A 4 × 4 Multiplier-Divider-less K-best MIMO Decoder up to 2.7 Gbps

Thi Hong Tran, Hiroshi Ochi
Grad. School of Comp. Science and Systems Eng., Kyushu Institute of Technology, 680-4 Iizuka, Fukuoka, Japan
Email: {hong, ochi}@dsp.cse.kyutech.ac.jp

Yuhei Nagao
Radrix co. ltd
Email: nagao@radrix.com

Abstract—This paper proposes a hardware architecture of K-best-based 4 × 4 MIMO decoder that supports up to 256-QAM. The novelties such as Direct Expansion, and 2D Sorter play important roles on reducing the complexity of the decoder. In addition, the most complex operators such as divider and multiplier are eliminated in this design. As compared to the previous works, the proposed decoder achieves the highest throughput (up to 2.7 Gbps), consumes the least power (56 mW), and obtains the best hardware efficiency (15.2 Mbps/Kgate).

1. Introduction

Due to high spectral efficiency, Multiple Input Multiple Output (MIMO) technology has attracted significant attention, and has been utilized in many IEEE standards such as 802.16 e/m, 802.11 n/ac [1]. One of the main challenges in exploiting the potential of MIMO systems is to design a low-complexity high-throughput and good bit error rate (BER) performance decoder. Among the large variety of MIMO decoder types, it is well known that Maximum Likelihood Detection (MLD) is the optimal approach in terms of Bit Error Rate (BER) performance. However, its complexity increases exponentially with the number of constellation nodes and with the number of spatial streams [2]. Several researches on suboptimal MLD algorithms, especially on the full K-best, have been done instead. If a MIMO system sends data via N spatial streams, the full K-best will process through N stages. In each stage, it firstly computes the euclidean distance from the received information to all of the constellation nodes (i.e., expansion task), and then sorts the obtained results (i.e., sorting task) to select K best nodes. If we denote W as the number of constellation nodes, complexity of the expansion and sorting tasks increases proportionally to W, and W², respectively.

In this study, we reform the K-best algorithm so that its complexity is insignificantly affected by W, and develop a hardware architecture of the 4 × 4 MIMO decoder based on the reformulated algorithm. The decoder supports up to 256-QAM, and does not implement neither divider nor multiplier.

The rest of this paper is organized as follows: Section II and III respectively describe the operation and the hardware architecture of the reformed 4 × 4 K-best MIMO decoder. Section IV evaluates the BER performance as well as ASIC synthesis results. The final section V is our conclusion.

II. The Reformed 4 × 4 MIMO K-best Decoder

In this paper, we assume that the channel estimation and the Sorted QR Decompose (SQRD) have been completed (refer to [3]). The decoder has the following input data: 1) A 4 × 4 upper triangle matrix R. 2) A 4 × 1 vector z. 3) Modulation size W and gain factor D, where W = 2, 4, 16, 64, 256 and D = 1, 1/√2, 1/√10, 1/√12, 1/√170 for cases of BPSK, QPSK, 16-256 QAM modulation, respectively. The decoder processes through four stages from 4 to 1.

\[
D_4 = \arg\min_{I} |z_i - r_{44}x_{i4}|^2
\]

\[
= \arg\min_{I} (z_i^4 - r_{44}x_{i4}^4)^2 + \arg\min_{Q} (z_i^Q - r_{44}x_{i4}^Q)^2
\]

\[
I_{D4}, Q_{D4}
\]

\[
D_n = \arg\min_{I} \sum_{j=n+1}^{W} r_{nj}x_j - r_{nn}x_n|^2
\]

\[
= \arg\min_{I} (f_{n}^I - r_{nn}x_{n}^I)^2 + \arg\min_{Q} (f_{n}^Q - r_{nn}x_{n}^Q)^2
\]

\[
I_{Dn}, Q_{Dn}
\]

Stage 4 computes (1) to find K = 21 smallest (best) values of \(D_4\) and the corresponding \(x_4\). These K values become the parent nodes of the next stage. Note that Ω, ΩI, and ΩQ are the set of W complex values, √W In-phase (IP), and √W Quadrature-phase (QP) values of constellation nodes; \(r_{ij}\) (i = 1, 2, 3; j ≥ i) are elements of \(R\), and \(x\), respectively.

In stage n (n = 3, 2, 1), the parent nodes are divided into three groups, K/3 = 7 parent nodes per group. Group 1, 2, and 3 contains the best, medium, and worst parent nodes, respectively. For each parent node of group 1, 2, and 3, equation (2) is computed to find \(L_1 = 4\), \(L_2 = 3\), and \(L_3 = 1\) best values of \(D_n\), respectively. In total, T = 56 values of \(D_n\) are generated in this stage.

