Dynamically Reconfigurable Soft Output MIMO Detector

Pankaj Bhagawat
Dept. of E.C.E
Texas A&M University
College-Station,TX-77840
pankaj-bhagawat@neo.tamu.edu

Rajballav Dash
Dept. of E.C.E
Texas A&M University
College-Station,TX-77840
rajballavdash@neo.tamu.edu

Gwan Choi
Dept. of E.C.E
Texas A&M University
College-Station,TX-77840
gchoi@ece.tamu.edu

***Abstract*— MIMO systems (with multiple transmit and receive antennas) are becoming increasingly popular, and many next-generation systems such as WiMAX, 3-GPP LTE and IEEE802.11n wireless LANs rely on the increased throughput of MIMO systems with up to four antennas at receiver and transmitter. High throughput implementation of the detection unit for MIMO systems is a significant challenge. This challenge becomes still harder, because the above mentioned standards demand support for multiple modulation and coding schemes. This implies that the MIMO detector must be dynamically reconfigurable. Also, to achieve required Bit Error Rate(BER) or Frame Error Rate (FER) performance, the detector has to provide soft values to advanced Forward Error Correction (FEC) schemes like Turbo Codes. This paper presents an ASIC implementation of a novel MIMO detector architecture that is able to reconfigure on the fly and provides soft values as output. The design is implemented in 45nm predictive technology library[16], and has a parallelism factor of four. The detector has many qualities of a systolic architecture and achieves a continuous throughput of 1Gbps for QPSK, 500Mbps for 16-QAM, and 187.5Mbps for 64-QAM. The total area is estimated to be approximately 70KGates equivalent, and power consumption is estimated to be 114mW.**

I. Introduction

The scarcity of available radio frequency spectrum combined with the increasing need for higher data rates has led to the use of multiple input-multiple output (MIMO) wireless systems, which offers higher throughput without any overhead in terms of bandwidth or transmitter power as compared to single input single output (SISO) wireless system. Future generation wireless standards such as IEEE802.11n, 3-GPP LTE, Wi-Max etc. all have MIMO as a key enabling technology. The physical layer (PHY) of these standards support multiple modulation schemes, multiple antennas configuration, variable code rate and multiple space-time coding schemes [9]. In this paper, we present a reconfigurable architecture for soft MIMO detection and its ASIC implementation estimates. The detector is able to reconfigure on the fly which is one of the prime requirements for above mentioned standards.

Practical implementation of MIMO systems pose a major challenge due to the complicated nature of the detection algorithms, this is particularly true for a soft output MIMO detector. A soft detector not only computes binary (or hard) estimates of the transmitted bits, but also provides reliability of the binary estimates. The "soft decisions" from the detector is then fed to FEC schemes like turbo-decoder, Low Density Parity Check (LDPC) decoder, or a Viterbi decoder. In all the cases soft values based FEC decoding provides much better BER performance than its hard counterpart[12]. Apart from providing soft values, the detector should be able to support various modulation schemes (like BPSK, QPSK,16-QAM, and 64-QAM), preferably on a single reconfigurable architecture. In the past, very few authors have addressed the issues of dynamic reconfigurability like [6,7,10] but they only deal with the hard decision detector. Also, only [10] provides a detector that can reconfigure on the fly.

The choice of algorithm and architecture has a significant bearing on the final hardware complexity and reconfigurability. Apart from the BER/FER performance, we focus on architectural issues like pipelining, parallelism, and reconfigurability. The detection algorithms can be broadly classified into linear, and non-linear. Linear algorithms like zero-forcing(ZF) [11], or Minimum Mean Squared Error(MMSE)[11] are low complexity but incur high penalty in BER/FER performance. Non-linear detectors like Successive Interference Cancellation(SIC)[11] are low complexity too, but provide only modest gain over their linear counterparts. Moreover, neither ZF nor SIC based receivers do well in a wireless channel with limited diversity. Authors in [11,14] provide excellent comparative study of various detectors in different channel conditions. It is clear that more sophisticated algorithms (tree search based) need to be considered for practical systems due their superior BER/FER performance. To get close to the optimum BER performance researchers have proposed many algorithms, that do non-exhaustive tree search, such as List Sphere Decoder (LSD)[12], however, its complexity is still too large, and is very hard to map onto a parallel, pipelined architecture . Also, LSD converges to a solution in a random fashion making it difficult to incorporate in a practical system. On the other hand, algorithms based on Breadth First Search (BFS) such as K-best, provides constant throughput but involves sorting operation which is very expensive, especially for higher order modulation schemes like 64-QAM. One of the reported implementation of a soft MIMO detector that supports 64-QAM is presented in [13]. Recently [14] presented an algorithm that takes BFS based approach to the problem, this algorithm is called Layered ORthogonal Lattice Detector(LORD) and

978-1-4244-2658-4/08/$25.00 ©2008 IEEE

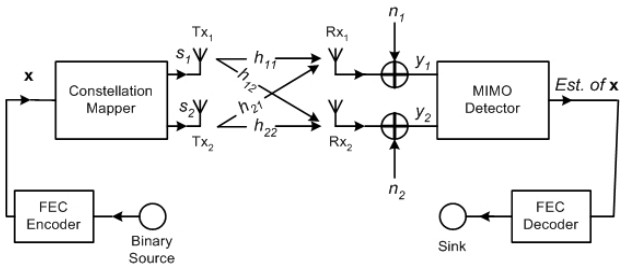

Fig. 1. End to End 2×2 MIMO System

provides close to optimal result. It can be implemented in a highly parallel and pipelined manner, has fixed throughput(for a given modulation scheme).

