An Energy-Efficient 64-QAM MIMO Detector for Emerging Wireless Standards

Nariman Moezzi-Madani and Thorlindur Thorolfsson
Electrical and Computer Engineering
North Carolina State University
Raleigh, USA
{nmoezzi, trthorol}@ncsu.edu

Joseph Crop and Patrick Chiang
Electrical Eng. and Computer Science
Oregon State University
Corvallis,USA
{cropj,pchiang}@eecs.oregonstate.edu

William Rhett Davis
Electrical and Computer Engineering
North Carolina State University
Raleigh, USA
wdavis@ncsu.edu

***Abstract*—A power/area aware design is mandatory for the MIMO (Multi-Input Multi-Output) detectors used in LTE and WiMAX standards. The 64-QAM modulation used in the MIMO detector requires more detection effort compared to the smaller constellation sizes widely implemented in the literature. In this work we propose a new architecture for the K-best detector, which unlike the popular multi-stage architecture used for K-best detectors, implements just one core. Also, we introduce a slight modification to the K-best algorithm that reduces the number of multiplications by 44%, and reduces the total power consumption by 27%, without any noticeable performance degradation. The overall architecture consumes only 24KGate, which is the smallest area compared to the other implementations in the literature. It also results in an at least 4-fold greater throughput-efficiency (Mbps/KiloGate) compared to the other detectors, while consuming a small power. The decoder implemented in a commercial 130nm process provides a data-rate of 107Mbps, and consumes 54.4mW.**

Keywords-MIMO; K-best; single-core; 64-QAM; LTE; WiMAX

I. INTRODUCTION

Multiple-input multiple-output (MIMO) technology increases capacity or diversity. The combination of MIMO and OFDM is a promising wireless access scheme. Different standards such as WiFi, WiMAX and 3GPP LTE have exploited the MIMO-OFDM system, where multiple antennas are used in the transmitter and the receiver. Extraction of transmitted data from the spatially multiplexed received data is a complex process and different algorithms are introduced in the literature to reduce the detection complexity.

Sphere decoders perform closely to the performance of ML. The three different sphere decoders are K-best, depth-first and fixed-complexity SD (FSD). The K-best algorithm avoids the variable/low throughput of the depth-first decoders and also provides a better BER performance than FSD when used for soft-output detection. K-best SD provides a BER performance close to ML, even in low SNRs. This algorithm is also very efficient for soft-output detection because it inherently generates the candidate list required to calculate the LLR (log-likelihood ratio) values for soft-output detection; however the architectures that implement this algorithm consume a large area. Area and power are the limiting factors for mobile applications. For example the 3GPP LTE is a MIMO-OFDM based standard designed for cellular communication systems. The implemented conventional K-best detectors are large and provide a data rate much greater than what is necessary for this standard.

We propose a single core architecture for the K-best algorithm that reduces the area significantly compared to other K-best detectors. This design is derived from the architecture we proposed in [10], but it reduces the complexity even more especially for large constellation sizes like 64-QAM. Furthermore, as discussed later, the small area of our design does not sacrifice throughput. The throughput-efficiency, which is defined as throughput divided by area (Mbps/KiloGate), of this design is at least 4 times greater than the 2x2 64-QAM decoders in literature [11], [12] and [14]. This architecture uses one core to process the received input vector, while the conventional K-best architectures use one or two cores for each stage of the tree to process the input vector.

Also, we modify the K-best algorithm to reduce the complexity and the power consumption of the baseband receiver. We reduce the number of the enumerated constellations from 8 to 4; it results in a 44% reduction in the number of multiplications, a 36% decrease in the number of additions compared to the original K-best algorithm, with negligible performance degradation. We also show that using MMSE-SQRD (minimum mean square error – sorted QR decomposition) technique in the channel processing unit instead of ZF-SQRD results in a smaller K, which eventually results in lower complexity. For example we show that a 10-best detector using MMSE-SQRD results in a better performance and less complexity than a 16-best detector using ZF-SQRD reported in [7]. Finally, we implement the design for the 64-QAM constellation size, which is rather challenging than the widely implemented 16-QAM, and considered the 2x2 antenna configuration which is used in the LTE, WLAN and WiMAX standards.

