VLSI IMPLEMENTATION OF LOW POWER RECONFIGURABLE MIMO DETECTOR

A Thesis

by

RAJBALLAV DASH

Submitted to the Office of Graduate Studies of
Texas A&M University
in partial fulfillment of the requirements for the degree of

MASTER OF SCIENCE

December 2009

Major Subject: Computer Engineering

VLSI IMPLEMENTATION OF LOW POWER RECONFIGURABLE MIMO DETECTOR

A Thesis

by

RAJBALLAV DASH

Submitted to the Office of Graduate Studies of
Texas A&M University
in partial fulfillment of the requirements for the degree of

MASTER OF SCIENCE

Approved by:

Chair of Committee,	Gwan S. Choi
Committee Members,	Peng Li
	Duncan M.H. Walker
Head of Department,	Costas N. Georghiades

December 2009

Major Subject: Computer Engineering

ABSTRACT

VLSI Implementation of Low Power Reconfigurable MIMO Detector. (December 2009)

Rajballav Dash, B.Tech., National Institute of Technology Rourkela

Chair of Advisory Committee: Dr. Gwan S. Choi

Multiple Input Multiple Output (MIMO) systems are a key technology for next generation high speed wireless communication standards like 802.11n, WiMax etc. MIMO enables spatial multiplexing to increase channel bandwidth which requires the use of multiple antennas in the receiver and transmitter side. The increase in bandwidth comes at the cost of high silicon complexity of MIMO detectors which result, due to the intricate algorithms required for the separation of these spatially multiplexed streams.

Previous implementations of MIMO detector have mainly dealt with the issue of complexity reduction, latency minimization and throughput enhancement. Although, these detectors have successfully mapped algorithms to relatively simpler circuits but still, latency and throughput of these systems need further improvements to meet standard requirements. Additionally, most of these implementations don't deal with the requirements of reconfigurability of the detector to multiple modulation schemes and different antennae configurations. This necessary requirement provides another dimension to the implementation of MIMO detector and adds to the implementation complexity.

This thesis focuses on the efficient VLSI implementation of the MIMO detector with an emphasis on performance and re-configurability to different modulation schemes. MIMO decoding in our detector is based on the fixed sphere decoding algorithm which has been simplified for an effective VLSI implementation without considerably degrading the near optimal bit error rate performance. The regularity of the architecture makes it suitable for a highly parallel and pipelined implementation. The decoder has intrinsic traits for dynamic re-configurability to different modulation and encoding schemes. This detector architecture can be easily tuned for high/low performance requirements with slight degradation/improvement in Bit Error Rate (BER) depending on needs of the overlying application. Additionally, various architectural optimizations like pipelining, parallel processing, hardware scheduling, dynamic voltage and frequency scaling have been explored to improve the performance, energy requirements and re-configurability of the design.

Dedicated to family and friends

A special dedication to my newly born niece Ashwanita

ACKNOWLEDGEMENTS

First, I would like to thank my advisor, Dr. Gwan Choi for his encouragement and support during the duration of my research and my committee members, Dr. Peng Li, and Dr. Hank Walker for their feedback. A special thanks to my friend and colleague, Pankaj Bhagawat who provided valuable philosophical and technical inputs during the course of my research work. I also take this opportunity to thank Dr. Sunil Khatri whose words of advice have been a great influence.

Thanks to my family for their tremendous belief in me. A final note of thanks to all my friends at Texas A&M University and outside, who have kept me entertained and motivated during my stay here at the university.

TABLE OF CONTENTS

Page

LIST OF FIGURES

Page

LIST OF TABLES

1. INTRODUCTION

The need for ubiquitous communication has made wireless communication one of the most important segments for technological growth. The huge success of portable devices like cell phones, personal digital assistants (PDAs), smart phones, net books etc provide ample evidence to the importance of wireless communication in the consumer market. This has led to the need for evolving wireless standards in order to cater to the increasing demands of the consumer market. The wireless communication market has slowly transitioned from being predominantly used for voice communication to widespread surge in data usage, video downloads, and multimedia messaging which enhances the need of high bandwidth wireless links [1]. In order to cater to these high bandwidth applications, the wireless standards put in place a range of specifications which should be met in order to provide quality support to end users. Hence, the evolution of standards and systems is driven by the emergence of new applications which continue to require better quality of service (QoS) and higher data rates and by the need to support the growing number of users. This vast user base creates bandwidth limitations, affects service costs and also influences the QoS.

A number of factors affect the quality and capacity of wireless communications systems. The scarcity of available bandwidth for wireless communication combined with the increasing demand for higher data rates puts an ever increasing demand for high

---

This thesis follows the style of *IEEE Transactions on Very Large Scale Integration (VLSI) Systems*.

speed communication systems. As increasing the spectrum available for wireless communication is not a viable solution, therefore utilizing the spectrum optimally with increased spectral efficiency is the key. This impairment is countered by improved communication systems which use advanced algorithms and efficient VLSI architectures to offer higher data rates and the required QoS.

1.1 MIMO Systems: An Overview

Multiple-input multiple-output (MIMO) systems [2] use multiple antennas at both the transmitter and at the receiver to allow for spatial multiplexing to increase channel bandwidth. This technology is believed to enable the increase in channel bandwidth efficiency in future generation wireless systems to cater to the growing number of users. MIMO Technology enables the use of higher number of antennas which essentially allows for higher spectral efficiency compared to single-input single-output (SISO) systems with single antenna at transmitter and receiver. The major advantages of MIMO are due to the following gain factors [3]:

1. Array gain: Multiple antenna at the receiver side helps in picking up a larger percentage of transmitted power from the transmitter which fundamentally increases the range of the communication system and helps in suppressing interference from other sources.
2. Diversity gain: Multiple receivers give us more information about the variations in the channel also called fading. This information allows us to better isolate signal component from noise thus increasing link-reliability and QoS.

3. Multiplexing gain: With multiple antennas at transmitter and receiver, multiplexed data streams can be transmitted together which allows for a linear increase in spectral efficiency and peak data rates in the same frequency band. The number of multiplexed data streams is limited by the number of antennas at the receiver and transmitter side.

A tradeoff exists between the above mentioned gains, as maximizing each of them requires different transmission schemes. The ability of MIMO technology although with the aforementioned tradeoffs, to provide these enormous gains has led to the use of MIMO in various wireless standards like UMTS, 3G, IEEE 802.11n WLAN and IEEE 802.16 WMAN among many others. Thus, MIMO technology theoretically offers significant increases in data throughput and link range without additional bandwidth or transmit power.

1.2 Implementation Challenge of MIMO Detector

Realizing the theoretical capacity boost of MIMO technology is only possible with the use of complex signal processing at the transmitter side & receiver side as compared to present day SISO systems. Spatially multiplexing the data streams in the transmit side requires more processing before the transmit stage in order to ease the decoding process. Also, a serial data stream needs to be pushed into a Serial Input Parallel Output (SIPO) buffer to be transmitted by multiple antennas. On the receiver side, these spatially multiplexed data streams needs to be separated by extremely complex signal processing techniques. Most of the MIMO detection techniques should cancel the interference of successive channels while keeping all the other advantages

associated with MIMO systems. The complexity of transmitter and receiver increases rapidly at a far higher rate than the increase in spectral efficiency with the increase in number of antennas. Sometimes owing to the complex signal processing required at the receiver and transmitter, the gains of linear spectral efficiency is more than countered by a higher than linear increase in decoder complexity, even with the most basic processing algorithms. In order to fully exploit the advantages of multi-antenna systems, algorithms with even higher complexity is needed. Implementing these complex algorithms in an efficient VLSI architecture on silicon is an extremely crucial design challenge. However, for the successful implementation and widespread use of MIMO systems highly integrated and affordable implementations of the MIMO detector is of paramount importance.

1.3 Current Generation MIMO Systems

MIMO detector is the most complex unit of the MIMO Communication system. When spatial multiplexing is used at the transmit side, MIMO detectors task is to separate the spatially multiplexed data streams at the receiver side. Earlier most of the comparisons of MIMO detectors used to be on the basis of the complexity analysis of the signal processing algorithms. This complexity analysis does not relate to the silicon complexity of the decoder itself although they are a good metric for analyzing the complexity of the decoding process. These analysis show that MIMO systems from 2~6 antennas are practical from communication standpoint. Hence, case analysis with successively increasing number of antennas in communication system doesn't give us any significant information which would help in understanding feasibility of MIMO

implementation. Another method used to do analysis and algorithm optimization for complexity reduction was with the use of digital signal processor (DSP) which has custom functions for signal processing applications. However, the throughput achieved through DSP implementations along with implementations on other software programmable processing architectures is not sufficient to meet the requirements of next generation wireless standards owing to computational overheads associated with programmable logic. The analysis of algorithm efficiency done using programmable logic proved to be considerably different from dedicated communication circuits. For this reason, many algorithms which promised to be low power and computationally efficient in DSP architectures turned out to be ill suited for application specific integrated circuits (ASICs). Hence, a need for dedicated VLSI architectures for MIMO decoding was necessary for successful and efficient implementation. Recently, there has been quite some work relating to the actual VLSI implementations of MIMO algorithms and of complete MIMO systems. The few presented algorithms and designs provide initial reference points for the silicon complexity of MIMO detectors and illustrate suitable hardware architectures. Still till date, implementations of MIMO systems are band limited due to decoder performance rather than by wireless channel capacity. The authors, provide a comprehensive comparison of the true silicon complexity of different detection schemes and the associated performance tradeoffs and VLSI architectures based on actual VLSI implementations. VLSI implementations of various signal processing algorithms for MIMO detection in [4], further deals with the exploration of the design space that is available on the algorithmic and architectural level for the ASIC

implementation of low-complexity hard-decision MIMO detection for spatial multiplexing. A hardware implementation perspective was presented and results for various algorithms were compared. Here is a brief summary of the VLSI implementations of the following algorithms:

1. Linear and Successive Interference Cancellation (SIC) Detection: With proper implementation strategies SIC algorithms are less costly to implement in terms of silicon area than completely linear detectors. In [4] and [5], a linear detection architecture is presented which achieves nearly hundred percent hardware utilization, low decoding latency and higher throughput. Optimizations for matrix inversion and matrix decomposition are considered to enhance the performance of these architectures. In particular different architectural and circuit-level tradeoffs are discussed for the implementation of QR decomposition.
2. Exhaustive Search Maximum Likelihood: It is shown how this algorithm which achieves optimum bit error rate performance, but with a complexity that grows exponentially in rate, can still be implemented economically for higher throughput [6]. The reasons for this are a number of lossless (in terms of bit error rate) algebraic transformations and an optimized VLSI architecture.
3. Iterative tree-search algorithms: Tree-search algorithms mostly refer to Sphere Decoding and K-Best decoding in terms of MIMO detection although other less known search strategies are prevalent as well. VLSI implementation of the K-Best algorithm is described for a 4 × 4 system with 16-QAM modulation which achieves a high throughput compared to other implementations [7]. Additionally,

in [8] a one-node-per-cycle VLSI implementation of Sphere Decoding is presented. The implementation is shown to operate directly on complex-valued constellation points without the use of costly transcendental functions in [9]. A new modified-norm implementation is introduced in [10] easing computational complexity in norm computation.

In the past, most of these implementations dealt on the issue of simplifying signal processing algorithms for reducing computational complexity, achieving higher throughput and reducing latency of MIMO systems. Although these are important metric for commercial use, but another important factor is the usability of the architecture over different modulation schemes, across different antenna configurations, support for soft output detection which enables its use with FEC decoders for better BER performance.

1.4 Contributions

In this thesis, we explore the design space that is available on the architectural level for the ASIC implementation of low-complexity hard decision MIMO detection for spatial multiplexing. Algorithm was chosen on the basis of re-configurability to various modulation schemes, hardware reuse, enhanced throughput and lower silicon complexity. VLSI architecture has been developed for an iterative tree search decoder based on a special case of fixed sphere decoding algorithm for MIMO detection called COSIC algorithm. Various architectural optimization techniques and transformation was explored to improve the performance of the implementation. Performance improvement in terms of silicon complexity, operating power and throughput were considered. The VLSI implementation results are insightful from an implementation standpoint as they

provide the right metric in deciding the impact of the underlying algorithm and the associated architectural transformations used for design optimization. Here is a summary of all the contributions in this thesis:

1. **Dynamic Reconfigurability in MIMO Detection:** Most of the MIMO detector implementations don't deal with the issue of adaptive modulation schemes supported by numerous wireless applications and standards. The proposed detector architecture [11] uses a modified form of FSD algorithm which ensures constant throughput for a particular modulation scheme. Finite state machine (FSM) based control logic has been developed around the FSD implementation which uses modulation data to reconfigure the decoder to various modulation schemes like QPSK, 16-QAM and 64-QAM modulation schemes for 4x4 MIMO systems. The detector architecture can be further fine-grained pipelined to achieve higher throughput [12] without any scheduling complexity. The proposed architecture is highly suitable for the next generation wireless standards because of its flexibility, reduced computational complexity and higher throughput.
2. **Design Space Exploration of Runtime Reconfigurable MIMO Detector for IEEE 802.11n:** The focus here is on wireless systems based on 802.11n standard. In particular; extensive architectural space exploration was done to address the issues of power consumption, area, and re-configurability between different modes of operation while meeting the standards throughput requirement. Ultimately, two optimized designs [13] that target low area and low

power respectively was proposed. This detector will also support on the fly re-configurability for QPSK, 16-QAM and 64-QAM modulation schemes. This architecture delivers close to optimal Maximum Likelihood (ML) BER performance with no reconfiguration latency, leading to uninterrupted detection of MIMO symbols. This will clearly present an example of the tradeoff limits of the design and the extent of tune-ability of the architecture based on various target applications.

3. **Low Power Reconfigurable MIMO Detector for Real-Time Mobile Applications:** In this design we target two major issues: lower energy consumption based on a DVFS to exploit the re-configurability of the architecture to different modulation schemes with lower complexity and fixed throughput of detector across different modulation schemes which is ideal for real-time multi-media applications. The decoder uses optimal voltage and frequency while processing buffered data frames resulting in significant energy gains suitable for portable devices. This technique of varying voltage and frequency is called dynamic voltage and frequency scaling (DVFS) [14]. The DVFS frequency controller calculates the number of bits and decoding iterations required to process frames, and based on this information sets the operating frequency of the detector. A voltage regulator [15] is used to generate the appropriate voltage for that particular frequency. The choice is made in such a way so as to decode each frame, within a fixed time period irrespective of modulation scheme. Thereby, making available the output of each frame

synchronized to the fixed rate at which the data is consumed in real-time application interface like 802.11n. This detector delivers quasi-optimal BER performance with no reconfiguration latency which guarantees the necessary Quality of Service (QoS) with uninterrupted processing of MIMO symbols.

1.5 Thesis Outline

In section 2 of this thesis, the MIMO system model is described. The section also lists the performance criteria which constitute the basis for the development and evaluation of algorithms and VLSI architectures. It also introduces the available algorithm choices for MIMO detection, together with their corresponding complexity scaling behavior. We also discuss the reasoning behind the choice of algorithm for our implementations. In section 3, the implementation of the reconfigurable sphere decoder is discussed. Details are presented as to how the COSIC algorithm is modified for re-configurability to various modulation schemes. In section 4, we see an architectural exploration of the hard detector aimed at IEEE 802.11n. We explore aspects such as parallel processing and pipelining such that an optimal detector based on throughput, area and power can be designed. Here we also present two designs, one optimized for power and other optimized for area. Section 5 deals with a scheduling algorithm presented for the reconfigurable detector such that it can use voltage and frequency optimally such that the working power envelope can be further squeezed. Section 6 wraps up the thesis with discussions on possible future work in MIMO, summary and conclusions.

2. BACKGROUND OF MIMO DETECTION

This section will provide a detailed background of MIMO system model, processing stages, and the various algorithms used for MIMO detection. The choice of algorithm is very important for any VLSI implementation. This section will provide a brief summary of the advantages and disadvantages of each algorithm and its associated implementation.

2.1 System Model and Processing Stages

Wideband MIMO communication systems can be reduced to a set of narrowband MIMO systems with proper modulation techniques such as OFDM or with proper equalization. As narrowband system model are much simpler, therefore it is straightforward to derive corresponding receivers for wideband MIMO communication systems based on the narrowband model. Also, the results and analysis derived from narrowband system model can be easily extendible to a wide range of communication scenarios and to provide a common basis for the comparison of different algorithms. Let us say the number of transmit antennas is given by M_T and the number of receive antennas is given by M_R. The example MIMO system shown in Figure 2.1 has M_T=2 and M_R=2. This system model can be extended for higher number of antennas. As we are only concerned about spatial multiplexing to enhance channel efficiency, we also assume $M_R \geq M_T$. Here 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, Convolution 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 (it is assumed that the gains h_{11}; h_{12} etc are known at the receiver through the channel estimation stage which happens during preprocessing). These bits are then fed to the FEC decoder to get back the bits generated by the binary source.

Transmitter: With spatial multiplexing, the modulation in the transmitter corresponds to choosing the entries of the transmitted signal vector **s** independently from a set of constellation points Ω as shown in Figure 2.2, according to the data to be transmitted, so that $\mathbf{s} \in \Omega^{M_T}$. The set Ω is defined by the modulation scheme for which a rectangular QAM modulation with $Q = |\Omega|$ bits per complex-valued scalar symbol and with Gray encoding is usually assumed. The rate of the corresponding MIMO system with M_T transmit antennas in spatial multiplexing mode is then given by $R = M_T \log_2 Q$ bits per channel use (bpcu). In all the sections discussed in this thesis, the corresponding constellation points are defined on an odd integer grid according to $\Omega = \{(1 + 2a) + j(1 + 2b)\}$ with $a, b \in Z$ as shown in Figure 2.2. For a fair comparison which is independent of the number of transmit antennas and of the modulation scheme, the signal vector **s** is normalized before transmission in such a way that the average transmitted power is one (i.e., $E\{||\mathbf{s}||^2\} = 1$).

