detection			
	Gram-Schmidt QR		Givens Rotation QR	
	No scaling	With scaling	No scaling	With scaling
$\mathbf{Q}_{SQ}$	19	14	16	14
$\mathbf{R}_{SQ}$	21	16	18	16

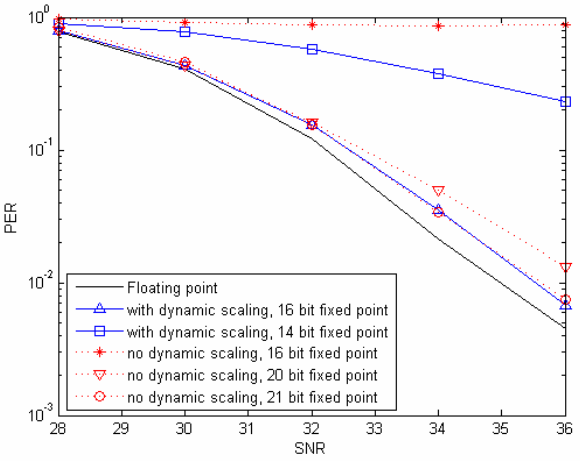


Figure 1 - Impact of Scaling on Gram-Schmidt QR

4. HARDWARE IMPLEMENTATION ON FPGAS

We select the modified Gram-Schmidt QR decomposition combined with Square-Root MMSE MIMO detection as the algorithm for our hardware implementation.

4.1 Multiplier Sharing Architecture

In this section, we introduce the multiplier sharing architecture as a technique to reduce the number of multipliers in the fully pipelined, Gram-Schmidt QR based Square-Root MMSE detector.

In the 802.11n testcase, a new 4x4 channel estimation matrix $\mathbf{H}$ is available to the detector at a maximum rate of ϕ = 14.4 *M instances per sec*. A fully pipelined detector must provide outputs of $\mathbf{W}_{MMSE}$ at a rate of ϕ = 14.4MHz without the need for a FIFO and unnecessary latency. Generally, the input/output rate ϕ is much lower than the maximum operating clock frequency of the hardware. Here, we define the *multiplier time sharing order* (γ) as in equation (7).

$$\textit{multiplier time sharing order } (\gamma) = \frac{\textit{Operating Clock Freq. in MHz}}{\phi} \quad (7)$$

In our FGPA, the multiplier time sharing order is 8 implying that the minimum operating clock frequency is 115.2 ($=8\times\phi$) *MHz* and a single multiplier processes 8 sets of inputs within a 14.4 *MHz* cycle. This specific number of multiplier time sharing order comes from the size (number of rows) of the compound matrix $\mathbf{A}_{SQ}^{1/2}$ for the Square-Root MIMO detector.

As a result, if the multiplier can run at a clock frequency of 115.2 *MHz*, the same multiplier can be shared within a single operation step ($\|\mathbf{v}\|^2$, $\mathbf{u}^* \cdot \mathbf{v}$, or $r_{ii} \cdot \mathbf{u}$) producing the output at the rate of 14.4 *M instances per sec*. With this multiplier sharing architecture, squared Euclidean norm ($\|\mathbf{v}\|^2$) and $\mathbf{v}$ vector update ($\mathbf{v} := \mathbf{v} - (\mathbf{u}^* \cdot \mathbf{v}) \cdot \mathbf{u}$) requires only 2 and 6 real multiplier instances respectively. A similar multiplier time sharing architecture with γ = 8 is applied to the scalar-matrix and the matrix-matrix multiplications in the remaining MIMO detection steps. Figure 2 shows the overall architecture of the fully pipelined 4x4 MIMO detector with the proposed multiplier sharing architecture. The square root and division operator in Figure 2 are instances of elements in the Xilinx Coregen library [16].

