

An FPGA based Co-Design Architecture for MIMO Lattice Decoders

Cao Liang, Jing Ma and Xinming Huang
Department of Electrical Engineering,
University of New Orleans, New Orleans, LA 70148
Email: {cliang1, jma, xhuang}@uno.edu

Abstract — *MIMO systems have attracted great attentions because of their huge capacity. The hardware implementation of MIMO decoder becomes a challenging task as the complexity of the MIMO systems increases. This paper presents hardware/software co-design architecture targeted on a single FPGA for two typical lattice decoding algorithms. Two levels of parallelisms are analyzed for an efficient implementation with the preprocessing part on embedded MicroBlaze soft processor and the decoder part on customized hardware. The system prototypes of the AV and VB decoders show that they support up to 34.2 Mbps and 3.15 Mbps data rate at 20dB SNR respectively on XUP Virtex-II pro developing board with an xc2vp30 FPGA, which are 19 and 16 times faster than their respective implementations on a DSP. The performance in terms of resource utilization and bit error rate are also compared between these two algorithms.*

Keywords — *Hardware/Software co-design, MIMO, Lattice decoder, decoding algorithms, FPGA, DSP*

1. Introduction

Multiple-input Multiple-output (MIMO) systems use multiple antennas in both transmitters and receivers to achieve huge capacity. Lattice theory and coding theory are applied in MIMO systems to improve the performance such as data rate and bit error rate (BER).

There are two typical lattice decoding algorithms. One is the Pohst strategy based algorithm developed by Viterbo and Boutros (VB) [1], and the other is the Schnorr-Euchner strategy based algorithm applied by Agrell, Eriksson, Vardy, and Zeger (AV) [2]. Due to the complexity of the lattice decoding algorithms and the high data dependency among the decoding procedures, the MIMO decoders are generally implemented on DSPs, such as Bell Labs layered space-time (BLAST) system [3][4].

Because of the flexible reconfiguration and support of parallelism, FPGAs are widely used in the signal processing field. In this paper, the

parallelism of the decoding procedures is explored and an efficient FPGA architecture is applied to improve the decoding rate. In addition, to reduce the complexity of the decoding part, the lattice generation matrix is often decomposed into a triangular matrix by certain matrix reduction algorithms [5][6], which is called the preprocessing of a lattice decoder. Because of the complexity of the calculations involved, the preprocessing is not suitable for hardware implementation.

Hardware/software co-design technique is a promising solution to combine the channel matrix preprocessing and the core decoding functions for both AV and VB lattice decoding algorithms on one FPGA. In this paper, the complex matrix operations and the lattice decoding procedures are partitioned into a MicroBlaze soft processor and customized hardware modules attached to the processor. Two levels of parallelisms are explored to develop efficient hardware/software co-design architecture: the parallel execution of one preprocessing core and multiple decoding cores, and the parallel execution of multiple states during the closest lattice point search.

The system is prototyped on the Xilinx University Program (XUP) Virtex-II pro developing board [7] which includes an xc2vp30 FPGA. The bit-accurate C-models of these two algorithms are simulated in software to verify the hardware implementation. The performances of data rate, BER, and FPGA resource utilization are compared between these two algorithms as well.

This paper is organized as follows. Section 2 reviews the principles of the AV and VB algorithms. The data dependency among the iterative lattice decoding procedures are also analyzed in this section. Two levels of parallel hardware/software co-design architectures for the decoding systems are explored in Section 3. The experimental performances are compared between these two algorithms in Section 4 followed by the conclusion in Section 5.

2. Decoding Algorithms

The maximum likelihood (ML) lattice decoding algorithm tries to find the closest point to the received vector in the lattice structure. AV and VB algorithms are both ML lattice decoding algorithms, which try to enumerate the lattice points inside a sphere, and find the best lattice point with minimum distance to the received point. The main difference between the two algorithms is the investigating order inside the lattice structure. The VB algorithm searches from the lower bound to the upper bound in each search layer and examines all possible lattice points falling into a certain sphere in the lattice structure with an initial radius \sqrt{C} . Meanwhile the AV algorithm spreads out from a nearby lattice point of the received point and terminates once the total distance is greater than the best distance and the search procedure reaches the bottom layer. No initial radius is needed in the AV algorithm, and it does not need to upgrade the upper and lower bounds of each layer using the time consuming square root functions as it needs in the VB algorithm. It is claimed that the AV algorithm is 2 to 8 times [2] more efficient than the VB algorithm even if a proper initial radius \sqrt{C} is applied to the VB algorithm. The *step-by-step* implementations for both decoders are presented in this section. Further details of the principles of these two algorithms can be referred to [1][2].