MIMO Channel: The equivalent baseband model of the MIMO wireless channel that yields the M_R-dimensional received vector **y** is given by the following input-output relation

$$\mathbf{y} = \mathbf{Hs} + \mathbf{n}. \tag{2.1}$$

The M_R-dimensional vector **n** models the thermal noise as independent identically distributed (i.i.d.) circular symmetric (proper) complex Gaussian with zero mean and variance σ^2 per complex dimension ($E\{\mathbf{n}\mathbf{n}^H\} = \sigma^2\mathrm{I}$). The $M_R \times M_T$ dimensional matrix H represents the complex-valued channel gains between each transmit and each receive antenna as discussed previously. For the simulations in the following sections, an i.i.d. Rayleigh fading channel model without correlation is assumed. Hence, the entries of H are chosen independently as zero mean proper complex Gaussian random variables with variance one per complex dimension. The SNR is defined in accordance with [3] as the ratio between the total transmitted power, which has been normalized to one, and the variance of the thermal noise according to

$$\boldsymbol{SNR = 1/\sigma^2} \tag{2.2}$$

Receiver: The M_R antennas at the receiver pick up the received signal vector y. Taking into account that the variance of the channel gains have unit variance, the *average* received signal-to-noise ratio (over channel realizations) per received antenna is immediately given by the SNR. The task of the MIMO detector at the receiver is to obtain the best possible estimate of the transmitted signal vector **s** based on the received vector **y**. Coherent modulation which is assumed in this thesis also requires that the receiver is provided with an estimate $\hat{\mathrm{H}}$ of the channel H. Such an estimate is usually

obtained during a separate training phase during preprocessing. Hence, broadly speaking MIMO processing can be split into two stages channel-rate and symbol-rate processing as illustrated in Figure 2.3. *Channel-rate processing* is often also referred to as *preprocessing*. The term comprises all operations that need to be carried out only when the channel estimate changes which happens when the system changes from one environment to another. *Symbol-rate processing* comprises all those operations that need to be carried out for each received symbol in order to estimate the transmitted vector symbol. We shall refer to this part of the receiver as the *detection unit*. Under low mobility scenarios when the channel doesn't change much, it is safe to assume that the channel remains same over a large number of symbols and hence, plays a less critical role in determining throughput of a general MIMO communication system. However, in high-mobility scenarios, under stringent latency constraints, or in wide-band MIMO

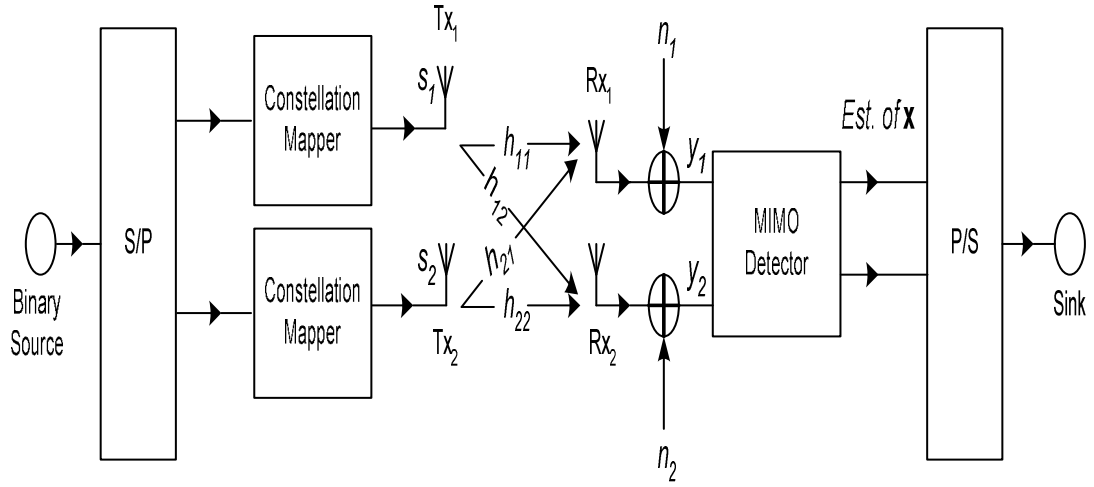

Figure 2.1: Example of a 2x2 MIMO Communication System with Channel Matrix

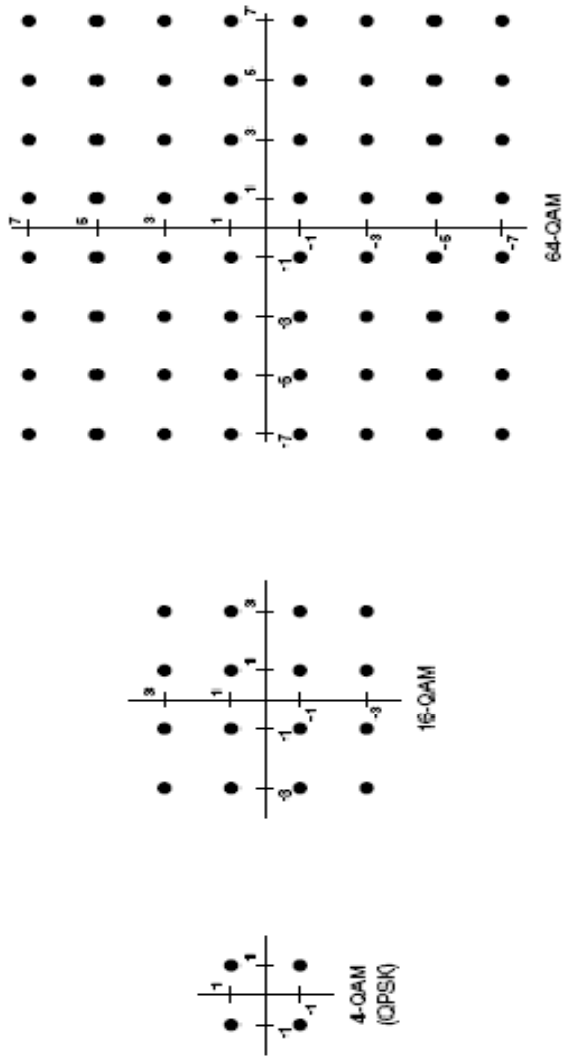

Figure 2.2: Constellation Point for QPSK, 16-QAM, 64-QAM Modulation Schemes

systems with frequency selective fading it is still justified, to consider the channel-rate processing complexity separate from the symbol-rate processing, as the frequency of the operation and the performance requirements are dictated by a completely different set of system parameters.

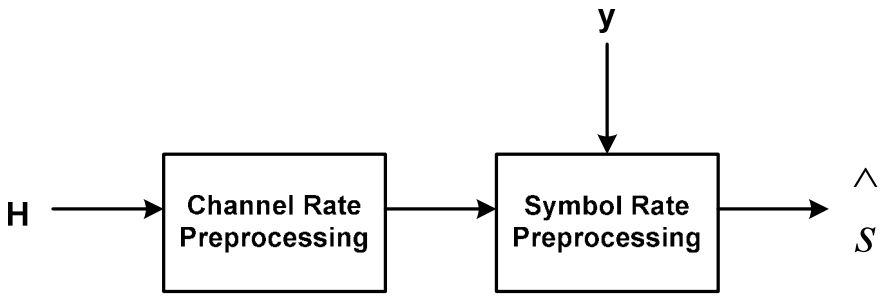

Figure 2.3: Processing Stages in General MIMO Detector

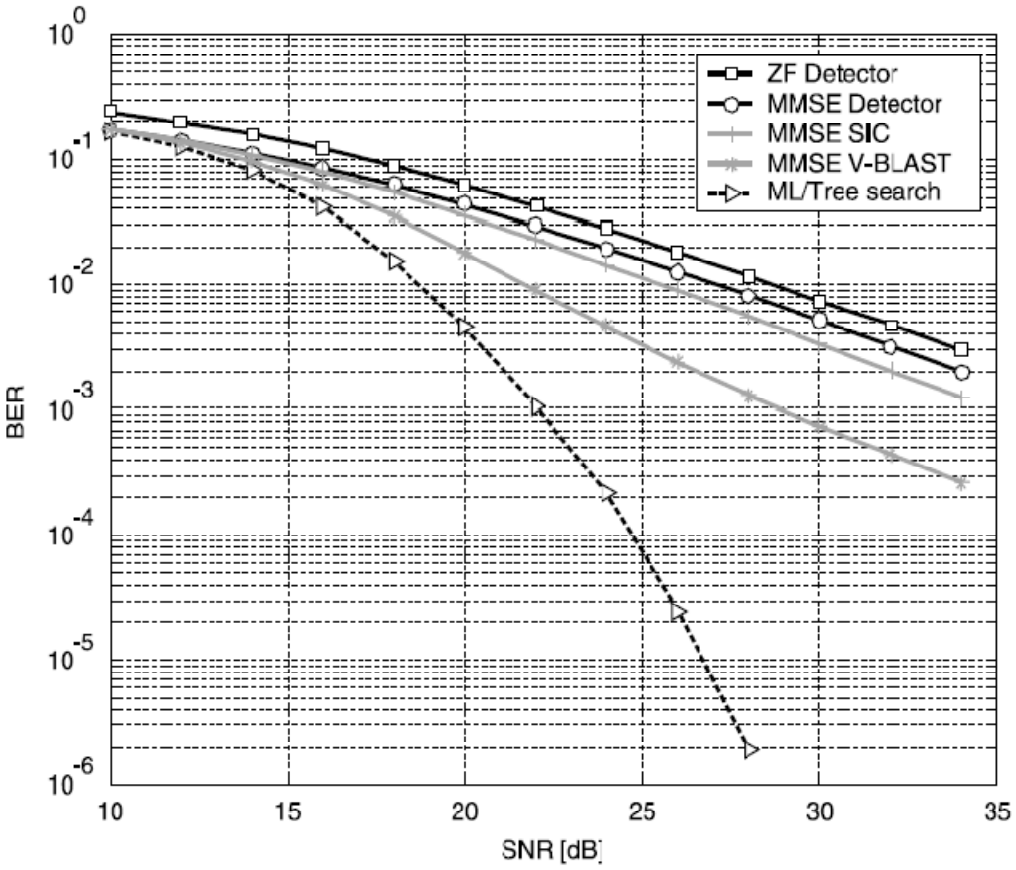

Figure 2.4: BER Performance of MIMO Detection Algorithms [4]

2.2 Maximum-Likelihood Detection and Sphere Decoding

As discussed earlier, there are many MIMO detection algorithms with different tradeoffs intrinsically embedded in each of them. A comparative study of the BER performance of

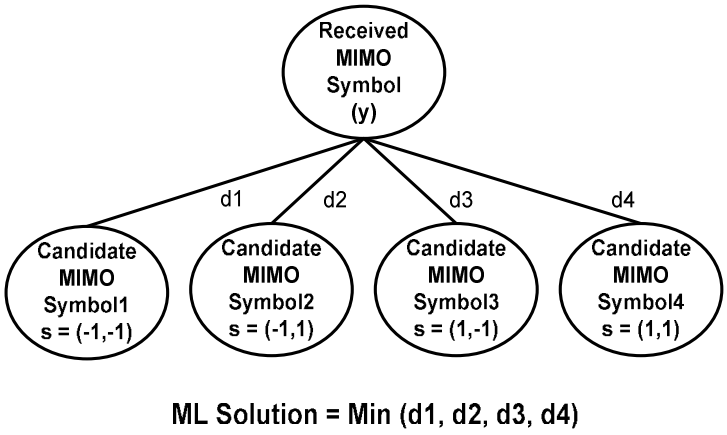

Figure 2.5: Example of ML Solution for 2x2 BPSK Systems

different algorithms is given in Figure 2.4. It shows a significantly higher BER performance of the Maximum-Likelihood (ML) detection algorithm than linear detection (ZF, MMSE) and Successive Interference Cancellation (SIC) algorithms. Along with its higher BER performance, ML detection algorithms as we will see in this section can be simplified for effective VLSI implementation.

The baseband system model for a MIMO system with M_T transmit and M_R receive antennas can be expressed as in (2.1) where **s** is M_T x 1 transmitted vector or vector symbol, **n** is M_R x 1 zero mean complex Gaussian noise vector, and H is M_T x M_R dimensional complex matrix. The $(i, j)^{th}$ element, h_{ij} , of the matrix H denotes the

complex channel gain from the j^{th} transmit antenna to the i^{th} receive antenna. In all our discussions, we assume $M_T = M_R = 4$ unless specified otherwise. The objective of ML detection is to search for a MIMO symbol over the entire set of possible MIMO symbols and find a set with minimum distance which in hard detection MIMO is taken as the most likely solution candidate. Therefore the objective of ML based MIMO detection to compute an estimate $\hat{s}$ of s such that:

$$\hat{\mathbf{s}} = \arg\min_{\mathbf{s}\,\varepsilon\,\Omega^{M_T}} ||\mathbf{y} - \mathbf{Hs}||^2$$

$$= \arg\min_{\mathbf{s}\,\varepsilon\,\Omega^{M_T}} \mathbf{d(s)} \text{ with } \mathbf{d(s)} = || \mathbf{y\text{-}Hs} ||^2 \qquad \textbf{(2.3)}$$

where Ω is set of complex entries from the QAM constellation as shown in Figure 2.2 and η is the cardinality of the set. As previously discussed straightforward approach to solving (2.3) is an exhaustive search over all possible candidate vector symbols as shown in Figure 2.5. However, since the number of possible solutions grows exponentially with M_T, the implementation of an exhaustive search becomes impractical as M_T increases. For example, in case of a 4x4 MIMO system with 16-QAM modulation an exhaustive search would require the evaluation of 65536 candidate vector symbols. As can be seen in [4], ML detection can be mapped to an Iterative Tree Search Problem which eases the search process. In each level of the tree, we compute a Partial Distance (PD) of each symbol from the set of QAM symbols in which we are searching. In this way, for 4x4 MIMO system we will have a tree of depth equal to four as we will have to detect four symbols at the receiver. The breadth of this tree depends on the value η for a η-ary modulation scheme. For modulation schemes which have more spectral efficiency like 16-QAM and 64-QAM we have to search for the ML solution in a tree of depth 4

and breadth 16, 64 respectively which is highly inefficient for hardware implementation. One way to circumvent the exhaustive search is to evaluate only a small subset of all the possible vectors. The objective of sphere decoding is to prune this tree with a radius update technique (as shown in Figure 2.6) such that the BER performance is not degraded but less number of nodes are processed. For effectively searching through this tree we need QR based preprocessing. This technique transforms the channel matrix (H) into a unitary matrix (Q) and upper triangular matrix (R): $\mathbf{H = QR}$. Hence, the cost function given by (2.3) can be rewritten as,

$$\mathbf{d(s) = \| y - Hs \|^2 = \| \hat{y} - Rs \|^2, \text{ and } \hat{y} = Q^H y} \qquad (2.4)$$

where, R is an upper triangular matrix, and Q^H is the Hermitian of a unitary matrix Q. Vector $\hat{y}$ as defined by (2.4) is the unconstrained zero forcing solution. The fact that R is upper-triangular ensures that each term in the summation depends only on the current level decision, as well as the history of the path to reach that level in the tree. Note that distance in (2.4) can now be rewritten to form summation across each transmit antenna.

$$\mathbf{d_i(s^{(i)}) = d_{i+1}(s^{(i+1)}) + | e_i (s^{(i)}) |^2} \qquad (2.5)$$

$$\mathbf{| e_i (s^{(i)})|^2 = | c_{i+1}(s^{(i+1)}) - R_{ii} \cdot s_i |^2} \qquad (2.6)$$

$$\mathbf{c_{i+1}(s^{(i+1)}) = \hat{y}_i - \sum_{j=i+1}^{M_T} R_{ij} \cdot s_j} \qquad (2.7)$$

The quantity $d_i(s^{(i)})$ is called the cumulative metric. The quantity $| e_i(s^{(i)}) |^2$ is called the incremental metric. The vector in (2.4)-(2.6) denotes a partial vector symbol candidate. This term $d_i(s^{(i)})$ for $i > 1$ is called PD which was discussed earlier and Distance (D) for i=1. Because the PD's depend only on $s^{(i+1)}$, they can be associated with corresponding

nodes in η-ary tree with M_T levels. For a 4x4 system, the equation for computation of each PD can be seen in Figure 2.7. As we can see, for a particular level of the tree the

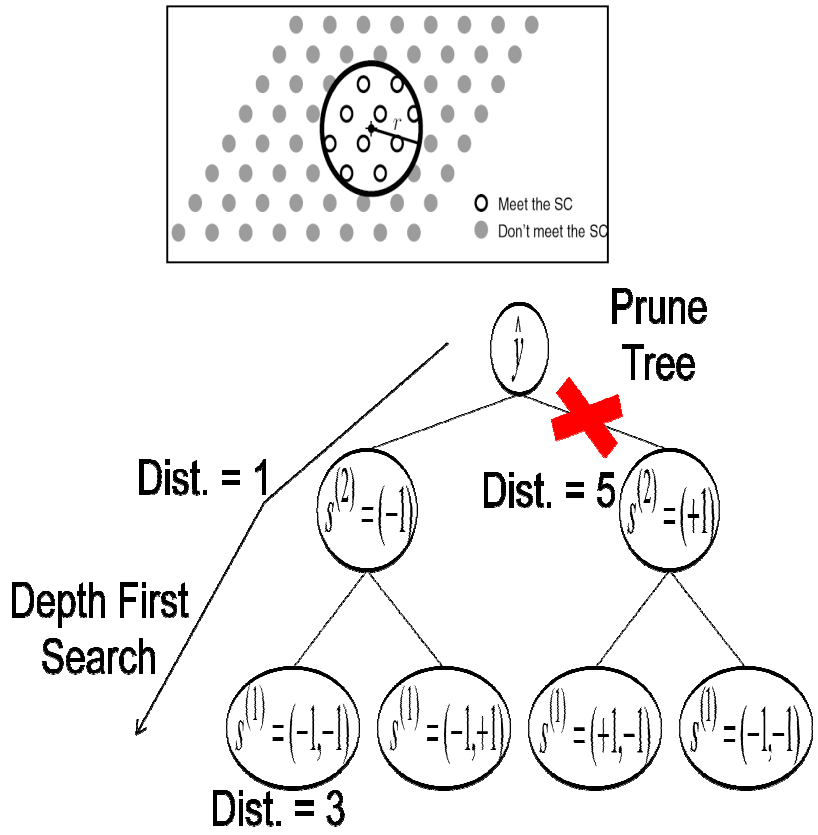

Figure 2.6: Example of Sphere Criterion and Tree Pruning

$$d_4\left(\mathbf{s}^{(4)}\right) = \left\|\hat{y}_4 - R_{44}.s_4\right\|^2$$

$$d_3\left(\mathbf{s}^{(3)}\right) = d_4\left(\mathbf{s}^{(4)}\right) + \left\|\hat{y}_3 - R_{34}.s_4 - R_{33}.s_3\right\|^2$$

$$d_2\left(\mathbf{s}^{(2)}\right) = d_3\left(\mathbf{s}^{(3)}\right) + \left\|\hat{y}_2 - R_{24}.s_4 - R_{23}.s_3 - R_{22}.s_2\right\|^2$$

$$d_1\left(\mathbf{s}^{(1)}\right) = d_2\left(\mathbf{s}^{(2)}\right) + \left\|\hat{y}_1 - R_{12}.s_2 - R_{13}.s_3 - R_{14}.s_4 - R_{11}.s_1\right\|^2$$

Figure 2.7: Computation of PD in Each Level of the SD Tree

computation of its PD depends on its own level and with the previously detected levels only. Also, noticeable is the fact that total distance of each leaf node (candidate MIMO symbol) is an additive sum of PDs at each level of the tree. Hence, PDs can be considered as an incremental distance at each level of the tree. Alternatively, the computation of the terms $d_i(s^{(i)})$ can be interpreted as a traversal of the tree from the root node to the leaf corresponding to **s**. Note that i = 1 correspond to leaf nodes. The estimate can now be obtained by searching the leaf with smallest D and returning the path from the top level ($i = M_T$) to that leaf node which will give $\hat{s}$. The PD's and D's in (2.4) are equivalently referred to as the node's metric in the sequel.

