High Speed Sphere Decoding Based on Vertically Incremental Computation

Se-Hyeon Kang and In-Cheol Park†
System LSI Division, Samsung Electronics, Republic of Korea
†Dept. of Electrical Engineering and Computer Science, KAIST, Republic of Korea
shkang@ics.kaist.ac.kr, icpark@ee.kaist.ac.kr

***Abstract*— Sphere decoding enables maximum likelihood (ML) detection with lower complexity than other decoding algorithms, but it still suffers from large computational delay. This paper proposes a vertical partial Euclidean distance (PED) computation method to reduce the critical path delay and computational resources. Since the proposed method computes ahead the PED of lower levels using upper level symbols, a high speed PED computation unit can be implemented with less hardware resources.**

I. INTRODUCTION

The ever-increasing demand on broadband mobile communications has motivated Multiple-input-multiple-output (MIMO) systems to achieve high spectral efficiency [1]. In rich-scattering environments, spatial multiplexing using multiple antennas can realize high spectral efficiency by utilizing multi-path propagation which was traditionally a pitfall for wireless communications. Thus the MIMO technique has been proposed as extensions to current wireless communication standards such as HSDPA and IEEE 802.11 and is part of emerging standards such as IEEE 802.16.

Since high data rate are enabled by employing many antennas, brute-force ML detection is infeasible for practical systems. Thus Zero-Forcing or V-BLAST is chosen as an alternative solution to trade off between BER performance and hardware complexity. Recently, the sphere-decoding algorithm has been proposed to limit the search space to a sphere centered at the received vector, leading to ML detection without exhaustive search [2, 3]. The search space is represented by a tree where all the possible symbols for an antenna become the child nodes of an upper node. The algorithm calculates the PED of each node and prunes a sub-tree which has larger PED than the radius of the sphere. To get a ML solution, the search process continues till all the other sub-trees are pruned.

The delay of PED computation is quite large and affects overall performance especially in a low SNR environment where several hundreds of decoding cycles are required. For example, IEEE 802.11n targeting 100Mbps should decode one symbol within 240ns if a 64-QAM 4×4 MIMO system is employed. To decode a received vector in one hundred cycles on the average the operating frequency should be more than 400MHz.

The conventional sphere decoder computes the PED of a node in one cycle [4]. PED computation involves cancelling the effects of other antennas, and thus the computation delay increases according to the number of antennas.

We propose a new PED computation method to compute ahead the effect of determined symbol which corresponds to the real or imaginary part of QAM symbol transmitted by one of antennas. As a result, only one antenna remains to be cancelled when computing the PED of a node. The proposed method reduces computation delay without performance loss, whereas recent researches reduce search cycles by sacrificing BER performance [5, 6, 7].

II. SPHERE DECODER

We consider a MIMO system with n_T transmit antennas and n_R receive antennas. It is assumed in this paper that the square M-QAM constellation is employed and the same constellation is used for all the sub-streams.

A. Sphere Decoding Algorithm

Each component of the transmitted vector is independently drawn from the same complex constellation. To get a real-valued expression for M-QAM MIMO systems, we transform the complex-valued matrix equation into real-valued matrix equation as $y = H \cdot x + w$, where w is a white Gaussian noise vector, x and y are real-valued expressions for the transmitted and received vectors, i.e., $x = [\mathrm{Re}(\tilde{x})\ \mathrm{Im}(\tilde{x})]^T$ and $y = [\mathrm{Re}(\tilde{y})\ \mathrm{Im}(\tilde{y})]^T$. A real-valued channel matrix H of size $m \times n$ ($m = 2n_T$ and $n = 2n_R$) can be obtained by treating real and imaginary part independently as expressed in Equation (1).

$$H = \begin{bmatrix} \mathrm{Re}\{\tilde{H}\} & -\mathrm{Im}\{\tilde{H}\} \\ \mathrm{Im}\{\tilde{H}\} & \mathrm{Re}\{\tilde{H}\} \end{bmatrix} \quad (1)$$

The ML detection is to find a vector $\hat{x}$ that has the smallest Euclidean distance to y as represented in Equation (2).

$$\hat{x} = \arg\min_{x \in S^m} \|y - H \cdot x\| \quad (2)$$

In other words, a vector x can be regarded as a coordinate of the lattice, the sphere decoding is equivalent to search a lattice point that is closest to a given point y. All the Euclidean distances from all possible lattice points, however, can not be calculated if the number of antennas and the order of modulation become large. The sphere decoding algorithm efficiently solves this problem by limiting the searching within a sphere centered at the received vector y.