The paper is organized as follows: the next section introduces the soft-output detection equation, tree search and the K-best algorithm. Section III explains the modification to the K-best algorithm. In Section IV we explain the proposed single-core architecture. Simulation and hardware results are shown in sections V and VI.

978-3-9810801-7-9/DATE11/©2011 EDAA

II. SOFT-OUTPUT K-BEST DECODING

Soft-output detection for MIMO detectors starts with QR decomposition of the channel matrix *H*. Then the sphere decoder performs the tree search and the pruning process as well as provides the candidate vectors and their computed norms to the soft-output computation unit (SOCU). This unit generates LLR (log-likelihood ratio) values and passes them to the channel decoder. Consider a MIMO system with M transmit and N receive antennas. The coded bit streams are modulated using a complex constellation O, each point is represented by Q bits. The modulated streams, *S,* are transmitted from the M antennas. The received signal at the baseband receiver is:

$$y = Hs + n \qquad (1)$$

where *H* is the NxM channel matrix and *n* is N dimensional i.i.d. Gaussian noise. The soft-output detector produces reliable information for each bit in transmitted vector *S,* by calculating the extrinsic Log-Likelihood ratio (LLR) values. With the Max-log approximation, the LLR-value denoted as *L(.)* becomes:

$$L(x_j) = \frac{1}{2\sigma^2}\{\min_{x\in X_j^{+1}}\|y-Hs\|^2 - \min_{x\in X_j^{-1}}\|y-Hs\|^2\} \qquad (2)$$

where X_j^{+1} and X_j^{-1} represent two vectors from set $X=[x_0, x_1, \ldots, x_{M.Q}]$ that make the norms minimum while have the *j*th bit equal to +1 and -1, respectively. To compute the LLR-values we need to generate a candidate list that includes the ML solution as well as candidates with smallest metrics. K-best algorithm inherently generates a candidate list with size K, while it searches for the ML solution.

A. Tree Search and K-Best Detector

Tree search algorithms explore the search space for the ML solution, which is an exhaustive search over all the possible vectors in O^M and the theoretically optimum solution for MIMO detection:

$$s_{ML} = \arg\min_{s\in O^M} \| y - Hs \|^2 \qquad (3)$$

To perform tree search, QR decomposition is needed. The Zero-Force QR decomposition technique (ZF-SQRD) decomposes matrix *H* according to *H* = *QR*, in which *R* is an upper triangular and *Q* is a unitary matrix. To improve the BER performance we exploited the MMSE-SQRD technique which decomposes the augmented channel matrix $\underline{H}=[H;\sigma_n I_M]$ according to $\underline{H} = \underline{Q}\underline{R} = [Q_1;Q_2]\underline{R}$ where σ_n is the noise variance. Overall, we extended channel matrix H by considering the noise variance. Multiplying both sides of (1) by Q_1^H gives:

$$\hat{y} = Q_1^H y = \underline{R}s + Q_1^H n - \sigma_n Q_2^H s \qquad (4)$$

It leads to solving the following:

$$s = \arg\min d(s), \quad d(s) = \| \hat{y} - Rs \|^2 \qquad (5)$$

$d(s)$ can be rewritten as a recursive sum of partial Euclidian Distances (PED), $d_i = d_{i+1} + |e_i|^2$; the distance increments are:

$$|e_i|^2 = \left| \hat{y}_i - \sum_{j=i+1}^{M} R_{ij} s_j - R_{ii} s_i \right|^2 \qquad (6)$$

The tree is developed based on equation (5) where d_{M+1} is located at the root and d_1 at the leaves. The sum of the distance increments from current and previous stages produces the Partial Euclidian Distance (PED) d_i. The smallest PED at the leaves reveals the ML solution. A tree search without pruning has the complexity of the ML solution; therefore sphere decoders are used to reduce the complexity of the tree search.

In tree search the number of the branches increases as the tree grows. The K-best algorithm keeps the number of the produced branches fixed by retaining just K branches with the smallest PEDs in each stage of the tree and discarding the other branches. Therefore, at the end of the tree, a list with K candidate vectors and their corresponding metrics is available, which will be used for soft-output computation. The performance of the system is dependent on the size of K and the pre-processing techniques. From now on we call the node under process a parent and its enumerated PEDs as children.

