J Sign Process Syst
DOI 10.1007/s11265-009-0360-5

Journal of Signal Processing Systems
Online First Version; 8 April 2009

A High Throughput Configurable SDR Detector for Multi-user MIMO Wireless Systems

Kiarash Amiri · Joseph R. Cavallaro · Chris Dick · Raghu Mysore Rao

Received: 19 October 2008 / Revised: 12 March 2009 / Accepted: 12 March 2009

Abstract Spatial division multiplexing (SDM) in MIMO technology significantly increases the spectral efficiency, and hence capacity, of a wireless communication system: it is a core component of the next generation wireless systems, e.g. WiMAX, 3GPP LTE and other OFDM-based communication schemes. Moreover, spatial division multiple access (SDMA) is one of the widely used techniques for sharing the wireless medium between different mobile devices. Sphere detection is a prominent method of simplifying the detection complexity in both SDM and SDMA systems while maintaining BER performance comparable with the optimum maximum-likelihood (ML) detection. On the other hand, with different standards supporting different system parameters, it is crucial for both base station and handset devices to be configurable and seamlessly switch between different modes without the need for separate dedicated hardware units. This challenge emphasizes the need for SDR designs that target the handset devices. In this paper, we propose the architecture and FPGA realization of a *configurable* sort-free sphere detector, *Flex-Sphere*, that supports 4, 16, 64-QAM modulations as well as a combination of 2, 3 and 4 antenna/user configuration for handsets. The detector provides a data rate of up to 857.1 Mbps that fits well within the requirements of any of the next generation wireless standards. The algorithmic optimizations employed to produce an FPGA friendly realization are discussed.

Keywords SDR · Configurable · MIMO · Multi-user · Sphere detection

K. Amiri (✉) · J. R. Cavallaro
Rice University, Houston, TX, USA
e-mail: kiaa@rice.edu

J. R. Cavallaro
e-mail: cavallar@rice.edu

C. Dick · R. M. Rao
Xilinx, San Jose, CA, USA
e-mail: chris.dick@xilinx.com

R. M. Rao
e-mail: raghu.rao@xilinx.com

1 Introduction

Multiple-input multiple-output (MIMO) communication systems and spatial division multiplexing (SDM) have recently drawn significant attention as a means to achieve tremendous gains in system capacity and link reliability. Moreover, spatial division multiple access (SDMA) has recently received attention for its promise to increase the sum data rate of different users in wireless networks, and creating a virtual MIMO between multiple users and a base station.

The optimal hard decision detection, in terms of BER performance, for all MIMO wireless systems is the maximum likelihood (ML) detector. However, direct implementation of ML grows exponentially with the number of antennas and the modulation scheme, making its ASIC or FPGA implementation infeasible for all but low-density modulation schemes using a small number of antennas. Sphere detection [1], and its K-best variation, has been proposed [2], analyzed [3] and implemented [4–9].

As MIMO solutions become more popular and are incorporated into different wireless standards, such as IEEE 802.11n, IEEE 802.16e and upcoming 3GPP

Published online: 08 April 2009

LTE, it is crucial to investigate methods to further reduce the complexity of detection while maintaining high BER performance. Conventional K-best MIMO detectors typically require long delay cycles for sorting steps. For instance, for a multi-stage real-valued based K-best detector for a 16-QAM MIMO system, a bubble sorter needs more than 40 cycles if the detector parameter, K, is set to 10. This long list size introduces a large delay for the processing of the next stage. Moreover, in order to achieve higher reliability, it is important to come up with a cost-free ordering scheme that would lead to a further error performance improvement of the system.

The wide range of developing wireless standards require handsets to support a wide variety of schemes with their limited available resources. Most upcoming standards, for example, require the handsets to support one to four antennas as well as QPSK, 16-QAM and 64-QAM modulations. This is a challenging task given that they need to be able to work with different standards and protocols. Therefore, given the area and power constraints of SDR handset devices, it is crucial that they are designed on a common hardware platform and utilize their real-time reconfiguration capability to communicate with heterogeneous networks.

This paper presents the architecture and the FPGA implementation of a configurable sphere detector called Flex-Sphere. Flex-Sphere supports three commonly used modulation schemes, 4, 16, 64-QAM, as well as a combination of 2, 3 and 4 antenna and/or user configuration, and can switch between all these parameters in a real-time fashion. These parameters are commonly used in the current and upcoming wireless standards, such as IEEE 802.16 and 3GPP LTE, and thus, were chosen as the implementation guidelines. However, it should be noted that the proposed architecture can be readily extended to higher order systems. The detector provides a data rate of up to 849.9 Mbps. The breadth-first search employed in our realization presents a large opportunity to exploit the parallelism of the FPGA in order to achieve high data rates. Algorithmic modifications to address potential sequential bottlenecks in the traditional breadth-first search-based SD are highlighted in the paper.

The initial results of this work were presented in [10, 11]; the simulation results for the hardware implementation of the full 4×4 system as well as FPGA synthesis results on the WARP platform, are added in this work. The rest of the paper is organized as follows: Section 2 presents the general system model. The proposed FPGA friendly architecture for the SDR MIMO detector is presented in Section 3. The complexity issues and comparisons are discussed in Section 4. Section 5 introduces the model-based design of the configurable Flex-Sphere for the SDR handset. The simulation results for floating point and FPGA fixed point of the system for different parameters are given in Section 6. Finally, the paper concludes with Section 7.

2 System Model

We assume a *virtual* MIMO system with n transmitters each with $L_r, r = 1, ..., n$ antennas such that $M_T = \Sigma_{r=1}^{n} L_r$, and a receiver, e.g. a basestation, with $M_R \geq M_T$ receive antennas. All the transmitters use the same channel to communicate simultaneously with the receiver. The input-output model is captured by

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

where $\widetilde{\mathbf{H}}$ is the complex-valued $M_R \times M_T$ channel matrix, $\widetilde{\mathbf{s}} = [\widetilde{s}_1, \widetilde{s}_2, ..., \widetilde{s}_{M_T}]^T$ is the M_T-dimensional transmitted vector from the n transmitters, where each $\widetilde{s}_j$, $j = 1, ..., M_T$, is chosen from a complex-valued constellation Ω_j of the order $w_j = |\Omega_j|$, $\widetilde{\mathbf{n}}$ is the circularly symmetric complex additive white Gaussian noise vector of size M_R and $\widetilde{\mathbf{y}} = [\widetilde{y}_1, \widetilde{y}_2, ..., \widetilde{y}_{M_R}]^T$ is the M_R-element received vector. Note that we do not restrict all the parallel M_T streams to use the same modulation order; rather, each stream, which corresponds to one of the antennas of one of the users, may be using either the 4, 16 or 64-QAM modulation.

The preceding MIMO equation can be decomposed into real-valued numbers as follows [12]:

$$\mathbf{y} = \mathbf{H}\mathbf{s} + \mathbf{n} \tag{2}$$

corresponding to

$$\begin{pmatrix} \Re(\widetilde{\mathbf{y}}) \\ \Im(\widetilde{\mathbf{y}}) \end{pmatrix} = \begin{pmatrix} \Re(\widetilde{\mathbf{H}}) & -\Im(\widetilde{\mathbf{H}}) \\ \Im(\widetilde{\mathbf{H}}) & \Re(\widetilde{\mathbf{H}}) \end{pmatrix} \begin{pmatrix} \Re(\widetilde{\mathbf{s}}) \\ \Im(\widetilde{\mathbf{s}}) \end{pmatrix} + \begin{pmatrix} \Re(\widetilde{\mathbf{n}}) \\ \Im(\widetilde{\mathbf{n}}) \end{pmatrix} \tag{3}$$

with $M = 2.M_T$ and $N = 2.M_R$ presenting the dimensions of the new model.