The \(n^{th}\) partial euclidean distance \(PED_n = PED_{n+1} + D_n\) is then computed, in which \(PED_4 = D_4\). The generated values of \(PED_n\) are sorted. K = 21 smallest values of \(PED_n\) and the corresponding \(\{x_4, \ldots, x_n\}\) are selected for the next stage (i.e., stage n − 1). Exceptionally, the sorting is not required in stage 1. Instead, all T = 56 best values of \(PED_1\) and the corresponding \(\{x_4, x_3, x_2, x_1\}\) are used for the final decision, which is whether hard or soft decision. To achieve high BER performance, we implement the soft decision where
the calculation of Log Likelihood Ratio (LLR) is necessary.

\[
    LLR(b_i, x_n) = LLR(b_i, x_n) - LLR_t(b_i, x_n) \quad (3)
\]

The LLR of the \(i\)th bit of \(x_n\), which is denoted by \(b_{ni}\), is calculated by (3). Where, \(LLR_0(b_i, x_n)\) and \(LLR_1(b_i, x_n)\) denote the smallest value of \(P_{ED1}\) in cases \(b_{ni} = 0\) and \(b_{ni} = 1\), respectively.

To determine \(K\) best nodes in stage 4 and \(L_{ij}\) best nodes per parent node in group \(g (g = 1, 2, 3)\) of stage 3 and 2, we propose a Direct Expansion method. To sort \(T = 56\) elements at the end of stage 3 and 2, we propose a Two-dimensional (2D) Sorter as follows.

**Direct expansion:**

In the first quarter of the constellation (where IP and QP parts are both non-negative), we divide the IP space into \(\sqrt{W} - 1\) sub-domains such as \([0, r_{nn}], [r_{nn}, 2r_{nn}], \ldots, ([\sqrt{W} - 2)r_{nn}, \infty)\). Each sub-domain is associated with \(cell(\sqrt{D})\) best values of \(x_i^T\). We do similarly for QP space and \(x_i^Q\) (refer to Fig. 1a for case of 16-QAM and \(L = 9\)). The 2D best child nodes per parent node are found as follows.

- **Step 1:** Calculate \(f_{nj}^T\) in (2) and compare it with values such as \(r_{nn}, 2r_{nn}, \ldots, ([\sqrt{W} - 2)r_{nn}\). Each sub-domain is associated with \(cell(\sqrt{D})\) best values of \(x_i^T\). We do similarly for QP space and \(x_i^Q\) (refer to Fig. 1a).

- **Step 2:** From \(cell(\sqrt{D})\) best values of \(D_1n\) and \(DQ_{nn}\), we calculate \(L\) best values of \(D_n\) by using \(D_n = D1_n + DQ_{nn}\).

To find \(L\) best child nodes per parent node, the previous works such as [4] must find the center node \(x_c = f_n/r_{nn}\). By comparing in step 1, we can eliminate the divider \(f_n/r_{nn}\).

**2D sorter:**

We put \(L_1 = 4\), \(L_2 = 3\), and \(L_3 = 1\) expanded child nodes of the \(i\)th parent nodes of three groups into the \(i\)th row of a \(7 \times 8\) matrix (see Fig. 1b). We sort 8 elements in each row, and locate the smallest value in the left of the matrix (i.e., row sorting). Once the row sorting of all rows are completed, we sort 7 elements in each column, and locate the smallest value in the top of the matrix (i.e., column sorting). After completing the row and column sorting, we select the top-left elements of the sorted matrix as the best values.

### III. Hardware Design

**A. Overview Architecture**

The overview hardware architecture of the decoder is shown in Fig. 2a. "STAGE 4" block computes \(K\) best values of \(P_{ED4}\) (i.e., \(P_{ED4} = D1\)) and the corresponding \(x_i\) in (1). Similarly, "STAGE 3" block computes \(K\) best values of \(P_{ED3}\) (i.e., \(P_{ED3} = P_{ED4} + D1\)) and the corresponding \(\{x_4, x_3\}\) in (2), \(n = 3\). "STAGE 2" block computes \(K\) best values of \(P_{ED2}\) (i.e., \(P_{ED2} = P_{ED3} + D2\)) and the corresponding \(\{x_4, x_3, x_2\}\) in (2), \(n = 2\). "STAGE 1" block computes \(T\) best values of \(P_{ED1}\) (i.e., \(P_{ED1} = P_{ED2} + D2\)) and the corresponding \(\{x_4, x_3, x_2, x_1\}\) in (2), \(n = 1\). "LLR" block computes the log likelihood ratio in (3). The "Multiplier-Less" block prepares necessary data so that no multiplier will be implemented in all above-mentioned blocks.

The decoder implements GAIN-MUX-based Multiplier and Resource Sharing techniques to reduce the complexity.

**B. GAIN-MUX-based Multiplier**

From (1) and (2), it can be seen that the decoder requires a large number of multipliers to compute \(r_{ij}x_j\) (\(i = 4, 3, 2, 1; j \geq i\)). For example, \(24K\) multipliers are needed to compute \(r_{44}x_4^4\) and \(r_{44}x_4^3\) in stages 4 (see (1)). While, multiplier costs large hardware resource.