2.3 Analysis Criteria for VLSI Implementations

Once the system level aspects and requirements are understood, one can start with the development of low-complexity MIMO receivers. The available design space comprises of a variety of algorithms choices each of which provides opportunities for further optimizations on both algorithm and VLSI architecture level. At the same time, these choices and optimizations often entail tradeoffs between silicon area, throughput and BER performance which need to be balanced by the designer. Hence, joint consideration of both algorithm and implementation aspects are crucial for achieving efficient, low-complexity implementations.

BER Performance: The quality of a MIMO detection algorithm and of its associated implementation can be assessed by its BER performance which is obtained from fixed-point computer simulations as corresponding analytical expressions are often not available or do not include non-idealities caused by implementation tradeoffs.

Diversity Gain: Diversity gain describes the behavior of an algorithm in the limit of high SNR, and the diversity order corresponds directly to the slope of the BER curve.

Error Floor: In practical systems, the additive thermal noise term **n** in the channel model in equation (2.1) does not accurately model the overall noise in an end-to-end system. Instead, other noise sources whose power does not degrade with increasing SNR also contribute to the effective overall noise power. At high SNR, these constant terms become the dominant factors and the BER curve shows an error floor.

Complexity Order of Algorithm: The complexity order of the MIMO detection algorithm provides description of the scaling behavior of its complexity in one or more design parameters in the limit of infinity. A complexity order of $O(n2)$ for example specifies that the fastest growing term in the expression for the corresponding complexity is quadratic in n.

VLSI Implementation Complexity: The *computational complexity* describes the complexity of an algorithm in terms of number of costly operations. However, in practice, the notion of what kind of operation qualifies as costly differs widely depending on the underlying implementation technology. VLSI implementations allow for replacing sequences of basic operations by much more efficient single-cycle custom composite operations and additional hardware resources can be allocated for parallel execution of more frequent or more time consuming operations. Thus, it is important to identify the complexity defining operations with a basic VLSI architecture in mind and to count the associated efforts individually. Such careful counting of operations (with VLSI architecture and the associated memory requirement) provides reasonable means

for the comparison of similar algorithms which call for similar underlying architectures and for assessing the impact of corresponding optimizations.

Silicon Complexity: Unfortunately, even smart ways of counting the number of operations tend to fail, when comparing fundamentally different algorithms or when attempting to accurately predict the capabilities of a final VLSI implementation. Moreover, counting of operations does not immediately provide information about the design tradeoffs between throughput and silicon area, as data dependencies, memory access bottlenecks and other potential impairments are not captured. The true silicon (or implementation) complexity of an algorithm is given by the area and the throughput or delay that is achieved with a particular VLSI architecture.

2.4 Discussion of Implementation/Simulation Methodology

As mentioned previously, the BER results are obtained from computer simulations based on the MIMO channel shown in Figure 2.1. This model is valid in rich scattering environments with sufficient spacing between the antennas on the order of one wavelength. The simulation results presented in this thesis assume perfect channel knowledge at the receiver, effectively setting $\hat{H}$=H, so that channel estimation and detection can be separated. All the BER results in this thesis are generated for a 4×4 MIMO system with the simulation setup and scaling values of noise for specific SNRs computed as described in [17]. Different architectures are explored with support for various modulation schemes and ideas presented can be easily extrapolated for different antenna configurations. The main motivation behind choosing $M_T = 4$ is the fact that four antennas already provide a considerable capacity improvement that is likely to cover the

needs for next generation wireless systems. Moreover, from a practical perspective, mounting more than four antennas with an appropriate distance with approximately one wavelength apart on a portable device appears difficult. In terms of the modulation scheme, it is also important to note that practical systems will employ adaptive modulation, predominantly using QPSK to 16-QAM for outdoor scenarios and QPSK to 64-QAM for indoor scenarios due to which algorithmic/architectural re-configurability to different modulation schemes are an important design criterion. Fixed point simulations were considered for finding design tradeoff with respect to BER and complexity. All the simulations for the algorithmic level exploration were done in bit accurate and cycle accurate code in MATLAB which is a good approximation of the implementation scenario. Once the fixed point accuracy and tolerance was verified, RTL was written in Verilog HDL to describe the functionality of the MIMO detector. RTL was verified using the same test points that were used for verification of the simulation model in MATLAB. RTL synthesis was done in Synopsys Design Compiler and post-synthesis simulations using cell libraries from Nangate 45nm PDK was done in Verilog XL. Cadence Silicon Encounter was used for the automated Place and Route of the design. All the results presented in this thesis are based on back annotated delays from cell libraries and delays from wire models. For the FPGA implementation, Xilinx integrated development environment was used for synthesis, timing closure and place and route.

3. FPGA IMPLEMENTATION OF RECONFIGURABLE MIMO DETECTOR

In this section, a reconfigurable architecture for MIMO detection and its FPGA implementation is presented based on a variant of the Sphere Decoding Algorithm discussed in Section 2.2. The design objective is to be able to reconfigure on the fly which is one of the prime requirements for future wireless standard. The previously discussed algorithms/architectures in Section 1.3 are relatively expensive from a re-configurability viewpoint. From the few re-configurable MIMO detectors that have been reported till date, one implementation [18] uses VBLAST based detection scheme that incurs significant BER degradation and an expensive Processor based Control Unit. On the contrary, we use a variant of the Fixed-Throughput Sphere Decoding Algorithm (FSD) [19] with modifications which is provided in [20] for our implementation. Many of the challenges discussed in previous sections have been addressed. Also, it can be implemented in a highly parallel and pipelined manner, has fixed throughput for a given modulation scheme, delivers quasi-ML BER performance and achieves on the fly reconfiguration. This detector supports on the fly re-configurability for QPSK, 16-QAM and 64-QAM modulation schemes with quasi-ML performance. The control logic has minimal complexity and is highly integrated with the data flow.

3.1 Fixed Sphere Decoding and COSIC

From an implementation point of view, the sphere decoding has two main drawbacks. Firstly, the detector complexity depends on the noise level and the channel conditions and, secondly, the sequential nature of the search limits the performance and the level of parallelism of a hardware implementation of the algorithm. A new fixed-

complexity sphere decoder (FSD) [19] is proposed to overcome those two problems by searching, independently of the noise level, over only a fixed number of lattice points H.s (2.1), generated by a subset of symbols around the received symbol. The FSD assigns a fixed number of candidates, n_i, to be searched per level independent of the initial radius. This can be explained as follows: whereas in the first level, $i = M_T$, more candidates need to be considered due to interference from the other levels, the decision-feedback equalization (DFE) performed on y_i and the increase in value of diagonal in later level reduces the number of candidates that need to be considered in the last levels. The total number of candidates whose Euclidean distance is calculated is, therefore, $N_S = \sum_{i=1}^{M_T} n_i$, where simulations show that quasi-ML performance is achieved with $N_S << \eta^{M_T}$. The n_i candidates on each level i are selected according to increasing distance to y_i, following the SE enumeration [21]. Figure 3.1 shows a hypothetical subset *S* in a 4x4 system with 4-QAM modulation where the number of points per level $n_s = (n_1; n_2; n_3; n_4)^T = (1; 1; 2; 3)^T$. In each level i, the n_i closest points to y_i are considered as components of the subset *S*. A trade-off exists between the complexity and the performance of the FSD. If more candidates are searched, the performance will be closer to that of the original SD but the required computational power will increase. That makes the FSD suitable for reconfigurable architectures where the number of candidates can be made adaptive depending on the MIMO channel conditions. The main problem with FSD ordering is its iterative preprocessing stages to find out the number of branches that should be selected at each level which has large computational complexity.

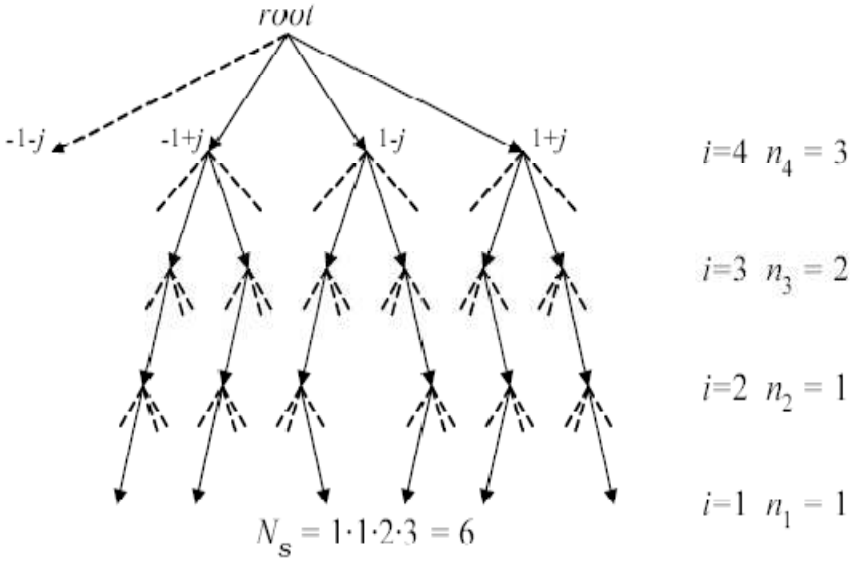

Figure 3.1: Example of (3, 2, 1, 1) FSD Ordering in [19]

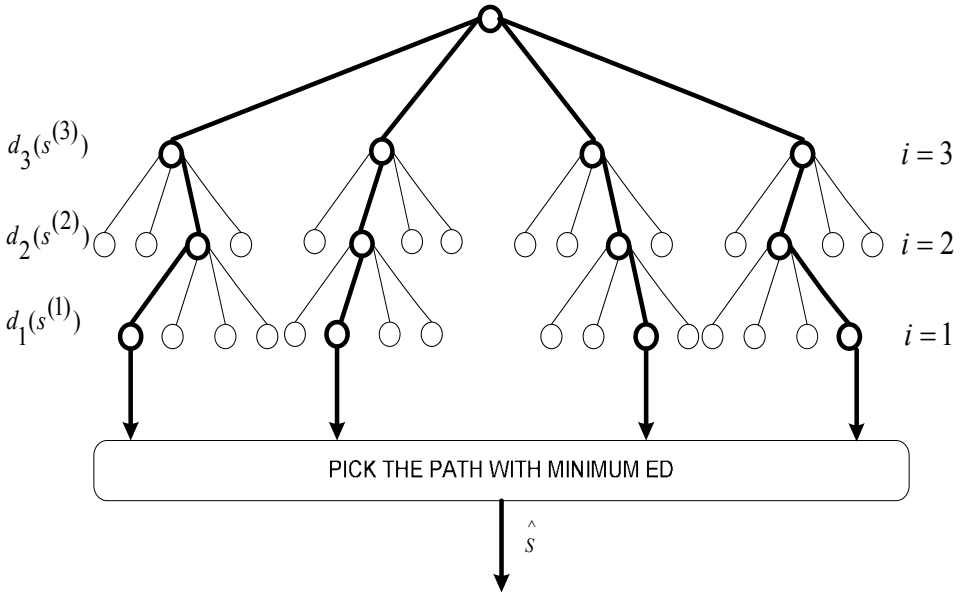

Figure 3.2: Tree Structure of COSIC Algorithm

This makes implementation of FSD tougher. In [20], a simple technique based on FSD is presented which performs near to Optimal in BER performance but has reduced complexity. Figure 3.2 shows the COSIC ordering technique in 4x4 system with QPSK modulation where the number of points per level $n_s = (n_1; n_2; n_3; n_4)^T = (1; 1; 1; 4)^T$. In COSIC, at the first stage η candidate solutions of the tree are considered and then in successive levels one the best solution candidate is considered as shown in Figure 3.2. This algorithm degrades slightly in BER performance as compared to the FSD algorithm but compensates in terms of reduced complexity which helps in VLSI implementation.

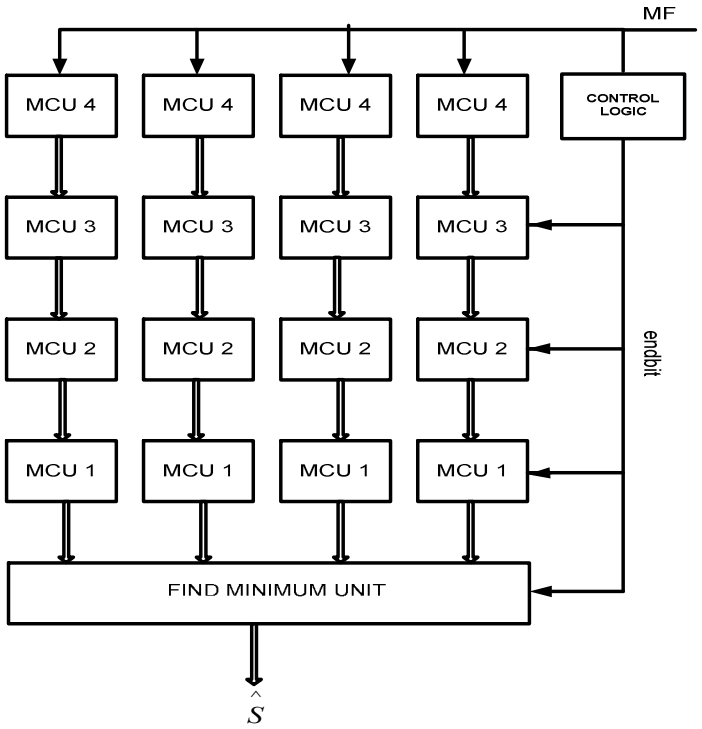

Figure 3.3: Data Path Parallelism and Control Structures

Simplified Norm Computation: The Euclidean norm or l^2 norm which needs to be computed at each level in (2.6) involves a squaring operation which requires multipliers. Multipliers are in general expensive in terms of hardware cost. In [9] it has been shown that the use of simplified norms leads to significant reduction in hardware cost with some BER degradation. In our design we have replaced the l^2 norm in (2.6) by l^1 norm. The l^1 norm approximation for is given by:

$$\mathbf{d_i(s^{(i)}) = d_{i+1}(s^{(i+1)})+| Re\{e_i(s^{(i)})\} | + | Im\{e_i(s^{(i)})\} |} \qquad \mathbf{(3.1)}$$

where Re{} and Im{} denote real and imaginary parts respectively. The use of l^1 norm causes the BER to degrade only by about 0.4dB-0.5dB [9].

3.2 Reconfigurable Sphere Decoder Architecture

Figure 3.3 shows the high level architecture of our decoder. The choice of 4-way parallelism was made because the smallest constellation supported on our decoder is QPSK (which has four symbols, η=4). If this architecture is pipelined with *m* stages then it has an initial latency of *m*+ η/4 clock cycles. Note that the FSD tree has η paths for a *η-ary* modulation scheme. Hence, the proposed architecture takes η/4 clock cycle to detect a *η-ary* modulated MIMO symbol. At each level of the FSD tree (Figure 3.2) we need to compute the $d_i(s^{(i)})$ metrics using (2.5)-(2.7). Each of these equations are computed by a Metric Computation Unit (MCU). The best child node at each level is picked using a slicing operation shown in Figure 3.4. Figure 3.5 shows the structure of the MCU at level-1 which is the lowest level in the tree. The upper box in the Figure 3.5 evaluates (2.7). Note that there is no need to implement the product terms in (2.7) using a multiplier. This product can be achieved by shift and add operation as shown in Figure

3.6, because the QAM constellation points only take on a finite number of integer values (e.g. in 16-QAM scheme the real and imaginary part of s_j ε {-3, -1, 1, 3}). The block named slicer as shown in Figure 3.7 picks the nearest QAM symbol to c_{i+1} as shown in Figure 3.4. The slicing operation involves independently comparing real and imaginary parts of c_{i+1} with appropriate decision thresholds. The decision thresholds are given by (-($\sqrt{\eta}$ - 2) + 2j) R_{ii}, where j is an integer such that 0≤ j≤ ($\sqrt{\eta}$ - 2)). Our decoder configures the slicer based on Modulation Format bits (MF), which indicates the modulation scheme of the current MIMO symbol. The control unit of our design is a simple FSM which takes in MF [1:0] (00 => QPSK, 01 => 16-QAM, and 10 => 64-QAM) and generates a signal '*Endbit*' every (η/4) clock cycle. This signal indicates the completion of decoding one MIMO symbol.