To the best of our knowledge no dynamically reconfigurable soft MIMO detectors have been reported in open literature. In this paper, we propose a novel architecture of a reconfigurable soft MIMO detector and its ASIC implementation estimates in 45nm predictive standard cell library. Our detector supports on the fly reconfigurability for QPSK, 16-QAM and 64-QAM modulation schemes. The control logic has low complexity and is highly integrated with the data flow. It delivers quasi-optimal FER performance with no reconfiguration latency, leading to uninterrupted detection of MIMO symbols.

The paper is organized as follows: Section II describes the basics of the channel model and the detection algorithm. This section concludes with description of the LORD algorithm, which provides soft outputs. In Section III we present the architectural details of the design. Section IV presents the ASIC implementation results, and Section V concludes the paper.

II. MIMO Detection

Fig.1 shows a simplified end to end MIMO system with two transmit and receive antennas(2×2 MIMO system). Binary source generates a sequence of information bits that is required to be transmitted over a wireless link. These bits are encoded by a FEC encoder(such as LDPC, Turbo codes, Convolutional Codes). The encoded bit sequence (**x**) is then modulated onto symbols (s_1, s_2 etc.) and sent to the transmitter, symbols from each transmitter undergoes independent *gains* (h_{11}, h_{12} etc.) before reaching the receiver. Hence, Rx_1 receives $s_1h_{11} + s_2h_{21}$, and Rx_2 receives $s_1h_{12} + s_2h_{22}$. Signals at Rx_1 and Rx_2 are further corrupted by noise (n_1,n_2). Obviously the big problem with this is that the receiver sees a combination of what was transmitted from both transmit antennas plus noise. The MIMO detector attempts to compute the estimate $\hat{s}$ of the most likely transmitted symbol sequence [s_1,s_2], and de-modulates $\hat{s}$ to give out the estimate of the encoded bit sequence (is is assumed that the gains h_{11}, h_{12} etc are known at the receiver). These bits are then fed to the FEC decoder to get back the bits generated by the binary source.

A. Channel Model and Optimal Hard MIMO detection

Above discussion can be generalized to M_T transmit and M_R receive antennas and expressed in terms of matrices as shown in eqn.1[1].

$$\mathbf{y} = \mathbf{Hs} + \mathbf{n} \tag{1}$$

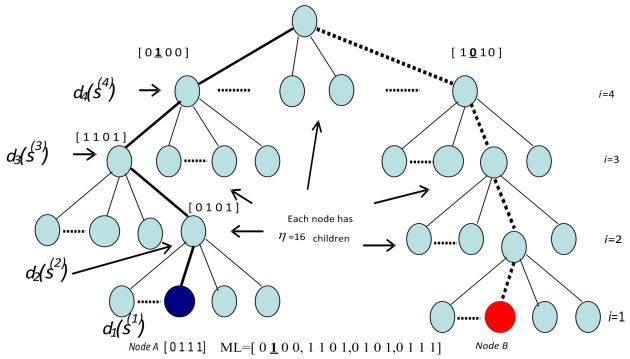

Fig. 2. Tree Structure for MIMO detection

where $\mathbf{y}=[y_1, y_2, ..., y_{M_R}]^T$ is a $M_R \times 1$ received vector, $\mathbf{s}=[s_1, s_2, ..., s_{M_T}]^T$ is $M_T \times 1$ transmitted vector (will be referred to as a MIMO symbol in the sequel), $\mathbf{n}$ is $M_R \times 1$ zero mean complex Gaussian noise vector, and $\mathbf{H}$ is a $M_R \times M_T$-dimensional complex matrix. The $(i,j)^{th}$ element, h_{ij}, of the matrix $\mathbf{H}$ denotes the complex channel gain from the j^{th} transmit antenna to the i^{th} receive antenna. As will be useful later, the i^{th} column of $\mathbf{H}$ is denoted as $\mathbf{h}_i$. In this paper we will assume $M_T = M_R = 4$, unless specified otherwise.

