Energy Efficient SQRD Processor for LTE-A using a Group-sort Update Scheme

Zhang, Chenxin; Prabhu, Hemanth; Liu, Liang; Edfors, Ove; Öwall, Viktor

Published in:
[Host publication title missing]

DOI:
10.1109/ISCAS.2014.6865098

2014

Energy Efficient SQRD Processor for LTE-A using a Group-sort Update Scheme

Chenxin Zhang, Hemanth Prabhu, Liang Liu, Ove Edfors, and Viktor Öwall
Department of Electrical and Information Technology, Lund University
Email: {Chenxin.Zhang, Hemanth.Prabhu, Liang.Liu, Ove.Edfors, Viktor.Owall}@eit.lth.se

***Abstract*—This paper presents an energy-efficient sorted QR-decomposition (SQRD) processor for 3GPP LTE-Advanced (LTE-A) systems. The processor adopts a hybrid decomposition scheme to reduce computational complexity and provides a wide-range of performance-complexity trade-offs. Based on the energy distribution of spatial channels, it switches between the brute-force SQRD and a low-complexity group-sort QR-update strategy, which is proposed in this work to effectively utilize the LTE-A pilot pattern. As a proof of concept, a run-time reconfigurable vector processor is developed to efficiently implement this adaptive-switching QR decomposition algorithm. In a 65 nm CMOS technology, the proposed SQRD processor occupies 0.71 mm^2 core area and has a throughput of up to 100 MQRD/s. Compared to the brute-force approach, an energy reduction of $5 \sim 33\%$ is achieved.**

I. Introduction

In multiple-input multiple-output (MIMO) systems, SQRD algorithm [1] has been employed as an efficient pre-processing step in data detection to improve performance. By optimizing the spatial processing order, SQRD is capable of significantly relieving the severe error propagation in sub-optimal detectors. However, the performance improvement is obtained at the cost of higher complexity. For instance, having a computational complexity of $\mathcal{O}(N^3)$ for an $N \times N$ matrix, state-of-the-art SQRD processors consume more energy than data detectors [2], e.g., 30 times in [3]. Additionally, MIMO is generally combined with orthogonal frequency division multiplexing (OFDM). As a consequence SQRD have to be performed at each subcarrier on every channel update, making SQRD an implementation bottleneck in MIMO-OFDM systems. To address the complexity issue, [4] proposed an approximated QRD method with a tracking-$\boldsymbol{R}$ (hold-$\boldsymbol{Q}$) scheme. However, there is a performance loss due to the out-of-date $\boldsymbol{Q}$ information, especially in fast-changing channels, and channel sorting was not considered.

In this paper, we present a hybrid decomposition scheme with group-sort QR-update strategy to efficiently implement a low-complexity SQRD processor for a 4×4 MIMO LTE-A system. By fully exploiting the property of LTE-A pilot pattern, i.e., CSIs of only antenna port 0 and 1 are changed during half-H renewals (Fig. 1), the proposed QR-update scheme computes exact $\boldsymbol{Q}$ and $\boldsymbol{R}$ matrices using only one Givens rotation. Compared to brute-force QRDs, this update strategy significantly reduces the computational complexity, while preserving the accuracy by avoiding the approximations as in the aforementioned tracking algorithms. To be able to obtain the low-complexity benefit of the introduced update scheme in the context of SQRD, we further propose an effective group-sort algorithm for channel reordering. The underlining idea is to restrict the sorting into groups of antenna ports, wherein a two-step (i.e., intra- and inter-group) sorting is applied to approximate the optimal detection order. Using the group-sort method, applicability of the QR-update is significantly expanded with negligible performance degradation compared to the precise sorting counterpart.