III. MODIFIED K-BEST

Like other K-best detectors first we use real value decomposition (RVD) in the channel processing unit [6]. RVD doubles the number of stages in the tree and changes the complex domain values to a real domain. In complex domain 64-QAM constellation, each parent produces 64 PEDs; however, in the real domain, each node produces 8 PEDs and the original K-best algorithm calculates the PEDs based on (6) for all the constellations. The search space for each symbol in the real domain is {-7,-5,-3,-1,1,3,5,7}. The fact is that instead of exploring all the constellations, we can explore the first few points that result in a smaller PED. The reason is that if we sort the PEDs of one parent, the last constellations always result in a large PED and will be discarded in the sort process. Therefore it is possible to reduce the complexity of the algorithm effectively by not exploring the entire constellation.

Figure 1 shows the simulation results for 2x2 64-QAM system using K = 10 and convolutional encoding with rate 0.5, in an AWGN channel. In this graph we have simulated the 2x2 system for 3, 4, and 8 children (the original K-best). A number of children equal to 4 results in a BER performance very close to the original K-best algorithm, while by using 3 children a performance degradation is observed. Reducing the number of children lower than 3 will result in significant performance degradation, and increasing it more than 4 is not necessary. The number of children is the same for all stages but stage 1; this number increases to six for the first stage because reducing the number of children lower than six for this stage reduces performance significantly. Here we discuss how we

can find the 4 children with smaller PEDs. First we estimate the first constellation (child) by using the following equation, which is obtained by making the right hand side of (6) zero:

$$\hat{s} = ((\hat{y}_i - \sum_{j=i+1}^{M} R_{ij} s_j) / R_{ii} \qquad (7)$$

The estimated point is shown in Figure 2. Then using a round function we find the closest point to the estimated value $\hat{s}$:

$$s_1 = round(\hat{s}) \qquad (8)$$

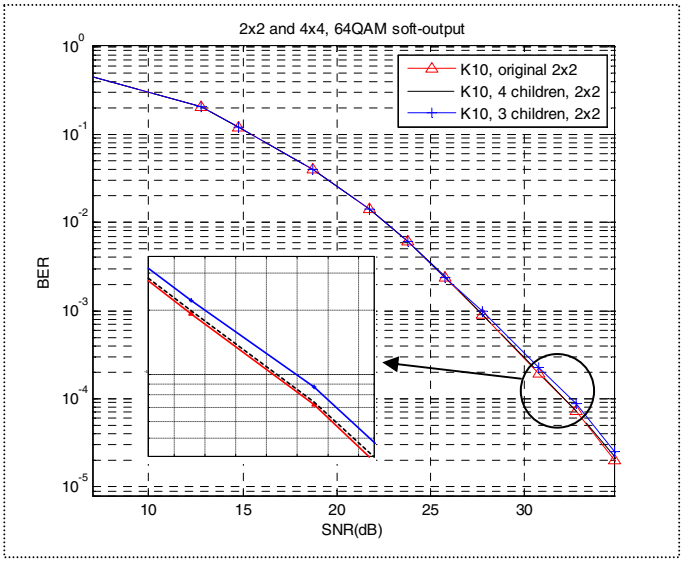

Figure 1. Modified K-best vs. original K-best.

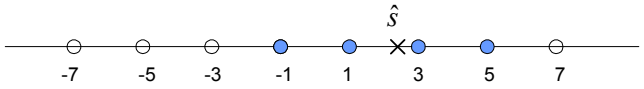

Figure 2. Exploring the first four constellation nodes.

However, the round function used here just rounds to the closest *odd* numbers. After finding the first symbol, we find the 2nd, 3rd and the 4th constellations in order, by adding with ± 2 and ± 4 . The complexity for this 4-bit adder is low, because one of the operands is always either 2 or 4.