Each entry s_i $(i = 1, 2, ..M_T)$ in the MIMO symbol $\mathbf{s}$, is drawn from a set Ω of cardinality η. In general the members of the set Ω are complex numbers with their real and imaginary parts of the form $\{-\sqrt{\eta} + (2k-1)\}$ where k=1,2,...$\sqrt{\eta}$. This scheme is called as a η-ary Quadrature Amplitude Modulation (QAM) and s_i are called QAM symbols. The QAM symbols are generated by mapping(using look-up table, for instance) a group (of size $\sqrt{\eta}$) of binary bits onto a symbol(a complex number) from Ω.

The objective of the MIMO detector is to estimate $\hat{\mathbf{s}}$ of $\mathbf{s}$(bits can be found using *de-mapping* the symbols in $\hat{\mathbf{s}}$) based on the the observation of $\mathbf{y}$ along with the knowledge of $\mathbf{H}$. It has been shown that the optimal or the Maximum Likelihood (ML) estimate $\hat{\mathbf{s}}_{ml}$ of $\mathbf{s}$ is given by eqn.2 [12]:

$$\hat{\mathbf{s}}_{ml} = arg \min_{\mathbf{s} \in \Omega^{M_T}} \|\mathbf{y} - \mathbf{Hs}\|^2 \tag{2}$$

A straightforward approach to solving eqn.2 is an exhaustive search over all possible candidate MIMO symbols. However, since the number of possible solutions grows exponentially with M_T, the implementation of an exhaustive search becomes impractical as M_T or η increases. For example, in case of a 4x4 MIMO system with 16-QAM modulation an exhaustive search would require the evaluation of $16^4 = 65536$ candidate vector symbols. One way to circumvent the exhaustive search is to evaluate only a small subset of all the possible vectors. This can be achieved by noting that $\mathbf{H}$ can be triangularized using QR decomposition: $\mathbf{H} = \mathbf{QR}$. where, $\mathbf{R}$ is an upper triangular matrix, and $\mathbf{Q}^H$ is the Hermitian of a unitary matrix $\mathbf{Q}$. Hence, the cost function given by (2) can now be rewritten as [5],

$$\hat{\mathbf{s}} = \|\mathbf{y} - \mathbf{Hs}\|^2 = \|\hat{\mathbf{y}} - \mathbf{Rs}\|^2, and\ \hat{\mathbf{y}} = \mathbf{Q^H y} \tag{3}$$

Vector $\hat{\mathbf{y}}$ as defined by (3) is the unconstrained zero forcing solution. (3) can be further expanded as shown in eqn.(4)-(6).

$$d_i(s^{(i)}) = d_{i+1}(s^{(i+1)}) + |e_i(s^{(i)})|^2 \tag{4}$$

$$|e_i(s^{(i)})|^2 = |c_{i+1}(s^{(i+1)}) - R_{ii}.s_i|^2 \quad (5)$$

$$c_{i+1}(s^{(i+1)}) = \hat{y}_i - \sum_{j=i+1}^{M_T} R_{ij}.s_j \quad (6)$$

The quantity $|e_i(s^{(i)})|^2$ will be called the Incremental Euclidean Distance (IED), and the term $d_i(s^{(i)})$ will be called Partial Euclidean Distance (PED) for $i > 1$, and Euclidean Distance (ED) for $i = 1$. The fact that **R** is upper-triangular ensures that each term on LHS of eqns.(4)-(6) depends only on the current level i, and the history of the path to reach that level(note that in eqn.(6), the index j runs from $i+1$ to M_T). Because the PED's depend only on $s^{(i+1)}$, they can be associated with corresponding nodes in a η-*ary* tree with M_T levels. The computation of the terms $d_1(s^{(1)})$ can then be interpreted as a traversal of the tree from the root($i = M_T$) to the leaf ($i = 1$)corresponding to **s**. The estimate can now be obtained by searching the leaf with the smallest ED and returning the path from the top level to that leaf as $\hat{\mathbf{s}}_{ml}$. The final hard output bits is provided by de-mapping the constituent QAM symbols of $\hat{\mathbf{s}}_{ml}$. The complexity of this tree search can be greatly reduced by noting that IEDs are always positive, and hence if the PED of a node exceeds a predefined threshold (called radius) the subtree rooted at that node can be excluded from further search. This approach is commonly known as sphere decoding, we will omit further discussion on sphere decoding for sake of brevity. Please refer to [2,4] and references therein for more information on sphere decoding.

Fig.2 shows the tree structure for a the MIMO detection for a 4×4, 16-QAM modulated system. Node A corresponds to the leaf with the least ED, and hence, the path from root to node A is the most likely **s** that was transmitted. Each node on the tree is in fact a QAM symbol(s_i) which was in turn created using 4 bits. The final *hard* output of the detector for the example shown would be [0 1 0 0 1 1 0 1 0 1 0 1 0 1 1 1]. The FEC decoder performs much better when provided with *reliabilities* or *soft output* associated with each hard output bit [12]. The computation of these reliabilities is the subject of discussion in next sub-section.