We call the ordering in Eq. 2, the conventional ordering. Using the conventional ordering, all the computations can be performed in real values, which would simplify the implementation complexity. Note that after real-valued decomposition, each $s_i, i = 1, ..., M$, in $\mathbf{s}$ is chosen from a set of real numbers, Ω_i', with $w_i' = \sqrt{w_i}$ elements. For instance, for a 64-QAM modulation, each s_i can take any of the values in the set $\Omega' = \{\pm 7, \pm 5, \pm 3, \pm 1\}$.

The general optimum detector for such a system is the maximum-likelihood (ML) detector which minimizes $\| \mathbf{y} - \mathbf{H}\mathbf{s} \|^2$ over all the possible combinations of the $\mathbf{s}$ vector. Notice that for high order modulations

and large number of antennas, this detection scheme incurs an exhaustive exponentially growing search among all the candidates, and is not practically feasible in a MIMO receiver. However, it is shown that using the QR decomposition of the channel matrix, the distance norm can be simplified [13] as follows:

$$D(\mathbf{s}) = \| \mathbf{y} - \mathbf{Hs} \|^2$$
$$= \| \mathbf{Q^H y} - \mathbf{Rs} \|^2 = \sum_{i=M}^{1} |y_i' - \sum_{j=i}^{M} R_{i,j} s_j|^2 \quad (4)$$

where $\mathbf{H} = \mathbf{QR}$, $\mathbf{QQ^H} = \mathbf{I}$ and $\mathbf{y}' = \mathbf{Q^H y}$. Note that the transition in Eq. 4 is possible through the fact that $\mathbf{R}$ is an upper triangular matrix.

The norm in Eq. 4 can be computed in M iterations starting with $i = M$. When $i = M$, i.e. the first iteration, the initial partial norm is set to zero, $T_{M+1}(\mathbf{s}^{(M+1)}) = 0$. Using the notation of [4], at each iteration the Partial Euclidean Distances (PEDs) at the next levels are given by

$$T_i(\mathbf{s}^{(i)}) = T_{i+1}(\mathbf{s}^{(i+1)}) + |e_i(\mathbf{s}^{(i)})|^2 \quad (5)$$

with $\mathbf{s}^{(i)} = [s_i, s_{i+1}, ..., s_M]^T$, and $i = M, M-1, ..., 1$, where

$$|e_i(\mathbf{s}^{(i)})|^2 = |y_i' - R_{i,i}s_i - \sum_{j=i+1}^{M} R_{i,j}s_j|^2 \quad (6)$$
$$= |b_{i+1}(\mathbf{s}^{(i+1)}) - R_{i,i}s_i|^2. \quad (7)$$

One can envision this iterative algorithm as a tree traversal with each level of the tree corresponding to one i value, and each node having w_i' children.

The tree traversal can be performed in a breadth-first manner. At each level, only the best K nodes, i.e. the K nodes with the smallest T_i, are chosen for expansion. This type of detector is generally known as the K-best detector. Note that such a detector requires sorting a list of size $K \times w'$ to find the best K candidates. For instance, for a 16-QAM system with $K = 10$, this requires sorting a list of size $K \times w' = 10 \times 4 = 40$ at most of the tree levels. This introduces a long delay for the next processing block in the detector unless a highly parallel sorter is used. Highly parallel sorters, on the other hand, consist of a large number of compare-select blocks, and result in dramatic area increase.

3 Flex-Sphere SDM/SDMA Detector

In order to simplify the sorting step, which significantly reduces the delay of the detector, we propose a novel MIMO detector. This detector is based on a sort-free strategy, and utilizes a new modified real-valued decomposition ordering (M-RVD) scheme.

3.1 Tree Traversal for Flex-Sphere Detection

In order to address the sorting challenge, we propose using a sort-free detector. With this technique, the long sorting operation is effectively simplified to a minimum-finding operation. The detailed steps of this algorithm are described below:

Input: $\mathbf{R}$, $\mathbf{y}'$
$T_{M+1}(\mathbf{s}^{(M+1)}) = 0$
$\mathcal{L} \leftarrow \emptyset$
$\mathcal{L}' \leftarrow \emptyset$
$i \leftarrow M$
\\ Full expansion of the first level:
- Compute T_i with Eq. 5,
- $\mathcal{L} \leftarrow \{(\mathbf{s}^{(i)}, T_i(\mathbf{s}^{(i)}))_j | j = 1, ..., w'\}$
- $i \leftarrow i - 1$
\\ Full expansion of the second level:
- **for** each $(\mathbf{s}^{(i+1)}, T_{i+1}(\mathbf{s}^{(i+1)})) \in \mathcal{L}$, **repeat**
  - compute $(\mathbf{s}^{(i)}, T_i(\mathbf{s}^{(i)}))_j$ children pairs, $j = 1, ..., w'$
  - $\mathcal{L}' \leftarrow \mathcal{L}' \cup \{(\mathbf{s}^{(i)}, T_i(\mathbf{s}^{(i)}))_j | j = 1, ..., w'\}$
- **end**
- $\mathcal{L} \leftarrow \mathcal{L}'$
- $\mathcal{L}' \leftarrow \emptyset$
\\ Minimum-based expansion of the next levels:
- **for** $i = M - 2$ down to $i = 1$, **repeat**
  - **for** each $(\mathbf{s}^{(i+1)}, T_{i+1}(\mathbf{s}^{(i+1)})) \in \mathcal{L}$, **repeat**
    - compute $(\mathbf{s}^{(i)}, T_i(\mathbf{s}^{(i)}))_j$ children pairs, $j = 1, ..., w'$
    - $(\mathbf{s}^{(i)}, T_i(\mathbf{s}^{(i)}))_{min} \leftarrow \underset{\{(\mathbf{s}^{(i)}, T_i(\mathbf{s}^{(i)}))_j | j=1,...,w'\}}{\text{argmin}} T_i(\mathbf{s}^{(i)})$
    - $\mathcal{L}' \leftarrow \mathcal{L}' \cup \{(\mathbf{s}^{(i)}, T_i(\mathbf{s}^{(i)}))_{min}\}$
  - **end**
  - $\mathcal{L} \leftarrow \mathcal{L}'$
  - $\mathcal{L}' \leftarrow \emptyset$
  - $i \leftarrow i - 1$
- **end**

- $(\mathbf{s}^{(i)}, T_i(\mathbf{s}^{(i)}))_{detected} \leftarrow \underset{\mathcal{L}}{\text{argmin}} \; T_i(\mathbf{s}^{(i)})$

An example of this algorithm is illustrated in Fig. 1 for a virtual 4×4, 64-QAM system. Note that as described above, the first two levels are fully expanded to guarantee high performance; whereas for the following levels, only the best candidate in the children list of a parent node is expanded. In other words, after passing the first two levels, w_{M_T} nodes are expanded, and for each of those w_{M_T} nodes, the best child node among its w_M' children nodes is selected as the survived node. Therefore, the new node list would contain w_{M_T} nodes in the third level. These w_{M_T} nodes are expanded in a similar way to the forth level, and this procedure continues until the very last level, where the minimum-distance node is taken as the detected node.