The radius of the sphere can be represented by Euclidean

1-4244-0921-7/07 $25.00 © 2007 IEEE.

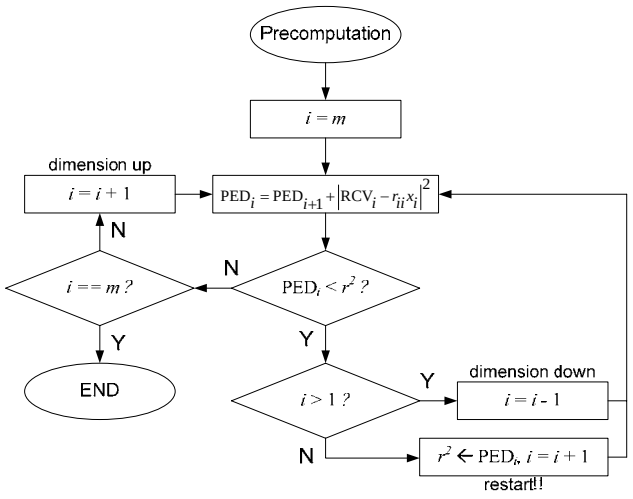

Fig. 1. Flow chart of sphere decoding algorithm.

distance between the received vector and a lattice point. In order to break the problem into sub-problems, it is useful to consider the QR factorization of matrix H, H = QR, where R is an $m \times m$ upper triangular matrix and Q is an $n \times m$ unitary matrix. Thus the radius of sphere can be represented as follows

$$r^2 = \|\mathrm{y} - \mathrm{H}\cdot\mathrm{x}\| = \|\mathrm{y} - \mathrm{QR}\cdot\mathrm{x}\| = \|\mathrm{Q}^{\mathrm{H}}\cdot\mathrm{y} - \mathrm{R}\cdot\mathrm{x}\| = \sum_{i=1}^{m}\left(z_i - \sum_{j=i}^{m} r_{ij}x_j\right)^2 = \sum_{i=1}^{m}\mathrm{INC}_i^{\,2} \quad (3)$$

, where $z = Q^H \cdot y$, z_i is an element of vector z, and x_i and r_{ij} represent an element of vector x and matrix R, respectively. Equation (3) can be re-written in an iterative form as shown in Equation (4).

$$\mathrm{PED}_i = \mathrm{PED}_{i+1} + \mathrm{INC}_i^{\,2} \quad (4)$$

PED_i is the partial Euclidean distance of level i, which is a result of accumulating INCs from m to i. Thus PED_1 becomes the Euclidean distance. Since R is an upper triangular matrix,

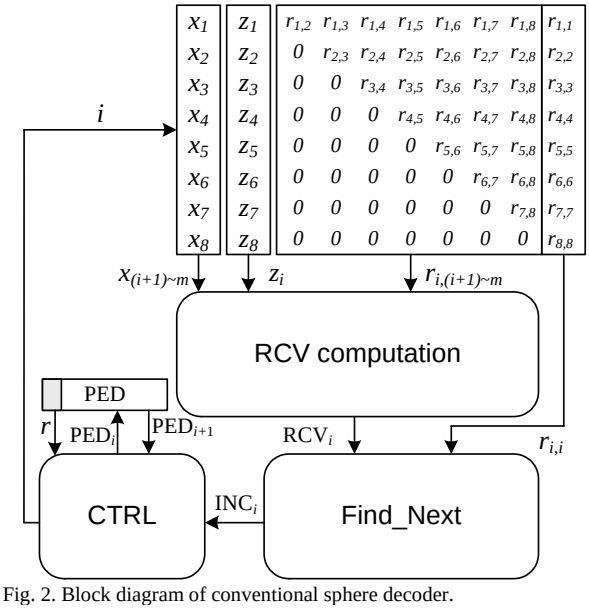

Fig. 2. Block diagram of conventional sphere decoder.

