# Rapid Prototyping of the Improved Cholesky Decomposition based MIMO Detector

Xuezheng Chu
*The Institute of Electronics, Communications and Information Technology, Queens University Belfast,*
xchu01@qub.ac.uk

Dr Khaled Benkrid
*School of Electronics and Engineering, The University of Edinburgh,UK*
K.Benkrid@ed.ac.uk

Dr John Thompson
*School of Electronics and Engineering, The University of Edinburgh,UK*
John.Thompson@ed.ac.uk

## Abstract

*Multiple Input Multiple Output (MIMO) has gradually become the most promising technique for the next generation wireless telecommunications systems. MMSE-VBLAST has a performance close to Maximum Likelihood with considerably low complexity. The improvements in the algorithm results in substantial computation and hence hardware savings as it avoids the hardware cost expensive square root and division operations. This improvement decreases the computational complexity of MMSE-VBLAST with no performance penalty compared to previous MMSE-VBLAST algorithmic. This has finally be validated for 2x2 and 4x4 MIMO systems using a rapid prototyping methodology that starts with full software formulation in MATLAB and ends with an optimized equivalent FPGA hardware implementation.*

## 1. Introduction

In the last ten years, the use of MIMO technology in wireless links has been extensively studied, mostly from the theoretical point of view, showing that significant capacity increases could be achieved under certain conditions by using multiple antennas at both transmitter and receiver. Vertical Bell Laboratories Layered Space Time coding (V-BLAST) [1] is a MIMO communication architecture proposed by Bell laboratories. For the uncoded MIMO case in the V-BLAST receiver, the Minimum Mean Square Error (MMSE) algorithm is widely considered as an efficient approach to obtain near-to -ML performance with reduced complexity.

Nowadays, the prototyping of those multiple-antenna systems has become increasingly important to verify the enhancements advanced by analytical results. However, in most cases, the target platform is rarely used as feedback to investigate ways of improving the algorithm. The main aim of the rapid prototyping methodology is to be able to verify the improvements from the algorithmic point of view using real-time prototype.

The paper starts from the generic MIMO system and presents the rapid prototyping methodology, adopted in the work, and shows how it can be applied to wireless MIMO system development. It then focuses on Cholesky decomposition and Triangular inversion algorithms, provides an improved solution and compares it with alternative techniques. Then, the paper analyzes two specific MIMO systems, namely 2x2 and 4x4 MMSE-VBLAST using the proposed solution, with simulation results provided. The work finally analyzes fixed point simulation of our solutions with 3-sigma automatic gain control (AGC). The prototyping of improved MMSE-VBLAST solution is presented at last. This validates the claimed efficiency of the improved solution.

## 2. MIMO System Model

An *MxN* MIMO system model is depicted in Figure 1.

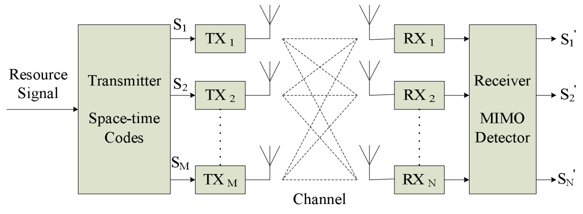

Figure 1. MIMO channel model

*M*, *N* represents the number of transmitter and receiver antennas, and $s_1, s_2 \ldots s_M, s_1', s_2' \ldots s_N'$, are the sub signal flows of transmitter and receiver respectively. The model can be formalized as:

$$\boldsymbol{r} = \boldsymbol{H}\boldsymbol{s} + \boldsymbol{n} \qquad (1)$$

where $\boldsymbol{r} = [r_1, r_2, \ldots, r_N]^T$ denotes the received signal *N*-vector, $\boldsymbol{H}$ is *MxN* signal matrix, $\boldsymbol{s} = [s_1, s_2, \ldots, s_M]^T$ denotes the transmitted signal *M*-vector, and $\boldsymbol{n} = [n_1, n_2, \ldots, n_N]^T$ represents *N*-vector of independent and identically distributed (i.i.d) complex additive white Gaussian noise (AWGN) samples with variance $N_0$ [2].

## 3. Rapid Prototyping Approach

The rapid prototyping methodology used in this work is actually a rapid functional validation method. It uses a MATLAB simulink environment (the Xilinx System Generator in this case) to prototype the system architecture and generates the corresponding relevant HDL files. Then, ModelSim and/or any vendor-specific simulation tool, e.g. Xilinx ISE in this case, are used to validate the functionality of the hardware produced. The latter should replicate the simulation results produced by the MATLAB-Simulink environment, and serves only as a checking/verification step. This rapid prototyping method saves considerable time especially for large-scale complex systems, and is thereby much more effective.

Using this rapid prototyping methodology allows for a high-level design to be quickly translated from algorithm design into system architecture. The researcher can initially realize a research idea or standard in the form of an algorithm written in software. The latter can then be used as a gold reference. From an implementation prospect, the use of this rapid prototyping methodology has the advantage of identifying the complexity issue and related costs in early development times. From the design prospect, this methodology has the advantage of quickly identifying bottlenecks, trade-offs between different design parameters, and ultimately identifying the necessary trade-offs for optimized solutions.