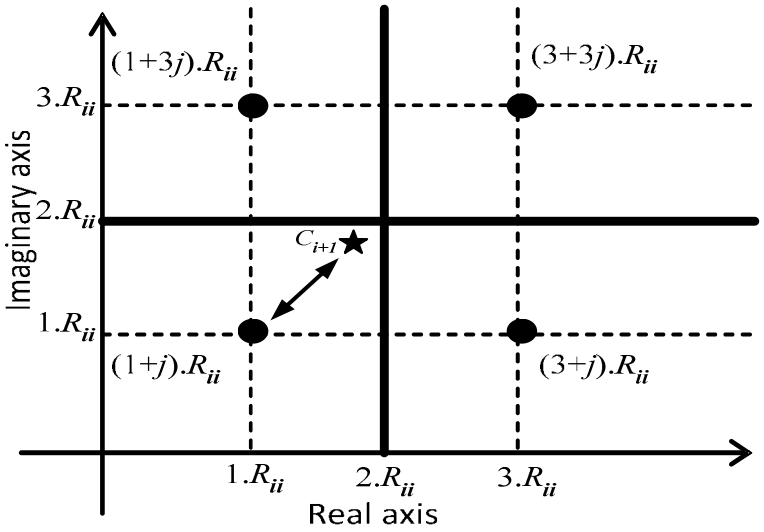

Figure 3.4: Example of Slicing Operation for 16-QAM

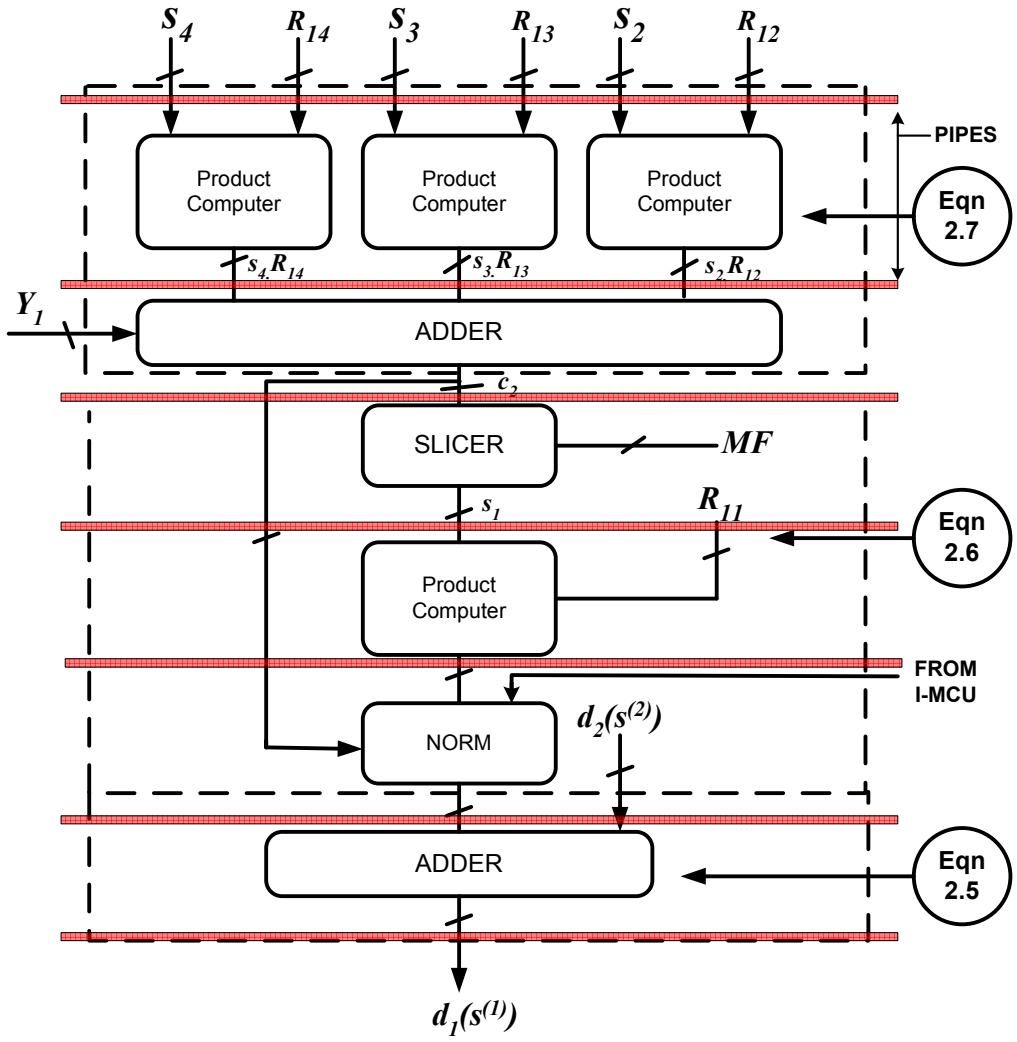

Figure 3.5: Metric Computation Unit of Level 1

The waveforms in Figure 3.7 show the relation of the control signals with respect to the MIMO symbol. The design of the control unit is independent of number of parallel processing units *m* and pipelines *k*, this implies that very little redesign effort is required

in case one wants to achieve very high throughput by increasing *m* (subject to latency constraint) as we will see in the next section.

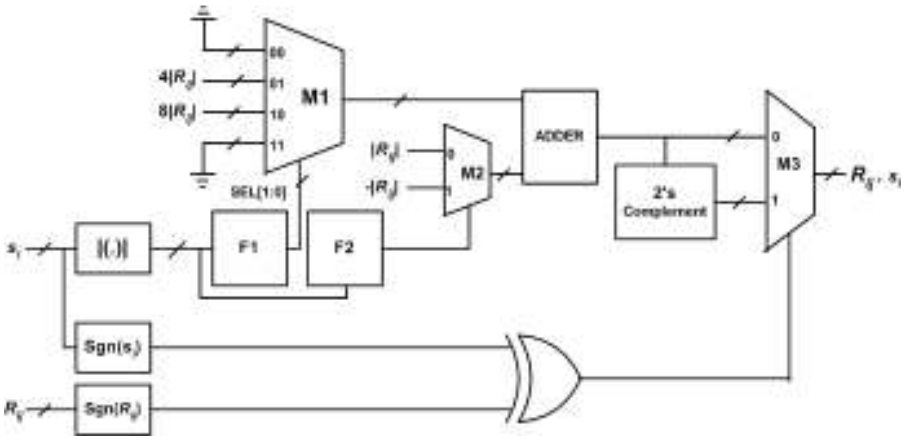

Figure 3.6: Product Computer Unit Using Shift and Add

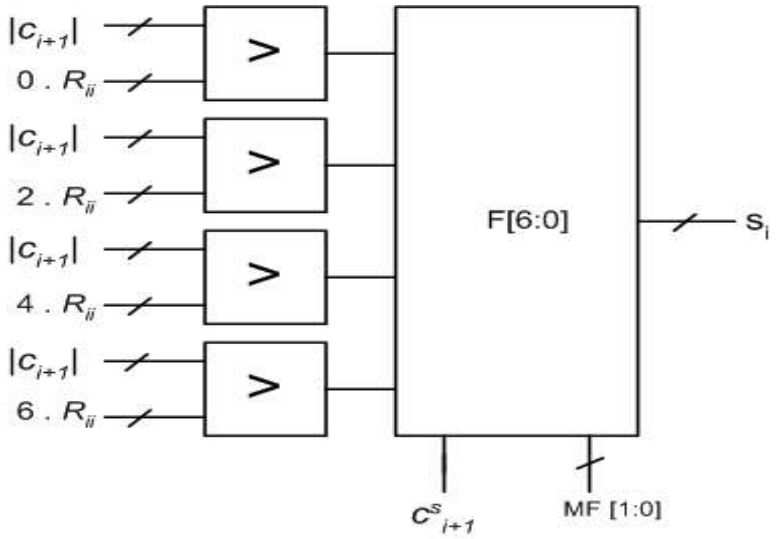

Figure 3.7: Slicer Logic which Performs Slicing Operation

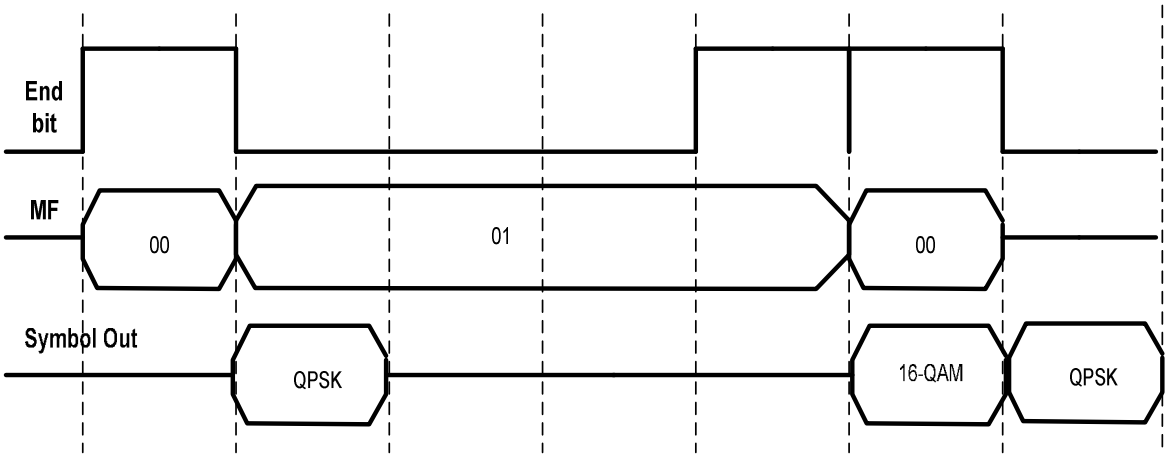

Figure 3.8: Output and Control Waveform

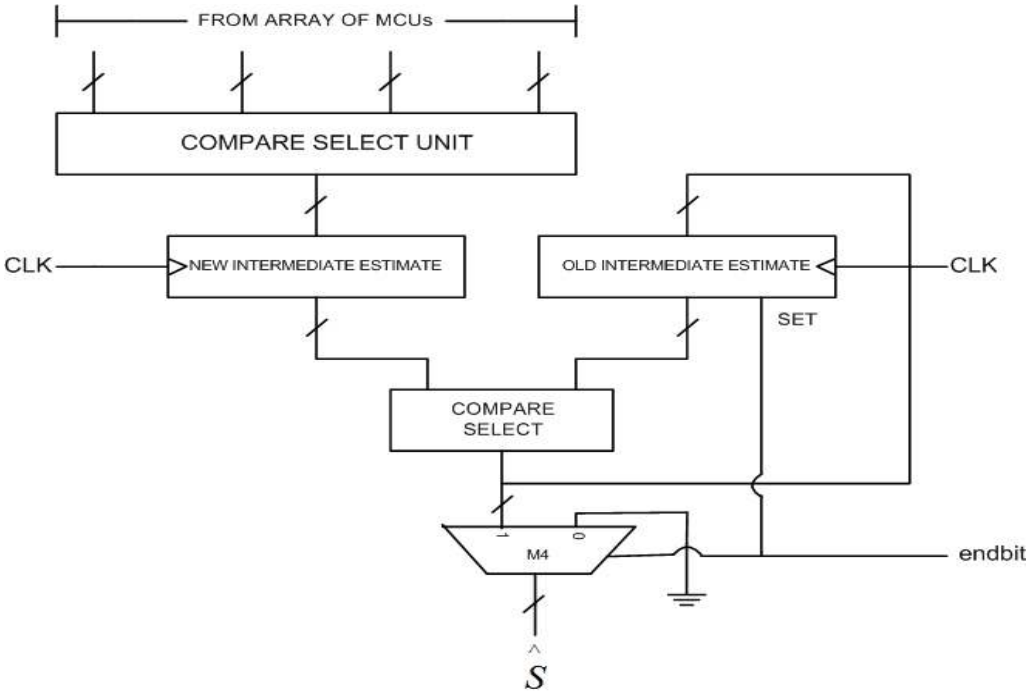

Figure 3.9: Find-Minimum Unit to Pick Minimum PD

The parallelism factor can also be increased (with corresponding changes in the control logic) to further increase the throughput. Figure 3.6 and Figure 3.7 shows the product computer and slicing unit used in our MCU. Figure 3.8 shows the underlying control signals generated to configure between different modulation schemes. Endbit is generated based on MF which is the detector input. Endbit is used to the minimum metric computed by the detector. Figure 3.9 shows the Find-Min unit which is required for hard decision of the MIMO symbol based on the computed distances.

TABLE 3.1

Comparison of Reconfigurable Architectures

Ref	QPSK	16-QAM	64-QAM	Dynamic ReconFigure	BER
[17]	No	Yes	No	No	Quasi-ML
[19]	No	Yes	No	No	Quasi-ML
[9]	No	Yes	No	No	ML
[16]	No	Yes	No	No	Quasi-ML
[22]	Yes	Yes	Yes	Yes	Sub-Optimal
[23]	No	Yes	No	No	Quasi-ML
[This]	Yes	Yes	Yes	Yes	Quasi-ML

3.3 FPGA Implementation Results

The known previous reconfigurable architectures are either non-dynamic [18] or uses on-chip processors [22] to achieve re-configurability which is unsuitable for ASIC

TABLE 3.2

FPGA Implementation Results

Target FPGA Device	xc4vfx60 (Xilinx Virtex-4)
Number of 4 Input LUTs	10745 (Utilization: 21%)
Number of Slice Flip flops	845 (Utilization: 1%)
Multipliers	None
Maximum Frequency	35 MHz
Max Decoding Rate: QPSK	280 Mbps
Max Decoding Rate: 16-QAM	140 Mbps
Max Decoding Rate: 64-QAM	52.5 Mbps
Total equivalent gate count	107, 458
Control Logic Overhead	0.3%

implementation. Moreover, the algorithm used in aforementioned designs degrades BER performance. In contrast, our decoder as shown in Table 3.1 supports on the fly reconfiguration, has a very simple control unit, with the control signals tightly integrated

with the datapath. More importantly, the proposed decoder delivers close to ML BER performance as shown in Figure 3.10. The decoding data flow is uninterrupted throughout the operation resulting in continuous detection. MATLAB was used to simulate bit accurate model of the decoder. We chose eleven bit fixed point quantization while maintaining internal precision) for negligible BER degradation. Based on this bit accurate MATLAB model, detail hardware was developed. RTL coding and synthesis

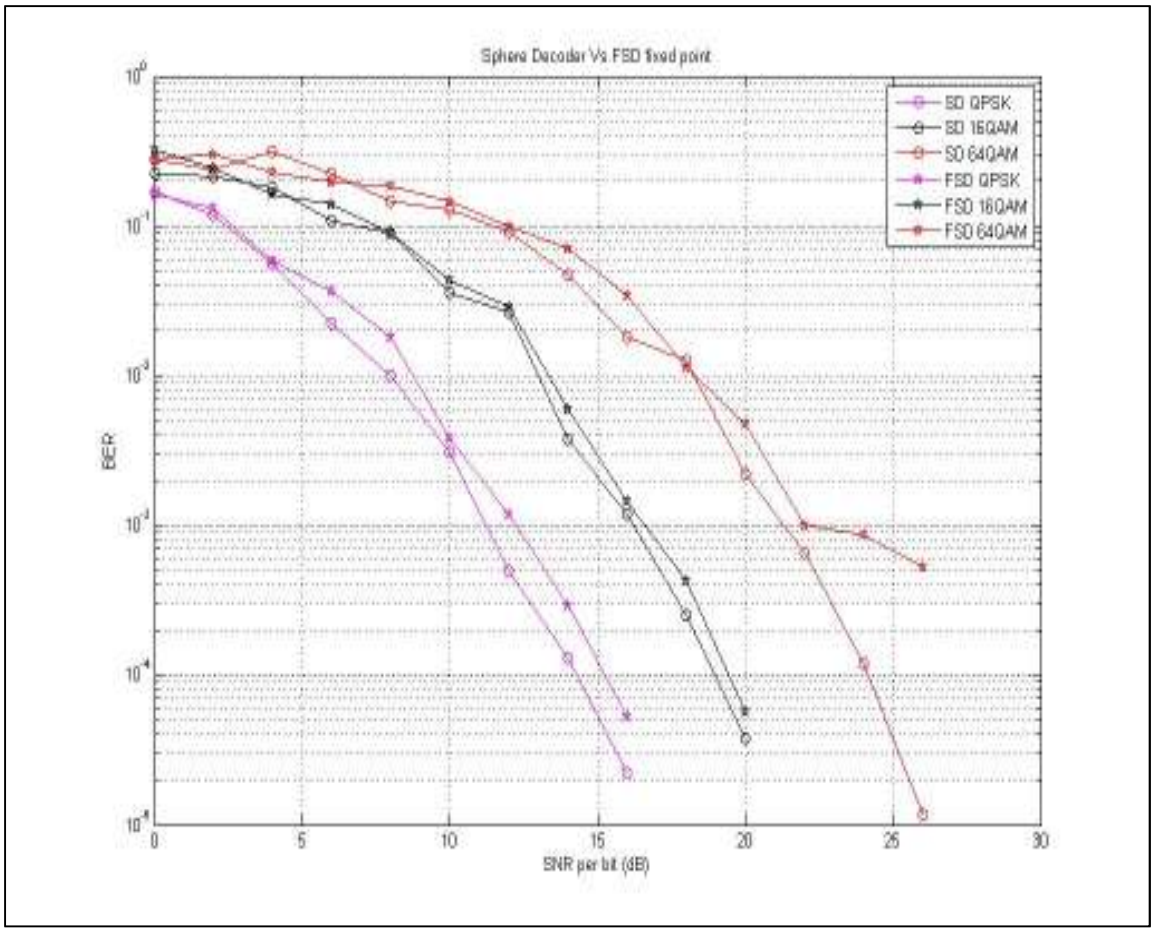

Figure 3.10: BER Performance Curve for SD (Floating Point/L2 Norm) vs. Implemented COSIC Algorithm with (Fixed-Point/L1 Norm)

was done using Verilog HDL and Xilinx ISE 8.1 Embedded Development Kit respectively. Xilinx Virtex-4 [xc4vfx60] device was used for mapping the synthesized netlist. Floor-planning, Place and Route (P&R) of the design was done using the integrated Xilinx Floor-planner and automatic P&R tool. The input test vectors were generated by the fixed-point MATLAB model. The hardware design was validated by carrying out simulations on these test vectors with the Post-P&R simulation model using ModelSim PE 6.3c. Table 3.2 shows the FPGA Implementation results for our architecture.

3.4 Summary of Results and Conclusion

The system architecture and FPGA implementation of a reconfigurable sphere decoder for MIMO detection is presented in this section. The detector is dynamically reconfigurable for QPSK, 16-QAM and 64-QAM modulation schemes for 4x4 MIMO systems. The decoder was implemented on a Xilinx [xc4vfx60] device. The decoder gives an un-coded throughput of 280 Mbps, 140 Mbps, and 52.5 Mbps for QPSK, 16-QAM, and 64-QAM respectively. The detector is further pipeline-able to achieve higher throughput. The proposed architecture is highly suitable for the next generation wireless standards because of its flexibility, reduced computational complexity and higher throughput.

4. ARCHITECTURAL SPACE EXPLORATION OF RECONFIGURABLE MIMO DETECTOR FOR IEEE 802.11n

The main objective of this section is to present an architectural exploration study done on the MIMO detector architecture that was developed in Section 3. The architecture described in the previous section can be parallelized and pipelined owing to its systolic-like nature. This section shows the results of exploration of this design space by searching through the tradeoff of pipelines and parallelism with the throughput for IEEE 802.11n providing the bounds for the operating clock frequency. As we saw in previous sections, the choice of algorithm and architecture has a significant bearing on the final hardware complexity and re-configurability. Besides the BER performance of an algorithm, our architecture focuses on issues like pipelining, parallelism, and re-configurability. The algorithm should be designed such that it lends to a highly pipelined and parallel architecture. Also, the algorithm/architecture should be amenable to dynamic reconfiguration with various modulation schemes which is a requirement of various wireless standards. Among the earliest algorithms used for signal detection for un-coded MIMO was VBLAST [24]. Although computationally efficient, VBLAST suffers from a substantial degradation of BER. Other approaches for MIMO detection as we discussed earlier are the Sphere Decoding (SD) algorithm [9], which is a Depth First Search (DFS) based algorithm, and the K-best algorithm which is a Breadth First Search (BFS) based algorithm [17]. SD algorithm provides optimal Maximum Likelihood (ML) BER performance, but it is unsuitable for parallel and pipelined implementation. Also SD algorithm converges to the optimal solution in random fashion [9] which makes it

unfit for practical systems. On the other hand, K-best algorithm provides constant throughput with quasi-optimal BER performance but involves sorting operation making it difficult to design a parallel and pipelined architecture. We use the Fixed-Throughput Sphere Decoding Algorithm (FSD) [19] with a modification called COSIC [20] algorithm for our implementation. It can be implemented in a highly parallel and pipelined manner, has fixed throughput (for a given modulation scheme) as seen in our implementation described in Section 3. FSD with COSIC achieves a quasi-optimal BER performance as can be seen from [20]. Here, it is also shown that the BER performance comparison for FSD and COSIC for 4x4 MIMO 64-QAM modulation scheme are quasi-optimal and close to optimal respectively. In this section, we focus on wireless systems specifically based on 802.11n standard. In particular we carry out extensive architectural space exploration to address the issues of power consumption, area, and re-configurability between different modes of operation while meeting the standards throughput requirement. Ultimately, we come up with two designs that target low area and low power respectively. Our detector supports on the fly re-configurability 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-ML BER performance with no reconfiguration latency, leading to uninterrupted detection of MIMO symbols.