To multiply \(r_{ij}\) with \(m\) best values of \(x_j\), it commonly needs \(m\) multipliers. In this design, we use GAIN and multiplexer (MUX) as be shown in Fig. 2b. The input data \(r_{ij}\) firstly goes into the "GAIN" block that amplifies \(r_{ij}\) by the modulation gain \(D\) and then by the values of the constellation nodes such as \(1, 3, 5, \ldots, 15\). Notice that all possible values of \(x_j\) are \(\{D, 3D, \ldots, 15D\}\). The outputs of "GAIN" blocks are then inputted to \(m\) MUX blocks. Each MUX is controlled by a select signal of \(x_j\) which is denoted by \(sel_{x_j}^{(m)}\). If values of \(x_j^{(m)}\) are \(\{D, 3D, \ldots, 15D\}\), values of \(sel_{x_j}^{(m)}\) will be \(\{0, 1, \ldots, 7\}\). Consequently, the outputs of MUX blocks are the results that we need. Meanwhile, hardware cost for MUX is much smaller than that of for multiplier.

The decoder needs multipliers to compute \(r_{44}x_4^4, r_{34}x_4^3, r_{24}x_4^2, \ldots\), while possible values of \(x_4^4, x_4^3, x_4^2\) are same. Thus, one "GAIN" block is shared to reduce the complexity. The "Multiplier-Less" block implements this "GAIN" block.

**C. Resource Sharing**

This technique is implemented in "STAGE 4", "STAGE 3", "STAGE 2", and "STAGE 1" blocks.

The "STAGE 4" block computes \(K\) best values of \(P_{ED4}\) and \(x_4\). Based on the Direct Expansion method, it finds \(cell(\sqrt{D}) = 5\) best values of \(D1\) and \(DQ\) and then adds these values together. Because the processes of finding \(D1\) and \(DQ\) are similar to each other, they share the same circuit. Fig. 2c shows the block diagram inside "STAGE 4". In which, "DI4 CAL" is shared to find the best values of \(D1, x_4\) and \(DQ, x_4^2\) in two clock cycles. In other words, the sharing factor of this block is 2. The design of "DI4 CAL" is shown in Fig. 2d. In which, "SIGN ABS" block determines the sign and absolute value of \(|z_4^4|\) (and \(|z_4^3|\). "CONS-LOCAT" block
per cycle. Block "2D SORT" is thus required only a one-row-sorting circuit. This circuit is shared to sort all 7 rows in 7 clock cycles. The sharing factor is 7. The hardware design of the "2D SORT" block is shown in Fig. 3. The "ROW-SORT" block sorts 8 outputs of "B1", "B2", and "B3" per clock cycle. Only 4 best data are obtained. In the "COL-SORT", the "1107" collects the best values from "ROW-SORT" in 7 cycles and sort them, "1106" block collects the 2nd best values from "ROW SORT" in 7 cycles, sorts them, and obtains 6 best data, so on. The designs of "ROW-SORT" and "1103" of "COL-SORT" are shown in Fig. 3b, and 3c, respectively.

The architectures of "STAGE 2" and "STAGE 1" are similar to "STAGE 3". The sharing factor of these blocks is 7. However, the "2D-SORT" block is not implemented in "STAGE 1". Instead, the results of "B1", "B2", and "B3" are directly passed to the "LLR" block.

IV. RESULTS AND COMPARISONS

The proposed decoder was designed by using Symphony tool of Synopsys, verified by combining with Matlab software program. The completed circuit was then converted to Verilog code and synthesized in Design Vision tool of Synopsis. The CMOS SAED 90 nm technology and saed90nm_min library was used. The applied voltage was 1.32 V. The ASIC synthesis results are shown and compared in Table I. All the designs are K-best-based 4 x 4 MIMO Decoders. From this table, the contribution of the proposed decoder can be seen as follows:

High throughput: The proposed decoder achieves the highest throughput among all designs. Comparing with the most recently work in [4], the proposed decoder’s throughput is 2 times higher.

Low power consumption: Among all the designs, the proposed design consumes the least power, which is about 56 mW.

Small area: Although supporting higher modulation (i.e., 256-QAM), and larger K (i.e., K = 21) than the most recently work in [4], the proposed decoder occupies less hardware area. It needs 180 K gates, which is almost half of [4].

High normalized hardware efficiency (NHE): The proposed design obtains the highest NHE. It is 15.2
MHz bandwidth, (16-256)-QAM, various decoder types.

It is 0.07 K (green line) outperforms the BLAST-MMSE and LRA-MMSE, which is better than [9], [8], [10], and [4] by 50.7, 29.2, 8.5, and 3.6 times, respectively.

In addition, as shown in Fig. 4, the proposed decoder (green line) outperforms the BLAST-MMSE and LRA-MMSE, and is close to the full K-best in terms of BER performance. Its hardware design achieves the highest throughput (2.7 Gbps), consumes the lowest power (56 mW), and low complexity (180 Kgate).

This research is expected to be applicable in many MIMO systems such as 802.11n/ac. Our future work is to upgrade the current decoder’s circuit so that it supports from 1 × 1 to 16 × 16 MIMO cases.

ACKNOWLEDGMENT

The authors would like to thank Assoc. Prof. Masayuki Kuroasaki, and Ms. Reina Hongyo for their helps on tool licenses and hardware design.

This work was partly supported by Japan Ministry of Education KAKENHI (22360156).

REFERENCES