Several prototyping systems have been developed in both academic research and professional development for MIMO systems. For example, a single-carrier MIMO system is described in [3], [4] and [5], and a MIMO-OFDM system is implemented in [6]. All of them focus on system integration and realize wireless channel testing. However, these are not suitable for rapid validation and of novel MIMO detection algorithms. The rapid prototyping method used in this work mainly concentrates on the signal processing side of MIMO systems and allow for the rapid validation of complex MIMO detection algorithms to find out the system bottlenecks in early in the development process.

The working process is described as follows. Firstly, the MATLAB language is used to simulate the complete MIMO system including transmitter, different MIMO algorithms and receiver, in double precision. The system's fixed point simulation is then deduced to decide on the precision needed for hardware implementation (as double precision and floating point arithmetic in general is generally prohibitively expensive in hardware). Automatic gain control should be used at this stage in order to normalize input data. Complicated mathematical analysis should be substituted by basic mathematical operations (i.e. add/sub, multiply, shift etc.), and exponent and logarithm operations should be replaced by their respective adequate hardware approximation functions e.g. Taylor and Maclaurin series. Performance results for different fixed-point precisions are then compared with the gold reference double precision simulation results. The fixed point precision that is closest to the double precision performance (according to a developer criterion) is then chosen. Trying to intensive fixed point simulation to each mathematical unit of the system, and finding the optimal fixed point precision to satisfy the overall precision requirement can save large amounts of hardware and leads to quicker implementations. The algorithm is then prototyped in Xilinx System Generator (a MATLAB-Simulink plug-in) and then compiled and synthesized to FPGA hardware. No hardware description language e.g. Verilog HDL/VHDL, is required to capture the system under study in System Generator as this is a graphical user environment with building blocks (hard or soft) linked in a data flow. Hardware optimized for a particular FPGA family (the target family) is generated automatically from such descriptions by System Generator, in the form of VHDL or Verilog. The latter is then synthesized, mapped and routed using Xilinx's ISE tools. These tools generate a number of reports which help us analyze the resource usage and timing performance of the resulting hardware configurations. Bottlenecks can arise at this stage of the development process.

This methodology has been used in this work in order to rapidly prototype MIMO detection algorithms in hardware. The methodology is distinguished by its speed, early functional validation, and fast route to hardware implementation.

## 4. Improved Cholesky Decomposition

### 4.1. Improved algorithm

With Cholesky Decomposition and Triangular Matrix Inversion [7], the *i*-th iteration of MMSE-VBLAST becomes as follows:

$$\begin{aligned}
&\boldsymbol{G}_i = \left(\boldsymbol{H}_i^H \boldsymbol{H}_i + \sigma^2 \boldsymbol{I}\right) \\
&\boldsymbol{L}_i = \text{cholesky}\left(\boldsymbol{G}_i\right) \\
&\boldsymbol{q}_i = \text{triangurlarInv}(\boldsymbol{L}_i) \\
&\boldsymbol{Q}_i = \boldsymbol{q}_i \boldsymbol{q}_i^H \\
&\boldsymbol{k}_j = \arg\min_j \left\|\boldsymbol{Q}_i\right\|^2 \\
&\boldsymbol{w}_i = \boldsymbol{Q}_j \boldsymbol{H}_i^H \\
&\hat{\boldsymbol{s}}_i = \frac{\boldsymbol{w}_i \boldsymbol{r}}{\boldsymbol{w}_i \boldsymbol{H}_j}
\end{aligned} \tag{2}$$

where $\boldsymbol{G}_i$ is the $\boldsymbol{G}$ matrix at iteration *i*, $\boldsymbol{L}_i$ is the Cholesky Decomposition of $\boldsymbol{G}_i$ and its triangular inversion is denoted by $\boldsymbol{q}_i$. Multiplying $\boldsymbol{q}_i$ and the relevenet complex conjugate transpose of $\boldsymbol{q}_i^H$, the pseudo inversion of $\boldsymbol{G}_i$ can then be transformed to $\boldsymbol{Q}_i$, Next, the nulling vector $\boldsymbol{w}_i$ calculated as the *j*-th row of $\boldsymbol{Q}_i$, and the *j*-th element with the minimum norm in $\boldsymbol{Q}_i$ is the strongest channel to be estimated in the current iteration. The estimated transmit symbol can be computed by $\hat{s}_i$ to match the closest multi-dimensional constellation point. After nulling the signal of the $j$-th channel from the received signal and cancelling the $j$-th column from the channel matrix $\boldsymbol{H}$, the computing steps into the ($i$+1)-th iteration.

Taking notice of the estimated symbol $\hat{s}_i$, the nulling vector $\boldsymbol{w}_i$ appears in both numerator and denominator, so any scale of $\boldsymbol{w}_i$ will not take effect of the estimated symbol. This is key to the algorithm improvement proposed here. Indeed, as the $\boldsymbol{Q}_i$ coming from the multiplication of the inverse of two triangular matrices is actually the Cholesky decomposition of a positive definite Hermitian matrix $\boldsymbol{G}_i$, several square root and division operations can be optimized away, which reduces the necessary hardware implementation resources drastically.