B. Soft MIMO Detection

The objective of a soft MIMO detector is to output the reliability associated with each hard output bit. This reliability is expressed in terms of the Log-Likelihood Ratio (LLR) of each bit, and is defined as $L(x_i|\mathbf{y}) = ln\frac{P(x_i=1|\mathbf{y})}{P(x_i=0|\mathbf{y})}$, where x_i is i^{th} bit of a binary vector **x**. This can be approximated [12] as:

$$L(x_i|\mathbf{y}) \approx \min_{x \in \mathbf{X}_{i,0}} \{\Gamma(\mathbf{x},\mathbf{y})\} - \min_{x \in \mathbf{X}_{i,+1}} \{\Gamma(\mathbf{x},\mathbf{y})\} \quad (7)$$

where $\Gamma(\mathbf{x},\mathbf{y})$=$|\mathbf{y} - \mathbf{H}\mathbf{x}|^2$, $\mathbf{X}_{i,0}$ and $\mathbf{X}_{i,+1}$ are two disjoint sets of bits with i^{th} member as 0 and +1 respectively.

In eqn.7 there are two minimization problems (eqn.2 has only one), i.e for each bit x_i it requires identification of the most likely transmit sequence where $x_i = 0$ and the most likely one where $x_i = 1$(first and second term in eqn.7 respectively). This fact is illustrated in Fig.2, where the path from root to node A correspond to the ML bit sequence. In order to compute LLR for the second bit of($x_2 = 1$ in this case) of ML bit sequence, it has to compute a bit sequence with $x_2 = 0$ *and* has minimum metric associated with it (this is shown by node B). The difference between these two metrics gives us the LLR value of x_2(Note that one of the two minima in eqn. 7 is always given by the metric associated with the ML bit sequence). To compute LLR values for all the bits we have to repeat this operation for each bit in the binary vector **x**, this is exactly what is done in Repeated Tree Search (RTS) algorithm [15]. Another algorithm which approximates the two terms in eqn.7 is List Sphere Decoder (LSD) [12]. LSD computes a set of tree leaves with *good* metrics, based on which it computes the LLR values. The complexity of RTS and LSD algorithms can be overwhelming even for a moderate sized QAM constellation. Also, these algorithms converge in a non-random fashion making it difficult to incorporate in practical systems.

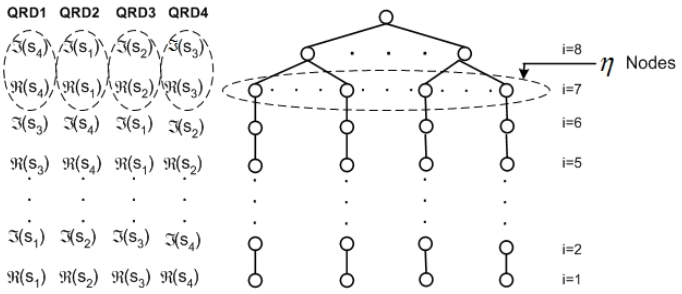

Fig. 3. LORD Algorithm Flow

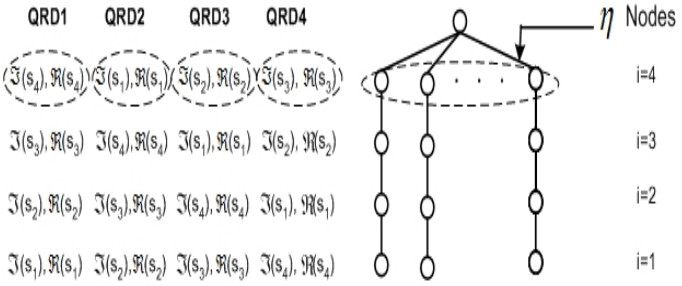

Fig. 4. Transformed Flow for LORD

C. Layered ORthogonal Decoding (LORD)

Recently a new algorithm called LORD has been proposed in [14]. This algorithm takes a suboptimal approach in searching for the two nodes that are required to compute the LLR value of a particular bit. Before we describe the LORD algorithm we briefly explain the real valued decomposition of the system model.

Real Valued Decomposition: Recall that in eqn.1 all the variables are complex numbers. In LORD, the model depicted by eqn.1 is converted such that all the variables are now real numbers. That is, **s** is replaced by s_r=$[\Re(s_1), \Im(s_1), ...\Re(s_{M_T}), \Im(s_{M_T})]$ (where $\Re$, and $\Im$ denote real and imaginary parts respectively). Vectors **y**, and **n** are expanded in terms of their real and imaginary parts in a similar manner.

Real equivalent of the matrix **H** can be written as H_r=[$\mathbf{h}r_1$,$\mathbf{h}r_2$,...$\mathbf{h}r_{2M_T}$]. Each pair of columns ($\mathbf{h}_{2k-1}$,$\mathbf{h}_{2k}$),

$k = \{1, ..., M_T\}$ of the real channel matrix H_r has the form: $\mathbf{h}_{2k-1} = [\Re(H_{1k}), \Im(H_{1k}), ...\Re(H_{M_R k}), \Im(H_{M_R k})]^T$, and $\mathbf{h}_{2k} = [-\Im(H_{1k}), \Re(H_{1k}), ... - \Im(H_{M_R k}), \Re(H_{M_R k})]^T$ Notice that after complex to real conversion the dimensionality of all vectors and matrices in eqn.1 will double. This implies that the tree depth is also doubled. However, the number children per node reduces to $\sqrt{\eta}$ (hence the total number of nodes in the tree remains same).