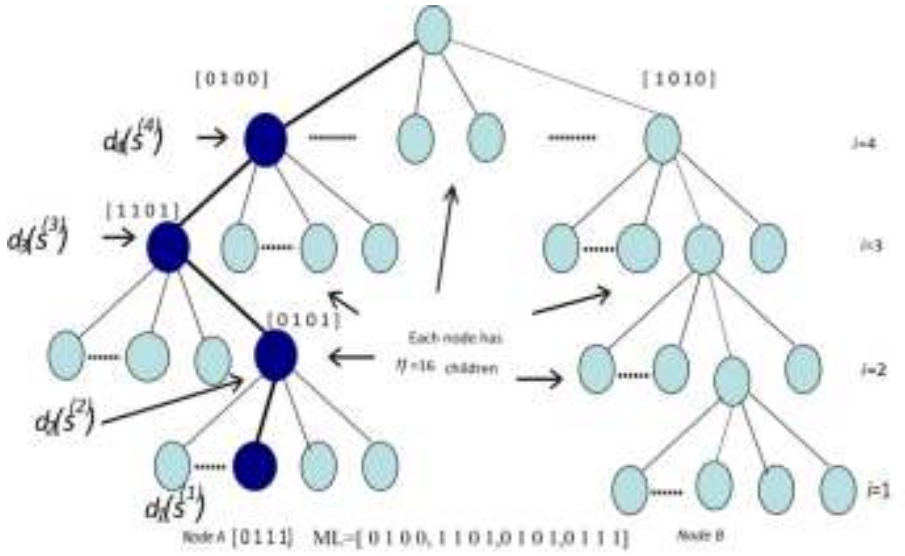

Figure 4.1: Tree Structure of FSD with COSIC Modification for 4x4 MIMO

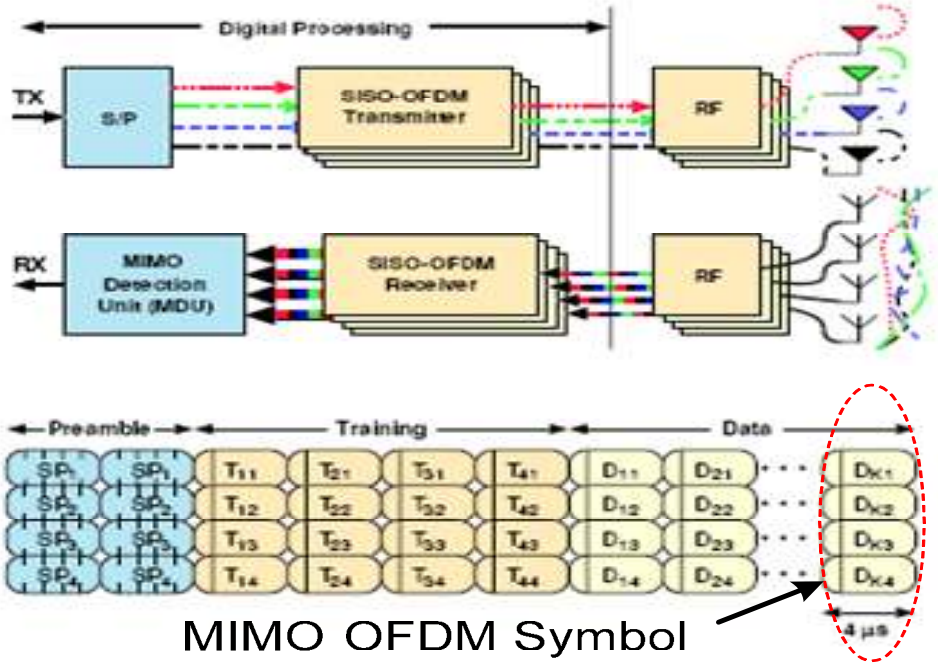

Figure 4.2: Packet Structure of IEEE 802.11n Systems

4.1 Architectural Flexibilities in FSD Implementation

As we saw in Section 3, the FSD algorithm with COSIC ordering maps the sphere decoding algorithm to a tree structure given in Figure 4.1. In COSIC, we consider η nodes in the first level and the best node in the next three levels of the FSD tree. MCU architecture for node processing in the last level (Level 1) was shown in Figure 3.5. The COSIC ordering technique if directly implemented then each of the branches of tree can map to η parallel processing MCUs which is a full parallel implementation of the COSIC algorithm. On the other extreme, if we reuse the same computing logic for computing each node of the tree then it is a complete sequential implementation of the same. The COSIC implementation in Section 3 has MCU units for each level of the tree. The depth of this logic can be pipelined owing to its systolic-like structure. Each added pipelining stage increases latency but also enhances the throughput of the detector. Hence, we can compensate for the loss of throughput in a sequential implementation by increasing the number of pipeline in our COSIC based architecture. The other constraint which is of importance is forced by the high level application e.g. IEEE 802.11n. As we will see later, the application enforces a hard constraint on the throughput required in the MIMO system. Hence, this section of the thesis describes methods in which we can exploit our architecture developed in Section 3 in terms of the number of pipelines/parallel processing nodes needed to meet the throughput requirements while providing optimal performance in terms of power and area. We show two optimized designs in this section. On design is optimized for low area and the other optimized for low power. Both our optimized designs meet applications throughput constraints.

4.2 High Level Architectural Space Exploration

This section begins with a brief discussion about the throughput limitations imposed by the 802.11n standard. We then develop a strategy to evaluate various architectural parameters that meet the requirements of the standard.

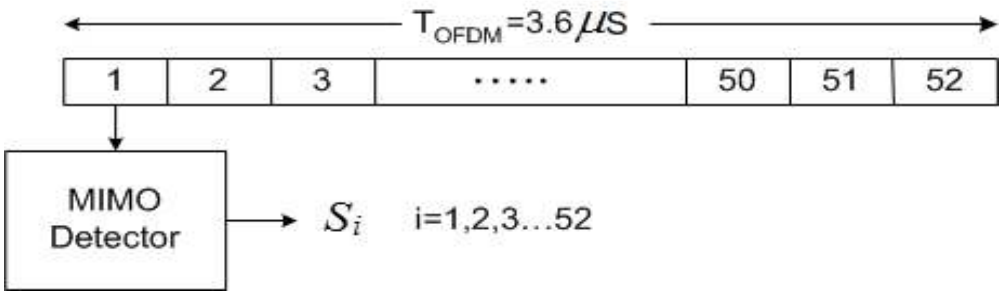

Figure 4.3: MIMO Detection Interface Timing as Required in IEEE 802.11n

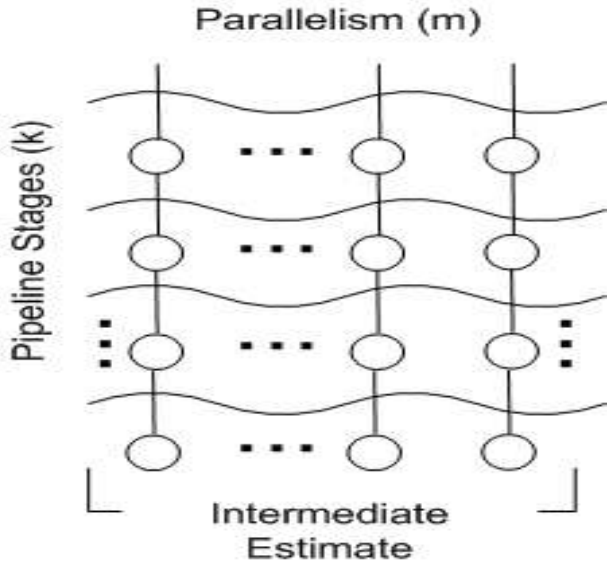

Figure 4.4: High Level Architecture of COSIC Based MIMO Detection

4.2.1 Requirements of the IEEE 802.11n standard

Figure 2.1 depicts a simplified MIMO-OFDM system. An IEEE 802.11n system and its associated processing stages and requirements are shown in Figure 4.2. In MIMO-OFDM systems such as IEEE 802.11n, OFDM is used to mitigate the affect of multi-path fading. There are 52 data tones (or subcarriers) to be processed at the receiver. Each tone carries a MIMO symbol; hence the detector has to process all 52 tones in stipulated time of 3.6μs (as imposed by the standard). This fact is shown pictorially in Figure 4.3. All 52 data tones are modulated using same modulation scheme, thus the decoder does not have to switch between modes within an OFDM symbol. The decoder is essentially an array of processors arranged as shown in Figure 4.4. It can process *m* candidate vectors in parallel. Furthermore, each data path in the array can be pipelined into *k* parts. The problem is to find *m* and *k* such that power/area is minimized as much as possible subject to the throughput constraint. Since, the decoder is reconfigurable this optimization has to be over all the supported modes.

4.2.2. Throughput Planning

First we will establish the relationship between the time taken, T_p, to process 52 MIMO symbols with *m* and *k* for η = 4, 16, and 64. Since the COSIC tree has η candidate vectors to be evaluated (for an η-ary modulation scheme), it takes η/m clock cycles to detect an η-ary modulated MIMO symbol in steady state. Hence, T_p is given by (4.1).

$$\mathbf{T_p = 52.\ \eta/m\ .\ freq} \qquad \mathbf{(4.1)}$$

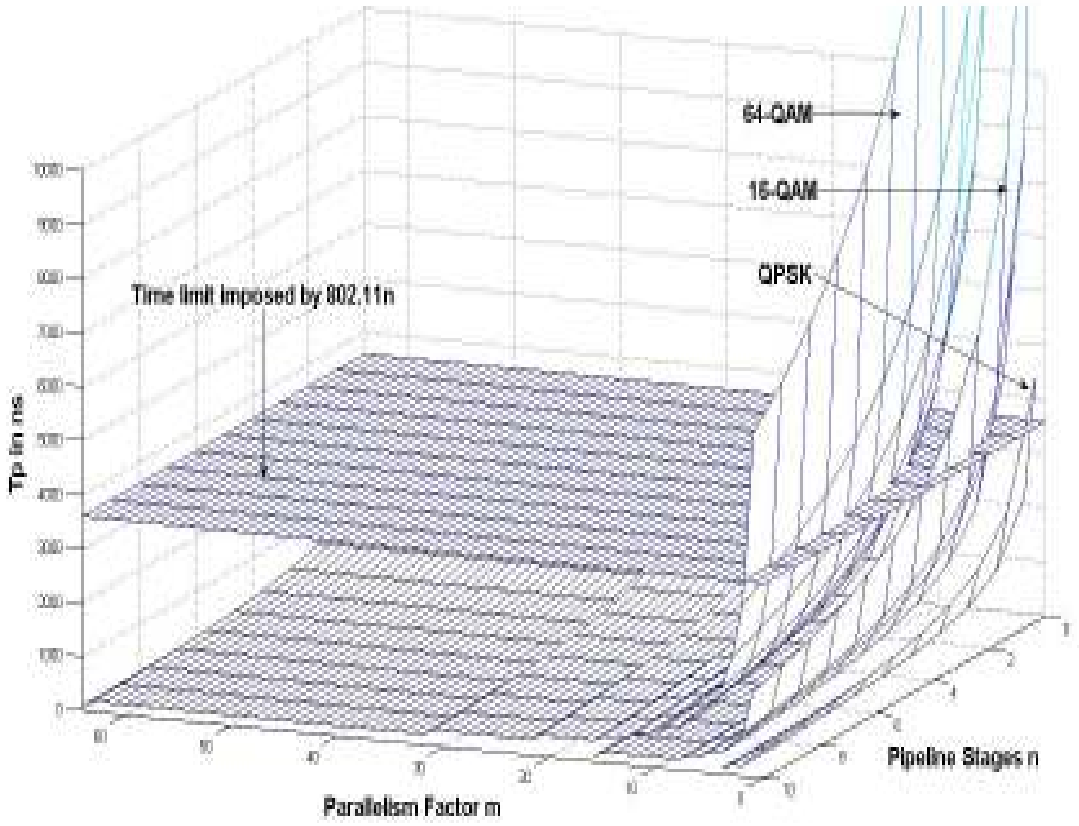

Figure 4.5: T_p vs. (m, k) Constraint Due to 802.11n

We assume that the critical delay of data-path after introducing *k* pipelines reduces to $C_d/(k+1)$. This assumption has been validated empirically for *k*=0 to 10, using Synopsys Design Compiler re-timing utility. Eqn. 4.1 thus becomes:

$$\mathbf{T_p = T_p = 52.\ \eta/m\ .\ C_d/(k+1)} \qquad \mathbf{(4.2)}$$

where the factor 52 corresponds to the number of tones, C_d is the combinational delay of the un-pipelined data-path. Since 802.11n requires that the processing on all 52 tones be over in 3600ns, Tp ≤ 3600ns. Using above discussed model in Figure 4.5, we show how

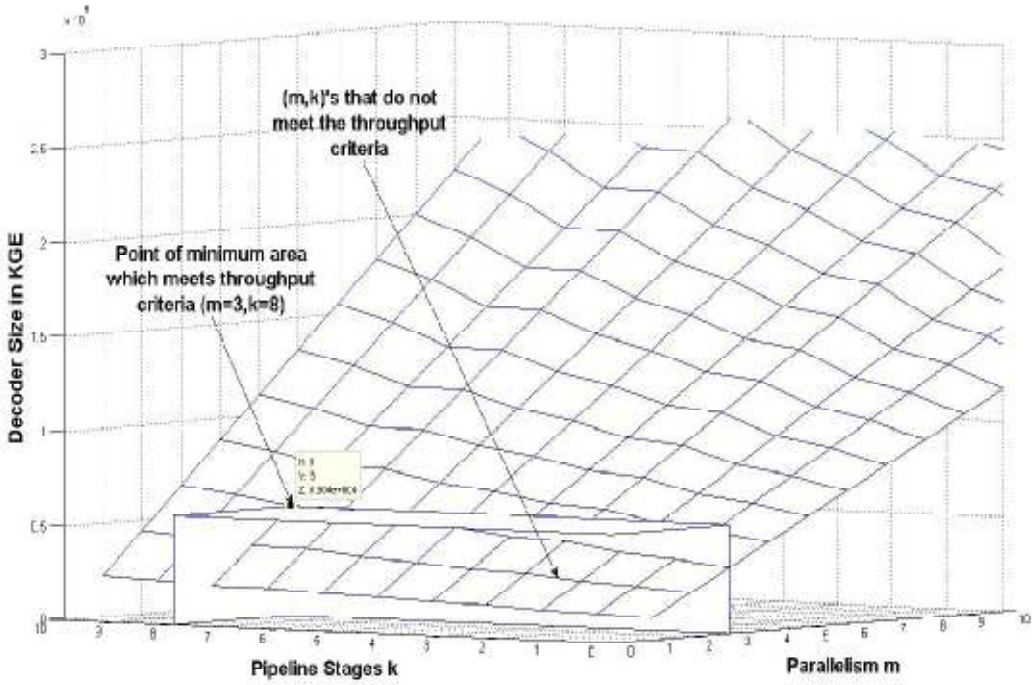

Figure 4.6: Area vs. T_p for 64-QAM

Tp behaves in the architecture space (with *m, k*) for η =4, 16 and 64. Figure 4.5 also shows the constraint imposed by the 802.11n standard on the maximum time allotted to process all MIMO symbols. Clearly, all the points above the constraint plane are unacceptable as they don't meet the throughput criteria. In practice we keep the constraint plane at 3000ns (rather than 3600ns), this is to accommodate 15-20% pessimism factor.

4.2.3. Power, Delay and Area Estimation

The power consumed comprises mainly of the core power (due to switching/leakage of logic gates), and due to clock network. The technology mapped

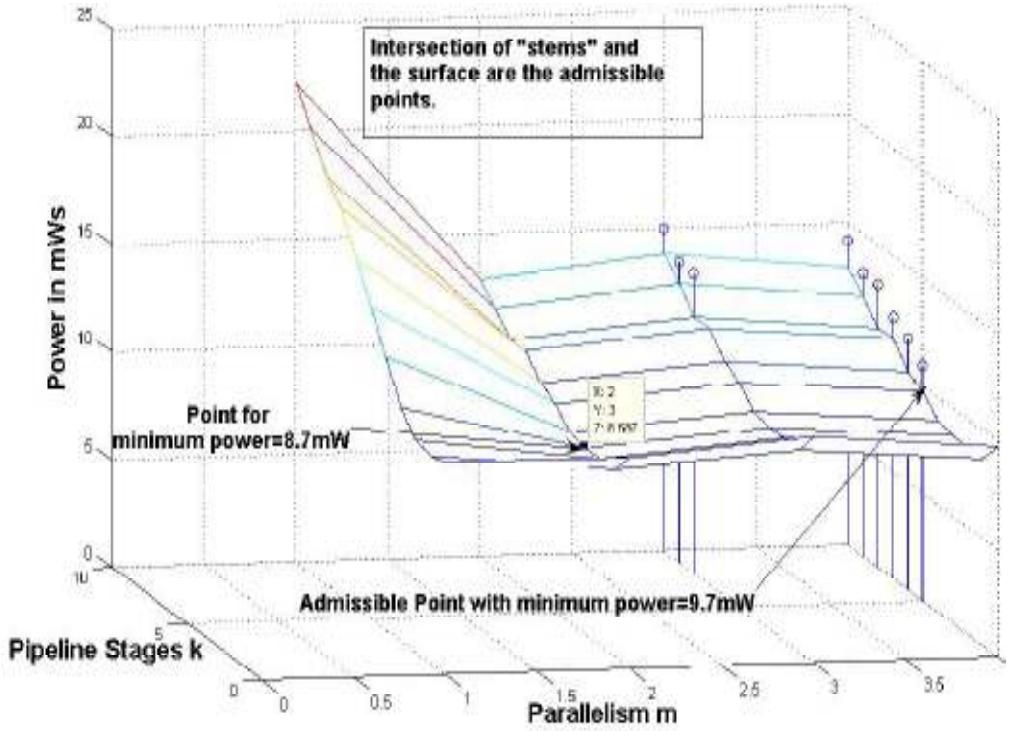

Figure 4.7: Aggregate Power vs. *m, k*

verilog netlist is analyzed for core power using Synopsys Design/Power Compiler. The technology library used was a composite current source (typical) based 45nm library from Nangate. The power consumed due to clock depends on number of flops and the geometry of the clock network. The clock network was modeled as a symmetrical mesh; the global clock network power was estimated using HSPICE. The local power was estimated using capacitive load due to the number of flip-flops driven by the local clock buffer (buffers were sized appropriately). Synopsys Design Compiler was used to estimate the delay of the circuit (re-timed circuit depending on the number of pipeline stages). Area estimates were also provided by Design Compiler.