The following briefly describe the computing of the 4th iteration of a 4x4 matrix case, in order to demonstrate the improvement claimed here.

$\boldsymbol{G}$ matrix is assumed as,

$$\boldsymbol{H}^H\boldsymbol{H}+\sigma^2\boldsymbol{I}=\begin{pmatrix} A_{11} & A_{12} & A_{13} & A_{14} \\ A_{21} & A_{22} & A_{23} & A_{24} \\ A_{31} & A_{32} & A_{33} & A_{34} \\ A_{41} & A_{42} & A_{43} & A_{44} \end{pmatrix}$$

The Cholesky decomposition of $\boldsymbol{G}$ is then:

$$\boldsymbol{L}=\text{cholesky}(\boldsymbol{G})=\begin{pmatrix} \frac{A_{11}}{\sqrt{A_{11}}} & 0 & 0 & 0 \\ \frac{A_{21}}{\sqrt{A_{11}}} & \frac{B_{22}}{\sqrt{B_{22}}} & 0 & 0 \\ \frac{A_{31}}{\sqrt{A_{11}}} & \frac{B_{32}}{\sqrt{B_{22}}} & \frac{C_{33}}{\sqrt{C_{33}}} & 0 \\ \frac{A_{41}}{\sqrt{A_{11}}} & \frac{B_{42}}{\sqrt{B_{22}}} & \frac{C_{43}}{\sqrt{C_{33}}} & \frac{D_{44}}{\sqrt{D_{44}}} \end{pmatrix} \tag{3}$$

which has the same factor in each column and can be factorized as:

$$\boldsymbol{L}=\begin{pmatrix} A_{11} & 0 & 0 & 0 \\ A_{21} & B_{22} & 0 & 0 \\ A_{31} & B_{32} & C_{33} & 0 \\ A_{41} & B_{42} & C_{43} & D_{44} \end{pmatrix}\times\begin{pmatrix} \frac{1}{\sqrt{A_{11}}} & 0 & 0 & 0 \\ 0 & \frac{1}{\sqrt{B_{22}}} & 0 & 0 \\ 0 & 0 & \frac{1}{\sqrt{C_{33}}} & 0 \\ 0 & 0 & 0 & \frac{1}{\sqrt{D_{44}}} \end{pmatrix} \tag{4}$$

where $A$ is the relevant element of $\boldsymbol{G}$ matrix, while $B$, $C$, $D$ are new elements computed by Cholesky decomposition. The complex conjugate transpose of $\boldsymbol{L}$ then can be factorized as:

$$\boldsymbol{L}^H=\begin{pmatrix} \frac{1}{\sqrt{A_{11}}} & 0 & 0 & 0 \\ 0 & \frac{1}{\sqrt{B_{22}}} & 0 & 0 \\ 0 & 0 & \frac{1}{\sqrt{C_{33}}} & 0 \\ 0 & 0 & 0 & \frac{1}{\sqrt{D_{44}}} \end{pmatrix}\times\begin{pmatrix} A_{11} & A_{12} & A_{13} & A_{14} \\ 0 & B_{22} & B_{23} & B_{24} \\ 0 & 0 & C_{33} & C_{34} \\ 0 & 0 & 0 & D_{44} \end{pmatrix} \tag{5}$$

Hence, we can easily deduce that:

$$\begin{aligned}\boldsymbol{Q}&=\boldsymbol{L}^{-1}\left(\boldsymbol{L}^H\right)^{-1}\\ &=\begin{pmatrix} A_{11} & A_{12} & A_{13} & A_{14} \\ 0 & B_{22} & B_{23} & B_{24} \\ 0 & 0 & C_{33} & C_{34} \\ 0 & 0 & 0 & D_{44} \end{pmatrix}^{-1}\times\begin{pmatrix} A_{11} & 0 & 0 & 0 \\ 0 & B_{22} & 0 & 0 \\ 0 & 0 & C_{33} & 0 \\ 0 & 0 & 0 & D_{44} \end{pmatrix}\\ &\times\begin{pmatrix} A_{11} & 0 & 0 & 0 \\ A_{21} & B_{22} & 0 & 0 \\ A_{31} & B_{32} & C_{33} & 0 \\ A_{41} & B_{42} & C_{43} & D_{44} \end{pmatrix}^{-1}\end{aligned} \tag{6}$$

As a result, the original squared root operations have been eliminated. Based on the Cholesky Decomposition algorithm, the relevant elements of the triangular matrix can be given by the following expressions

$$B_{ij}=\frac{1}{A_{11}}\bar{B}_{ij},\, C_{ij}=\frac{1}{A_{11}^2B_{22}}\bar{C}_{ij},\, D_{ij}=\frac{1}{A_{11}^4B_{22}^2C_{33}}\bar{D}_{ij} \tag{7}$$