Preprocessing: As discussed in section II, MIMO detection is preceded by a preprocessing stage. This preprocessing stage (called QR decomposition or QRD) facilitates detection by mapping the detection process on to a tree search. It can be seen from the system model (eqn.1) that by permuting the columns of **H** matrix before QRD, we can precisely control which QAM symbols will be processed first (at top level of the tree).

LORD algorithm computes the LLR of the bits corresponding to the nodes at top two levels. For this it processes all nodes at top two levels, and evaluates only the best children of these nodes for levels below(Fig.3). To compute LLR values for bits at levels below the top two levels it recomputes the QRD with the columns of **H** permuted. Obviously, QRD has to be performed M_T times in order to compute LLR values for bits at all levels of the tree. Basically it processes all nodes at top two levels of the tree, but computes metrics only for the best child (node with least PED) until it arrives at leaf nodes, LLR values are then computed using eqn.7. Fig.3 shows the operation of LORD for a 4×4 system with η-ary QAM modulation. After each QRD, LLR values for bits *riding* at level 8 and 7 are computed. Thus, during QRD1 the bits associated with $\Im(s_4)$, and $\Re(s_4)$ will have their LLR values computed. During QRD2 the bits associated with $\Im(s_1)$, and $\Re(s_1)$ will have their LLR values computed. At the end of QRD4, we have LLR values for all the QAM symbols. The number of MIMO symbols evaluated per QRD is η. Also, QRD has to be performed 4 times, hence total number of MIMO symbols evaluated is 4η.

The result of this real valued decomposition is that after QRD of **H**, the upper triangular matrix **R** has a property that makes processing of real and imaginary parts of a QAM symbol completely independent of each other, hence they can be processed in parallel(interested reader is referred to [14] for more mathematical details). This fact can be used to transform the LORD tree into a tree as shown in Fig.4. The transformed LORD tree is again of height 4 because real and imaginary parts can now be processed in parallel. This transformation is shown explicitly by dotted ellipses in Fig.3 and 4.

III. Reconfigurable MIMO Detector

As discussed earlier LORD needs 4 different QRDs to be performed, this means the preprocessing complexity of LORD is 4 times that of other MIMO detectors. However, in packet based wireless systems such as 802.11n this additional processing has to performed only once per packet(which typically contain several hundred or even thousands of MIMO symbols). Thus, the additional complexity is averaged over a large number of MIMO symbols. This implies that per MIMO symbol preprocessing complexity is nominal.

A. High Level Architecture

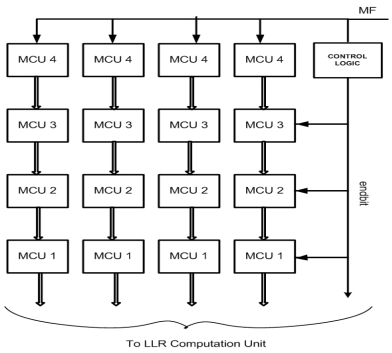

Fig. 5. High Level Architecture

Fig.5 shows the high level architecture of the proposed decoder. It consists of an array of Metric Computation Units (MCUs). It can process four candidate vectors in parallel. It can be seen from Fig.5, that the data flow, including the control signals, is always forward flowing. An important implication of this fact is that this detector can be operated in a *continuous* manner, even when it is switching between different modulation schemes. Since the detector has to evaluate $4.\eta$ candidate MIMO symbols (for an η-ary modulation scheme), and because the detector incurs no reconfiguration latency, it takes η clock cycles to compute LLR values for an η-ary modulated MIMO symbol. Another important result of above mentioned qualities of the architecture is that the control system is independent of the number of pipeline stages. Hence, very high throughput is achievable by introducing more pipeline stages. Also, the parallelism factor can be increased(this requires some changes to be made in the control system) to further increase the throughput.

The Metric Computation Units (MCUs) consists of two identical units, Real part MCU (R-MCU) and a Imaginary part MCU (I-MCU). Both compute $d_i(\mathbf{s}^{(i)})$ metrics using eqns.(4)-(6). Following subsection describes MCU in greater details.