The proposed modification reduces the number of multiplications and additions used for generating and sorting the PEDs. We compared this complexity reduction for a 2x2 64-QAM system. The results are shown in Table I. We use the 16x16 multipliers to calculate square functions and the 4x16 multipliers for multiplications by candidate symbols, which use 4 bits. We have used adders to implement the comparators in the sort unit. As a result of the algorithm modification, the number of multiplications reduces by 44% and additions by 36% if just 4 out of 8 children are used, as shown in Table II. This table also shows the SNR loss due to the introduced modifications, extracted from Figure 1. The SNR loss due to using 3 children is 0.3dB while this number decreases to less than 0.05dB for 4 children. Therefore we implement the detector using 4 constellation points, as it has a negligible BER performance degradation. The hardware area and power reduction due to this modification are shown in the next section. Later we show that this technique also increases the throughput of our single-core architecture.

TABLE I. COMPLEXITY COMPARISON; K-BEST VS. MODIFIED K-BEST

	Original K-best	Modified K-best (3 children)	Modified K-best (4 children)
16x16 Mult.	263	115	141
4x16 Mult.	290	140	166
Additions	434	188	230
Comparison	700	460	486

TABLE II. COMPLEXITY AND PERFORMANCE COMPARISON WITH THE ORIGINAL K-BEST

	Modified K-best (3 children)	Modified K-best (4 children)
Mult. Reduction	53%	44%
Add. Reduction	57%	36%
SNR Loss	0.3dB	<0.05dB

IV. PROPOSED ARCHITECTURE

In this section, the single-core architecture is proposed. This core process all layers of the tree consecutively using one core. The conventional multi-stage architectures [4]-[7] for K-best algorithm use one or two (dependent on using RVD) cores for each stage of the tree, and therefore result in a large area, and a large throughput which is usually much more than the required throughput. Furthermore, multi-stage architectures are not flexible, and not practical for large number of antennas such as 12x12. The advantages of the single-core architecture are the low-area even for large antenna sizes, low-power consumption, high throughput-efficiency, and flexibility for different number of antennas. Also this architecture is a promising candidate for multi-core architectures, as it is small and has a high-throughput. Moreover as reported in [15] if we incorporate extra cores in the design, they can be used as spare cores to increase the manufacturing yield as well as the reliability and availability of the chip in the field.

A. Units

The single-core architecture includes four blocks: metric calculation unit (MCU), merge unit, trace back and soft-output computation unit (SOCU). MCU calculates PEDs, and implements (6) as shown in Figure 3. As it can be seen, the whole architecture processes one parent in each cycle. We used 16x4 bit multipliers for the multiplication of symbols by matrix coefficients. The "round" unit is used to determine the first four candidates out of eight. Next the PEDs are calculated in parallel for the four candidates (children). Because the PEDs from the parent are already sorted (as explained in previous section) a merge unit can sort all the PEDs that come from the other parents in the next cycles. It should be mentioned that the complexity of the merge operation is so much lower than sort operation. The "dashed" lines in Figure 3 are the parts of the original K-best architecture circuit eliminated by using the proposed algorithm modification in section III. We used a 10 by 4 odd-even merge network for the merge operation [9]. This unit has a parallel scheme and utilizes 20 comparators to merge the two pre-sorted inputs.