Considering a MIMO system with $N \times M$ channel matrix H , the received signal \bar{r} is given by:

$$\bar{r} = H\bar{x} + \bar{n} \quad (1)$$

where $\bar{x} \in C^M$ is the transmitted vector, and \bar{n} is an $N \times 1$ random vector of white Gaussian noise. The ML decoding algorithm finds:

$$\hat{u} = \arg \min_{\bar{u} \in C^M} \|\bar{r} - H\bar{u}\|^2 \quad (2)$$

where \hat{u} is the decoded vector. The ML based lattice decoding system can be summarized as:

Input: The channel lattice generation matrix H and the received signal vector \bar{r} .

Output: An $M \times 1$ vector \hat{u} such that $H\hat{u}$ is a lattice point that is closest to \bar{r} .

• AV algorithm

Step 1: Preprocessing and initialization:
Transforms H into a lower triangular matrix by QR [5] decomposition algorithm. Initializes dimension index $k=N$. Finds the bounded index u_N and the orthogonal distance d from \bar{r} to the layer with index u_N . Sets $d_{best} = \infty$.

Step 2: Finite State Machine (FSM)
Upgrades d_{new} using d .
If $d_{new} < d_{best}$ and $k > 1$ goes to state A;
If $d_{new} < d_{best}$ and $k = 1$ goes to state B;
If $d_{new} > d_{best}$ goes to state C.

Step 3: State A:
Expands the layer into $(k-1)$ -dimensional sublayer and finds d and bounded u_k . Goes to step 2.

Step 4: State B
Records the currently best distance and the lattice points. Sets $k=2$, and finds d and bounded u_k . Goes to step 2.

Step 5: State C
Stops if $k=N$, otherwise moves the procedure one step up $k=k+1$, and finds d and bounded u_k . Goes to step 2;

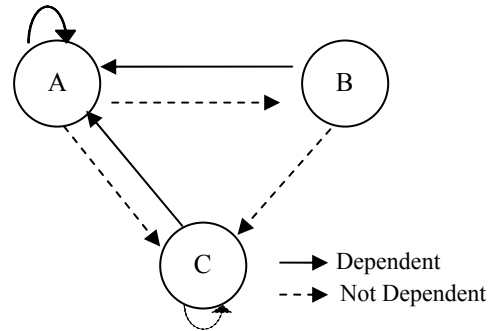


Figure 1 Dependency graph of the AV algorithm

There are three states involved in the decoding procedure as described in the *step-by-step* demonstration for the AV algorithm. Figure 1 shows the data dependency among all possible state transitions. State A is dependent on both states B and C if the search procedure switches to A from B or C, because the orthogonal distance d calculated in B or C is used to upgrade the layer index u_k in state A. Similarly, State A is self dependent. State B has no data dependency on A because these two states always work on different search dimensions if B follows A during the

search. For the same reason, state C is not dependent on any other states that could jump to C. Based on the data dependency analysis, the possibility of the parallelism among these three states is found as:

$A \parallel B$, $A \parallel C$, $B \parallel C$, $C \parallel C$. These conditions are used to implement the *state level* parallelism presented in Section 3.

• **VB algorithm**

Step 1: Preprocessing and initialization:
Transforms H into an upper triangular matrix by Cholesky [6] decomposition algorithm. Initializes the sphere radius \sqrt{C} by an adaptive method [8]. Sets dimension index $k=N$ and $d_{best}=\sqrt{C}$. Finds the upper bound L_N and index u_N .

Step 2: Finite State Machine (FSM)
Upgrades $u_k=u_k+1$.
If $u_k < L_k$ and $k>1$ goes to state A;
If $u_k < L_k$ and $k=1$ goes to state B;
If $u_k > L_k$ goes to state C.

Step 3: State A:
Expands the layer into $(k-1)$ -dimensional sublayer and finds the parameters used to upgrade u_k and L_k . Goes to state D.

Step 4: State B
Upgrades d_{new} .
If $d_{new} < d_{best}$, records the currently best distance and the best points. Sets $k=N$. Goes to state D.
If $d_{new} > d_{best}$, goes to step2.

Step 5: State C
Stops if $k=N$, otherwise moves the procedure one step up $k=k+1$, and goes to step 2;

Step 6: State D
Upgrades u_k and L_k , and goes to step 2.