So the $\boldsymbol{Q}$ can be factorized as,

$$\begin{aligned}\boldsymbol{Q}&=\begin{pmatrix} A_{11} & A_{21}' & A_{31}' & A_{41}' \\ 0 & \bar{B}_{22} & \bar{B}_{32}' & \bar{B}_{42}' \\ 0 & 0 & \bar{C}_{33} & \bar{C}_{43}' \\ 0 & 0 & 0 & \bar{D}_{44} \end{pmatrix}^{-1}\times\begin{pmatrix} A_{11} & 0 & 0 & 0 \\ 0 & A_{11}\bar{B}_{22} & 0 & 0 \\ 0 & 0 & A_{11}\bar{B}_{22}\bar{C}_{33} & 0 \\ 0 & 0 & 0 & A_{11}\bar{B}_{22}\bar{C}_{33}\bar{D}_{44} \end{pmatrix}\\ &\times\begin{pmatrix} A_{11} & 0 & 0 & 0 \\ A_{21} & \bar{B}_{22} & 0 & 0 \\ A_{31} & \bar{B}_{32} & \bar{C}_{33} & 0 \\ A_{41} & \bar{B}_{42} & \bar{C}_{43} & \bar{D}_{44} \end{pmatrix}^{-1}\end{aligned} \tag{8}$$

where $(.)'$ denotes the transpose value of corresponding element, and $\overline{(\cdot)}$ denotes the denominator of the relevant expression. At this stage, the Cholesky decomposition has been reformulated to become the product of the inverse of two triangular matrixes (each is the transpose the other) and one diagonal matrix, with no need for the square root and division operations. The only remaining computation is the inversion of the triangular matrix.

Based on the triangular matrix inversion algorithm mentioned in [7], the inverse of the two triangular transpose matrices are also transpose, and the inversion of the lower and upper triangular matrix is also a lower and upper triangular matrix, respectively. Therefore, on the assumption of the lower triangular matrix in (8), the inversion of triangular matrix can be basically computed by back substitution as the following form,

$$\boldsymbol{q} = \boldsymbol{p}^{-1} = \begin{pmatrix} X_{11} & 0 & 0 & 0 \\ X_{21} & X_{22} & 0 & 0 \\ X_{31} & X_{32} & X_{33} & 0 \\ X_{41} & X_{42} & X_{43} & X_{44} \end{pmatrix} \quad (9)$$

From the formula deduction, each element of q multiplies the common factor of $A_{11}\bar{B}_{22}\bar{C}_{33}\bar{D}_{44}$, saving the division operations. As a result, the pseudo matrix inversion of ***G*** reduces to a multiplication of three matrices without any expensive square root and division operations. It should be noted at this stage that the technique presented above for the case of a 4x4 system is applicable to any *NxN* system. From (7) each element of the Cholesky decompostion is dealt with one by one, which can be pipelined in hardware in order to increase throughput. In the back substitution triangular inversion, however, there is no correlation between the computations of each element and what only need is the elements factorized by Cholesky decomposition, which means that we can employ instruction parallelism to speed up the computation.

Table 1. Operation complexity of the two algorithms

| Algorithms | Cholesky Decomposition | Improved Cholesky Decomposition |
|---|---|---|
| **Addition** | $n^3 - \frac{1}{2}n^2 - \frac{1}{2}n$ | $n^3 - \frac{1}{2}n^2 - \frac{1}{2}n$ |
| **Multiplication** | $\frac{2}{3}n^3 + n^2 - \frac{2}{3}n$ | $\frac{2}{3}n^3 + \frac{3}{2}n^2 - \frac{1}{6}n - 2$ |
| **Division** | $2n-1$ | 0 |
| **Square Root** | $n$ | 0 |

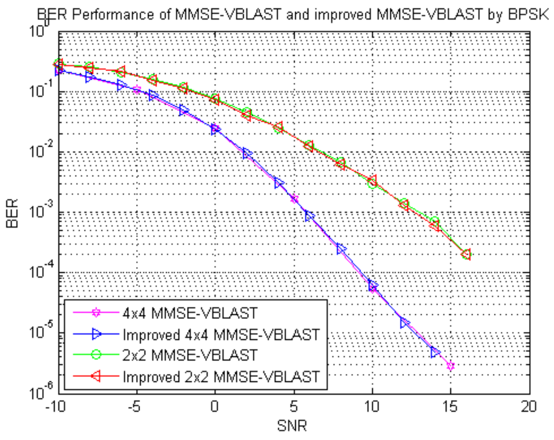

Figure 2. BER Performance for our improved MMSE-VBLAST VS original MMSE-VBLAST formulation, for 2x2 and 4x4 MIMO systems with BPSK modulation

Table 1 illustrates the addition, multiplication, division, square root operation for each of the improved Cholesky Decomposition and the original one, supposing nxn squared target matrix here. The operation complexity is O(n3) for both of them. The improved one use 1/2n2 additional multiplications though, it totally frees 2n-1 and n division and square root operations respectively. Considering nearly 30 times more hardware cost of the division and square root operations than that of addition and multiplication, the improvement would benefit itself on hardware cost. Figure 2 demonstrates the double precision simulation of the MMSE-VBLAST with the original formulation versus double precision improved formulation for the case of a 2x2 and 4x4 MIMO systems using BPSK modulation. It clearly reveals that the original 2x2 and 4x4 MMSE-VBLAST systems and equivalent improved MMSE-VBLAST system have the same performance.