To demonstrate the effectiveness of the solution, we implement the proposed group-sort QR-update scheme on a vector processor,

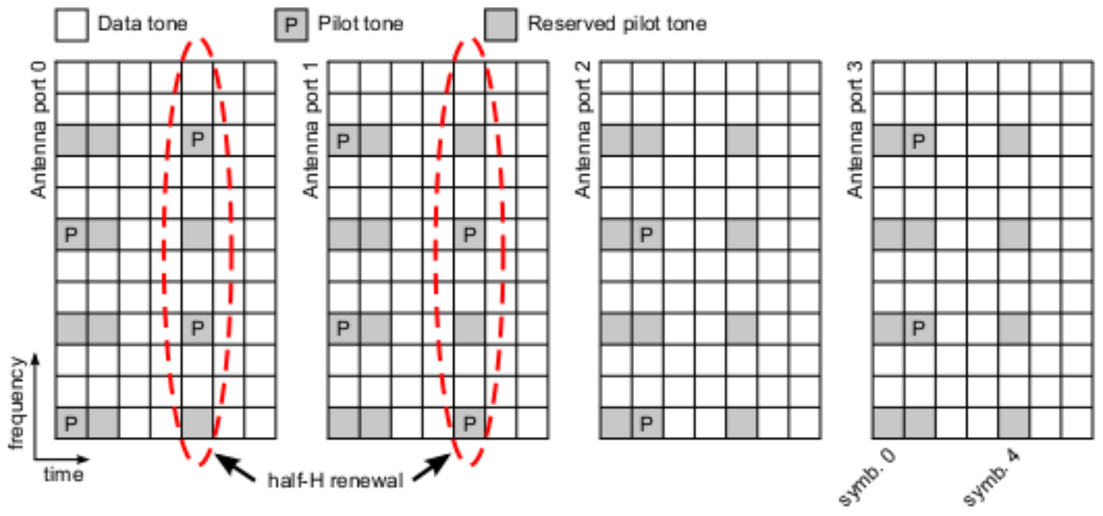


Fig. 1. Scattered pilot pattern for four antenna ports in one LTE-A resource block. Symbol positions of half-H renewals are circled in dashed lines.

developed based on the reconfigurable array framework [5]. Depending on run-time channel energy distribution, the processor is capable of dynamically switching between the QR-update and the brute-force algorithm. Using the proposed group-sort QR-update scheme, an energy reduction of around 17% is achieved with only 0.2 dB performance degradation compared to the brute-force approach.

II. System model

Considering an $N \times N$ MIMO system, the received vector $\boldsymbol{y}$ can be expressed as

$$\boldsymbol{y} = \boldsymbol{H}\boldsymbol{x} + \boldsymbol{n}, \tag{1}$$

where $\boldsymbol{x}$ is the transmit vector, $\boldsymbol{n} \sim \mathcal{CN}(0, \sigma^2 \boldsymbol{I}_N)$ is the complex Gaussian noise, and $\boldsymbol{H}$ is the complex-valued channel matrix. In this work, we consider a 4×4 LTE-A downlink operating in 5 MHz bandwidth and normal cyclic prefix mode. The error-correcting code scheme is a rate 1/2 parallel concatenated turbo code with 6 decoding iterations. Perfect signal synchronization and channel estimation are assumed at the receiver and frame-error-rate (FER) is used as the metric to evaluate system performance.

In LTE-A, data transfers are carried out using resource blocks, each containing 12 consecutive sub-carriers and 7 OFDM symbols allocated in a time-frequency grid. Pilot tones are distributed over the grid to assist CSI estimation. According to the scattered pilot pattern in Fig. 1, we observe that pilot tones allocated in the middle of each resource block are only available for antenna ports 0 and 1. This corresponds to an update of half of columns (e.g., 2 out of 4) in channel matrix $\boldsymbol{H}$, denoted as half-H renewal with respect to full-H counterpart that takes place at the beginning of each resource block.

III. Hybrid SQRD algorithm

SQRD is capable of improving detection performance by optimizing processing order based on energy of spatial channels. SQRD starts with a column permutation to the original channel matrix $\boldsymbol{H}$,

$$\tilde{\boldsymbol{H}} = \boldsymbol{H}\boldsymbol{P}, \tag{2}$$

where $\tilde{\boldsymbol{H}}$ and $\boldsymbol{P}$ denote the sorted channel and a permutation matrix, respectively. After sorting, a QRD is performed on $\tilde{\boldsymbol{H}}$ to obtain the orthogonal matrix $\boldsymbol{Q}$ and upper-triangular matrix $\boldsymbol{R}$. In the following, we focus on computational complexity of the QRD on $\tilde{\boldsymbol{H}}$.