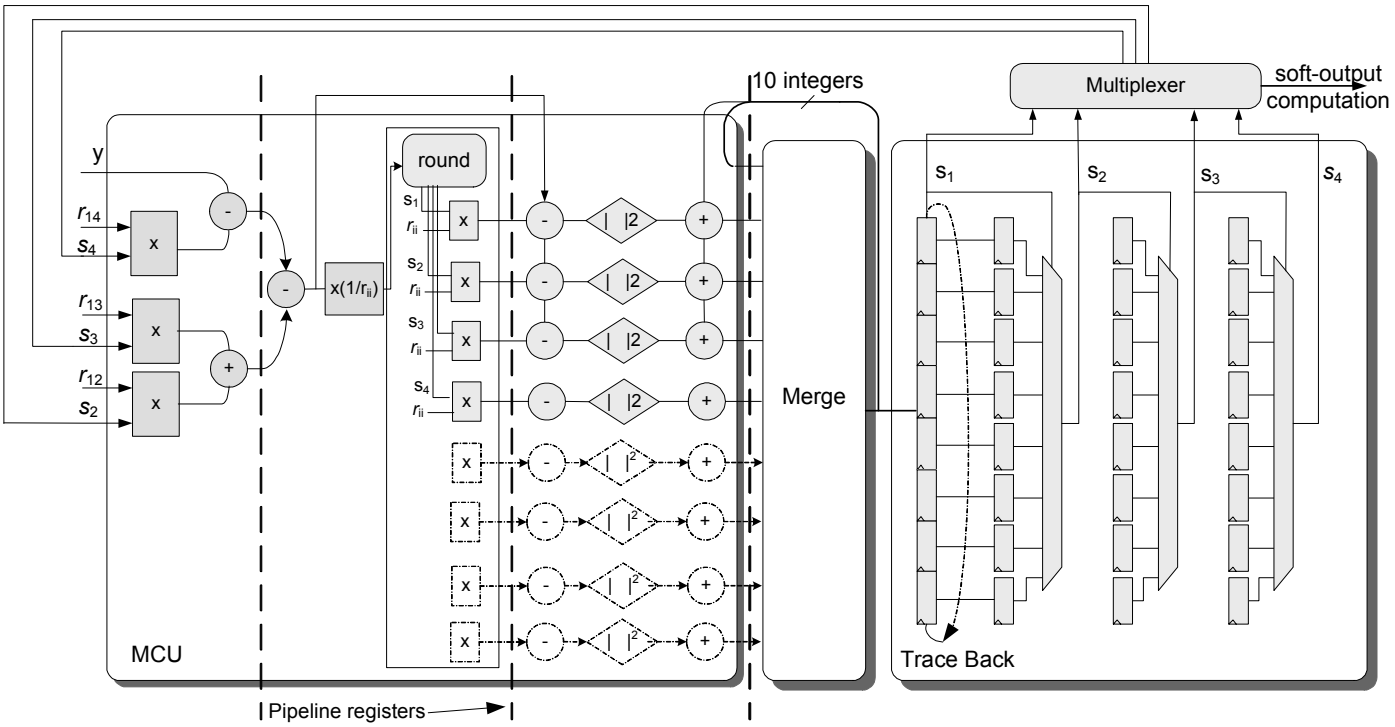

Figure 3. Proposed single-core architecture.

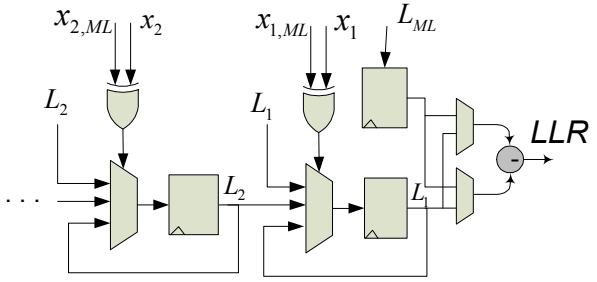

Figure 4. Soft-output computation unit.

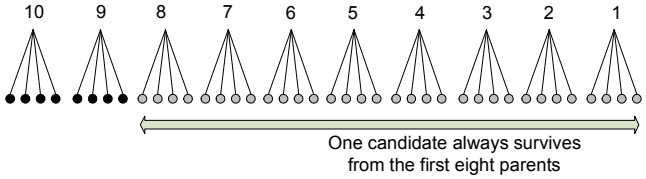

Figure 5. Scheduling strategy.

The trace back unit consists of four register banks, which restores the survived candidates at the end of each stage. As you can see there is a shift operation for the first register stack shown with a dashed line, which happens once after the merge operation for each layer is finished. While this shift operation occurs the multiplexers determine the symbols (from previous stages) associated with the current detected symbol and send them back to the MCU for the PED generation process of the next stage of the tree.

Finally, when the leaves of the tree are reached, the trace back unit sends the restored candidates along with their metrics to the soft-output computation unit. SOCU is shown in Figure 4. This unit implements (2) and it takes ten cycles to compute the LLRs, one cycle for each symbol candidate. As shown in Figure 4, the upper right hand side register restores the ML metric (L_{ML}), which is always one of the metrics in (2), and the registers on the lower side of the SOCU restore the second metric (L_j) needed to calculate (2) for bit j of the transmitted vector. If one of the metrics is not identified, a pre-defined value L_{max} will be used. The XOR gates decide if the metric associated with the current symbol should be restored in register (L_j) or not. After 10 cycles the right hand side adder calculates the LLR values by calculating $\pm(L_{ML} - L_j)$ for each bit.