## 4.2. 3-Sigma Automatic Gain Control

This section presents the automatic gain control (AGC) method involved in the project called the 3-sigma method [8]. Here, sigma (σ) represents the statistical term for the Standard Deviation of a distribution. It is known as the "68-95-99.7 rule" or the "empirical rule" [9] that 99.7% of the input data should fall in the range (-3σ, 3σ) for normal distribution. As the input data of MMSE-VBLAST detector follows a normal distribution with zero mean, and assuming a system precision has been chosen, $E[r]=0$ and $E[\boldsymbol{s}^2]=1$, $\sigma^2$ can be derived as follows:

$$\sigma_r^2 = E\left[\boldsymbol{r}^2\right] = h^2 E\left[\boldsymbol{s}^2\right] + E\left[\boldsymbol{n}^2\right] = h^2 + \sigma_n^2$$
$$E\left[\mathrm{Re}\{\boldsymbol{r}^2\}\right] = \left[\mathrm{Re}\{h\}\right]^2 + \frac{\sigma_n^2}{2}$$
$$E\left[\mathrm{Im}\{\boldsymbol{r}^2\}\right] = \left[\mathrm{Im}\{h\}\right]^2 + \frac{\sigma_n^2}{2} \quad (10)$$

Taking into account of the "68-95-99.7 rule", the main point of this method is to scale the input signal to cover the range(-3σ, 3σ), which is called dynamic range [10] in hardware. Without considering the integer and fractional part of the input signal, if the hardware precision is *precision*, the largest range can be expressed by this precision is $(-2^{precision-1}, 2^{precision-1})$, then the system desired sigma value would be $\sigma_D=(2^{precision}-1)/6$. The channel model of the system has 0 mean and 1 variance, so the variance of the real and imagery part of the channel matrix is 1/2, hence, the scaled factor based on σ can be obtained from the desired σ of ***r*** divided the actual σ, the factor for real and imagery part is the same which is:

$$\lambda_{3\sigma} = \frac{2^{precision} - 1}{6\sqrt{1/2 + \sigma_n^2/2}} \quad (11)$$

Multiplying this factor, ***r*** and ***H*** can be scaled into the system dynamic range. The method scales the input signal to suit the system bandwidth from a statistical point of view. Meanwhile, on the aim of scaling input data to the range is $(-2^{precision\text{-}1}, 2^{precision\text{-}1})$, the computing process could be further optimized by avoiding the consideration of which part is integer or fractional.

### 4.3. Fixed Point Arithmetic

Figure 3 shows fixed point simulation results for 2x2 MMSE-VBLAST MIMO system with BPSK symbol modulation using the 3-sigma AGC method.

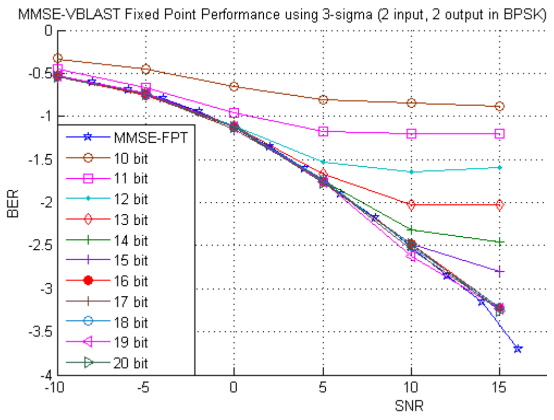

Figure 3. 2x2 MMSE-VBLAST Fixed Point Performance with the 3-sigma AGC method

Here, the system is quantized from 10 bit to 20 bit, and 16 bit precision is shown to satisfy the desired system performance. Although the 16 bit precision is incapable of fully matching the performance of the double precision floating point implementation under a SNR equal to 15dB, the BER performance is approximately equal to $10^{-3.3}$ which is considered acceptable.

using the 3-sigma method, Figure 4 shows the simulation result of a 4x4 MMSE-VBLAST fixed point implementation with SNRs ranging from -10dB to 15 dB with BPSK symbol modulation.

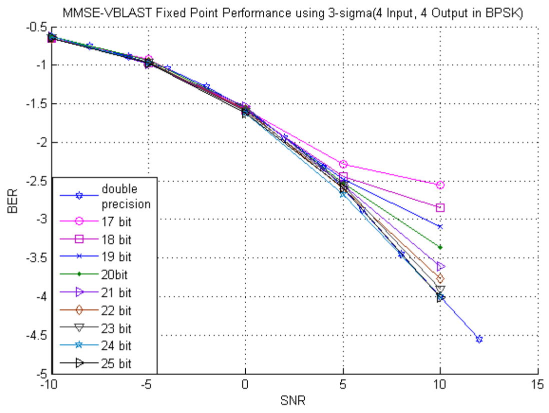

Figure 4. MMSE-VBLAST Fixed Point Performance using 3-sigma method in BPSK, the system contains 4 transmit antennas and 4 receive antennas

Here again, although the 23-bit precision cannot perfectly match the double precision floating point implementation when the SNR is higher than 15 dB, the BER performance of 23-bit is lower than $10^{-3}$ and can thus be acceptable. It is finally worth mentioning that with respect to instances where input data follow a Gaussian distribution and the performance of the 3-sigma method is not deemed satisfactory, 4-sigma or even 6-sigma could be used instead.