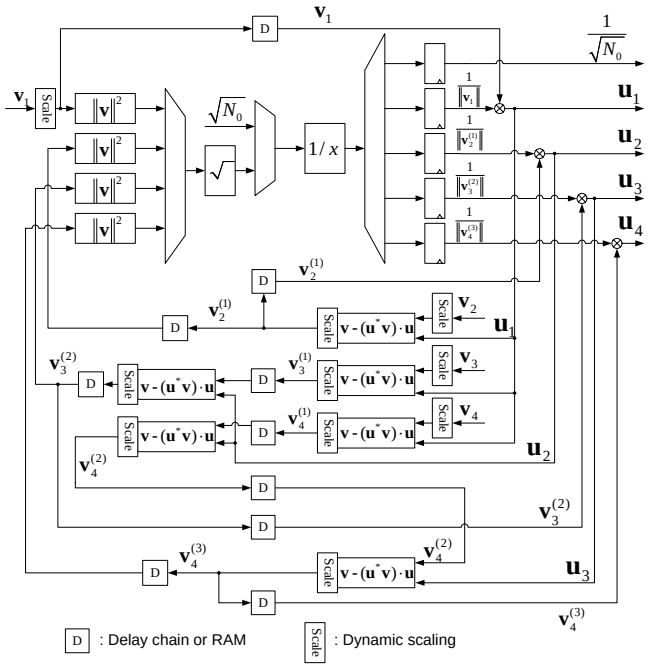
4.2 Multiplier Saving Techniques

The modified Gram-Schmidt QR decomposition circuit in Figure 2 does not make use of the sparsity pattern of the $\mathbf{A}_{SQ}^{1/2}$ matrix. Since the lower half of $\mathbf{A}_{SQ}^{1/2}$ is sparse ($\mathbf{Q}_2$ is upper triangular), the multipliers in the $\|\mathbf{v}\|^2$ and $\mathbf{v} - (\mathbf{u}^* \cdot \mathbf{v}) \cdot \mathbf{u}$ computation are not active all the time. This can be exploited to save some multipliers when the orthogonalization is performed on the columns of $\mathbf{A}_{SQ}^{1/2}$.

In Figure 3, real multipliers in the $\|\mathbf{v}_1\|^2$ computation are *active* (shaded rectangles) during only 5 out of 8 clock cycles, and complex multipliers in the $\mathbf{u}_1^* \cdot \mathbf{v}_j$ computation have 4 *inactive* slots (unshaded rectangles) out of 8. Meanwhile, only 5 complex multiplications are required to compute $(\mathbf{u}_1^* \cdot \mathbf{v}_j) \cdot \mathbf{u}_1$ and, the 5th

component of $\mathbf{u}_1$ is a real number. Therefore, $(\mathbf{u}_1^* \cdot \mathbf{v}_j) \cdot u_{1,1} \sim (\mathbf{u}_1^* \cdot \mathbf{v}_j) \cdot u_{1,4}$ can be computed using the *inactive* cycles of the multipliers for $\mathbf{u}_1^* \cdot \mathbf{v}_j$ computation, while $(\mathbf{u}_1^* \cdot \mathbf{v}_j) \cdot u_{1,5}$ can use the *inactive* slots in $\|\mathbf{v}_1\|^2$ computation. This technique allows us to save 9 real multipliers in Figure 2 at no significant cost.

In addition, some multipliers in the scalar-matrix or matrix-matrix computation can be saved by exploiting the fact that diagonal elements of $\mathbf{Q}_2$ are real numbers. Although there are 10 non-zero components in $\mathbf{Q}_2$, 4 of them are real and the remaining 6 are complex. Hence, when we compute $\frac{1}{\sqrt{N_0}} \cdot \mathbf{Q}_2$ and $\frac{\mathbf{Q}_2}{\sqrt{N_0}} \cdot \mathbf{Q}_1^*$, 2 and 13 real multipliers should be sufficient respectively.



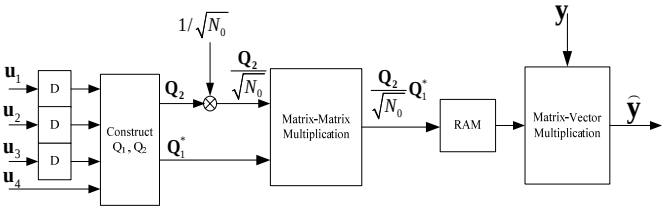


Figure 2 – Square-Root MMSE Detector Circuit

5. FPGA IMPLEMENTATION RESULTS

The MMSE MIMO detector design ($N_T = N_R = 4$) was successfully synthesized then placed and routed on commercial FPGAs such as Xilinx's Virtex-II and 4 series. These chips have a number of dedicated hardware multipliers and two-port block RAMs. Table 3 shows the implementation result of the IEEE 802.11n compatible MIMO detector including resource utilization from the place and route report.