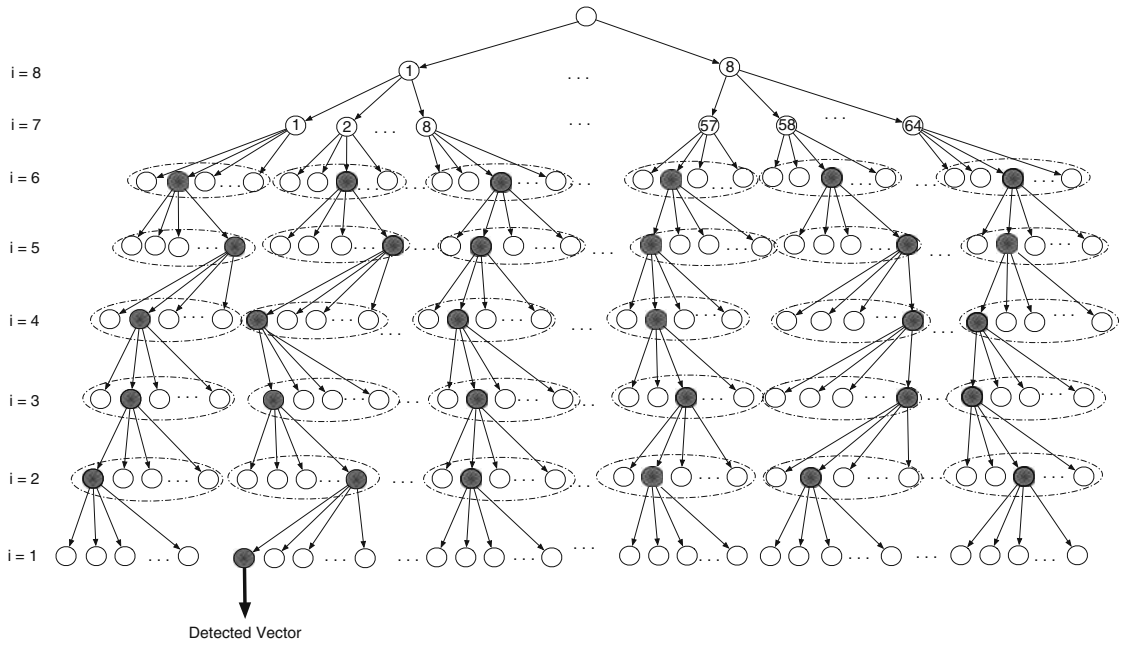

Figure 1 Flex-Sphere algorithm for a 64-QAM, 4×4 system. The topmost two levels are fully expanded. The nodes marked with black are the minimum in their own set, where each set is denoted by dashed line. Note that because of the real-valued decomposition, each node has only $\sqrt{64} = 8$ children. Also, the number of tree levels are $M = 2 \times M_T = 8$.

Moreover, from the Schnorr-Euchner (SE) ordering [14], we know that finding

$$(\mathbf{s}^{(i)}, T_i(\mathbf{s}^{(i)}))_{min} \leftarrow \underset{\{(\mathbf{s}^{(i)}, T_i(\mathbf{s}^{(i)}))_j | j=1,\ldots,w_i'\}}{\operatorname{argmin}} T_i(\mathbf{s}^{(i)})$$

basically corresponds to finding the real-valued constellation point closest to $\frac{1}{R_{ii}} b_{i+1}(\mathbf{s}^{(i+1)})$; see Eq. 7. Thus, the long sorting of K-best is avoided.

3.2 Modified Real-Valued Decomposition (M-RVD) Ordering

For the sort free detector described in the preceding section, we propose using a novel real-valued decomposition (M-RVD) ordering which improves the BER performance compared to the ordering given in Eq. 2. The new decomposition is summarized as:

$$\widehat{\mathbf{y}} = \widehat{\mathbf{H}}\widehat{\mathbf{s}} + \widehat{\mathbf{n}} \tag{8}$$

or,

$$\begin{pmatrix} \Re(\widetilde{y}_1) \\ \Im(\widetilde{y}_1) \\ \Re(\widetilde{y}_2) \\ \Im(\widetilde{y}_2) \\ . \\ . \\ . \\ \Re(\widetilde{y}_{M_R}) \\ \Im(\widetilde{y}_{M_R}) \end{pmatrix} = \widehat{\mathbf{H}} \begin{pmatrix} \Re(\widetilde{s}_1) \\ \Im(\widetilde{s}_1) \\ \Re(\widetilde{s}_2) \\ \Im(\widetilde{s}_2) \\ . \\ . \\ . \\ \Re(\widetilde{s}_{M_T}) \\ \Im(\widetilde{s}_{M_T}) \end{pmatrix} + \begin{pmatrix} \Re(\widetilde{n}_1) \\ \Im(\widetilde{n}_1) \\ \Re(\widetilde{n}_2) \\ \Im(\widetilde{n}_2) \\ . \\ . \\ . \\ \Re(\widetilde{n}_{M_R}) \\ \Im(\widetilde{n}_{M_R}) \end{pmatrix} \tag{9}$$

where $\widehat{\mathbf{H}}$ is the permuted channel matrix of Eq. 3 whose columns are reordered to match the other vectors of the new decomposition ordering in Eq. 8. It is worth noting that since the difference between RVD and M-RVD is the grouping of the signals, there is no extra computational cost associated with this novel ordering.

Note that with the modified real-valued decomposition (M-RVD) ordering, the first two levels correspond to the in-phase and quadrature parts of the same complex symbol; whereas in the conventional real-valued

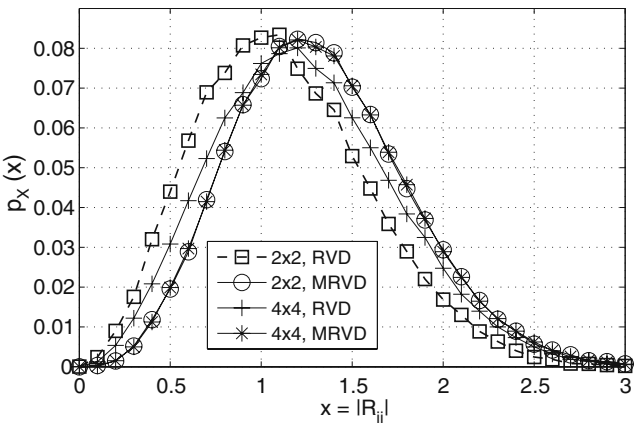

Figure 2 Probability density function of the $R_{6,6}$ for 4×4 and $R_{2,2}$ for 2×2 when either conventional RVD or the proposed RVD are used. Note the shift of the curves when M-RVD is used.

decomposition scenario, the first two levels of the tree correspond to the quadrature parts of two different complex symbols. A careful look at the tree traversal scheme of the preceding section shows that since the first two levels of the tree are fully expanded, the error performance of the scheme heavily depends on the third level of the tree. Therefore, rather than using the magnitude of $R_{M,M}$ as a metric to choose the decomposition ordering scheme, which justifies the conventional real-valued decomposition (RVD) [15], we need to look at the behavior of the third lowest diagonal element of the **R** matrix. As demonstrated in Fig. 2, there is an increase in the magnitude of $R_{M-2,M-2}$ when using M-RVD, hence M-RVD is a better choice than the conventional RVD. The impact of M-RVD on the BER performance is discussed in the next sections.

4 Complexity Comparison

In order to compare the complexity of the proposed MIMO detector, described in the preceding section, versus the conventional K-best technique, we consider the number of operations, the relative latency reduction, and the architecture advantages of the proposed detector.