B. Throughput

This architecture processes one parent at a time; the number of parents for layers 1 to 4 is 1, 6, 10, and 10 respectively (the number of parents in stage 2 is equal to the number of children generated in stage 1, and in section 3 we explained that the optimum number of children for the first stage is 6). Therefore it should take 10 cycles to process the third and the fourth stages, 6 cycles to process the second stage and 4 cycles to complete the first stage. The reason for taking 4 cycles for the first stage instead of 1, is the resource sharing. As shown in Figure 3, MCU can generate 4 PEDs at a time; therefore to produce 6 PEDs, 2 cycles are needed by the MCU and 2 more cycles for scheduling. As we explain here, unexpectedly it takes 13 cycles instead of 10 to process stages 3 and 4. By using a new sort-strategy we reduce the 13 cycles to 11, without any performance degradation.

Three pipeline levels are used in MCU (shown in dashed lines). An examination of the architecture depicts that the

added pipeline levels in the MCU reduce the throughput, because they impose three delay cycles when the process of the next stage starts. The reason is that it takes 3 cycles for the PEDs restored in trace back to enter the merge unit. To circumvent this problem, we introduce a scheduling plan that avoids two delay cycles: The algorithm modification introduced in section 3 decreases the number of children from eight to four. Therefore, in stages 3 and 4, we have ten parents, each with four children. If we identify one survivor at the end of cycle 8, we have compensated two cycle delays imposed by the MCU pipeline stages.

Figure 5 demonstrates that always one of the survived derived from the first eight parents; because K = 10, and 10 is greater than the sum of the children of the last two parents (which is eight). Therefore, using the first output of the merge unit at the end of the cycle eight (instead of cycle 10), two delay cycles are removed. We should also mention that no pipeline level can be inserted inside the merge unit. The reason is the loop exploited in the merge unit that makes pipelining inside this unit in-efficient.

Overall, 32 (4+6+11+11) total cycles for the modified single-core sphere decoder are necessary to process the input vector, and the throughput can be calculated according to $(12\times clock\ freq)/32$ for the 2x2 64-QAM configuration.

TABLE III. POWER CONSUMPTION OF THE ORIGINAL AND MODIFIED 10-BEST UNITS USING THE SINGLE-CORE ARCHITECTURE @ 284MHZ

	Original K-best, 8 children	Modified K-best, 4 children
MCU	45.4	27.79
Merge	16.03	14.82
SOCU	6.51	5.19
Trace-back	6.69	6.32
Other logic	0.77	0.275
Total power	75.4mW	54.4mW

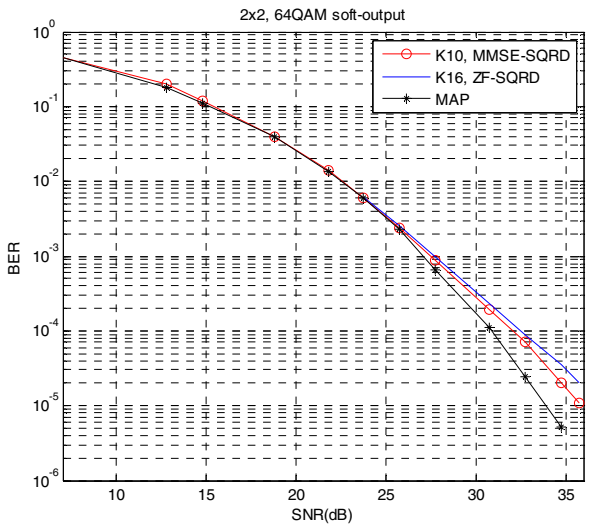

Figure 6. MMSE-SQRD vs ZF-SQRD channel processing.