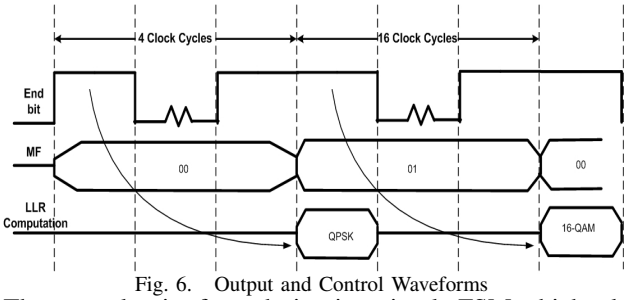

Fig. 6. Output and Control Waveforms

The control unit of our design is a simple FSM which takes in MF[1:0], where MF indicates the modulation format on the current MIMO vector (00=>QPSK, 01=>16-QAM,and 10=>64-QAM), and generates a signal 'endbit' every (η) clock cycle. This signal indicates the completion of detection for one MIMO symbol. The waveforms in Fig.6 shows the relation of the control signals with respect to the MIMO symbol. The architecture operates in a continuous manner

within and between MIMO symbols. The timing diagram for the decoding process is shown in Fig.7. The architecture has many qualities of a systolic architecture, in that these MCUs have mostly local connections and the pipeline is never broken even while switching between modulation schemes.

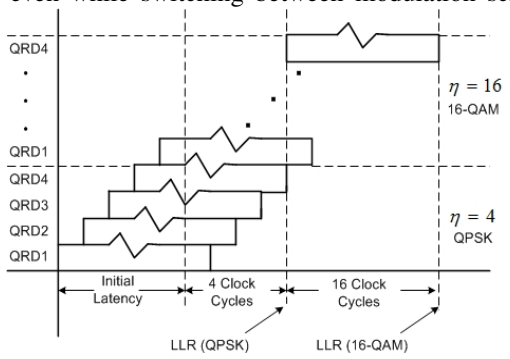

Fig. 7. Timing Diagram

The throughput of the decoder can be very easily computed as follows. Recall that we are considering a 4×4 MIMO system, and an η-ary QAM symbol is constructed using $log_2(\eta)$ bits. It takes η/4 clock cycles to compute LLR values for bits at top level in a MIMO symbol. Since there are 4 levels, to calculate LLR values for all the bits will take η cycles. Hence, the throughput is given by $(4log_2(\eta)/\eta)f$, where f is the clock frequency.

B. MCU architecture

The MCU computes eqns.(6)-(4) in that order. Fig.8 shows the detailed structure of an R-MCU at level 1.(lowest level in tree, corresponding to $i = 1$). The upper dotted box in Fig.8 evaluates (6). The R-MCU in turn is composed of 1) Product Computers (PCs) 2) Adders 3) a Slicer 4)a Norm Computer (NC).

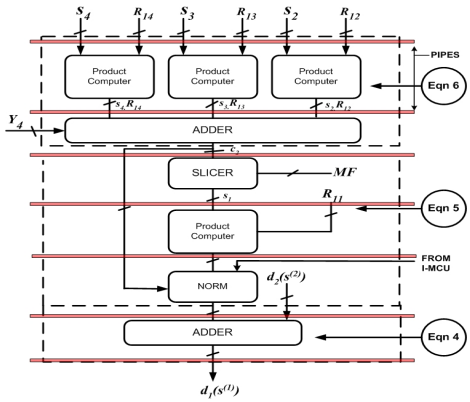

Fig. 8. Real Part Metric Computation Unit (R-MCU) of Level 1

Product Computer (PC): Fig.9 shows details of a PC, this unit computes the product of R_{ii} and s_i as required in eqn.5 and eqn.6. There is no need to implement the product terms in eqn.6 using a multiplier, this can be implemented simply a by shift and add operation, because the QAM constellation points only take on a finite number of integer values (e.g. in 64-QAM scheme the real and imaginary part of $s_j \epsilon \{-7, -5, -3, -1, 1, 3, 5, 7\}$). For example, $7R_{ii}$ can be implemented by shifting R_{ii} left 3 times and subtracting R_{ii}

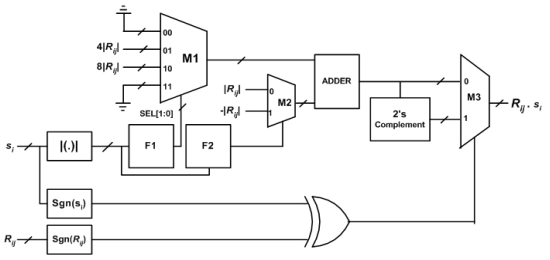

Fig. 9. Product Computer

from it. Blocks labeled F1 and F2 implement a combinational function which generate select signals for MUX M1 and MUX M2 respectively. F1 basically checks for 3 conditions: 1) Is $|s_i|$=1?, or $|s_i|$=3,5?, or $|s_i|$=7?, based on these condition selection is made on the inputs of M1. F2 checks for 2 conditions: 1) Is $|s_i|$=1,5?, or $|s_i|$=3,7?, based on this M2 chooses appropriate output to be added with output of M1. Based on signs of R_{ij} and s_i the final "'product"' is corrected for sign(we have utilized the symmetry inherent in a QAM constellation).