## 5. System Simulation and Implementation

The double precision floating point and fixed point simulations for the improved 2x2 and 4x4 MMSE-VBLAST systems have been given above. Further discusses of the hardware architecture prototyping using Xilinx System Generator tool shows Xilinx FPGA hardware design at the Simulink level.

### 5.1. 2x2 System Architecture

The previous analysis of the double precision floating point and fixed point simulation for the improved 2x2 MMSE-VBLAST shows that 16-bit precision performs very closely to double precision floating point. Partitioning the 2x2 MMSE-VBLAST program between MATLAB and simulated FPGA parts is the first step towards implementing the system in Xilinx System Generator. Figure 5 pictures a simple and intuitionistic approach to partitioning handling.

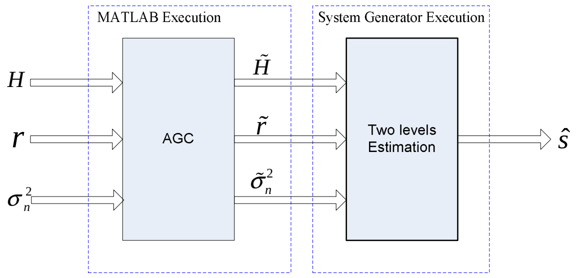

Figure 5. Partitioning of the 2x2 MMSE-VBLAST between MATLAB and the System Generator

Functionally, MATLAB performs the AGC block once per frame and sends the scaled input data to Gatewayin port (see Figure 6). The System Generator executes two iterations of the VBLAST algorithms in a pipelined fashion, both of which include ***G*** matrix computation, nulling vector computation and decision. The signal is then sent out from the Gatewayout port for output. Figure 6 shows the blocks diagram of the complete System Generator implementation of 2x2 MMSE-VBLAST detector.

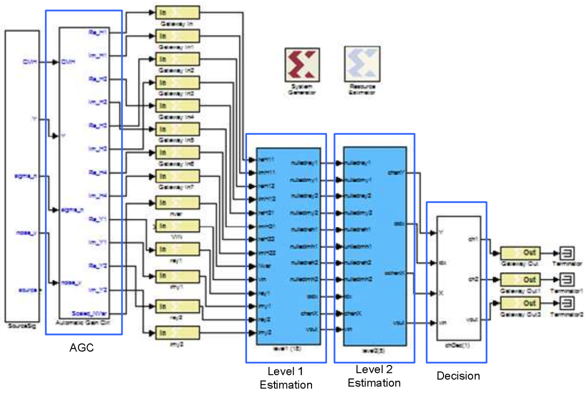

Figure 6. System Generator block diagram of 2x2 MMSE-VBLAST

## 5.2. 4x4 System Architecture by Rapid Prototyping

Figure 7 shows the block diagram of the System Generator implementation of a 4x4 MMSE-VBLAST MIMO detector. The input signal ***r*** and ***H*** are scaled by 3-sigma AGC and quantized as 24 bit signed integers. AGC block is performed once per frame in Simulink, while Xilinx FPGA blocks executes 4 iterations of the estimation, all of which contain ***G*** matrix computation, Cholesky decomposition, triangular inversion, nulling vector computation, minimum search and decision. Though the Cholesky decomposition and triangular inversion cannot be avoided in 4x4 system, an improved solution has been proposed previously. Functions of different blocks of the design are described below.

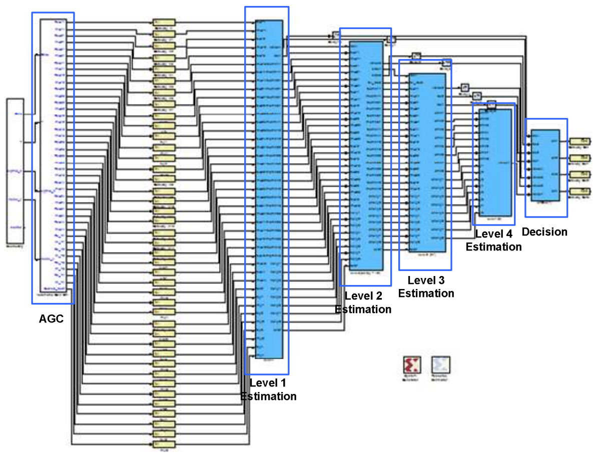

Figure 7. System Generator block diagram of 4x4 MMSE-VBLAST

*Level 1 to 4 Estimation*: These blocks perform the signal estimation for each iteration, where the channel number checked is 4, 3, 2 and 1 respectively.