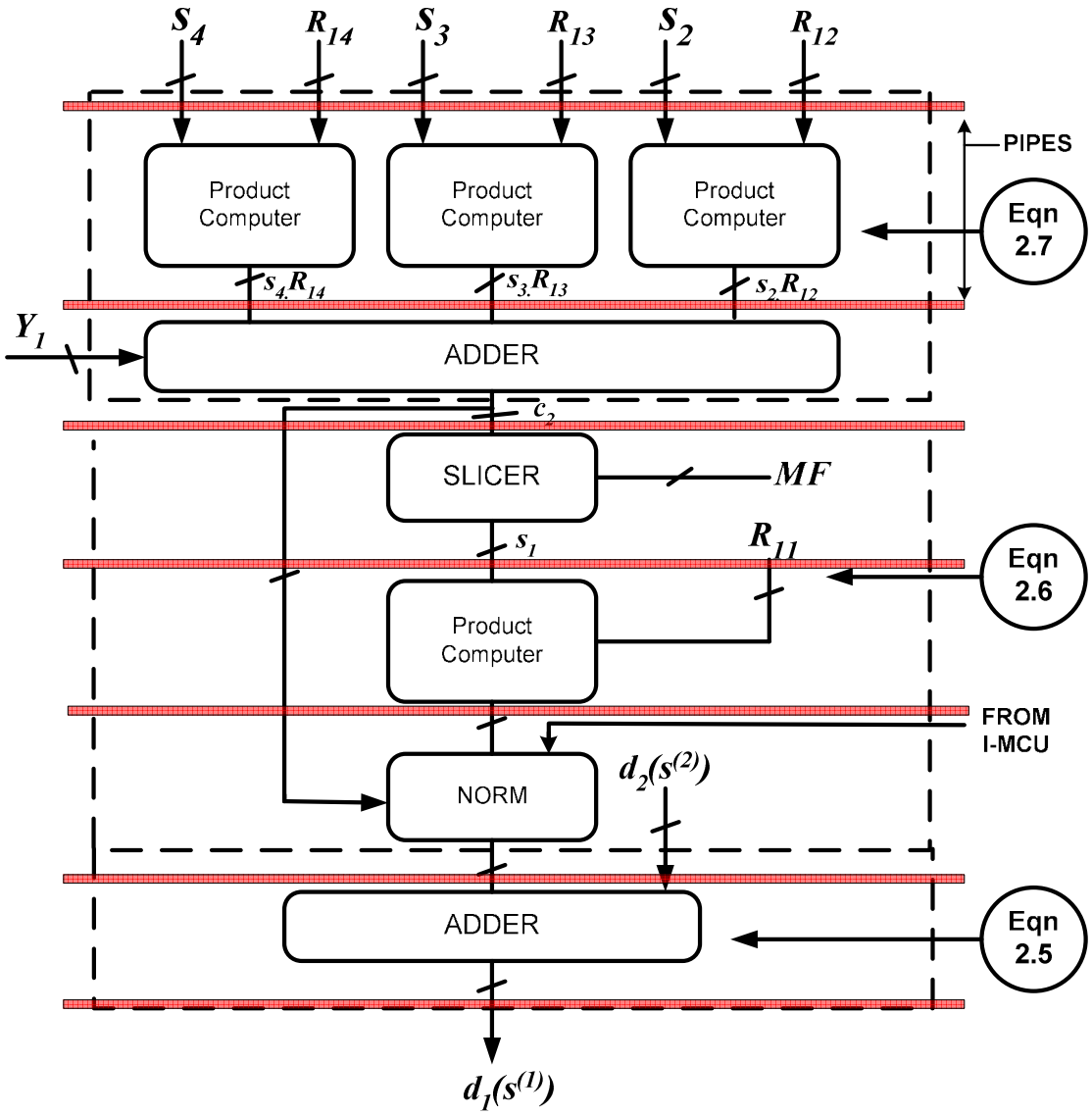

Figure 4.8: MCU Architecture of Level 1 in the MIMO Detector

4.2.4 Architectural Exploration for Low Area

In this subsection we describe our exploration procedure for low area, based on (m, k) constrained to throughput requirements of 802.11n. Figure 4.4 shows the

exploration space with respect to *m* and *k*. The points below the constraint plane are called admissible points. Since 64-QAM is computationally most expensive, the admissible points for 64-QAM must be admissible for 16-QAM and QPSK. Hence, to find (*m*, *k*) for area optimized decoder we only need to meet throughput requirements for 64-QAM with minimum hardware. Figure 4.6 shows variation of area of the decoder with *m* and *k.* In particular *m*=3 and *k*=8 meets the throughput requirement with least area as shown in Figure 4.6.

4.2.5 Architectural Exploration for Low Power

Exploring the same space for power is more complicated, because different modulation schemes have different power consumption profiles while achieving the required throughput. Power consumption has to be optimized over all modes of operation. This means we have to pick (*m, k*) such that the 'aggregate power" is minimized. We define aggregate power as Pow_{agg}= Prob(QPSK) * Pow(QPSK)+ Prob(16-QAM) * Pow(16-QAM)+ Prob(64-QAM) * Pow(64-QAM), where Prob(QPSK) is the probability of the decoder being reconfigured to process QPSK MIMO symbols, and Pow(QPSK) is the power consumed by it while processing a QPSK MIMO symbol etc. Since there is no apriori knowledge of the probabilities, we assume them to be equally likely i.e. Prob(QPSK)=Prob(16-QAM)=Prob(64-QAM)=1/3. Pow_{agg} is a function of (*m, k*) as shown in Figure 4.7. However the point corresponding to the least power, does not meet the throughput criteria. Figure 4.7 shows the points (using stems) that meet the throughput criteria. Hence, (*m, k*) needs to be searched among these points. Hence, in our power optimized design uses *m*=4 and *k*=5 as shown in Figure 4.7.

TABLE 4.1

Comparison of Performance with Existing Architectures

Reference	Supported Modes	BER	Area (KGE)	Power (mW)	Technology (nm)	Throughput (Mbps)
[9]	16-QAM	Q-Opt	50K	473	250	169
[17]	16-QAM	Q-Opt	91K	626	350	52
[25]	QPSK	Opt	685K	N/A	180	28.8
[11]	16-QAM	Q-Opt	175K	407	180	160

4.3 Details of the MIMO Detection VLSI Architecture

The Metric Computation Units (MCUs) computes $d_i(s^{(i)})$ metrics using (2.5)-(2.7).

4.3.1 MCU Architecture

The MCU computes equations (2.5)-(2.7) in that order. Figure 4.8 shows the detailed structure of a MCU with only combinational logic at level 1 (lowest level in tree, corresponding to i = 1). The upper dotted box in Figure 4.8 evaluates (2.7). There is no need to implement the product terms in (2.7) using a multiplier. This product can be implemented simply by shift and add operation, because the QAM constellation points only take on a finite number of integer values (e.g. in 16-QAM scheme the real and imaginary part of $s_j \varepsilon \{-3, -1, 1, 3\}$). The middle dotted box implements (2.5); this box is primarily composed of two blocks: a Slicer, and a block for computing the norm. As discussed earlier the COSIC algorithm picks the all children of the root node, and the

TABLE 4.2

ASIC Implementation Details

Design Parameters	**Area Optimized Design**	**Power Optimized Design**
Target Technology Library	Nangate 45nm PDK	Nangate 45nm PDK
Pipeline Stages (k)	8	5
Parallelism (m)	3	4
Gate Equivalent	58.2K	67.7K
Power Consumption	11.91mW	9.7mW
Frequency: QPSK	38.8MHz	18MHz
Frequency: 16-QAM	116.3MHz	71.8MHz
Frequency: 64-QAM	426.6MHz	287.3MHz
Throughput Requirement QPSK	115.6Mbps	115.6Mbps
Throughput Achieved QPSK	155Mbps	144Mbps
Throughput Requirement 16-QAM	231.1Mbps	231.1Mbps
Throughput Achieved 16-QAM	310.13Mbps	287.2Mbps
Throughput Requirement 64-QAM	346.7Mbps	346.7Mbps
Throughput Achieved 64-QAM	465.38Mbps	430.95Mbps

'best' child (nodes with least $|e_i|^2$) of these nodes thereafter. From (2.5) 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 Figure 4.8. This operation involves independently comparing real and imaginary parts of c_{i+1} with appropriate decision thresholds. For example, in Figure 4.8 the nearest scaled QAM symbol to c_{i+1} is $(1+j).R_{ii}$, since real and imaginary parts of c_{i+1} are both less than $2R_{ii}$. In general, the decision thresholds are given by $(-(\sqrt{\eta} - 2) + 2j)$ R_{ii}, where j is an integer such that $0 \leq j \leq (\sqrt{\eta} - 2)$). The decoder configures the slicer based on the MF bits (MF bits are essentially representative of η). The block named 'NORM' computes (2.6), this involves squaring operations, however, we use suboptimal norm in order to avoid multipliers as will be described next. Simplified l^1 norm computation as shown earlier in Section 3 was also used instead of l^2 norm to reduce the number of multipliers in the design. Each individual blocks in the design can be seen in Section 3.

4.4 ASIC Implementation Results

Table 4.1 shows the comparison between various existing designs (q-opt stands for quasi-optimal). In Table 4.2 we present implementation results of our two designs (area and power optimized) and where they stand with respect to the existing designs. MATLAB was used to simulate bit accurate model of the decoder. We chose eleven bit fixed point quantization (internal precision was maintained) for negligible BER degradation. Detailed hardware architecture was then developed. The RTL coding was done using Verilog HDL. Nangate 45nm CMOS predictive standard cell library was

used for the design flow. Synopsys Design Compiler was used to synthesize the gate level net-list and retime the circuit. MATLAB generated test vectors that were used to validate the design.

4.5 Results, Summary and Conclusion

Two designs for reconfigurable MIMO detector are presented in this section. First design pushes for least area, and the second one targets lower power consumption. The detector is dynamically reconfigurable for QPSK, 16-QAM and 64-QAM modulation schemes for 4x4 MIMO systems as was discussed in Section 3 as well. The proposed architecture is highly suitable for the IEEE 802.11n wireless standard because of its flexibility, and ability to meet the required throughput.

5. LOW POWER RECONFIGURABLE MIMO DETECTOR FOR REAL-TIME MOBILE APPLICATIONS

This section presents a low power runtime reconfigurable architecture for MIMO detection and its ASIC implementation. The design is able to reconfigure on the fly to various modulation schemes as we have seen in earlier sections with fixed throughput for each modulation scheme and achieves quasi-optimal BER performance. The detectors discussed earlier have variable throughput for different modulation schemes which doesn't work well with wireless systems which need adaptive modulation schemes along with fixed throughput decoding for all modulation schemes. In this section, a decoder architecture which shows significant savings in term of energy as compared to conventional decoder architectures is presented. The low power strategy is based on the optimal parallel processing/pipelining arrangement (as shown in Section 4) combined with a dynamic voltage and frequency scaling (DVFS) technique. Current demands for higher data rates has led to the use of MIMO wireless systems, which offers higher throughput without any overhead in terms of bandwidth as compared to single input single output wireless system due to its higher spectral efficiency. Nevertheless, the high complexity of the MIMO decoder hardware results in significant power consumption which is unacceptable for portable applications. These applications also require support for multiple modulation schemes, multiple antennas configuration, variable code rate and multiple space-time coding schemes [17]. The complicated nature of the MIMO detection algorithms combined with the need for re-configurability pose a major challenge to the implementation of MIMO detectors [26] as shown in Section 3.

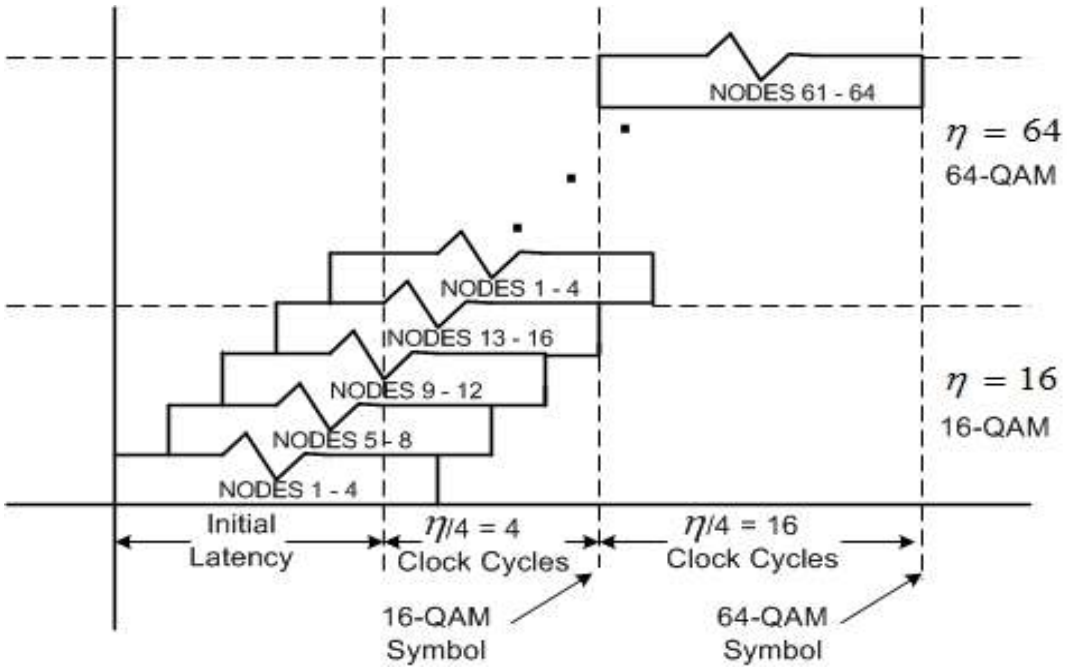

Figure 5.1: Decoding Iterations Verses Modulation Scheme

Architectures amenable to deep pipelines and high degree of parallelism are more desirable for practical implementation due to their superior throughput and lower energy consumption levels. Among the many algorithms considered for implementation, our implementation is based on the Fixed-Throughput Sphere Decoding Algorithm with COSIC arrangement as it can be implemented in a highly parallel and pipelined manner, and as the name suggests, has fixed throughput for a particular modulation scheme which was explored in Section 4. COSIC also achieves a quasi-optimal BER performance as can be seen from [19]. Earlier in the thesis in Section 3, we present a dynamically reconfigurable MIMO detector architecture. This architecture dynamically reconfigures based on the modulation scheme of the frame. Each frame takes variable

number of iterations through the detector to complete processing depending on the η of the modulation scheme as shown in Figure 5.1, resulting in unpredictable frame completion time which makes interfacing with the application modules rather difficult because high level applications pose restrictions on the duration of MIMO detection as can be seen in Figure 4.2. In addition, power consumption is also an important design constraint for the mobile applications which mostly run on battery powered devices. In this section, we propose a DVFS based reconfigurable MIMO detector and its ASIC implementation in TSMC 0.18μm technology which addresses three major issues: lower energy consumption and re-configurability to different modulation schemes with lower complexity and fixed throughput across modulation schemes. Our detector supports on the fly re-configurability for QPSK, 16-QAM and 64-QAM modulation schemes with constant throughput for all modulation schemes which are ideal for real-time multi-media applications. The decoder uses optimal voltage and frequency combinations while processing data frames resulting in significant energy gains (nearly 24% per decoded bit) suitable for portable devices. This technique is called dynamic voltage and frequency scaling (DVFS) [14][15]. DVFS brings down the system power dissipation as it is proportional to occurrence of activity and quadratic voltage supply. The modulation format bit input determines the number of decoding iterations of the hardware, based on which the operating frequency and voltage is chosen. The choice is made in such a way so as to decode each frame, within a fixed time-period. Thereby, making available the output of each frame synchronized to the fixed rate at which the data is consumed in real-time application interface. The control logic has low complexity (8% overhead in

power and 5% overhead in area) and is highly integrated with the data flow. This detector delivers quasi-optimal BER performance with no reconfiguration latency which guarantees the necessary Quality of Service (QoS) with uninterrupted processing of MIMO symbols.