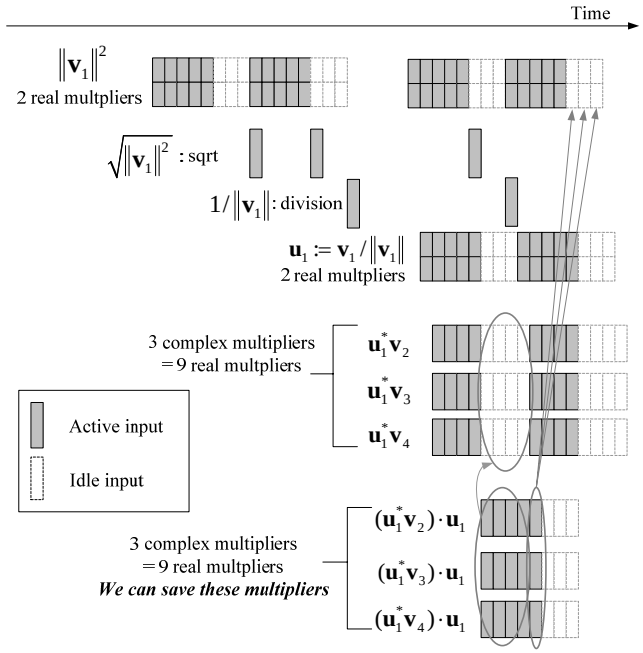


Figure 3 - Scheduling for Multipliers

The latency of the implemented detector corresponds to 2.77*us* when f_{clk} is 140MHz, which is shorter than a single OFDM symbol in IEEE 802.11n draft proposal **[3]**. The computation throughput for $\mathbf{W}_{MMSE}$ and $\hat{\mathbf{y}}$ are the same for f_{clk} / 8 *instances per second.* Either 140MHz (for Virtex-II) or 170MHz (for Virtex-4) operating clock frequency can provide sufficient throughput to meet our specification of 14.4*M instances per sec*. The $\hat{\mathbf{y}}$ throughput of f_{clk} / 8 *vectors per second* can support up to 420 *Mbps* when the number of spatial stream is 4, f_{clk} = 140*MHz*, and 64QAM is used for each OFDM sub-carrier. The FPGA slice resource usage is less than 25% of the total number of slices available on a Virtex-II 6000 FPGA and the design utilized about 43% of the total dedicated multipliers available in the same FPGA.

Table 3 - Place and Rout Report

Target FPGA	Slices	Number of Multipliers	BRAMs
Virtex-II xc2v6000 (speed grade -6)	8513 out of 33792	64	23
Virtex-4 xc4vlx160 (speed grade -12)	7500 out of 67854		
Target FPGA	**Max Clock Freq. (f_{clk})**	**Latency (Clocks)**	**Throughput $\mathbf{W}_{MMSE}$, $\hat{\mathbf{y}}$**
Virtex-II xc2v6000 (speed grade -6)	140 MHz	388	f_{clk} / 8 instances per second
Virtex-4 xc4vlx160 (speed grade -12)	170 MHz		

Table 4 and Table 5 show the resource usage, throughput and latency comparison result between this work and other references. In fact, none of [8] – [11] in Table 4 and Table 5 is designed to produce complete MMSE solution $\hat{\mathbf{y}}$ and some of them compute

only $(\mathbf{H}^*\mathbf{H} + N_0 \cdot \mathbf{I})^{-1}$ rather than $\mathbf{W}_{MMSE}$ as the output. For comparison purpose, we explicitly specified the resource usage of our design for computing $1/\sqrt{N_0} \cdot \mathbf{Q}_2$, which correspond to that of $(\mathbf{H}^*\mathbf{H} + N_0 \cdot \mathbf{I})^{-1}$ in matrix inversion based MMSE detection (see equations (2) and (5)). Table 4 shows that our design utilized fewer slices than [8] and [11] while the number of multipliers used in our design is only 50% of those in [9]. Meanwhile, the throughput of our design is at least 3 times higher than [9], [10] and [11] as shown in Table 5. The throughput of [8] is not given but its latency is too high to meet our design requirement.

Table 4 - Resource Comparison

	Computation	Slices	Multiplier	BRAM
This Work	$1/\sqrt{N_0} \cdot \mathbf{Q}_2$ which corresponds to complexity of $(\mathbf{H}^*\mathbf{H} + N_0 \cdot \mathbf{I})^{-1}$	6487 (Virtex-II)	45	15
	$\mathbf{W}_{MMSE}$	7679 (Virtex-II)	58	19
[8]	$\mathbf{W}_{MMSE}$	16865 (Virtex-II)	44	101
[9]	$\alpha(\mathbf{H}^*\mathbf{H} + N_0 \cdot \mathbf{I})^{-1}$ α scaling will require additional multipliers and dividers	4446 (Virtex-II)	101	N/A
[10]	$(\mathbf{H}^*\mathbf{H} + N_0 \cdot \mathbf{I})^{-1}$	86% of Virtex-II [1]	N/A	N/A
[11]	$(\mathbf{H}^*\mathbf{H} + N_0 \cdot \mathbf{I})^{-1}$	9117 (Virtex-4)	22	9