*Cholesky decomposition*: Based on (7), the elements of Cholesky decomposition can be obtained sequentially. It employs the inherent parallelism present in the algorithm, and its deterministic structure makes the pipelining of the algorithm feasible. Figure 8 shows the System Generator block diagram of the Cholesky decomposition.

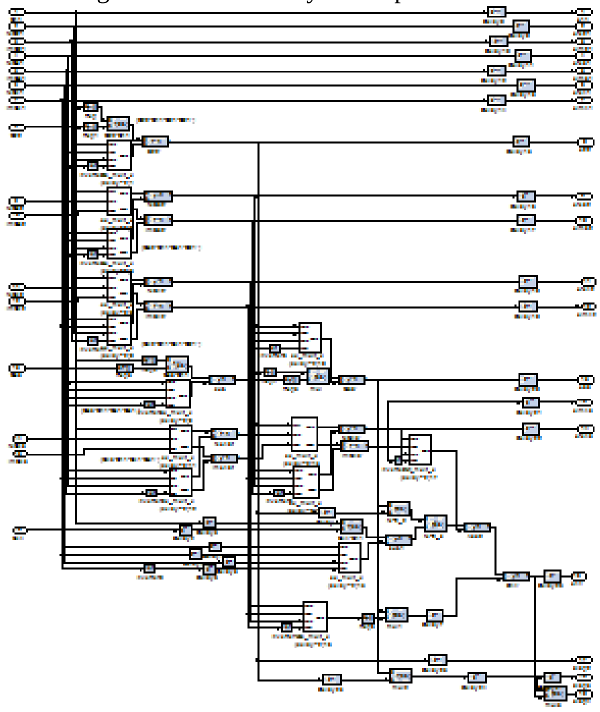

Figure 8. System Generator Implementation of Cholesky Decomposition

Figure 9 presents the time diagram of the Cholesky decomposition algorithm on System Generator. It shows how different parts of pipeline stages of the algorithm start processing valid data in sequence. For the sake of simplicity, information about latency of each stage is not depicted.

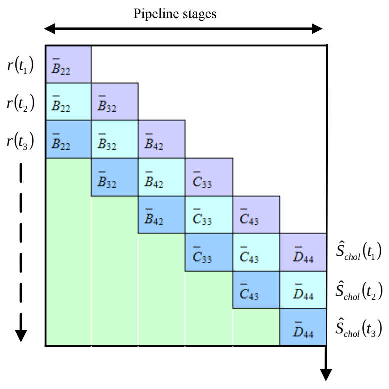

Figure 9. Pipeline stages in the case of 4x4 MMSE-VBLAST MIMO detector

The detection process is specified for three time instants to show how symbols are being outputted from the Cholesky decomposition block at a constant rate. The white area indicates the different pipeline stage parts waiting for valid data to fill in. The green area denotes all the filled pipeline stages and symbols in parallel processing for different time instants. In this way, once the pipeline stages are filled, all blocks of the Cholesky decomposition are active every clock cycle and perform at full speed.

*Triangular Inversion*: Figure 10 pictures the System Generator Implementation block diagram of the triangular inversion based on back substitution. Here all elements are concurrently computed. Minimum search is constructed by 4 Muxes to compare the minimum norms of all required channels and indicate the index of the channel number.

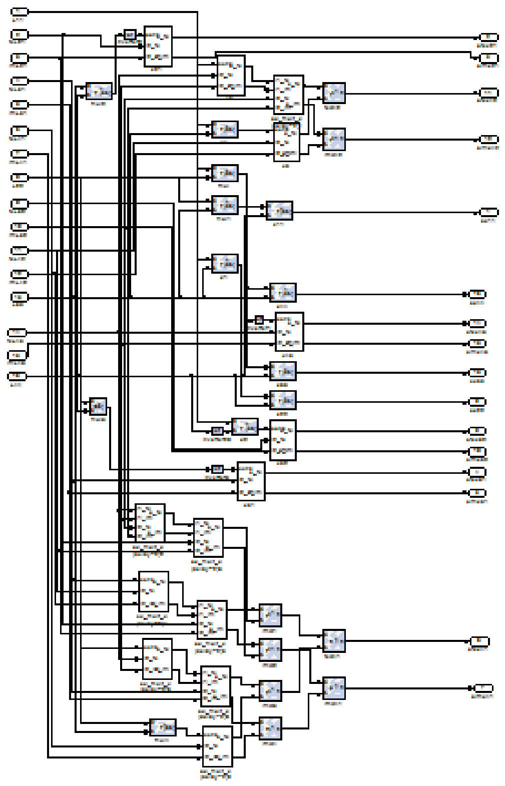

Figure 10. System Generator Implementation of Triangular Inversion

## 5.3. Hardware Implementation Results

The MMSE-VBLAST has been prototyped for 2x2 and 4x4 system with BPSK modulation. The System Generator design has been compiled into FPGA hardware in the form of Verilog. The latter was in turn synthesized into FPGA configurations using Xilinx ISE tool. Functional simulation is performed at the System Generator level, and the Verilog level using ModelSim. The Xilinx ISE synthesis report gives the FPGA resource usage (area) and speed information, which is reported in Table 2 for 2x2 and 4x4 MMSE-VBLAST MIMO detectors using BPSK modulation, on a Xilinx Virtex4 VC4VSX55 FPGA chip. Note that the resource usage is also estimated by the Resource Estimator blockset of the Xilinx System Generator tool, and the results are the same as those reported by Xilinx ISE tool.