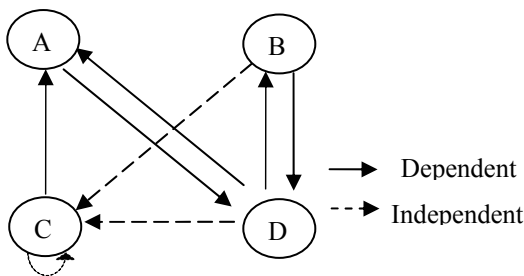


Figure 2 Dependency graph of the VB algorithm

Similarly to the AV algorithm, Figure 2 gives all possible state transitions and the data dependency of the VB algorithm. It shows that only state C can be executed simultaneously with states B, C or D. Therefore, the parallel states are: $B \parallel C$, $C \parallel C$, $D \parallel C$.

3. Architecture

This section presents hardware/software co-design architecture for both AV and VB algorithms. Two levels of parallelisms are analyzed and illustrated.

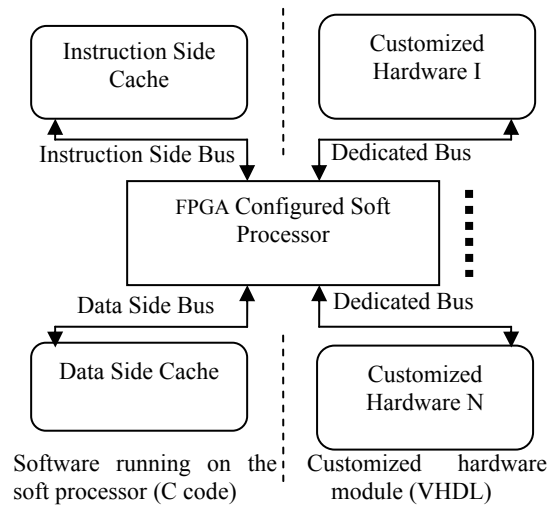


Figure 3 FPGA based hardware/software co-design architecture

Hardware/software co-design technique is applied to partition the decoding algorithm into the soft processor and the customized hardware modules. With a flexible and high speed interface between the hardware and software cores, this architecture can greatly speed up the application because of the high speed hardware calculations and the support of the flexibility and programmability of the soft core. Figure 3 shows the co-design architecture based on the XUP Virtex-II pro developing board. It includes an xc2vp30 FPGA and an embedded MicroBlaze™ (MB) soft processor, which is a 32-bit RISC processor, implemented using general logic primitives [9]. The On-chip Peripheral Bus (OPB) is chosen as the interface between MicroBlaze and Customized hardware. It is synchronous to the MicroBlaze processor and can achieve a high data transfer rate. The maximum number of the customized modules that can be attached to the

soft processor is determined by the size of the hardware core and the available resources on an FPGA.

- **Processor level parallelism**

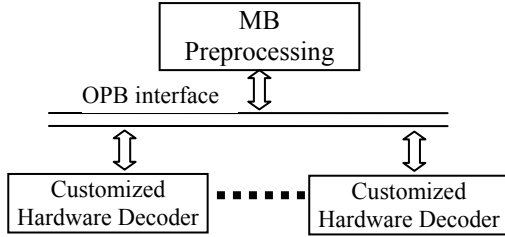


Figure 4 Processor level parallelisms

The hardware/software architecture for both AV and VB algorithms is illustrated in Figure 4. Because of the complicated matrix decomposition operations including matrix inversion and divisions, the preprocessing part is instantiated on MicroBlaze soft processor for both algorithms. The core decoding functions are implemented in customized hardware modules to speed up the decoding procedures. The OPB interface is used to transfer data between them. Furthermore, multiple decoding blocks are mapped on FPGA in parallel to improve the decoding rate. This is achievable because the lattice channel matrix is generally static within a frame length (typically 20ms), and all the signals received within the same frame length are decoded using the same matrix decomposed by the preprocessing part on MicroBlaze. Thus the *processor level* parallelism can decode multiple received signals simultaneously. Based on the FPGA resource usage shown in Section 4, Virtex-II pro developing board can implement 4 hardware decoders for the AV algorithm and 2 decoders for the VB algorithm in parallel to accelerate the decoding speed.