only x_i should be determined to get minimum PED_i at level i. x_j whose j is greater than i is already determined in the upper level. PED_i can be re-written by defining RCV_i as shown in Equation (5), where RCV_i can be computed by subtracting the effect of already determined symbols from z_i. To get INC_i, we have to select a symbol x_i that yields the smallest INC_i among the remaining symbols.

$$\mathrm{RCV}_i \triangleq z_i - \sum_{j=i+1}^{m} r_{ij}x_j \quad (5)$$

$$\mathrm{PED}_i = \mathrm{PED}_{i+1} + \left|\mathrm{RCV}_i - r_{ii}x_i\right|^2 \quad (x_i \in S)$$

Sphere decoding is analogous to the depth-first search. All the possible values of x_i become sub-trees for x_{i+1}. At every level, it selects the best candidate for x_i. As going down the tree, the partial Euclidean distance keeps increasing to the exact Euclidean distance of a lattice point as indicated in Equation (4). Since PED keeps increasing, the lattice points in that sub-tree can never be inside the sphere and thus it can be pruned. Every time a valid lattice point is found, the search is restricted further by reducing the radius of the sphere to the Euclidean distance of the lattice point. The overall flow of the sphere decoding algorithm is depicted in Fig. 1.

B. Conventional PED Computation

The hardware architecture for the sphere decoding algorithm can be directly derived from the depth-first search. PED_i is evaluated for a node at every cycle. If it is larger than the current radius, it goes up one level. Otherwise, it goes down one level. The conventional sphere-decoding architecture for a 4×4 MIMO system is depicted in Fig. 2. The rectangles in the upper side of Fig. 2 are memories to store the determined symbol x_i, the received symbol z_i and the channel coefficients r_{ij}. The RCV computation unit reads them to compute RCV_i. The Find_Next unit computes the minimum INC_i by selecting a symbol x_i among all the possible symbols of level i. The control unit adds the minimum INC_i to PED_{i+1} and compares it with the current radius to decide whether to go up or down.

The RCV computation unit cancels the effects of already determined symbols as defined in Equation (5). Since R is an upper triangular matrix, the number of channel coefficients to be calculated increases as the level goes down as shown in Fig 3. The RCV computation becomes a burden in the lower level to meet the cycle constraints because the conventional sphere decoder computes a node in one cycle.

$$\begin{aligned} r^2 = {} & |\, z_8 - r_{88}x_8 \,| \\ & + |\, z_7 - r_{78}x_8 - r_{77}x_7 \,| \\ & + |\, z_6 - r_{68}x_8 - r_{67}x_7 - r_{66}x_6 \,| \\ & + |\, z_5 - r_{58}x_8 - r_{57}x_7 - r_{56}x_6 - r_{55}x_5 \,| \\ & + |\, z_4 - r_{48}x_8 - r_{47}x_7 - r_{46}x_6 - r_{45}x_5 - r_{44}x_4 \,| \\ & + |\, z_3 - r_{38}x_8 - r_{37}x_7 - r_{36}x_6 - r_{35}x_5 - r_{34}x_4 - r_{33}x_3 \,| \\ & + |\, z_2 - r_{28}x_8 - r_{27}x_7 - r_{26}x_6 - r_{25}x_5 - r_{24}x_4 - r_{23}x_3 - r_{22}x_2 \,| \\ & + |\, z_1 - r_{18}x_8 - r_{17}x_7 - r_{16}x_6 - r_{15}x_5 - r_{14}x_4 - r_{13}x_3 - r_{12}x_2 - r_{11}x_1 \,| \end{aligned}$$

Fig. 3. Conventional RCV computation.

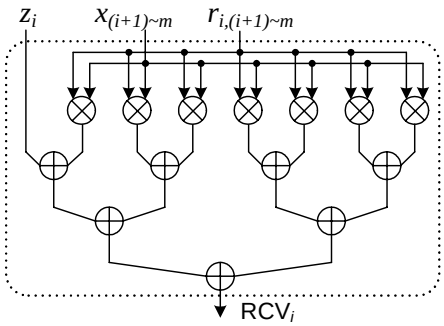

Fig. 4. Conventional RCV computation unit.