5.1 Circuit Power Estimation and Reduction

There are three major sources of power dissipation in CMOS circuit:

$$\mathbf{P_{Total} = P_{Switching} + P_{SC} + P_{Leakage}} \quad \mathbf{(5.1)}$$

The individual power components are switching power, short circuit power and leakage power can be expressed using the following equation:

$$\mathbf{P_{Total} = \alpha.C_{L}.\Delta V.V_{dd}.f_{clk} + I_{SC}.V_{dd} + I_{leakage}V_{dd}} \quad \mathbf{(5.2)}$$

where $P_{switching}$ represents the switching power due to the charging and discharging parasitic capacitances in the circuit. C_L is the loading capacitance, f_{clk} is the clock frequency, and α is the node transition factor defined as the probability that a power consuming transition occurs. In most cases, the voltage swing ΔV is the same as the supply voltage V_{dd}. The short circuit power P_{SC} is caused by direct-path short circuit current I_{SC} which arises when both NMOS and PMOS are simultaneously turned on. This is caused by the finite rising and falling time of input signal. The short circuit power can be kept within 15% of the switching power if carefully designed [27]. $P_{leakage}$ is the leakage component of power, where $I_{leakage}$ is the total leakage current in CMOS circuit. Also for CMOS circuits, delay of the circuit increase with decreased voltage supply, as shown in (5.3):

$$\mathbf{\zeta = 1/ f_{clk} = [C_L V_{dd}/ I_{dsat}] \propto [V_{dd}/ (V_{dd} - V_{th})^{1.3}]} \quad (5.3)$$

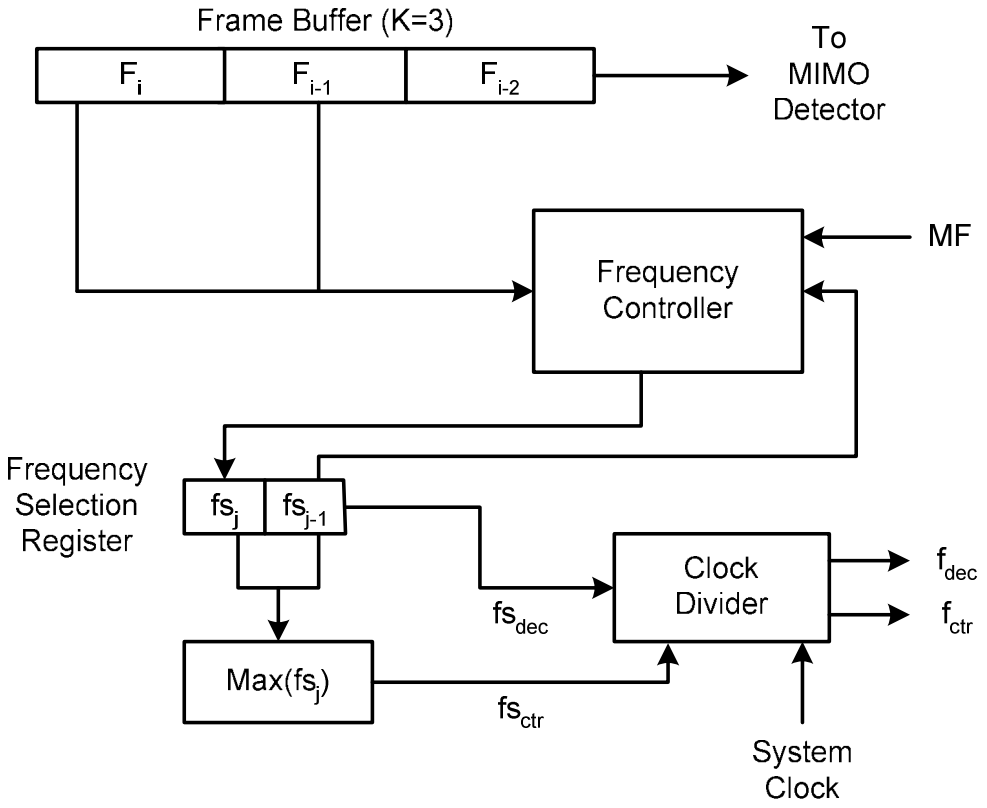

Figure 5.2: Frequency Controller Unit for DVFS

Typically, switching power is the main source of power dissipation in the circuit especially in baseband processing circuits which continuously process data. It should be noted that while power consumption decreases linearly with the operation frequency, the time for finishing the certain workload increases. As a result, the total energy consumption remains constant for the same workload if the power supply is not changed. Dynamic voltage and frequency scaling is an effective method to address this energy consumption problem, especially under wide variations in workload.

5.2 DVFS Based Low Power Decoder Architecture

5.2.1 Reconfigurable Sphere Decoder Architecture

Figure 4.8 in the previous section shows the high level architecture of the decoder implementation which has been discussed in detail earlier. The choice of 4-way parallelism is ideal for power considerations because the smallest constellation supported on our decoder is QPSK (which has four symbols, $\eta=4$) as was seen in Section 4. If this architecture is pipelined with m stages then it has an initial latency of $m+\eta/4$ clock cycles. Note that the FSD tree has η paths for a η-ary modulation scheme. Hence, the proposed architecture takes $\eta/4$ clock cycle to detect a η-ary modulated MIMO symbol. At each level of the FSD tree (Figure 5.1) we need to compute the $d_i(s^{(i)})$ metrics using (2.5)-(2.7) as was discussed earlier. Each of the equations is computed by MCUs. Figure 4.8 shows the structure of the MCU at level 1 (lowest level in tree). The detailed architecture of the detector is described in Section 3 and it high level exploration in terms of optimal parallelism and pipelining is presented in Section 4.

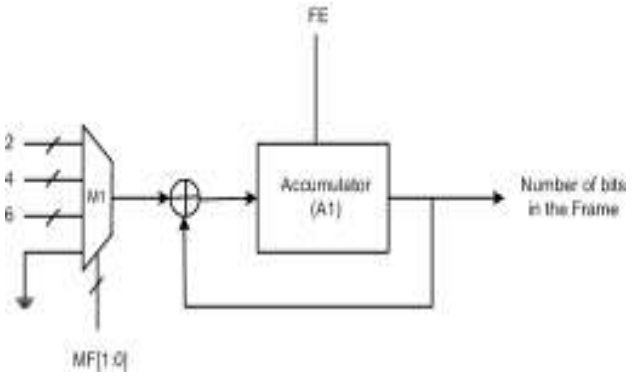

Figure 5.3: Bit Computer Unit

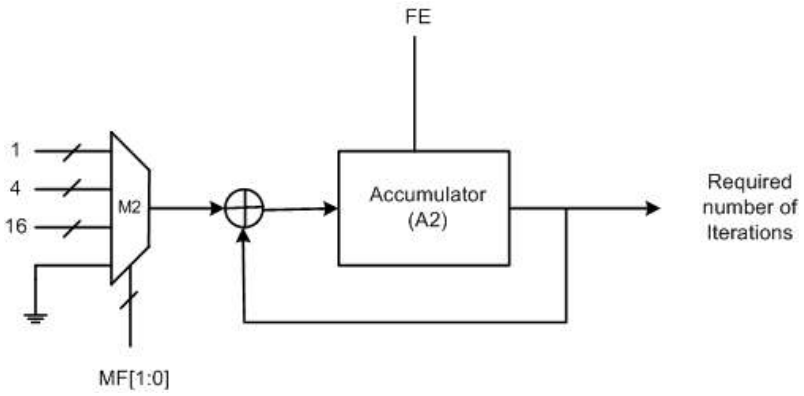

Figure 5.4: Iteration Computer Unit

5.2.2. Dynamic Voltage and Frequency Scaling Policy

The key observation made for adaptive decoding is that for the reconfigurable decoder architecture described above, QPSK data frames are decoded at the rate of 1 iterative cycle of hardware per symbol, 16-QAM data frames are decoded at 4 iterative cycles of the hardware per symbol and 64-QAM data frames are decoded at 16 iterative cycles of hardware per symbol because the decoding process in QPSK and 16-QAM is finished in lesser number of decoding iterations than 64-QAM data frames resulting in variable throughput of the decoder for different modulation schemes as shown in Figure 5.1. Hence, significant energy saving can be achieved by lowering the decoder performance level especially while processing QPSK and 16-QAM data frames. In this way, the 64-QAM decoding process can borrow time from the other modulation schemes. The modulation format (MF) bit (which was discussed earlier in Section 3) of each symbol is used to pre-determine the total number of bits to be decoded as shown in Figure 5.3 and the total number of decoding iterations needed to process that particular

ALGORITHM I

Dynamic Voltage and Frequency Scaling Policy

```
i = 0
while i < 3 do
    if MF == 0 then
        Tot_No_Bit = Tot_No_Bit + 2;
        Num_Dec_Iteration = Num_Dec_Iteration + 1;
    end if
    if MF == 1 then
        Tot_No_Bit = Tot_No_Bit + 4;
        Num_Dec_Iteration = Num_Dec_Iteration + 4;
    end if
    if MF == 2 then
        Tot_No_Bit = Tot_No_Bit + 6;
        Num_Dec_Iteration = Num_Dec_Iteration + 16;
    end if
i = i + 1;
end while
Reqd_Dec_Time  = Tot_No_Bit * Throughput;
Opr_Freq = Num_Dec_Iteration /Reqd_Dec_Time;
```

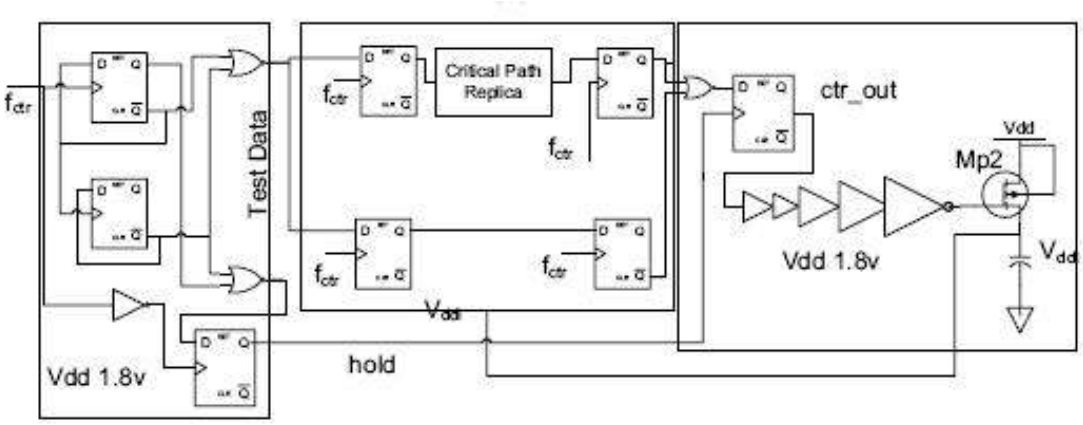

Figure 5.5: Buck Converter Circuit for Voltage Regulation

data frame is computed as shown in Figure 5.4. Based on the throughput requirement and total number of bits in a data frame, we determine the frequency-voltage combination to be used for decoding. The maximum number of decoding iteration is set to be close to optimum in terms of energy and detection performance, without violating the real-time constraints. The policy is described in the pseudo code given in Algorithm I. It should be noted that Num_Dec_Iteration is the required number of decoding iterations for that particular data frame; Tot_No_Bit is the total number of bits embedded in the buffered data frame. Note that each symbol of QPSK has 2 bits, 16-QAM has 4 bits and 64-QAM has 6 bits embedded in it. A data frame buffer is used to accumulate 3 symbols. The data in the buffer is operated upon as given in Algorithm I and the operational frequency is picked by the frequency controller (setting the frequency selection registers) in such a way that it is just greater than the required computed frequency.

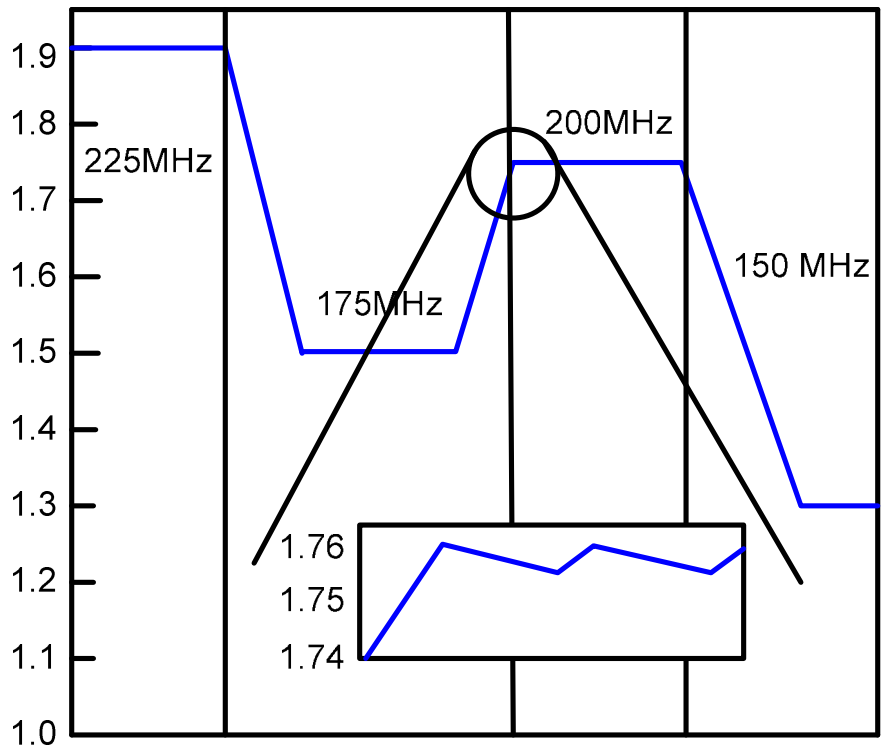

Figure 5.6: Voltage Output of the Voltage Controller

5.2.3. DVFS Controller Design

The above policy can be implemented with extremely low hardware complexity. Even though it has been reported [28] that for modern VLSI technology, the leakage power is becoming so significant that the best solution for managing power is maintaining the highest performance as long as possible and then turning the circuit into sleep mode. In the case of MIMO decoder, however, this is not feasible because of the real-time constraints, constantly incoming data, as well as power overhead associated with turning off and on the circuit. The clock frequency is determined by the constant

decoding time which is pre-determined for each successive frame. Figure 5.2 shows the diagram of the

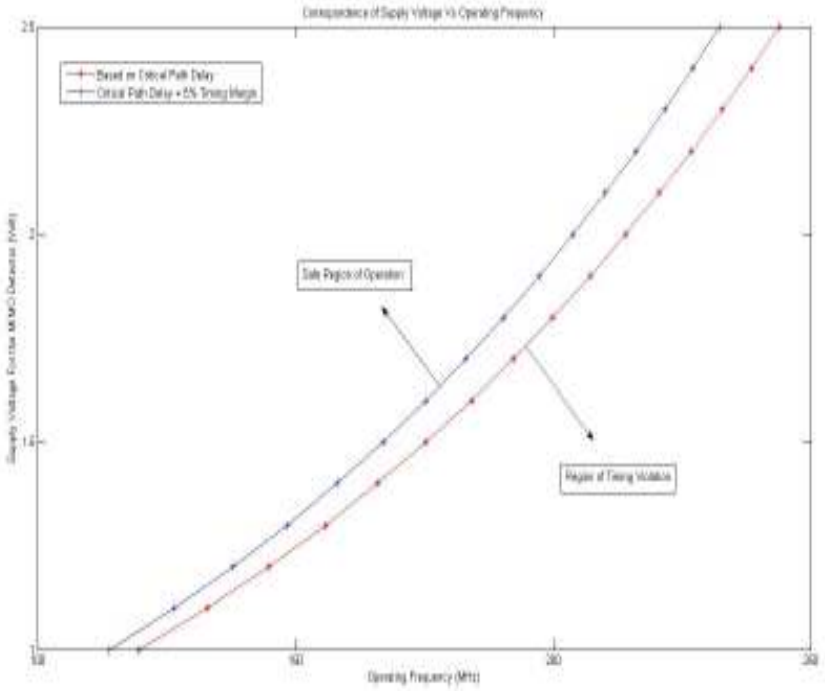

Figure 5.7: Voltage Supply Requirement with Different Frequency

frequency controller in the design which generates the operating frequency of the successive frames. Operating at low clock frequency, the voltage supply can be lowered correspondingly to the frequency which is done by the voltage regulator shown in Figure 5.5. The voltage output response of the regulator for different operating frequencies in shown in Figure 5.6 which responds fast enough to meet the timing margin shown in Figure 5.7. The different symbols in the data frame are stored in a data frame buffer. As previously discussed, the total number of bits in the frame can be calculated using the

Bit Computer Unit as shown in Figure 5.3. The frame end bit (FE) of the Bit Computer Unit resets the accumulator so that it starts counting when the next frame is buffered. The Bit Computer Unit uses an accumulator to keep track of the number of bits embedded in each symbol stored in the frame. Once the number of bits to be decoded in

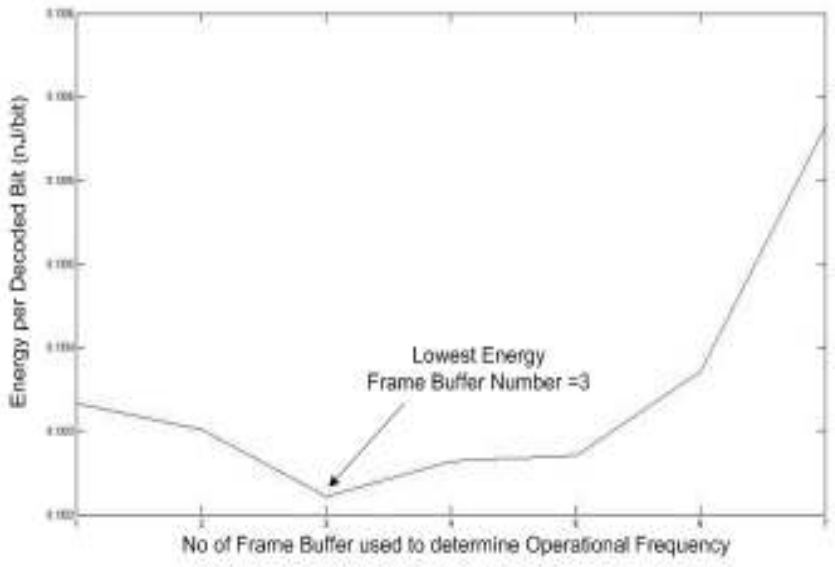

Figure 5.8: Number of Frame Buffer Verses Energy per Decoded Bit

a frame and the required throughput for an application is known (which is a given in practical systems), the frame decoding time is computed by the frequency controller. The number of decoding iterations needed to decode a frame in calculated by the Iteration Computer Unit as shown in Figure 5.4. The frame decoding time combined with the total number of required iterations is used to compute the operational frequency.

Once a frame is decoded, the voltage & frequency level needs to be changed according to the needs of the next frame. Because level of the voltage supply cannot be changed instantly, inter-frame buffers are required for storing the incoming channel data and a frequency selection buffer is also required to store the decoding frequency information for corresponding data frames. The data frame size M is chosen based on the energy per decoded bit vs number of frame buffer curve shown in Figure 5.8. In our design, we have chosen data frame buffer size as three symbols (M=3) i.e. three symbols will be stored together in the data frame buffer to estimate the frequency of operation of the decoder while processing that frame. The inter frame buffer size K is determined by the time response of voltage supply V_{ddl} as well as the overall throughput required in the application. As presented later in this section, buffer size of 1 frame is needed typically in the detector for synchronization purposes. The overhead is buffer of size 2 frame since a buffer size of 1 frame is intrinsic for the decoder. Clock divider is preferred over other designs such as phase-loop locked (PLL) in [14], because it provides reasonable frequency resolution for the decoding policy and capability to change immediately. F_{dec} clocks the decoder for current frame, and f_{ctr} is sent to the voltage scaling controller for adjusting the voltage for the next frame once the frame processing is over. F_{ctr} is conservatively generated as the fastest clock such that the voltage supply will be within safe region of operation (Figure 5.6). A variety of VLSI implementations of the voltage-scaling controller have been reported in the literatures. In our implementation, we use the design in [15] of the buck converter, because it is of reasonable complexity and suits the needs of this design. Other control schemes can also be used. Secondly, a

reconfigurable architecture based on the FSD algorithm [19] is used for the MIMO decoder. The critical path of the decoder is extracted and replicated for the voltage controller. Load capacitor of the voltage controller should be large enough to maintain a steady voltage level in presence of sudden change in the output current. The capacitor is chosen to be 0.65μC. The power transistor Mp2 is 400μm in width, which is driven by five stages of buffer, with a scaling up factor of 4 [15], considering the minimum power consumption. Figure 5.5 shows the diagram of the circuitry.