- **State level parallelism**

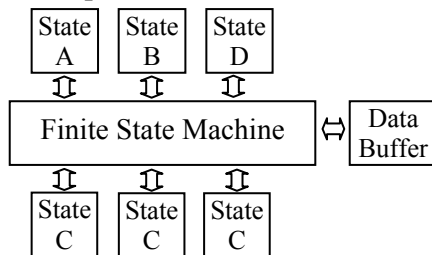


Figure 5 State level parallelism for the VB decoding algorithm

To further improve the decoding rate, a *state level* parallelism is developed and implemented on the customized hardware modules. According to the data dependency analysis in Section 2, multiple states can be executed simultaneously in the decoding procedure. As an example, Figure 5 presents the *state level* parallelism for the VB based decoding algorithm. A finite state machine (FSM) is designed as a central controller to be responsible for the state transitions and synchronization between the states. For a 4-antenna MIMO system, six hardware state modules are created, with one for each state and other two duplicated C states. This is because up to three state Cs can be executed at the same time with states B and D based on the flow analysis and experimental results. For instance, when the current state is B, FSM enables not only state B but also all three state C modules. The results from these modules can be directly used if this state B is followed by multiple state Cs. Data buffer unit is used to temporally store the data during the lattice decoding procedure.

The state parallel structure for the AV algorithm is similar to that of the VB algorithm. There are five state modules for a 4-antenna MIMO system compared to Figure 5, because state D does not exist in the AV algorithm.

4. Experimental Results

Both AV and VB algorithms are developed in EDK [10] and prototyped on XUP Virtex-II pro developing board with an xc2vp30 FPGA. The resource utilization and the maximum frequency of the decoders are compared between the two algorithms. The bit-accurate models for both algorithms are simulated in Matlab, and are used to verify the results from the hardware/software co-design implementations. The two algorithms are also implemented on DSP for performance comparison.

- **Soft core prototype performance**

As described in Section 3, the matrix decomposition is implemented on the MicroBlaze soft processor. The QR and Cholesky reduction algorithms are applied to decompose the channel matrix in AV and VB decoders respectively. Both algorithms involve matrix inversion and transpose operations during preprocessing. From synthesis results, the MicroBlaze soft processor can operate

at 100MHz when prototyped on the xc2vp30 FPGA. The timing performance of the preprocessing part for a 4-transmit and 4-receive antenna MIMO system is shown in Table 1.

Table 1 Soft core timing performance

	AV algorithm	VB algorithm
Device	Xilinx MicroBlaze	
Frequency	100MHz	
Decomposition	36,277cycles (QR)	4,621 cycles (Cholesky)
Inversion	12,287 cycles	
Transpose	1,129 cycles	
Total Time	53,422 cycles 0.534ms	32,548cycles 0.325ms

It is demonstrated from the results that the preprocessing for a 4 by 4 lattice decoder can be finished within 0.54ms, which is much less than the typical frame length (20ms) of a MIMO system. An 8-dementional MIMO system has also been examined. The results show that the preprocessing can be completed within 2.7ms for both algorithms. These results prove the feasibility of the *processor level* parallelism architecture designed in Section 3.

• **Hardware core prototype performance**

Based on the hardware/software co-design architecture described in Section 3, the core decoding functions are mapped on the logic cells to accelerate the decoding procedures. The FPGA resource utilization and the post place-and-route (PAR) frequencies are compared between the AV and VB algorithms for a single 4 by 4 lattice decoder as shown in Table 2.

Table 2 FPGA resource utilization for single lattice decoder

	AV algorithm	VB algorithm
Target FPGA	xc2vp30	
No. of SLICES	3586 out of 13696	3171 out of 13696
No. of bonded IOBs	109 out of 556	109 out of 556
No. of MULT 18X18s	14 out of 136	78 out of 136
Max Freq.	86.8MHz	69.3MHz

The results show that the VB based decoder uses more embedded multipliers than the AV based decoder because the VB algorithm involves more complicated calculations. Meanwhile the AV decoder can achieve a higher maximum frequency than the VB decoder.

Considering the *processor level* parallelism, the experimental results show that up to 4 and 2 hardware decoding modules can be mapped on the xc2vp30 device for the AV and VB algorithms respectively. The maximum frequency is 74.4MHz for the 4-module AV decoder and 55.2MHz for the 2-module VB decoder. These hardware modules are expected to decode the received signal simultaneously, however a slight overhead existed before starting a new module because the soft processor needs to calculate different received signal vector for each module. This overhead is not significant because only the multiplication of a $1 \times N$ vector with an $N \times N$ orthogonal matrix is involved.

• **Decoding rate**

In this subsection, the decoding rate calculated using a single lattice decoding module is compared between these two algorithms first. The advantage of the *processor level* parallelism is shown later.