V. SIMULATIONS

The performance of the MIMO detector is highly dependent on the pre-processing unit. All the K-best SDs use the popular Zero-Force QR decomposition. We show in this section that MMSE-SQRD (used by different types of MIMO detectors) improves the BER of the system compared to the ZF-SQRD effectively. Two other popular pre-processing techniques used in most K-best SDs are real value decomposition (RVD) and sorted QR decomposition (SQRD), which we have also exploited in the QRD process. We refer you to [1] for more information regarding these two techniques. These techniques result in a better BER performance, and eventually in a smaller K that directly affects area, throughput and power.

We simulated the K-best detector in a MIMO-OFDM system including a convolutional encoder with a rate of 0.5. Figure 6 shows the simulation results for the 2x2 64-QAM configuration. This figure shows that the detector with $K_{MMSE-SQRD} = 10$ has a better BER performance than the detector with $K_{ZF-SQRD} = 16$ in [7]. K = 10 also provides a performance close to the MAP detector with less than 2dB loss at BER = 10^{-5}. The MMSE-SQRD has more complexity than ZF-SQRD, but because the processing rate for channel decomposition is much less than the processing rate of the MIMO detector, the complexity overhead is negligible (1 to 64 for OFDM frame with 64 symbols).

VI. RESULTS

We implemented the soft-output 2x2 64-QAM MIMO detector using a low-threshold-voltage commercial 130nm CMOS technology with vdd = 1.5v. We synthesized the design using Synopsys Design Compiler for both the original and the modified K-best. We have also included the implementation result on FPGA.

A. Original K-best Versus Modified K-best

The modified K-best uses 24Kilo gates (22KGate for the core and 2KGate for the channel coefficients) and consumes 54.4mW at the maximum data rate of 107Mbps. The original K-best consumes 30.7 kilo gates and consumes 75.4mW at maximum data rate of 92Mbps. The breakdown of the power consumed by different modules is shown in Table III. As expected, the majority of the power saving due to the modified K-best occurs in the MCU where all the multiplications and the majority of the additions occur. The throughput of the original K-best is lower than the modified K-best because of the pipeline levels in MCU; we introduced a scheduling strategy that can be used effectively with the modified K-best single-core architecture and eliminate two out of three delay cycles enforced by the MCU pipeline levels. Therefore the algorithmic modification introduced in section 3 results in a 27% power reduction, a 22% area reduction and a 16% increase in throughput compared to the original K-best.

For comparison purposes we also implemented our design on a Xilinx Virtex-IIv6000 FPGA; our design consumes 1066 slices and 9 embedded multipliers, where as another k-best 2x2 64-QAM design [7], uses 8192 slices and 31 embedded multipliers.

B. Comparison to Other Detectors

Implementation metrics for different 2x2 64-QAM detectors are shown in Table IV. In order to create a fair comparison, we scaled the throughput of different designs by the technology process length. The MFCSO (modified fixed complexity soft output) [10] is based on the FCSO [13], in which the entire constellation is enumerated in the exact marginalization. However, MFCSO searches over only a subset of constellation points around an initial estimate $\hat{s}$, instead of searching the full constellation. The number of multiplications and additions reported in this work is more than the number of multiplications and additions reported in our design in Section 4. The LORD algorithm [12] is similar to FCSO with only one layer used in the exact marginalization. The hardware complexity of these algorithms grows exponentially with the number of antennas, which creates inefficiency in the 4x4 antenna configuration. The Selective Spanning with Fast Enumeration (SSFE) based detector [14] implemented the detector for different configurations, and the 2x2 64-QAM results are extracted from this paper for comparison.

Our design consumes 24KG which is 40% smaller than the smallest architecture in this table [14]. Also, we compared the throughput-efficiency of different designs defined as throughput divided by area (Mbps/KGate). The throughput-efficiency of our design is at least 4-fold greater than the other designs, which suggests that although our design is very small, but it provides a high throughput.

TABLE IV. ASIC IMPLEMENTATION RESULT FOR 64-QAM DESIGNS.