The conventional architecture of the RCV computation unit is depicted in Fig. 4, where the product terms, determined symbols multiplied by channel coefficients, are added. Since the number of the product terms to be added is seven at level 1, the RCV computation unit needs 7 multipliers and 7 adders. For this worst case, the critical path delay is equal to 1 multiplication plus 3 additions. At higher levels that do not need 7 multipliers, some of the multipliers do null operations by putting zero channel coefficients. Since the RCV computation occupies about 40% of the overall delay, reducing the delay of RCV computation is significant in improving the overall performance.

III. Proposed Sphere Decoder

This section describes a new PED computation method proposed to compute the PED in vertical direction. Since all the symbols required to compute RCV$_i$ are determined during the previous cycles, the effect of each symbol is subtracted right after it is determined.

A. Vertical PED Computation

When x_i is determined at level i, the proposed computation method is to compute all the relevant terms located vertically in matrix H at the next cycle. The vertical PED computation method is shown in Fig. 5, where relevant terms are shaded. The proposed method partially computes RCV$_i$ every cycle whereas the conventional method computes it at once. Let RCV$_i$(j) be a partial value computed by subtracting the effects of m-th to j-th symbols from the received symbol z_i. The proposed method can be expressed as Equation (6). RCV$_i$(m) is simply the received symbol and RCV$_i$(i) becomes the RCV$_i$ of the conventional method. Once RCV$_i$ is computed, the other parts are the same as the conventional one. Since the proposed

$$
\begin{aligned}
r^2 = & |\, z_8 - r_{88}x_8 \,| \\
& + |\, z_7 - r_{78}x_8 - r_{77}x_7 \,| \\
& + |\, z_6 - r_{68}x_8 - r_{67}x_7 - r_{66}x_6 \,| \\
& + |\, z_5 - r_{58}x_8 - r_{57}x_7 - r_{56}x_6 - r_{55}x_5 \,| \\
& + |\, z_4 - r_{48}x_8 - r_{47}x_7 - r_{46}x_6 - r_{45}x_5 - r_{44}x_4 \,| \\
& + |\, z_3 - r_{38}x_8 - r_{37}x_7 - r_{36}x_6 - r_{35}x_5 - r_{34}x_4 - r_{33}x_3 \,| \\
& + |\, z_2 - r_{28}x_8 - r_{27}x_7 - r_{26}x_6 - r_{25}x_5 - r_{24}x_4 - r_{23}x_3 - r_{22}x_2 \,| \\
& + |\, z_1 - r_{18}x_8 - r_{17}x_7 - r_{16}x_6 - r_{15}x_5 - r_{14}x_4 - r_{13}x_3 - r_{12}x_2 - r_{11}x_1 \,|
\end{aligned}
$$

Fig. 5. Vertical PED computation.

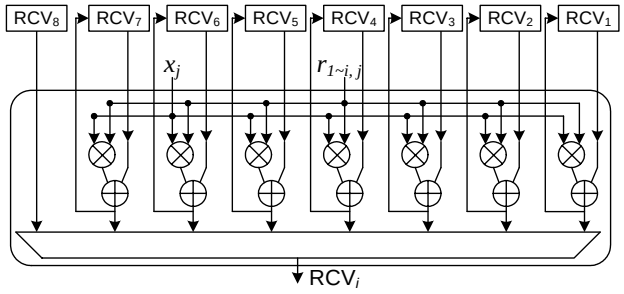

Fig. 6. Vertical RCV computation unit.

method computes RCVs of lower levels incrementally, the critical path delay of RCV computation is reduced to adding only one product term regardless of the number of antennas as represented in Equation (6).

$$\mathrm{RCV}_i(j) = \mathrm{RCV}_i(j+1) - r_{i,j+1}x_{j+1} \qquad (j \geq i) \quad (6)$$

The architecture of the proposed vertical PED computation is depicted in Fig. 6, where each RCV is computed incrementally using its own multiplier and adder. RCV$_i$ is selected to compute PED$_i$. As explained above, the critical path is reduced to one multiplier plus an adder and a multiplexer.