4.1 Number of Operations

In this section, we compute the number of operations required to complete the detection process. Since the channel matrix typically changes at a much slower rate than the received signal vector, we make the assumption that simple channel matrix operations, e.g. $R_{ij}s_j$ computations, are performed in a separate preprocessing unit. Note that this simply involves shift-add operations with $s_j \in \Omega'$. Also, as suggested in [4], we make the assumption that all the PED norms are approximated by ℓ^1-norms to avoid the squarers and multipliers. Therefore, the only major high rate detector operations, are compare-select for either sorting or minimum-findings, addition and multiplication.

Note that in order to achieve minimum latency, we make the assumption that both detectors use cascaded minimum-finders to sort a list. Therefore, in order to find the best K elements of a list of size l; K cascaded minimum finders are required. So, the number of operations required to sort the best K candidates of a list of size l, denoted by $f_K(l)$ in Table 1, is given by

$$f_K(l) = K \times l - \frac{K(K+1)}{2}. \tag{10}$$

Given the above assumptions, the total number of operations for the K-best scenario and the proposed Flex-Sphere scheme are given in Table 1.

1. Compare-Select: The K-best method requires finding the best K nodes among Kw' candidates in $(M-3)$ of the levels, i.e. $f_K(Kw')(M-3)$ operations; whereas, the Flex-Sphere only needs to compute the minimum nodes among w' nodes for w' groups in those $M-3$ levels, i.e. $w' f_1(w')(M-3)$ operations. Moreover, the best node is chosen among Kw' nodes, i.e. $f_1(Kw')$ operations, in the last level of the K-best tree, and among $w'^2 = w$ nodes, i.e. $f_1(w)$ operations, in the Flex-Sphere. While the second level requires finding the best K nodes among the w children, i.e. $f_K(w)$ operations, in the K-best structure, the Flex-Sphere does not need such sorting since it is fully expanding that level.
2. Addition: For the K-best scenario, assuming that $K > w'$, which ensures higher performance, and based on Eq. 6, level $i = M$ requires w' addition operations, level $i = M-1$ requires $w'(1+w')$ operations, and the rest of the levels each need $K(M-i+w')$ addition operations. Moreover, based on Eq. 5, level $i = M-1$ needs w addition operations, and each of the remaining levels, $i = M-1, ..., 1$, need Kw' operations. Therefore, the total number of adders needed for the K-best detection scheme is given in Table 1. A similar

Table 1 Comparison of the latency and the operation counts between the conventional K-best and the proposed Flex-Sphere detector.

	K-best	Flex-Sphere detector
Compare-select	$f_K(Kw')(M-3) + f_K(w) + f_1(Kw')$	$w' f_1(w')(M-3) + f_1(w)$
Addition	$2w' + 2w + 2Kw'(M-2) + K(M(M-1)/2 - 1)$	$2w' + w + w.w'(M(M+1)/2 - 3)$
Multiplication	$w' + w + Kw'(M-2)$	–
Latency	$\sum_{m=0}^{K-1} \lceil \log(Kw' - m) \rceil$	$\lceil \log w' \rceil$
Example (16-QAM, $K = 4$)	16	2
Example (16-QAM, $K = 5$)	24	2

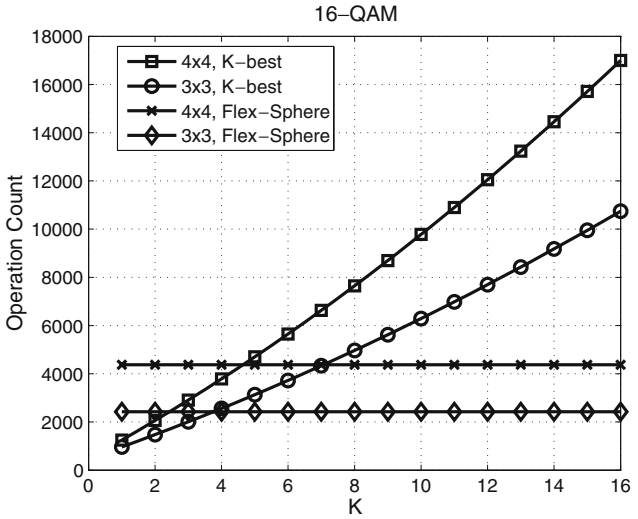

Figure 3 Comparison of the number of operations between the proposed scheme and K-best for different values of K and different number of antennas. The 16-QAM modulation is assumed.

approach will yield the total number of additions required for the Flex-Sphere detection.

3. Multiplication: The Flex-Sphere uses ℓ^1-norms, and thus, does not need to use the FPGA multipliers; whereas, the K-best scheme needs to compute w' ℓ^2-norms in the first level, w norms in the second level, and Kw' norms in the remaining $(M-2)$ levels.

In order to compute the final operation count, comparators are assumed to have unit complexity, and adders to have twice complexity as that of comparators. Multipliers are needed to implement the squarers, and for the wordlengths that we are interested in, i.e. 16 bits, they can be assumed to be ten times more complex than additions. It is worth noting that other relative complexity coefficients would yield similar general results. Based on these relative complexities, the number of operations are plotted for different numbers of antennas in Fig. 3. The operation count increases for higher K values because higher K means higher number of visited nodes per level; therefore, higher K requires larger computations. Note that except for small K values, the computation overhead of the conventional K-best scheme is considerably more than the proposed Flex-Sphere scheme. More details on the BER performance comparisons will be presented in Section 4.4.

4.2 Latency

High latency decreases the data rate in feedback based receivers. For instance, for iterative detector/decoder structures, where the detector uses the feedback data from the decoder to improve the detection performance, higher detection/decoding latency reduces the data rate significantly. A similar argument applies to the overall receiver throughput when the interaction between the physical layer and MAC layer takes more cycles due to the higher physical layer latency. We compare the latency overhead of our proposed detector versus the conventional K-best detector, and show that the Flex-Sphere technique introduces significant latency reduction.

Note that if the detectors are fully parallelized for enhancing data rates, the conventional K-best detector requires K successive minimum finders. The first minimum finder needs to find the minimum among Kw' candidates, therefore, has a latency of $Kw'-1$. The second one needs to find the minimum among $Kw'-1$, therefore has a latency of $Kw'-2$, and so on. The proposed Flex-Sphere detector, however, requires only one level of minimum finder as it only needs to find the minimum, i.e. sorting with $K=1$. Thus, if we assume full parallelism for both types of detectors, the latency of the sorter that connects one of the middle levels of the tree to the next level is given in Table 1.

Notice the significant latency reduction that the proposed Flex-Sphere detector promises for the sorting after each level. Also, note that Table 1 represents only the latency of one level; thus, for a 4×4 system, there would be $M-3=2M_T-3=5$ of such sorters, see Table 1.

4.3 Architecture

The common K-best sorting requires a bubble-sort architecture [8]. In this architecture, all the nodes need to be passed into the sorter sequentially, and the process of the next level of the tree can not start until all the $K\times w'$ nodes are passed through the sequential sorter. Even semi-parallel sorters, still require large area and cycles, to finish the detection process, see Table 1 and Fig. 3. With the Flex-Sphere technique, all the long size sortings are avoided. Moreover, the Flex-Sphere technique is amenable to parallelizing with less overhead than the K-best technique.