In the 2x2 system, a total of 56 adders, 77 registers and 3 comparators are used. The minimum clock period is 5.890ns and the maximum frequency is 169.776MHz. In the 4x4 system, however, the minimum period is 11.539ns and the maximum frequency is 86.657 MHz.

Comparing the 2x2 and the 4x4 systems, it can find out that the 4x4 uses over 10 times the amount of slices, 6 times the amount of flip-flops, 4 times the amount of

DPS48s and 13 times the LUTs resources of the2x2 system. This is due to the fact that the 4x4 system has to compute Cholesky decomposition and triangular inversion explicitly, which is costly in terms of resources. The channel bandwidth of currently designed WCDMA and CDMA 2000 systems are up to 20MHz [11]. For a 4x4 MIMO detector, the designed clock frequency is 80MHz, which means 4 clock cycles are used to process a set of data. The maximum frequency of 4x4 system showed in Table 2 indicates that the design can meet the system demands, and so is the 2x2 case.

Table 2. Estimated FPGA Resource Use of 2x2 and 4x4 MMSE-VBLAST

| VC4VSX55 | 2x2 | % | 4x4 | % | Total |
|---|---|---|---|---|---|
| Number of Slices | 1,792 | 7% | 21,836 | 88% | 24,576 |
| Number of Slice Flip Flops | 3,183 | 6% | 18,373 | 37% | 49,152 |
| Number of 4 input LUTs | 2,713 | 5% | 33,512 | 68% | 49,152 |
| Number of bonded IOBs | 273 | 42% | 593 | 92% | 640 |
| Number of DSP48s | 104 | 20% | 426 | 83% | 512 |
| Number of GCLKs | 1 | 3% | 1 | 3% | 32 |

## 6. Conclusion and future work

An FPGA implementation of the improved MMSE-VBLAST using rapid prototyping methodology has been presented in this paper. The advantages of the rapid prototyping methodology are the flexibility that provides to analyze in detail the hardware implementation of the algorithm while designing. For this purpose, the MMSE-VBLAST algorithm has been chosen as a candidate to optimize on hardware. This resulted in an improved implementation of the algorithm which reduces its complexity. FPGA hardware implementations have then been achieved albeit in simulation.

The future work would be real hardware implementation of the 2x2 and 4x4 MMSE-VBLAST algorithm on an FPGA board. This implementation should perform further improvements on the hardware to exploit the hardware resources fully and maximise the throughput.

## Acknowledgement

I express my grateful thanks to Khaled Benkrid and John Thompson for their valuable guidance. My thanks also extend to my current supervisors John McAllister and Roger Woods for their great advices on the part of complexity analysis.

## 7. References

[1] P. W. Wolniansky, G. J. Foschini, G. D. Golden *et al.*, "V-BLAST: an architecture for realizing very high data rates over the rich-scattering wireless channel." pp. 295-300.

[2] E. Biglieri, *MIMO wireless communications*, Cambridge: Cambridge University Press, 2007.

[3] H. Thomas, F. Andreas, G. Holger *et al.*, “Real-time signal processing for multiantenna systems: algorithms, optimization, and implementation on an experimental test-bed,” *EURASIP J. Appl. Signal Process.*, vol. 2006, no. 1, pp. 136-136.

[4] R. M. Rao, Z. Wiejun, S. Lang *et al.*, “Multi-antenna testbeds for research and education in wireless communications,” *Communications Magazine, IEEE*, vol. 42, no. 12, pp. 72-81, 2004.

[5] P. Murphy, F. Lou, A. Sabharwal *et al.*, "An FPGA based rapid prototyping platform for MIMO systems." pp. 900-904 Vol.1.

[6] C. Mehlfuhrer, M. Rupp, F. Kaltenberger *et al.*, "A scalable rapid prototyping system for real-time MIMO OFDM transmissions." p. 7 pp.

[7] G. H. Golub, and C. F. Van Loan, *Matrix computations*, 3rd ed., Baltimore: Johns Hopkins University Press, 1996.

[8] Mini-circuits. "http://staff.ee.sun.ac.za/deswardt/RF/General/Application%20Note%20Amplifier.htm."

[9] Wikipedia. "Normal distribution, http://en.wikipedia.org/wiki/Gaussian_distribution."

[10] B. Mulgrew, P. M. Grant, and J. Thompson, *Digital signal processing : concepts and applications*, 2nd ed., Houndmills, Basingstoke, Hampshire [Eng.] ; New York, N.Y.: Palgrave Macmillan, 2003.

[11] N. Parameshwar, and R. Rajagopalan, "A comparative study of cdma2000 and W-CDMA." pp. 15.1-15.9.