B. Resource sharing

The vertical RCV computation unit shown in Fig. 6 reduces the critical path delay, but does not reduce the numbers of multipliers and adders. As the multipliers and adders for level i have nothing to do after RCV$_i$ is completed, they can be shared to reduce the required resources. For example, the leftmost multiplier and adder are used only in level 7 and the next ones only in level 7 and 6. These units can be shared to compute RCVs of the right side.

Resource allocation examples are depicted in Fig. 7. Fig. 7 (a) shows the original vertical PED computation. Two numbers in a box represent indices of the corresponding channel coefficient. The first index i represents the time limit of the

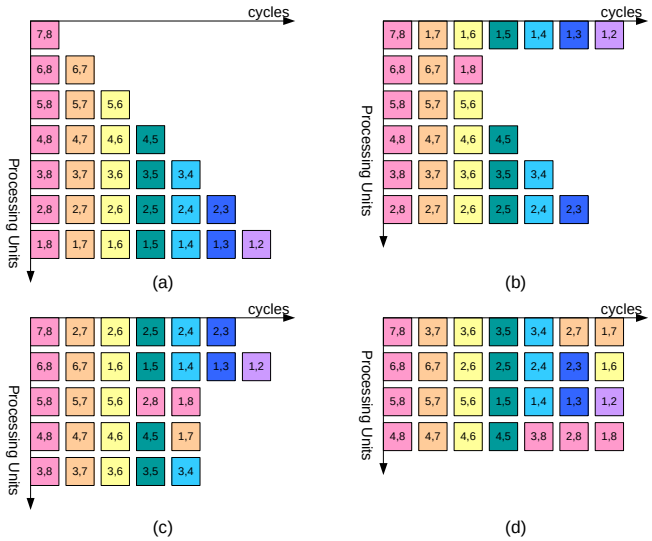

Fig. 7. Resource allocation of RCV computation unit with (a) 7 (b) 6 (c) 5 (d) 4 multipliers and adders.

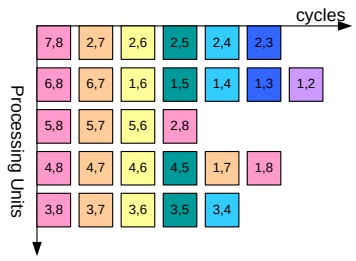

Fig. 8. Optimal resource allocation results.

coefficient multiplication, and the second index j means that x_j is multiplied by the coefficient. The boxes in a column are to be processed at the same cycle and the boxes in a row are to be processed in the same computation unit. Therefore, the boxes in the same column should be processed with different computation units. If the computation units are reduced to less than 7, some computation units have to take charge of other computations in different rows. Thus channel coefficients in low rows have to be moved to empty spaces. There are some rules to be kept in moving channel coefficients. Since the first index i represents the time limit, the coefficient should be processed earlier than cycle $(n - i + 1)$. As the coefficients associated with the same second index j are related to x_j, it is better to place them at the same row to have them computed in the same processing unit. Fig. 7 (b), (c), and (d) are made by moving coefficients under these rules.

If the number of processing units is reduced to 5, for example, two bottom rows should be moved up. Since coefficients with index i=2 has more tight time limit, these rows are moved to the top row. Not all the coefficients, however, can be moved to a row because of the time limit. The remaining coefficients, (2, 8), (1, 8) and (1, 7) are placed at the third and forth rows, leading to Fig. 7 (c). Careful placing of the remaining coefficients will help reduce the delay and save hardware resources. First, the number of coefficients with the same index i should be minimized in a column. If coefficients with the same index i are placed in the same column, additional adders are required to sum up their product terms. Thus Fig. 7 (c) can be improved by moving (1, 8) to the 6th column to minimize additional adders as shown in Fig. 8. Second, only one coefficient, if possible, should be placed at the last cycle of RCV_i computation. Since RCV_i is delivered to the Find_Next unit and used to compute PED_i, additional addition increases the critical path delay. In that sense, Fig. 7 (d) increases critical path delay because it needs to sum up four product terms in the last cycle. Fig. 9 shows the final architecture of the proposed vertical PED computation and resource sharing, which reduces two multipliers and one adder without sacrificing the performance. One adder is used to sum two product terms with the index i=1, 2 in 4th, 5th and 6th columns. Since the additional summation can be processed in parallel with the Find_Next unit, the overall delay is almost comparable to that of Fig. 6.