4.4 Simulation Results

For the BER simulations, the Rayleigh fading channel model is assumed, and the channel matrix is independent for each new transmission. The BER results of 4×4 and 3×3 systems are compared for a 16-QAM modulation scheme. Note that in order to conduct a fair performance comparison, the K values are chosen such that the K-best technique has similar number of

operations as that of the proposed Flex-Sphere scheme, see Fig. 3. Therefore, based on the results shown in Fig. 3 and Table 1, K is set to 5 and 4 for the 4×4 and 3×3 systems, respectively.

The BER simulation results of Fig. 4 suggest that the proposed Flex-Sphere scheme can improve the BER performance more than 5 dB compared to the conventional K-best technique in higher SNR regimes. Note that it was shown in the preceding sections that for a 4×4 case, the $K = 5$ scheme requires similar computational complexity as that of the Flex-Sphere scheme, and it requires 12 times more latency for sorting in each level compared to the proposed sort-free scheme. A similar argument holds for a 3×3 system when $K = 4$. It is also worth noting that in both cases, the M-RVD ordering plays an important role in improving the performance.

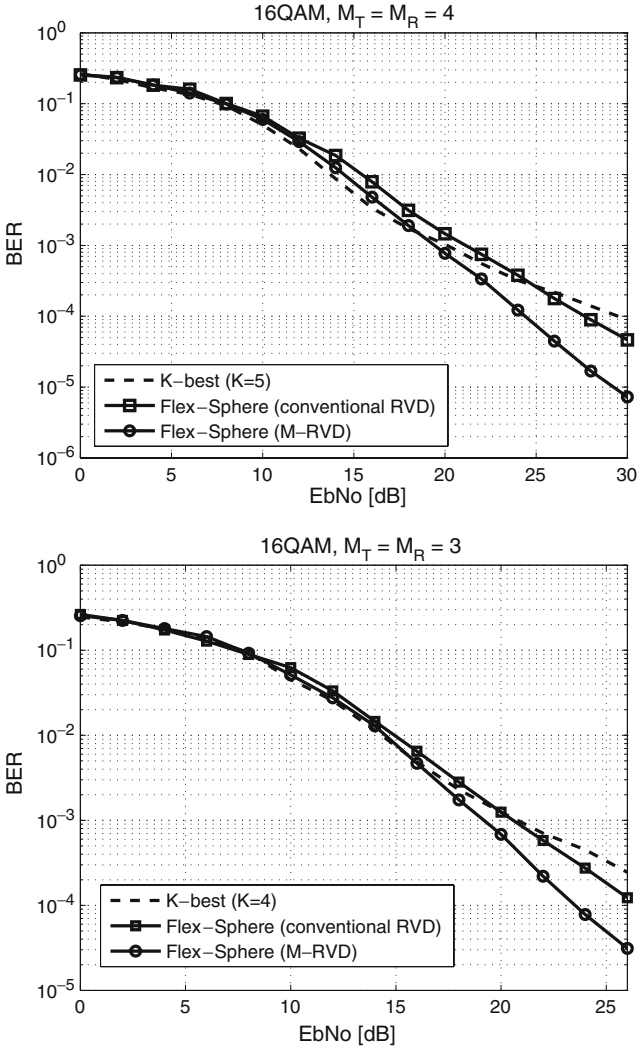

Figure 4 BER performance of the proposed detector with and without the novel ordering (M-RVD) described in Section 3.2 assuming a 16-QAM modulation for both $M_T = M_R = 4$ and $M_T = M_R = 3$. The K-best implementation for $K = 5$ and $K = 4$ has similar computational complexity as that of the sort-free schemes for $M_T = 4$ and $M_T = 3$, respectively.

5 FPGA Design of the Configurable Detector for SDR Handsets

In this section, the main features of the architecture and the FPGA implementation of the SDR handset detector are presented. We use Xilinx System Generator [16] to implement the proposed architecture. In order to support all the different number of antenna/user and modulation orders, the detector is designed for the maximal case, i.e. $M_T \times M_R$, 64-QAM case, and configurability elements are introduced in the design to support different configurations.

5.1 PED Computations

Computing the norms in Eq. 7 is performed in the PED blocks. Depending on the level of the tree, three different PED blocks are used: The PED in the first real-valued level, PED_1, corresponds to the root node in the tree, $i = M = 2M_T = 8$. The second level consists of $\sqrt{64} = 8$ parallel PED_2 blocks, which compute 8 PEDs for each of the 8 PEDs generated by PED_1; thus, generating 64 PEDs for the $i = 7$ level. Followed by this level, there 8 parallel general PED computation blocks, PED_g, which compute the closest-node PED for all 8 outputs of each of the PED_2s. The next levels will also use PED_g. At the end, the Min_Finder unit detects the signal by finding the minimum of the 64 distances of the appropriate level. The block diagram of this design is shown in Fig. 5.

5.2 Configurable Design

In order to ensure the configurability of the Flex-Sphere, it needs to support different M_T as well as different modulation orders for different users. The configurability of the detector is achieved through two input signals, M_T and $q^{(i)}$, which control the number of antennas and the modulation order, respectively. These two inputs can change based on the system parameters at any time during the detection procedure. Therefore, this configurability is a real-time operation.

5.2.1 Number of Antennas

The M_T determines the number of detection levels, and it is set through M_T input to the detector, which in turn, would configure the Min_Finder appropriately.

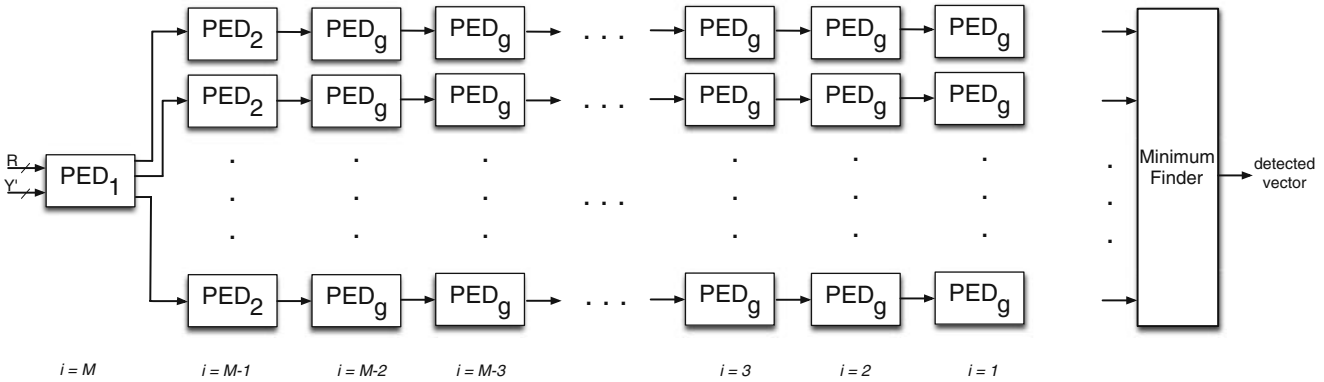

Figure 5 The block diagram of the Flex-Sphere. Note that there are M parallel PEDs at each level. The inputs to the Min_Finder is fed from the appropriate PED block, as described in Section 5.2.1.

Therefore, the minimum finder can operate on the outputs of the corresponding level, and generate the minimum result. In other words, the multiplexers in each input of the Min_Finder block, choose which one of the four streams of data should be fed into the Min_Finder. Therefore, the inputs to the Min_Finder would be coming from the $i = 5, 3$ or 1, if M_T is 2, 3 or 4; respectively, see Fig. 5.