Table 3 Data rate for FPGA-based 4 by 4 single decoder at 20 dB SNR

	AV Algorithm	VB Algorithm
Hardware	xc2vp30 FPGA	
Max Freq	86.8MHz	69.3MHz
Cycles/Iteration	12 cycles	20 cycles
# of iterations	5.8	14
Bits/Dimension	2	
Dimension	4	
Decoding rate	9.98Mbps	1.98Mbps

The data rate for a single decoder MIMO system with N transmit and N receive antennas is determined as:

$$Rate = \frac{Freq * Bits / Dimension * N}{Cycles / Iteration * Iteration_num} \quad (3)$$

In a parallel architecture, the *Iteration_num* is the total number of visits to any state. Two or more states executed at the same time are counted as one visit. *Cycles/Iteration* denotes the average clock cycles for a state visit. It is calculated as:

$$Cycles / Iteration = \Sigma(P_{State} * (Cycles / State)) \quad (4)$$

where P_{state} denotes the state visit probability for each state, which is calculated from software simulation. *Cycles/State* is the clock cycles used

for each state captured from the FPGA simulation. Simulation results show that the average processing time of a state visit is 12 clock cycles for the AV algorithm and 20 clock cycles for the VB algorithm. Based on these parameters, the decoding rate for a 4 by 4 single decoder MIMO system at 20dB signal-to-noise ratio (SNR) is given in Table 3. The results show that the AV based decoder is more than 5 times faster than the VB based decoder for this MIMO system. These results match the analysis given in Section 2.

The decoding rate of these two algorithms can be improved greatly if multiple decoding modules are instantiated. According to the FPGA resource usages shown in Table 2, up to 4 and 2 decoding hardware blocks can be implemented in a single xc2vp30 FPGA device for the 4-antenna AV and VB decoders. These hardware blocks can decode the receiving data bits concurrently. The experimental results illustrate that the decoding rate for a 4 by 4 MIMO system can reach up to 34.2Mbps and 3.15Mbps for the AV and VB algorithms correspondingly considering the received signal calculation overhead.

These two algorithms are also implemented on DSP in comparison with the proposed FPGA-based decoders. A fixed-point DSP TI TMS320C6201 is chosen as the benchmark platform, which operates at a fixed frequency 200MHz. Figure 6 gives data rate comparisons of decoding cores between the FPGA and DSP implementations. The results show that the DSP-based AV and VB decoders can only reach about 1.8Mbps and 0.2Mbps respectively for a 4 by 4 MIMO system at 20dB SNR. Comparing with the FPGA-based performance, the key reason for the low decoding rate is the lack of parallelism support in DSP. All states in the decoding procedure must be executed sequentially, which greatly increases the *Cycles/Iteration* and *Iteration_num*. As shown in Figure 6, the DSP based VB decoder takes 474 clock cycles in average for a state visit, which takes only 20 clock cycles in the corresponding FPGA implementation. The *Iteration_num* of the DSP implementation is larger than that of the FPGA implementation because DSP does not support the *state level* parallelism designed in Section 3. Furthermore the DSP-based decoders can not achieve the *processor level* parallelism. Although the system clock frequency is higher, the decoding

rate of the DSP-based lattice decoders is still almost 20 times slower than the FPGA-based lattice decoders.

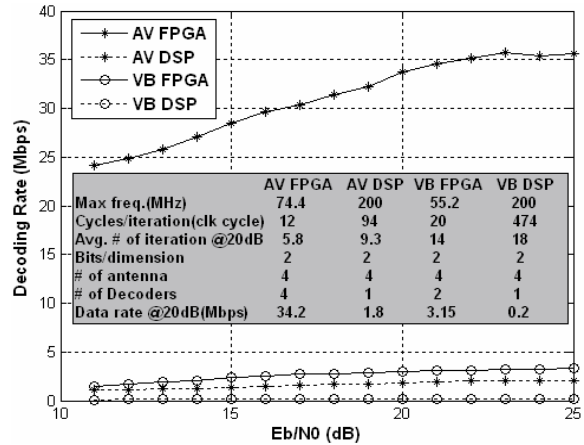
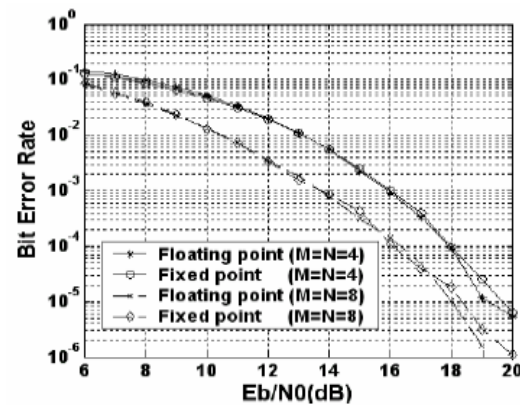
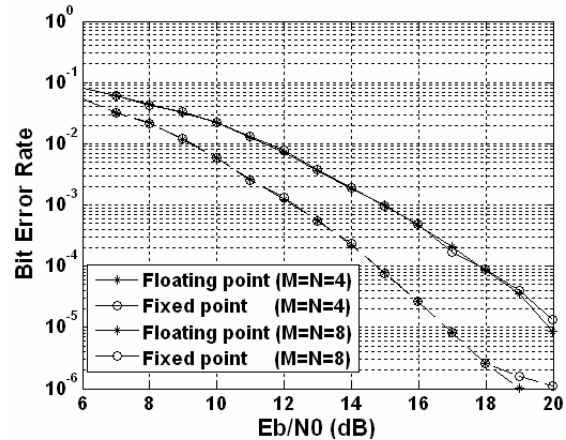