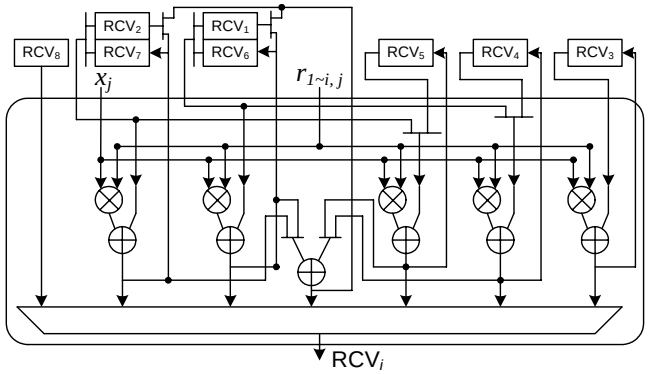

Fig. 9. Optimal architecture of the RCV computation unit.

IV. Implementation Results And Conclusion

Based on the proposed vertical PED computation method, a sphere decoder is designed for a 64-QAM 4×4 MIMO system using a 0.18 um 4-Metal CMOS process. It occupies an area of 0.74 × 0.74 mm^2 and operates at 210MHz. As summarized in Table I, the delay and the area are reduced by 27.7% and 22.5%, respectively, compared to the conventional one.

TABLE I
PERFORAMNCE COMPARISON OF SPHERE DECODERS

Performance	Conventional Decoder	Proposed Decoder
Gate count	39,582	30,655
Delay	12.24 ns	8.84 ns

Sphere decoding is an essential part of high performance MIMO communication systems, but it still suffers from large computational delay in VLSI implementation. This paper has proposed a new vertical PED computation method in order to reduce the critical path delay and hardware resources. The proposed method computes PED in vertical direction using upper level symbols, and makes the delay independent of the number of antennas and the order of modulation scheme. In addition, it enhances hardware utilization by sharing multipliers and adders.

Acknowledgement

This work was supported by Institute of Information Technology Assessment through the ITRC and by IC Design Education Center (IDEC).

References

[1] G. Foschini and M. Gans, "On the limits of wireless communications in a fading environment when using multiple antennas," *Wireless Personal Communications*, vol. 6, pp. 311-335, 1998.

[2] E. Viterbo and J. Boutros, "A universal lattice code decoder for fading channels," *IEEE Trans. Inf. Theory*, vol. 45, no. 5 pp. 1639-1642, July 1999.

[3] M. O. Damen, H. E. Gamal, and G. Caire, "On maximum-likelihood detection and the search for the closest lattice point," *IEEE Trans. Inf. Theory*, vol. 49, no. 10, pp. 2389-2402, Oct. 2003.

[4] A. Burg, M. Borgmann, M. Wenk, M. Zellweger, W. Fichtner and H. Bölcskei, "VLSI implementation of MIMO detection using the sphere decoding algorithm," *IEEE Journal of Solid-State Circuits*, vol. 40, Jul. 2005, pp. 1566-1577.

[5] K.-W. Wong, C.-Y. Tsui, R.S.-K. Cheng and W.H.Mow, "A VLSI architecture of a K-best lattice decoding algorithm for MIMO channels," in *Proc. of ISCAS*, vol. 3, 2002, pp. 273-276.

[6] R. Gowaikar and B. Hassibi, "Efficient statistical pruning for maximum likelihood decoding," in Proc. *ICASSP*, vol. 5, 2003, pp. 49-52.

[7] A. Chan and I. Lee, "A new reduced-complexity sphere decoder for multiple antenna systems," in *Proc. of ICC*, 2002, pp. 460-463.