The M_T input can change on-the-fly; thus, the design can shift from one mode to another mode based on the number of streams it is attempting to detect at anytime. Moreover, as will be shown later, the configurability of the minimum finder guarantees that less latency is required for detecting smaller number of streams.

5.2.2 Modulation Order

In order to support different modulation orders per data stream, the Flex-Sphere uses another input control signal $q^{(i)}$ to determine the maximum real value of the modulation order of the i-th level. Thus, $q^{(i)} \in \{1, 3, 7\}$. Moreover, since the modulation order of each level is changing, a simple comparison-thresholding can not be used to find the closest candidate for Schnorr-Euchner [14] ordering. Therefore, the following conversion is used to find the closest SE candidate:

$$\widetilde{s} = g\left(2\left\lceil\frac{b+1}{2}\right\rceil - 1\right) \tag{11}$$

where $\lceil . \rceil$ represents rounding to the nearest integer, $b = (1/R_{ii}) \cdot b_{i+1}$ of Eq. 7, and $g(.)$ is

$$g(x) = \begin{cases} -q^{(i)} & x \le -q^{(i)} \\ x & -q^{(i)} \le x \le q^{(i)} \\ q^{(i)} & x \ge q^{(i)} \end{cases} \tag{12}$$

All of these functions can be readily implemented using the available building blocks of the Xilinx System Generator, see Fig. 6. Note that the multiplications/divisions are simple one-bit shifts.

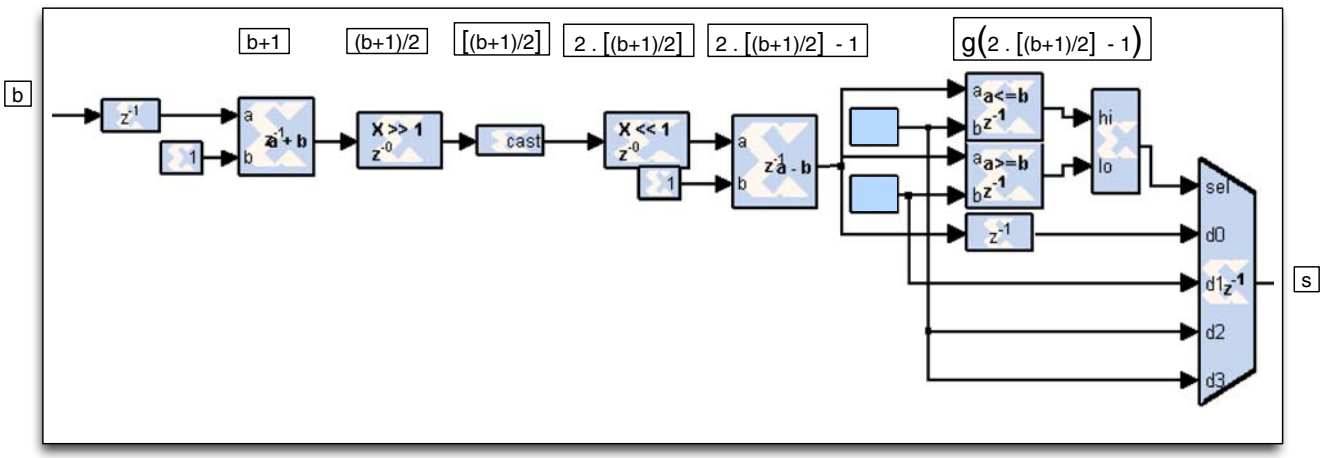

Figure 6 The pipelined System Generator block diagram for Eq. 11 in the PED$_g$ to support different modulation orders.

For the first two levels, which corresponds to the in-phase and quadrature components of the last antenna, the PED of the out-of-range candidates are simply overwritten with the maximum value; thus, they will be automatically discarded during the minimum-finding procedure.

5.3 Modified Real Valued Decomposition (M-RVD)

Using the real-valued decomposition, the two extra adders that are required per each complex multiplication, can be avoided; thus, avoiding the unnecessary FPGA slices on the addition operations. Moreover, while using the complex-valued operations require the SE ordering of [4], which would be a demanding task given the configurable nature of the detector; with the real-valued decomposition, the SE ordering can be implemented more efficiently and simply for the proposed configurable architecture as described earlier. Also, note that even though some of the multiplications can be replaced with shift-adds in an area-optimized ASIC design, as discussed in Section 4; for an FPGA implementation, the appropriate design choice is to use the available embedded multipliers, commonly known as XtremeDSP and DSP48E in Virtex-4 and Virtex-5 devices.

It is noteworthy that if the conventional real-valued decomposition of Eq. 3 were employed; then, the results for a 2×2 system would have been ready only after going through all the in-phase tree levels and the first two quadrature levels. However, with the modified real-valued decomposition (M-RVD), every antennas is isolated from other antennas in two consecutive levels of the tree. Therefore, there is no need to go through the latency of the unnecessary levels. Thus, using the M-RVD technique, offers a latency reduction compared to the conventional real-valued decomposition.

5.4 Timing Analysis

Each of the PED_g blocks are responsible for expanding 8 nodes; thus, the folding factor of the design is $F = 8$. In order to ensure a high maximum clock frequency, several pipelining levels are introduced inside each of the PED computation blocks. The latency of the PED_1, PED_2 and PED_g blocks are 7, 17 and 22, respectively.

Table 2 Latency for different values of M_T.

M_T	Latency
$M_T = 2$	$8 + \text{PED}_1 + \text{PED}_2 + 2 \cdot \text{PED}_g + \text{Min\_Finder} = 84$
$M_T = 3$	$8 + \text{PED}_1 + \text{PED}_2 + 4 \cdot \text{PED}_g + \text{Min\_Finder} = 128$
$M_T = 4$	$8 + \text{PED}_1 + \text{PED}_2 + 6 \cdot \text{PED}_g + \text{Min\_Finder} = 172$

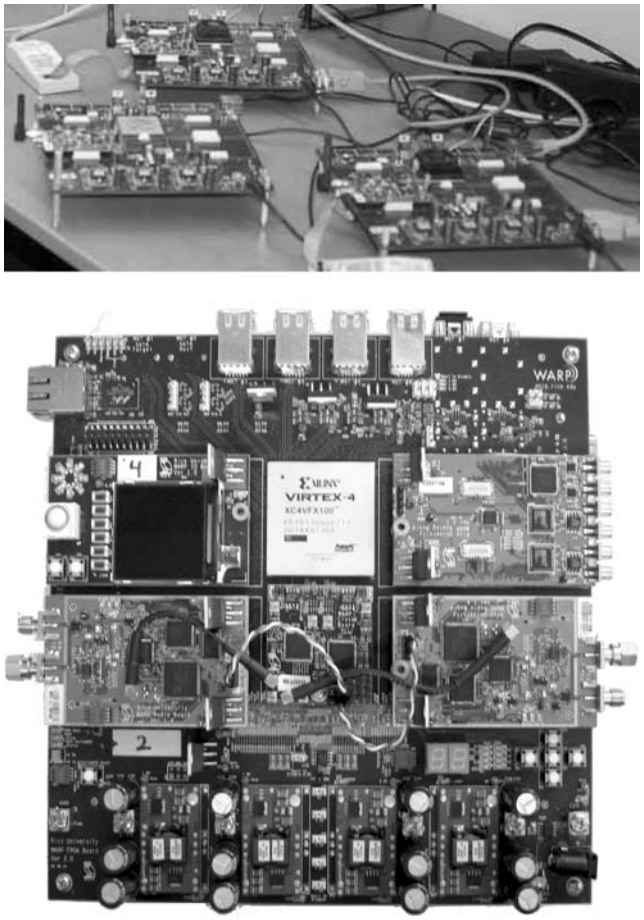

Figure 7 The next generation WARP board with four daughtercard slots. The board can support up to four radio daughtercards.