Figure 6 Decoding rate comparisons for a 4 by 4 MIMO system

• Bit Error Rate



(a)



(b)

Figure 7 BER performance for a 4 by 4 MIMO system (a) AV decoder (b) VB decoder

The BER performance is evaluated for different lattice dimensions, different lattice generation matrices with Gaussian distribution and different SNRs. Figure 7 shows the BER performance for both AV and VB based decoders. The performances between 4 by 4 and 8 by 8 antenna systems are also compared. The experimental results show that the FPGA-based lattice decoders designed in this paper match the performances from the bit-accurate software simulation.

5. Conclusion

Hardware/software co-design architecture for two typical lattice decoders has been designed and implemented in this paper. The FPGA-based decoders are mapped on the customized hardware modules with a MicroBlaze soft processor for the channel matrix preprocessing. Two levels of parallel structures are analyzed in this co-design architecture. The proposed system architecture is prototyped on the XUP Virtex-II pro developing board. The experimental results show that the AV and VB based decoders can reach up to 34.2Mbps and 3.15Mbps respectively for a 4 by 4 MIMO system at 20dB SNR. They are about 19 and 16 times faster than their respective implementations in a DSP processor. The BER performance of the experimental prototype matches with the software simulation results.

References

- [1]. E. Viterbo and J. Boutros, "A universal lattice code decoder for fading channels", *IEEE Trans. on Inf. Theory*, pp. 1639-1642, July 1999.
- [2]. E. Agrell, T. Eriksson, A. Vardy and K. Zeger, "Closest point search in lattices", *IEEE Trans. on Inf. Theory*, pp. 2201-2214, August 2002.
- [3]. A. Adjoudani, E. C. Beck, A. P. Burg, G. M. Djuknic, T. G. Gvoth, D. Haessig, S. Manji, M. A. Milbrodt, M. Rupp, D. Samardzija, A. B. Siegel, T. Sizer, H. C. Tran, S. Walker, S. A. Wilkus, and P. W. Wolniansky, "Prototype experience for MIMO BLAST over third-generation wireless system", *IEEE Journal on Selected Areas in Communications*, Vol. 21, No. 3, pp. 440-451, April 2003.
- [4]. M. Guillaud, A. Burg, M. Rupp, E. Beck, and S. Das, "Rapid prototyping design of a 4x4 BLAST-over-UMTS system", *35th Asilomar Conference on Signals, Systems and Computers*, pp.1256-1260, Pacific Grove, CA, USA, November 4-7, 2001.
- [5]. W. W. Hager, "Applied Numerical Linear Algebra", Englewood Cliffs, NJ: Prentice-Hall, pp. 208-236, 1988.
- [6]. H. Cohen, "A course in Computational Algebraic Number Theory", Berlin, Germany: Springer-Verlag, pp.102-104, 1993.
- [7]. <http://www.xilinx.com/univ/xupv2p.html>
- [8]. J. Ma, C. Liang and X. Huang, "A comparison of lattice decoding algorithms in hardware implementation", *Proc. SPIE*, Vol. 5819, pp. 200-210, June 2005.
- [9]. http://www.xilinx.com/xlnx/xebiz/designResources/ip_product_details.jsp?key=micro_blaze
- [10]. http://www.xilinx.com/bvdocs/ipcenter/data_sheet/EDK_Sell_Sheet.pdf