Slicer: As discussed earlier the LORD algorithm picks all the children of the root node(Fig.4), and the 'best' child (nodes with least $|e_i|^2$) of these nodes thereafter (see subsection II.C). From eqn.(5) it can be easily seen that, in order to minimize $|e_i|^2$, we need to compute s_i such that the *distance* between c_{i+1} and $R_{ii}s_i$ is minimized. The slicer block picks the nearest *scaled* QAM symbol($R_{ii}s_i$) to c_{i+1} as shown in Fig.10. This operation involves independently comparing real and imaginary parts of c_{i+1} with appropriate decision thresholds. For example, in Fig.10 the nearest *scaled* QAM symbol to c_{i+1} is $5R_{ii}$, and hence the best child is s_i=5. In general, the decision thresholds are given by $(-(\sqrt{\eta} - 2) + 2k)R_{ii}$,

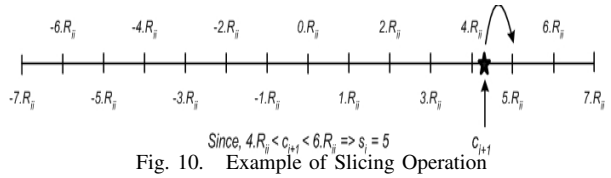

Fig. 10. Example of Slicing Operation

where k is an integer such that $0 \leq k \leq (\sqrt{\eta} - 2)$. The decoder configures the slicer based on the MF bits (MF bits are essentially representative of η).

Norm Computer:The Euclidean norm or l^2 norm involves a squaring operation which requires multipliers. Multipliers are in general expensive in terms of hardware cost. In [4] it has been shown that the use of simplified norms leads to significant reduction in hardware cost with marginal BER degradation (for hard decoding). In our design we have replaced the l^2 norm in eqn.(5) by l^1 norm. The l^1 norm approximation for eqn.(4) is given by:

$$d_i(s^{(i)}) = d_{i+1}(s^{(i+1)}) + |\Re\{e_i(s^{(i)})\}| + |\Im\{e_i(s^{(i)})\}| \quad (8)$$

The use of l^1 norm causes the FER to degrade only slightly(Fig.11).

The LLR computation unit shown in Fig.5 calculates the LLR values using eqn.7.

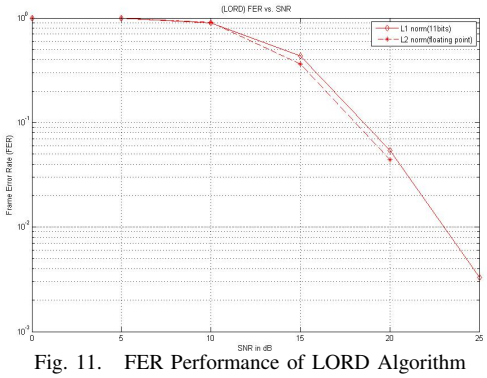

Fig. 11. FER Performance of LORD Algorithm

IV. Implementation Results

Previously reported architectures are either non-dynamic[7] or uses on-chip processors[6] to achieve reconfigurability, which is not best suited for ASIC implementation. Moreover, the algorithm used in aforementioned designs significantly degrades the BER performance. The design in [6] incurs reconfiguration latency resulting in a slower throughput and increased memory requirements to store the intermediate data. The design in [10] is able to reconfigure on the fly but supports only hard detection. In contrast, our decoder supports on the fly reconfiguration, and provides soft values as output. We note that no fair quantitative comparison is possible with existing designs, since all the aforementioned designs are either FPGA based or support only hard detection only. Hence, in Table.I we provide a qualitative comparison of existing work with ours ($\eta = 4$, $\eta = 16$, and $\eta = 64$ corresponds to QPSK, 16-QAM, and 64-QAM resp.) MATLAB was used to simulate bit accurate model of the decoder. We chose eleven bit fixed point quantization(internal precision was maintained) for negligible FER degradation (Fig.12). A detailed hardware architecture was then developed. The RTL coding was done using Verilog HDL. OSU PDK 45nm CMOS predictive standard cell library was used for the design flow. Synopsys Design Compiler was used to synthesize the gate level net-list. The synthesis results are provided in Table II.

TABLE I

COMPARISON OF EXISTING ARCHITECTURES

Ref.	$\eta = 4$	$\eta = 16$	$\eta = 64$	Dyn. Recon.	BER/FER	Soft Detect.
[2]	No	Yes	No	No	Quasi-ML	No
[3]	No	Yes	No	No	Quasi-ML	No
[4]	No	Yes	No	No	ML	No
[5]	No	Yes	No	No	Quasi-ML	No
[6]	Yes	Yes	Yes	Yes	Sub-Opt.	No
[8]	No	Yes	No	No	Quasi-ML	No
[13]	No	No	Yes	No	Quasi-ML	Yes
$[Ours]$	Yes	Yes	Yes	Yes	Quasi-ML	Yes

Table.II summarizes the implementation results. The design operates at 500Mhz clock frequency. As a result the proposed detector design is able to achieve very high throughput even for high order modulation scheme like 64-QAM (187.5Mbps), whereas, QPSK and 16-QAM are detected at 1Gbps and 500Mbps respectively. This gives us maximum energy per detected bit of 608pJ/bit.