Note that the larger latency of the PED_g blocks is due to more multiplications required to compute the PEDs of the later levels. The Min_Finder block has a latency of 8.

As mentioned earlier, different values of M_T require different number of tree levels, which incurs different latencies. The latencies of the three different configurations of M_T are presented in Table 2. In computing the latencies, an initial 8 cycles are required to fill up the pipeline path.

Table 3 FPGA resource utilization summary of the proposed Flex-Sphere for the Xilinx Virtex-4, xc4vfx100-10ff1517, device.

No. of antennas	2, 3
Modulation order	{4, 16, 64}-QAM
Max. data rate	562.5 Mbps
Number of slices	18,825/42,176 (44%)
Number of slice FFs	23,961/84,352 (28%)
Number of LUTs	30,297/84,352 (35%)
Number of DSP48E	129/160 (80%)
Max. freq.	250 MHz

Table 4 Comparison of the system support and FPGA resource utilization of the proposed Flex-Sphere vs. optimized FSD-B [18].

Design	Flex-Sphere	Optimized FSD-B [18]
Device	XC5VSX95	XC2VP70
No. of antennas	2, 3, 4	4
Modulation order	{4, 16, 64}-QAM	64-QAM
Max. data rate	857.1 Mbps	450 Mbps
BER $= 10^{-4}$ @ SNR $=$	$= 25$ dB	$= 25$ dB
Number of slices	11,604/14,720 (78 %)	24,815/33,088 (74 %)
Number of registers/FFs	27,115/58,880 (46 %)	39,800/66,176 (60 %)
Number of slice LUTs	33,427/58,880 (56 %)	31,759/66,176 (47 %)
Number of DSP48E/multipliers	321/640 (50 %)	252/328 (88 %)
Number of block RAMs	0 (0 %)	88/328 (26 %)
Max. freq.	285.71 MHz	150 MHz

5.5 Implementation Results on WARP

Wireless Open-access Research Platform (WARP) [17], which is a scalable and extensible programmable wireless platform, is a suitable platform for prototyping the detection algorithms. Each board can support up to four antennas, and if the boards are stacked together to form a bigger node, they can support even more antennas. Several architecture-friendly wireless algorithms for handsets have been implemented and verified on this testbed, see Fig. 7. The new version of this board is based on Virtex-4 FPGA, and Table 3 presents the System Generator implementation results of the Flex-Sphere on a Xilinx Virtex-4 FPGA, xc4vfx100-10ff1517 [16] for 16-bits precision. The maximum number of detectable streams is set to $M_T = 3$. The maximum achievable clock frequency is 250 MHz. Since the design folding factor is set to $F = 8$, the maximum achievable data rate, i.e. $M_T = 3$ and $w_i = 64$, is

$$D = \frac{M_T \cdot \log w}{F} \cdot f_{max} = 562.5 \text{ [Mbps]}. \tag{13}$$

5.6 Implementation Results for $M_T = 4$

Table 4 presents the System Generator implementation results of the Flex-Sphere on a Xilinx Virtex-5 FPGA, xc5vsx95t-3ff1136 [16] for 16-bits precision and $M_T = 4$. The maximum achievable clock frequency is 285.71 MHz. Since the design folding factor is set to $F = 8$, the maximum achievable data rate, i.e. $M_T = 4$ and $w_i = 64$, is

$$D = \frac{M_T \cdot \log w}{F} \cdot f_{max} = 857.1 \text{ [Mbps]}. \tag{14}$$

This table also presents the implementation results of a previously reported 64-QAM, 4×4 system [18]. While the the proposed Flex-Sphere is implemented on a different FPGA device, due its relatively larger size, it can support different number of antennas and modulation orders, and achieves high data rate requirements of various wireless standards.

Table 5 summarizes the data rates for all of the different scenarios of the $M_T = 4$, Virtex-5, implementation.

Table 5 Data rate for different configurations of the 4×4, Table 4, implementation.

	4-QAM	16-QAM	64-QAM
$M_T = 2$	142.7 Mbps	285.7 Mbps	428.4 Mbps
$M_T = 3$	214.1 Mbps	428.4 Mbps	642.7 Mbps
$M_T = 4$	285.7 Mbps	571.4 Mbps	857.1 Mbps

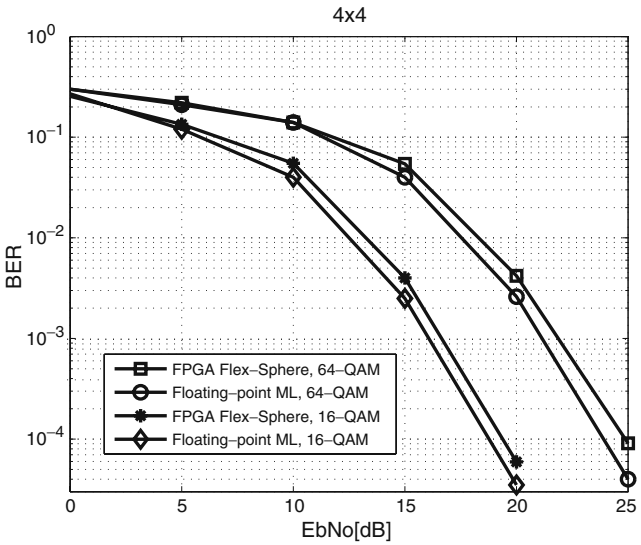

Figure 8 BER plots comparing the performance of the floating-point maximum likelihood (ML) with the the FPGA implementation. Note that the channel pre-processing of [19] is employed to improve the performance.

6 Simulation Results

In this section, we present the simulation results for the Flex-Sphere, and compare the performance of the FPGA fixed-point implementation with that of the optimum floating-point maximum-likelihood (ML) results. Prior to the M-RVD, introduced in Section 3, we employ the channel ordering of [19] to further close the gap to ML. Also, we make the assumption that all the streams are using the same modulation scheme. We assume a Rayleigh fading channel model, i.e. complex-valued channel matrices with the real and imaginary parts of each element drawn from the normal distribution.

In order to ensure that all the antennas in the receiver have similar average received SNR, and none of the users messages are suppressed with other messages, a power control scheme is employed. Figure 8 shows the simulation results for the maximal 4×4 configuration. As can be seen, the proposed hardware architecture implementation performs within, at most, 1 dB of the optimum maximum-likelihood detection.

7 Conclusion and Future Work

In this paper, we presented a configurable architecture for multi-user MIMO detection, which can support different number of antennas and modulation orders required by a wide variety of different standards in a real-time way. The proposed architecture enhances the performance of SDR handsets for next generation wireless standards. We also presented the FPGA implementation results of the 3×3 and 4×4 configurations, and the simulation results suggest that the performance can be made considerably close to the optimum ML detector. It is worth noting that even though the presented results are for hard detection, they can be readily extended to support configurable soft detection scenarios, required for soft iterative detection-decoding schemes [3]. This can be achieved by developing a configurable soft computation block that uses the list of the symbols of the last level for computing the soft information. Comparing the performance of this soft detection strategy with other soft detection strategies forms the next step of the work.