A. QR-update scheme

In case only parts of the matrix columns alter over time, QRD of the new matrix can be performed in a more efficient way than a brute-force computation (referred to as Case-I), i.e., starting from scratch [6]. Inspired by this, we propose a low-complexity QR-update scheme during half-H renewals. Specifically, the proposed scheme starts with the brute-force SQRD during full-H renewals, expressed with a subscript "old" as

$$\tilde{\boldsymbol{H}}_{\text{old}} = \boldsymbol{Q}_{\text{old}} \boldsymbol{R}_{\text{old}}. \tag{3}$$

During half-H renewals, $\tilde{\boldsymbol{H}}_{\text{new}}$ is obtained by updating two columns of $\tilde{\boldsymbol{H}}_{\text{old}}$. Although orthogonal vectors in $\boldsymbol{Q}_{\text{old}}$ may no longer triangularize $\tilde{\boldsymbol{H}}_{\text{new}}$, it may still have vectors pointing in the correct directions. As a consequence, the new $\boldsymbol{R}$ matrix, denoted as $\tilde{\boldsymbol{R}}_{\text{new}}$, can be expressed using $\tilde{\boldsymbol{H}}_{\text{new}}$ and $\boldsymbol{Q}_{\text{old}}$ as

$$\tilde{\boldsymbol{R}}_{\text{new}} = \boldsymbol{Q}_{\text{old}}^{H} \tilde{\boldsymbol{H}}_{\text{new}}. \tag{4}$$

Due to the outdated $\boldsymbol{Q}_{\text{old}}$, $\tilde{\boldsymbol{R}}_{\text{new}}$ is no longer an upper-triangular matrix but may still reveal some upper-triangular properties depending on the positions of the two renewed columns. Specifically, in cases where column changes take place at the right-most of $\tilde{\boldsymbol{H}}_{\text{new}}$, only one element in the lower triangular part of $\tilde{\boldsymbol{R}}_{\text{new}}$ (i.e., $\tilde{r}_{\text{new}}(4,3)$) becomes non-zero. This implies that triangularization of $\tilde{\boldsymbol{R}}_{\text{new}}$ can be significantly simplified by nulling the single non-zero element instead of operating on all columns afresh as

$$\boldsymbol{G}\tilde{\boldsymbol{R}}_{\text{new}} = \boldsymbol{G}\boldsymbol{Q}_{\text{old}}^{H} \tilde{\boldsymbol{H}}_{\text{new}}, \tag{5}$$

where

$$\boldsymbol{G} = \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & c & s^{*} \\ 0 & 0 & -s^{*} & c^{*} \end{bmatrix}. \tag{6}$$

In (6), $(\cdot)^{*}$ is the complex conjugation, c and s are defined as

$$\begin{aligned} c &= \tilde{r}_{\text{new}}^{*}(3,3)/z, \\ s &= \tilde{r}_{\text{new}}^{*}(4,3)/z, \\ z &= \left(|\tilde{r}_{\text{new}}(3,3)|^{2} + |\tilde{r}_{\text{new}}(4,3)|^{2}\right)^{1/2}. \end{aligned} \tag{7}$$

After triangularizing $\tilde{\boldsymbol{R}}_{\text{new}}$, exact $\boldsymbol{Q}_{\text{new}}$ and $\boldsymbol{R}_{\text{new}}$ in the proposed QR-update scheme are obtained, expressed as

$$\boldsymbol{Q}_{\text{new}} = (\boldsymbol{G}\boldsymbol{Q}_{\text{old}}^{H})^{H}, \tag{8}$$

$$\boldsymbol{R}_{\text{new}} = \boldsymbol{G}\tilde{\boldsymbol{R}}_{\text{new}} = \boldsymbol{G}(\boldsymbol{Q}_{\text{old}}^{H}\tilde{\boldsymbol{H}}_{\text{new}}). \tag{9}$$