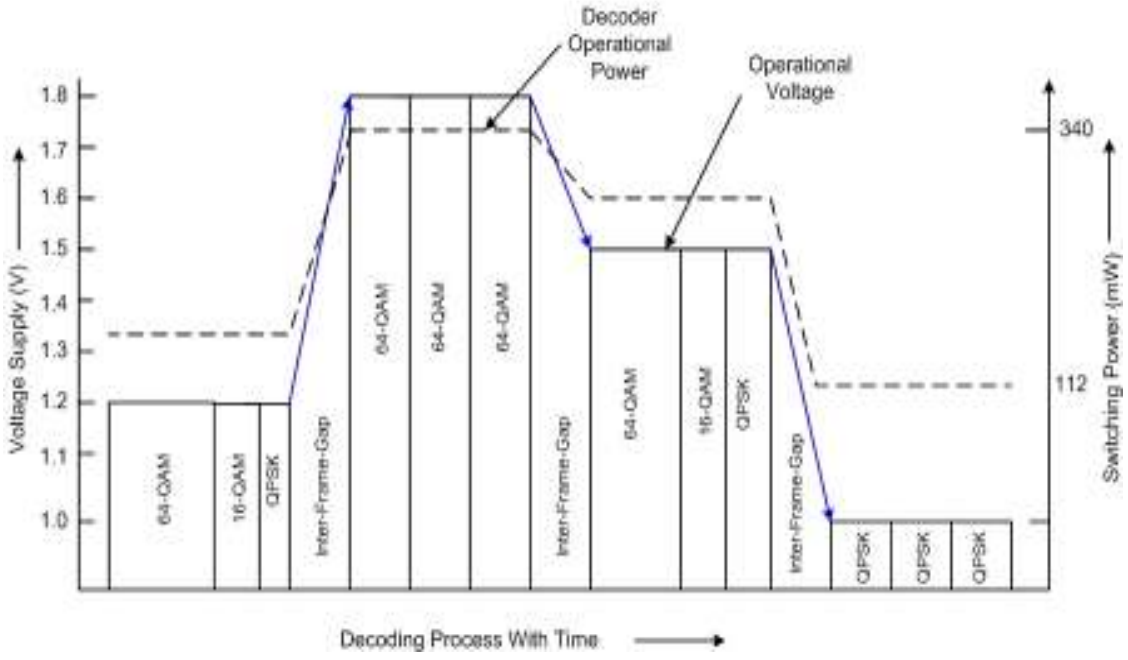

Figure 5.9: Frame-Wise Dynamic Voltage/Power Profiles

5.3 Implementation Results

The design of voltage-scaling controller has been simulated using TSMC 0.18μm technology. With 1.9V voltage supply, the decoder can be clocked as fast as 250MHz with a 21 stage pipeline. Extra 5% timing margin has been added to the critical path replica of the voltage controller to accommodate for variations as shown in Figure 5.6. Figure 5.9 demonstrates voltage response of the converter with respect to the change in load of the data frames. The buck converter can scale up the output voltage level to a maximum of 60mV/μs which was seen using SPICE simulation using Synopsys HSPICE. Hence, the result presented in [15] aligns with your simulation results.

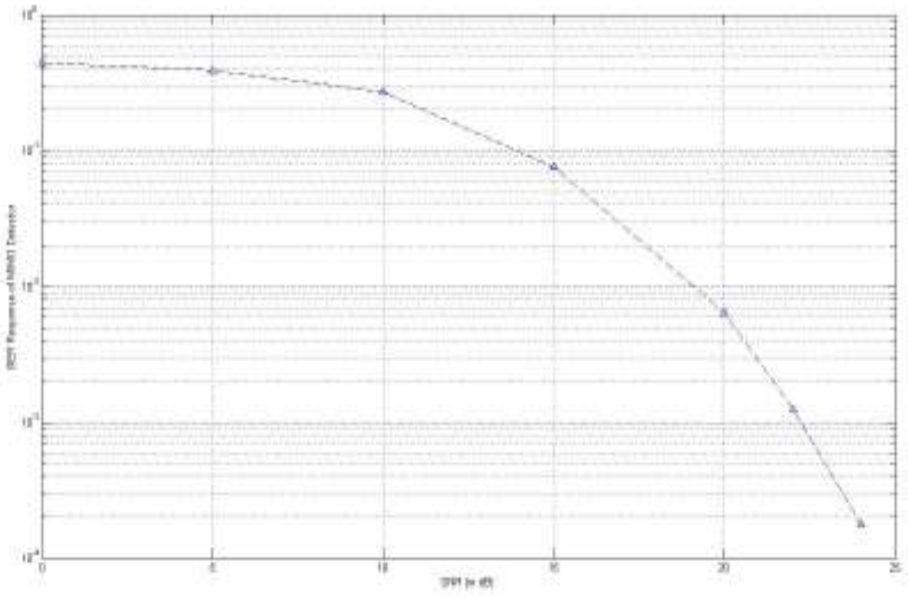

Figure 5.10: BER Response of the presented DVFS based MIMO Detector

Assuming a 450MHz system clock, it can be divided into 225MHz, 200MHz, 175MHz and 150MHz for the decoder. The voltage supply varies from 1.9V to 1.0V within this frequency range. It takes about 15μs to scale the voltage up by 0.9V as shown in Figure 5.6. A inter frame-buffer size of 3 in our case suffices to meet the voltage transition requirements and also provides the right tradeoff between energy saving because of DVFS and number of frame buffers. Current through the PMOS power transistor constitutes the majority of power overhead of the controller. It is simulated to be in the order of 15mW as shown in Table 5.1, which is small compared to maximum

TABLE 5.1
ASIC Implementation Details and Comparison

Comparison Criteria	**DVFS Arch**	**Non-DVFS Arch**
Target Technology	0.18μm	0.18μm
Throughput: QPSK	225 Mbps	225 Mbps
Throughput: 16-QAM	225 Mbps	450 Mbps
Throughput: 64-QAM	225 Mbps	675 Mbps
Equivalent Gates	74 KGE	70 KGE
Energy/Decoded Bit	130 pJ/bit	170 pJ/bit
Energy Gain (%)	24 %	N/A
Control Area Overhead	5%	N/A
Control Power Overhead	8%	N/A

power dissipation of the decoder which is around 340mW when operated at a voltage of 1.9V and frequency of 225MHz for a continuous 64-QAM decoding sequence. If we assume the equal probability of occurrence of each modulation scheme in a data frame, then the average power dissipation of the detector is further lowered to around 262mW approximately. The numbers of decoding iterations for 64-QAM, 16-QAM and QPSK are 16, 4 and 1 respectively, based on the detector architecture used with a MCU parallelism of four. These numbers are chosen based on complexity of design and consideration of performance requirement. The frequency selections are such that the above numbers of decoding iterations with frequency and voltage scaling yield constant-time detection which is useful for real-time applications. It is clearly seen in Table 5.1 that up to 24% energy per decoded bit is saved without bit-error degradation as shown in Figure 5.9. In summary, the presented adaptively decoding scheme will achieve significant saving in decoding energy. Figure 5.8 also shows the timeline of the decoder operation with three sample frames having different combination of modulation schemes. The power profile is seen to vary with time because of reduced performance during duration of reduced load. It can also be observed that all the frames are decoded within the same time interval providing constant throughput which are a requirement of many wireless standards.

5.4 Results, Summary and Conclusions

A low power MIMO detector scheme suitable for portable battery powered device in real-time mobile communication is presented. Incoming data symbols are processed before decoding to determine the frequency of operation and voltage to be

used for the decoding process based on the throughput requirements. Power overhead of the adaptively decoding control unit mainly stems from the power transistor, and it is found to be small compared to the power saved because of the frequency & voltage scaling of the decoding process. Up to 24% energy saving per decoded bit in the decoding process is achieved without performance degradation.

6. SUMMARY, CONCLUSIONS AND FUTURE WORK

MIMO enables spatial multiplexing to increase channel bandwidth which requires the use of multiple antennas in the receiver and transmitter side. The increase in bandwidth comes at the cost of high silicon complexity of MIMO detectors which is a result of the intricate algorithms required for the separation of these spatially multiplexed streams. Previous implementations of MIMO detector have mainly dealt with the issue of complexity reduction, latency minimization and throughput enhancement. Although, these detectors have successfully mapped algorithms to relatively simpler circuits but still, latency and throughput of these systems need further improvements to meet standard requirements. Additionally, most of these implementations don't deal with the requirements of re-configurability of the detector to multiple modulation schemes. This necessary requirement provides another dimension to the implementation of MIMO detector and adds to the implementation complexity. This thesis focuses on the efficient VLSI implementation of the MIMO detector with an emphasis on low power, performance and re-configurability to modulation schemes. MIMO decoding in our detector is based on the FSD decoding algorithm which has been simplified for an effective VLSI implementation without considerably degrading the near optimal bit error rate performance. The regularity of the architecture makes it suitable for a highly parallel and pipelined implementation. The decoder has intrinsic traits for dynamic re-configurability to different modulation and encoding schemes. This decoder architecture can be easily tuned for high/low performance requirements with slight degradation/improvement in Bit Error Rate (BER) depending on needs of the

overlying application. Additionally, various architectural optimizations like pipelining, parallel processing, hardware scheduling, dynamic voltage and frequency scaling have been explored to improve the performance, energy requirements and re-configurability of the design.

Future work in this field can extend to the implementation a soft output MIMO detector [29] which can be fed to Forward Error Correction Schemes like Turbo Codes, LDPC etc. for better BER performance which may be required in many high integrity wireless applications. Iterative Soft MIMO detector with feedback from FEC decoding stage will be an interesting design challenge. Configurability of MIMO architectures for multiple antenna configurations which are required for future generation standards can be explored as well.

REFERENCES

[1] S. Cherry, "Edholm's law of bandwidth," *IEEE Signal Proc. Magazine*, vol. 41, no. 7, pp. 58–60, July 2004.

[2] G. Foschini and M. Gans, "On limits of wireless communications in a fading environment when using multiple antennas," *Wireless Personal Communications*, vol. 6, no. 3, pp. 311–334, 1998.

[3] A. Paulraj, R. Nabar, and D. Gore, *Introduction to Space-Time Wireless Communications*. New York: Cambridge University Press, 2003.

[4] A. Burg, "VLSI Circuits for MIMO communication systems," A Dissertation at ETH Zurich, 2006.

[5] S. Haene, A. Burg, D. Perels, P. Luethi, N. Felber, and W. Fichtner, "Silicon implementation of an MMSE-based soft demapper for MIMO-BICM," in *Proc. IEEE Int. Symp. on Circuits and Systems (ISCAS)*, May 2006, pp 2597-2600.

[6] A. Burg, N. Felber, and W. Fichtner, "A 50 Mbps 4 × 4 maximum likelihood decoder for multiple-input multiple-output systems with QPSK modulation," in *Proc. IEEE Int. Conf. on Electronics, Circuits, and Systems (ICECS)*, vol. 1, Dec. 2003, pp. 332–335.

[7] M. Wenk, M. Zellweger, A. Burg, N. Felber, and W Fichtner, " K-best MIMO detection VLSI architectures achieving up to 424 Mbps throughput," in *Proc. of IEEE Int. Symp. on Circuits and Systems,* May 22, 2006, pp. 1151-1154.

[8] A. Burg, M. Wenk, M. Zellweger, M. Wegmueller, N. Felber, and W. Fichtner, "VLSI implementation of the sphere decoder algorithm," in *Proc. IEEE European Solid-State Circuits Conf. (ESSCIRC)*, Sept. 2004, pp. 303-306.

[9] A. Burg, M. Borgmann, M. Wenk, M. Zellweger, W. Fichtner, and H. Bolcskei, "VLSI implementation of MIMO detection using the sphere decoder algorithm," *IEEE J. Solid-State Circuits*, vol. 40, issue 7, pp. 1566-1577, 2005.

[10] A. Burg, M. Borgmann, C. Simon, M. Wenk, M. Zellweger, and W. Fichtner, "Performance tradeoffs in the VLSI implementation of the sphere decoding algorithm," *in Proc. IEEE 3G Mobile Comm. Tech. Conf.*, Oct. 2004, pp. 93-97.

[11] P.Bhagawat, R.Dash, and G.Choi, "Architecture for reconfigurable MIMO detector and its FPGA Implementation," in *Proc. of 15th IEEE Int. Conf. on Electronic Circuit and Systems (ICECS)*, Malta, 2008, pp. 61-64.

[12] P.Bhagawat, R.Dash, and G.Choi, “High performance on the fly reconfigurable MIMO detector,” in *Proc. 42nd Asilomar Conf. on Signals, Systems and Computers*, Pacific Grove, California, 2008, pp. 1849-1850.

[13] P.Bhagawat, R.Dash, and G.Choi, "Array like runtime reconfigurable MIMO detectors for 802.11n WLAN: A design case study,” in *Proc. of 14th Asia and South Pacific Design Automation Conference (ASP-DAC)*, Japan, 2009, pp. 751-756.

[14] P.Macken, M.Degrauwe, M. van Paemel, and H.Oguesy, “A voltage reduction technique for digital systems,” in *Proc. of 37th IEEE International Solid-State Circuits Conference(ISSCC)*, San Francisco, CA, 1990, pp. 238-239.

[15] Tadahiro Kuroda, Kojiro Suzuki, Shinji Mita, Shinji Mita, Fumihiko Sano, et al. “Variable supply-voltage scheme for low-power high-speed CMOS digital design,” *IEEE J. of Solid-State Circuit*, vol. 33, no. 3, pp. 359-378, May 1994.

[16] P. Bhagawat, S. Ekambavanan, S. Das, G. Choi, and S. Khatri, “VLSI implementation of a staggered sphere decoder design for MIMO Detection,” in *Proc. of 45th Annual Allerton Conf.*, University of Illinois at Urbana-Champaign, 2007, pp. 228-235.

[17] Z. Guo and P. Nilsson, "Algorithm and implementation of the K-best sphere decoding for MIMO detection," *IEEE Journal on Selected Areas in Communications*, vol. 24, no. 3, pp. 491-503, March 2006.

[18] H. Wang, P. Leray, and J. Palicot, *Reconfigurable Computing: Architectures and Applications*. Berlin: Springer, 2006.

[19] L. Barbero and J.Thompson, "Rapid prototyping of a fixed-throughput sphere decoder for MIMO systems," in *Proc. of IEEE International Conf. on Comm.* (ICC '06), Istanbul, Jun. 2006, vol. 7, pp. 3082-3087.

[20] C.Hess, M.Wenk, A.Burg, P.Luethi, C.Studer, N.Felber, and W.Fichtner, "Reduced-complexity MIMO detector with close-to-ML error rate performance," in *Proc. of Great Lakes Symposium on VLSI*, 2007, pp. 200-203.

[21] C. P. Schnorr and M. Euchner, "Lattice basis reduction: Improved practical algorithms and solving subset sum problems," Mathematical Programming, vol. 66, pp. 181–199, 1994.

[22] H.Wang, P.J.Delahaye, P. Leray, and J.Palicot, "Managing dynamic reconfiguration on MIMO Decoder," in *Proc.* of *Parallel and Distributed Processing Symposium*, March 2007, pp. 26-30.

[23] Huang, X., Liang, C., and 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.

[24] W. Wolniansky, G. Foschini, G. Golden, and R. Valenzuala, "V-BLAST: An architecture for realizing very high data rates over the rich-scattering wireless channel," in *Proc. of IEEE Int. Symp. on Signals Systems and Electronics (ISSSE)*, 1998, pp.295-300.

[25] R. Jenkal, and W. Rhett Davis, "An architecture for energy efficient sphere decoding," in *Proc. of International Symposium on Low Power Electronics and Design* (ISLPED), Portland, Oregon, Aug 27-29, 2007, pp. 267-270.

[26] Shariat-Yazdi, and R.Kwasniewski, "Challenges in the design of next generation WLAN terminals," in *Proc. of Canadian Conf. on Electrical and Computer Engineering (CCECE)*, April 2007, pp. 1483-1486.

[27] M.Pedram, and J.Rabaey, *Power Aware Design Methodologies*. Norwell, MA: Kluwer Academic Publishers, 2002.

[28] R.Jejerikar, C.Pereira, and R.Gupta, "Leakage aware dynamic voltage scaling for real-time embedded systems," in *Proc. of Design Automation Conference*, 2004, pp. 275-280.

[29] P.Bhagawat, R.Dash, and G.Choi, "Dynamically reconfigurable soft output MIMO detector," in *Proc. of 26th IEEE International Conference on Computer Design (ICCD),* 2008, Lake Tahoe, CA**,** pp. 68-73.

VITA

Name: Rajballav Dash

Address: NVIDIA Corp., 2701 San Tomas Expy, Santa Clara, CA-95050.

Email Address: rajballavdash@gmail.com

Education:

M.S. in Electrical and Computer Engineering — Dec, 2009
Texas A&M University, Texas, U.S.A

B.Tech. in Electronics & Inst. Engineering — May, 2007
N.I.T. (formerly R.E.C.), Rourkela, India.

Research Interests:

VLSI Design, Communication Systems, Electronic Design Automation

Related Peer-Reviewed Publications:

[1] Pankaj Bhagabat, **Rajballav Dash**, Gwan Choi, "Systolic Like Soft-Detection Architecture for 4x4 64-QAM MIMO System," *Design Automation and Test in Europe (DATE09)*, Nice, France. Apr 22-23, 2009.

[2] **Rajballav Dash**, Rajesh Garg, Sunil Khatri, Gwan Choi, "SEU Hardened Clock Regeneration Circuits," *10th International Symposium on Quality Electronic Design (ISQED)* San Jose, CA. Mar 16-18 2009.

[3] Pankaj Bhagabat, **Rajballav Dash**, Gwan Choi, "High Performance On the Fly Reconfigurable MIMO Detector," *42nd Asilomar Conference on Signals, Systems and Computers*, Oct 26th - 29th, 2008, Pacific Grove, California, USA.

[4] Pankaj Bhagabat, **Rajballav Dash**, Gwan Choi, "Array Like Runtime Reconfigurable MIMO Detectors for 802.11n WLAN: A Design Case Study," *14th Asia and South Pacific Design Automation Conference (ASP-DAC)*, Jan 19-22, 2009, Yokohama, Japan.

[5] Pankaj Bhagabat, **Rajballav Dash**, Gwan Choi, "Dynamically Reconfigurable Soft Output MIMO Detector," *26th IEEE International Conference on Computer Design (ICCD) 2008*, Oct 12th-15th, Lake Tahoe, California, USA.

[6] Pankaj Bhagabat, **Rajballav Dash**, Gwan Choi, "Systolic Like Soft-Detection Architecture for 4x4 64-QAM MIMO System," *Design Automation and Test in Europe (DATE09)*, Nice, France. Apr 22-23, 2009.