Acknowledgements This work was supported in part by Xilinx Inc., and by NSF under grants EIA-0321266, CCF-0541363, CNS-0551692, and CNS-0619767.

References

1. Fincke, U., & Pohst, M. (1985). Improved methods for calculating vectors of short length in a lattice, including a complexity analysis. *Mathematics of Computation, 44*(170), 463–471.
2. Viterbo, E., & Boutros, J. (1999). A universal lattice decoder for fading channels. *IEEE Transactions on Information Theory, 45*(5), 1639–1642.
3. Hochwald, B., & ten Brink, S. (2003). Achieving near-capacity on a multiple-antenna channel. *I EEE Transactions on Communications, 51*, 389–399.
4. Burg, A., Borgmann, M., Wenk, M., Zellweger, M., Fichtner, W., & Bolcskei, H. (2005). VLSI implementation of MIMO detection using the sphere decoding algorithm. *IEEE Journal of Solid-State Circuits, 40*(7), 1566–1577.
5. Barbero, L. G., & Thompson, J. S. (2006). Performance analysis of a fixed-complexity sphere decoder in high-dimensional MIMO systems. *IEEE Conference on Acoustics, Speech and Signal Processing, 4*, 557–560.
6. Amiri, K., & Cavallaro, J. R. (2006). FPGA implementation of dynamic threshold sphere detection for MIMO systems. *40th Asilomar conf. on signals, systems and computers, Nov*.
7. Guo, Z., & Nilsson, P. (2006). Algorithm and implementation of the K-Best sphere decoding for MIMO detection. *IEEE JSAC, 24*(9), 491–503.
8. Wong, K., Tsui, C., Cheng, R. S., & Mow, W. (2002). A VLSI architecture of a K-best lattice decoding algorithm for MIMO channels. *IEEE International Symposium on Circuits and Systems, 3*, 273–276.
9. Jiang, M., Ng, S. X., & Hanzo, L. (2006). Hybrid iterative multiuser detection for channel coded space division multiple access OFDM systems. *IEEE Transactions on Vehicular Technology, 55*, 115–127.
10. Amiri, K., Dick, C., Rao, R., & Cavallaro, J. R. (2008). Novel sort-free detector with modified real-valued decomposition (M-RVD) ordering in MIMO systems. *Proc. of IEEE Globecom*.
11. Amiri, K., Dick, C., Rao, R., & Cavallaro, J. R. (2008). Flex-sphere: An FPGA configurable sort-free sphere detector for multi-user MIMO wireless systems. *Proc. of SDR Forum, Dec*.
12. Guo, Z., & Nilsson, P. (2005). A 53.3 Mb/s 4×4 16-QAM MIMO decoder in 0.35μm CMOS. *IEEE International Symposium on Circuits and Systems, 5*, 4947–4950.
13. Damen, M. O., Gamal, H. E., & Caire, G. (2003). On maximum likelihood detection and the search for the closest lattice point. *IEEE Transactions on Information Theory, 49*(10), 2389–2402.
14. Schnorr, C. P., & Euchner, M. (1994). Lattice basis reduction: Improved practical algorithms and solving subset sum problems. *Mathematical Programming, 66*(2), 181–191.
15. Burg, A. (2006). *VLSI circuits for MIMO communication systems*. PhD thesis.
16. Xilinx (2008) Xilinx homepage. http://www.xilinx.com/.
17. WARP: http://www.warp.rice.edu/.
18. Barbero, L. G., & Thompson, J. S. (2006). FPGA design considerations in the implementation of a fixed-throughput sphere decoder for MIMO systems. *Field programmable logic and applications, 2006. FPL '06. International conference on, Aug*.
19. Barbero, L. G., & Thompson, J. S. (2006). A fixed-complexity MIMO detector based on the complex sphere decoder. *IEEE 7th workshop on signal processing advances in wireless communications, 2006. SPAWC '06, Jul*.

Kiarash (Kia) Amiri received his B.S. degree in Electrical Engineering from Sharif University of Technology in 2005 and his M.S. degree in Electrical and Computer Engineering from Rice University, Houston, Texas, in 2007. He is currently a PhD candidate in Electrical and Computer Engineering at Rice University where he is a member of the Center for Multimedia Communication (CMC) lab. His research focus is in the area of physical layer design and hardware architecture for wireless communication. During the summer and fall of 2007, Kia worked on developing and implementing MIMO algorithms as part of the Advanced Systems Technology Group in Xilinx, San Jose, CA.

Joseph R. Cavallaro received the B.S. degree from the University of Pennsylvania, Philadelphia, Pa, in 1981, the M.S. degree from Princeton University, Princeton, NJ, in 1982, and the Ph.D. degree from Cornell University, Ithaca, NY, in 1988, all in electrical engineering. From 1981 to 1983, he was with AT&T Bell Laboratories, Holmdel, NJ. In 1988, he joined the faculty of Rice University, Houston, Tex, where he is currently a Professor of electrical and computer engineering. His research interests include computer arithmetic, VLSI design and microlithography, and DSP and VLSI architectures for applications in wireless communications. During the 1996–1997 academic year, he served at the USA National Science Foundation as Director of the Prototyping Tools and Methodology Program. He was a Nokia Foundation Fellow and a Visiting Professor at the University of Oulu, Finland in 2005 and continues his affiliation as an Adjunct Professor there. He is currently the Associate Director of the Center for Multimedia Communication at Rice University. He is a Senior Member of the IEEE. He was Co-chair of the 2004 Signal Processing for Communications Symposium at the IEEE Global Communications Conference and General Co-chair of the 2004 IEEE 15th International Conference on Application-Specific Systems, Architectures and Processors (ASAP).

Chris Dick is the DSP Chief Architect at Xilinx and the engineering manager for the Xilinx Wireless Systems Engineering team. Chris has worked with signal processing technology for over two decades and his work has spanned the commercial, military and academic sectors. Prior to joining Xilinx in 1997 he was a professor at La Trobe University, Melbourne Australia for 13 years and managed a DSP Consultancy called Signal Processing Solutions. Chris holds the positions of adjunct professor at Rice University in Houston and Santa Clara University and he is a senior member of the IEEE.

Chris' work and research interests are in the areas of fast algorithms for signal processing, digital communication, MIMO, OFDM, software defined radios, VLSI architectures for DSP, adaptive signal processing, synchronization, hardware architectures for real-time signal processing, and the use of Field Programmable Arrays (FPGAs) for custom computing machines and real-time signal processing. He holds a bachelor's and PhD degrees in the areas of computer science and electronic engineering.

Raghu Mysore Rao is a Senior Staff Communication Systems Engineer and System Architect with the Wireless Systems Engineering Team, Xilinx Inc. working on digital communication algorithms and architectures for FPGA implementation, in particular MIMO and OFDM systems such as 3GPP-LTE and WiMax. He is an IEEE senior member. He has a Ph.D. in wireless communications from UCLA. He received his B.E degree in Electronics and Communications from The National Institute of Engineering, Mysore, India in 1987 and his M.Tech degree in Computer Science from the University of Hyderabad, Hyderabad, India in 1989. From 1989–1992 he was with Texas Instruments, India developing EDA algorithms for FPGAs. From 1992–1999 he was with Exemplar Logic Inc., initially working on timing analysis and timing optimization algorithms and then as a Director of Engineering responsible for all product development. In 1999 he joined UCLA to pursue a Ph.D. degree in wireless communications. His interests are in digital communication and signal processing algorithms and architectures for their efficient implementation on FPGAs.