Graphics	**[11]**	**[12]**	**[14]**	**This work**
Technology	65nm	65nm	65nm	130nm
Algorithm	MFCSO	LORD	SSFE	K-best K =10
Configuration	2x2 64QAM	2x2 64QAM	2x2 64QAM	2x2 64QAM
Max. clock freq. (MHz)	300	80	350	287
Gate Count (KG)	55	135	40	24
Scaled Throughput	57	82	8	107
Throughput-efficiency (Mbps / KG)	1.04	0.6	0.29	4.45

VII. CONCLUSION

In this work we proposed a new architecture for the K-best algorithm. This single-core architecture reduces area significantly, and consumes a small amount of power. Also we modified the K-best algorithm by just exploring four out of eight children of each parent. This technique reduces the number of tree paths with negligible performance degradation. The overall power reduction due to this modification is 27% compared to the original K-best detector. Furthermore we compared our design to the state of the art; our design increases the throughput-efficiency at least 4-fold compared with other 2x2 64-QAM detectors, and consumes 54.4mW at the data rate of 107Mbps. The single-core architecture does not grow proportionally with an increase in the number of antennas. The low power consumption along with the high throughput-efficiency of this design makes it highly promising for mobile applications.

VIII. REFERENCES

[1] A. Burg, M. Borgmanr, M. Wenk, C. Studer, H. Bolcskei, "Advanced receiver algorithms for MIMO wireless communications," *IEEE Design, Automation, and Test in Europe (DATE)*, vol. 1, pp. 130, March 2006.

[2] K. Amiri, C. Dick, R. Rao and J. R. Cavallaro, " A High Throughput Configurable SDR Detector for Multi-user MIMO Wireless Systems", Springer Journal of Signal Processing Systems, 2009.

[3] L. G. Barbero, T. Ratnarajah, and C. Cowan, "A low-complexity soft mimo detector based on the fixed-complexity sphere decoder," in *ICASSP '08*, Las Vegas, Apr. 2008.

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

[5] 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.

[6] M. Wenk et al.,"K-Best MIMO Detection VLSI architectures achieving up to 424 Mbps," *In proc. IEEE ISCAS*, vol. 3, pp. 1151-1154, May 2006.

[7] J. Ketonen, M. Juntti, "SIC and K-best LSD Receiver Implementation for a MIMO-OFDM System," in *Proc. 16th EUSIPCO*, Aug. 2008

[8] P. Luethi, A. Burg, S. Haene, D. Perels, N. Felber, and W. Fichtner, "VLSI implementation of a high-speed iterative sorted MMSE QR decomposition," in *Proceedings of IEEE International Symposium on Circuits and Systems (ISCAS)*, pp. 1421–1424, New Orleans, 2007.

[9] K. E. Batcher, "Sorting networks and their applications," *AFIPS Proceedings on Spring Joint Computer Conference*, pp. 304-314, 1968.

[10] N. Moezzi-Madani, T. Thorolfsson, and W.R. Davis "A low-area flexible MIMO detector for WiFi/WiMAX standards," *Design, Automation & Test in Europe Conference & Exhibition (DATE)*, pp.1633-1636, March 2010.

[11] D. Wu, J. Eilert, R. Asghar, M. Ge and D. Liu ,"VLSI Implementation of a Multi-Standard MIMO Symbol Detector for 3GPP LTE and WiMAX," *IEEE International Conference on Electronics, Circuits and Systems*, Tunisia, 2009.

[12] T. Cupaiuolo, M. Siti and A. Tomasoni "Low-Complexity High Throughput VLSI Architecture of Soft-Output ML MIMO Detector," *IEEE Dessign, Test and Automation in Europe*, Germany, Dresden, March, 2010.

[13] E. G. Larsson and J. Jalden, "Soft MIMO detection at fixed complexity via partial marginalization", in *IEEE Transactions on Signal Processing*, vol. 56, pp. 3397-3407, Aug. 2008.

[14] R. Fasthuber, M. Li, D. Novo, P. Raghavan, L. V. D. Perre, and F. Catthoor, "Novel energy-efficient scalable soft-output ssfe mimo detector architectures," in *International Symposium on Systems, Architectures, Modeling, and Simulation*, July 2009, pp. 165–171.

[15] S. Shamshiri, P. Lisherness, S.-J. Pan, and K.-T. Tim Cheng, "A cost analysis framework for multi-core systems with spares," *IEEE International Test Conference (ITC)*, pp. 1-8, Oct. 2008.