V. Conclusion

A novel reconfigurable MIMO detector architecture and its ASIC implementation is presented in this paper. The detector is dynamically reconfigurable for QPSK, 16-QAM and 64-QAM modulation schemes for 4×4 MIMO system. The decoder was implemented in 45nm predictive technology library [16]. The decoder achieves throughput of 1Gbps, 500Mbps, and 187.5Mbps for QPSK,16-QAM, and 64-QAM respectively. The proposed architecture is highly suitable for the next generation wireless standards because of its flexibility, support for soft output and higher throughput.

TABLE II

SYNTHESIS RESULTS

Target Tech. Library	OSU 45nm PDK[16]
Gate Equivalent	70,000
Power Consumption	113.9mW
Maximum Frequency	500MHz
Throughput: QPSK	1Gbps
Throughput: 16-QAM	500Mbps
Throughput: 64-QAM	187.5Mbps

References

[1] W. Wolniansky, et al., *"V-BLAST:An architecture for realizing very high data rates over the rich-scattering wireless channel"*, Proc. IEEE ISSSE 1998, pp.295-300, Sept. 1998.

[2] Z. Guo and P. Nilsson, *"Algorithm and implementation of the K-best sphere decoding for MIMO detection"*, IEEE Journal on Selected Areas in Communications, Volume 24, Issue 3, March 2006, pp 491-503.

[3] L. Barbero and J. Thompson, *"Rapid Prototyping of a Fixed-Throughput Sphere Decoder for MIMO Systems"*, in IEEE International Conference on Communications (ICC '06), Istanbul, Jun. 2006.

[4] Burg, A.,et al., *"VLSI implementation of MIMO detection using the sphere decoding algorithm"*, IEEE Journal Solid State Circuits, vol.40, pp 1566-1577, July 2005.

[5] Bhagawat,P., Ekambavanan,S., Das,S., Choi,G., Khatri.S, *"VLSI Implementation of a Staggered Sphere Decoder Design for MIMO Detection"*, Forty-Fifth Annual Allerton Conference, September 26-28, 2007, University of Illinois at Urbana-Champaign, IL, USA.

[6] Wang,H., et al., *" Managing dynamic reconfiguration on MIMO Decoder"*, Parallel and Distributed Processing Symposium,26-30 March 2007.

[7] Wang,H., Leray,P., Palicot, J., *"A Reconfigurable Architecture for MIMO Square Root Decoder"*, Lecture Notes in Computer Science, Reconfigurable Computing: Architectures and Applications,August 03, 2006.

[8] Huang,X., Liang,C., Ma, J., *"System Architecture and Implementation of MIMO Sphere Decoders on FPGA"*, IEEE Transactions on Very Large Scale Integration (VLSI) Systems, Vol 16, No.2, pp. 188-197, Jan.2008.

[9] Shariat-Yazdi, R.Kwasniewski, T., *"Challenges in the Design of Next Generation WLAN Terminals"*, Canadian Conference on Electrical and Computer Engineering(CCECE), pp. 1483-1486,April.2007.

[10] Bhagawat,P., Dash,R., Choi, G., *"Architecture for Reconfigurable MIMO detector and its FPGA Implementation"*, accepted for publication in 15th IEEE International Conference on Electronics, Circuits, and Systems, ICECS 2008.

[11] Michalke,C., Zimmermann,E., Fettweis, G., *"Linear Mimo Receivers vs. Tree Search Detection: A Performance Comparison Overview"*, IEEE 17th International Symposium on Personal, Indoor and Mobile Radio Communications (PIMRC),pp.1-7, Sept. 2006.

[12] Hochwald,B. M., TenBrink,S., *"Achieving Near-Capacity on a Multiple-Antenna Channel"*, IEEE Trans. on Commun., 51:389399, Mar. 2003.

[13] Chen,S., Zhang,T., Xin, Y., *"Relaxed K-best MIMO Signal Detector Design and VLSI Implementation"*, IEEE Transactions on Very Large Scale Integration (VLSI) Systems, vol. 15, issue 3, pp. 328-337, March 2007

[14] Siti, M., Fitz, M.P., *"A Novel Soft-Output Layered Orthogonal Lattice Detector for Multiple Antenna Communications"*, IEEE International Conference Communications, 2006. ICC '06.

[15] Wang, R., Giannakis, G., *"Approaching MIMO channel capacity with reduced-complexity soft sphere decoding,"*, in Proc. of IEEE Wireless Communications and Networking Conf. (WCNC), vol. 3, Mar. 2004, pp.16201625.

[16] J. Stine, et al., *"FreePDK: An Open-Source Variation-Aware Design Kit."*, Proceedings of the 2007 IEEE International Conference on Microelectronic Systems Education, 2007.