Table 5 - Throughput and Latency Comparison

	Output	Throughput *(instances per second)*	f_{clk} MHz	Latency clks
This Work	$\mathbf{W}_{MMSE}$, $\hat{\mathbf{y}}$	17.50 *M* (Virtex-II)	140	388
		21.25M (Virtex-4)	170	
[8]	$\mathbf{W}_{MMSE}$	N/A	N/A	3000
[9]	$\alpha(\mathbf{H}^*\mathbf{H} + N_0 \cdot \mathbf{I})^{-1}$	[2] 6.75 *M* (Virtex-II)	108	64
[10]	$(\mathbf{H}^*\mathbf{H} + N_0 \cdot \mathbf{I})^{-1}$	[2] 6.25 *M* (Virtex-II)	100	350
[11]	$(\mathbf{H}^*\mathbf{H} + N_0 \cdot \mathbf{I})^{-1}$	[3] 0.13 *M* (Virtex-4)	115	933

6. CONCLUSION

In this paper, we studied hardware friendly algorithms that avoid matrix inversion for linear MMSE MIMO detection. We suggested a dynamic scaling technique for the modified Gram-Schmidt QR decomposition increasing the numerical stability of the fixed point design. We have demonstrated the successful FPGA implementation with multiplier sharing architecture and multiplier saving techniques. The designed 4x4 linear MMSE MIMO detector is capable of complying with the proposed IEEE 802.11n standard.

[1] The exact slice count is not available since its FPGA part name is not given.

[2] The throughput is not specified in the reference. However, it can be computed from its architecture.

[3] [11] is a floating point design.

REFERENCES

[1] Arogyaswami J. Paulraj, Dhananjay Gore, Rohit U. Nabar, and Helmut Bolcskei, "An Overview of MIMO Communications – A Key to Gigabit Wireless", *Proceeding of the IEEE,* Vol, 92, No.2, Feb 2004.

[2] Weijun Zhu, Babak Daneshrad, Jatin Bhatia, Jesse Chen, Hun-Seok Kim, Karim Mohammed, Omar Nasr, Sandeep Sasi, Anish Shah, Minko Tsai, "A Real Time MIMO OFDM Testbed for Cognitive Radio & Networking Research", *in preparation*

[3] IEEE TGn Working Group, "Joint Proposal: High throughput extension to the 802.11 Standard"

[4] C.M. Rader, "MUSE-a systolic array for adaptive nulling with 64 degrees of freedom, using Givens transformations and wafer scale integration", *Application Specific Array Processors, 1992, Proceedings of the International Conference on*, Aug. 1992

[5] Kalavai Raghunath, Keshab Parhi, "A 100MHz Pipelined RLS Adaptive Filter", *Acoustics, Speech, and Signal Processing, ICASSP, International Conference on, May 1995*

[6] Babak Hassibi, "An Efficient Square-Root Algorithm for BLAST", *Acoustics, Speech, and Signal Processing, 2000. ICASSP '00. Proceedings. 2000 IEEE International Conference on*

[7] Iain B. Collings, Michael R.G. Butler and Matthew R. McKay, "Low Complexity Receiver Design for MIMO Bit-Interleaved Coded Modulation", *ISSSTA2004*, Sep, 2004.

[8] Markus Myllyla, Juha-Matti Hintikka, Joseph R. Cavallaro and Markku Juntti**,** "Complexity Analysis of MMSE Detector Architectures for MIMO OFDM Systems"**,** *Signals, Systems and Computers, 2005. Conference Record of the Thirty-Ninth Asilomar Conference on*

[9] Isabelle LaRoche, Sebatien Roy, **"**An Efficient Regular Matrix Inversion Circuit Architecture for MIMO Processing**",** *Circuits and Systems, 2006. ISCAS 2006. Proceedings IEEE International Symposium on*

[10] Fredrik Edman, Viktor Öwall, "A Scalable Pipelined Complex Valued Matrix Inversion Architecture", *Circuits and Systems, 2005. ISCAS 2005. IEEE International Symposium on*

[11] Marjan Karkooti, Joseph R. Cavallaro, "FPGA Implementation of Matrix Inversion Using QRD-RLS Algorithm", *Signals, Systems and Computers, 2005. Conference Record of the Thirty-Ninth Asilomar Conference on*

[12] Zhan Guo, Peter Nilsson, "A Low-Complexity VLSI Architecture for Square Root MIMO Detection" *Personal, Indoor and Mobile Radio Communications, 2003. PIMRC 2003. 14th IEEE Proceedings on*

[13] Ronald Bohnke, Dirk Wubben, Volker Kuhn, and Karl-Dirk Kammeyer, "Reduced Complexity MMSE Detection for BLAST Architectures", *GLOBECOM 2003*

[14] Gene H. Golub, Charles F. Van Loan, "Matrix Computations", *The Johns Hopkins University Press, 3rd Edition*

[15] IEEE TGn Working Group, "TGn Channel Models"

[16] http://www.xilinx.com/ipcenter/index.htm