By combining the traditional brute-force approach (i.e., computing QRD from scratch during half-H renewals) and the QR-update scheme, a hybrid decomposition algorithm is formed which dynamically switches between the two schemes to reduce the computational complexity, depending on run-time conditions of the channel reordering. Obviously, the complexity reduction depends on the applicability of the QR-update. Intuitively, we could fix the position of antenna ports 0 and 1 to the right-most part of $\tilde{\boldsymbol{H}}_{\text{new}}$ in order to obtain a maximum complexity gain, since it completely avoids brute-force computation during half-H renewals. However, the advantage of channel reordering (for improving detection performance) is lost and we refer to this as Case-II. On the other hand (Case-III), where channel columns are permuted based on the optimal detection order without considering the position of renewed channel columns, the applicability of the QR-update is dramatically reduced. For example, considering the 4×4 MIMO LTE-A, only $(2!2!)/4! = 1/6$ of sorting combinations meet the required update condition, thus limiting the complexity reduction. As a consequence, a smart scheduling strategy is needed to explore the low-complexity potential of the QR-update, while still retaining the performance gain of the optimal channel reordering.

TABLE I
CASE-I–IV OF THE PROPOSED HYBRID SQRD ALGORITHM

	Channel reordering	Brute-force	QR-update
Case-I	Optimal ordering with precise sorting	100%	0%
Case-II	Fixed order for antenna port 0 and 1	0%	100%
Case-III	Optimal ordering with precise sorting	83.33%	16.67%
Case-IV	Group-sort	50%	50%

B. Group-sort algorithm

To fulfil the aforementioned requirement, we propose an effective group-sort algorithm for channel reordering. Instead of operating on individual columns, sorting of $\boldsymbol{H}$ is applied on two virtual groups, wherein columns associated with antenna ports 0 and 1 are tied together. This way, combinations of "columns" is reduced from 4! to 2!. Consequently, the probability of having both altered columns at the right-most part of $\tilde{\boldsymbol{H}}_{\text{new}}$ is increased by 3 times, i.e., from $1/6$ to $1/2$. To reduce errors due to sub-optimal sorting sequences, a two-step sorting scheme is adopted. First, the sorting between groups is based on the total energy of bundled columns as

$$\mathcal{I} = \arg\max_{i=\{0,1\},\{2,3\}} \sum\nolimits_{i} \|\boldsymbol{h}_i\|^{2}, \tag{10}$$

where $\mathcal{I}$ contains inter-sorted group indexes, e.g., $\mathcal{I} = \{0,1\}$ if antenna ports 0 and 1 correspond to the strongest channels. Second, the two columns within each group, e.g., indexes within $\mathcal{I}$, are intra-sorted based on the energy of individual columns. To conclude, Table I summaries all four cases of the hybrid SQRD algorithm and their corresponding applicability, wherein we denote the proposed group-sort method as Case-IV.

C. Algorithm evaluation

To illustrate the effectiveness of the proposed algorithm, the 3GPP EVA channel model with a maximum Doppler frequency of 70 Hz is used. Operating at a 2.6 GHz carrier frequency, this corresponds to a speed of 29 km/h. In each FER simulation, 5000 LTE-A subframes are transmitted and decoded using a fixed-complexity sphere decoder [7]. Performance of the proposed group-sort QR-update and aforementioned cases are shown in Fig. 2. Note that Case-III has the same performance as the brute-force approach and is used as a reference for FER measurements. Compared to the one where no QRDs are performed during half-H renewals (upper curve in Fig. 2), it clearly shows the importance of performing CSI and QR updates even for channels with moderate Doppler shifts. Additionally, adoption of channel reordering during QR decomposition improves performance to that of the fixed-order approach, e.g., 1.1 dB difference between Case-II and III at FER $= 10^{-2}$. Furthermore, the group-sort approach has only small performance degradation of about 0.2 dB compared to Case-III, however, with a large complexity reduction as analyzed in the following.

Table II summaries complexity ($\mathcal{C}$) of computations (3)–(5) for an $N\times N$ MIMO system. To perform the brute-force decomposition (3), we consider Gram-Schmidt algorithm [6] that has a total complexity

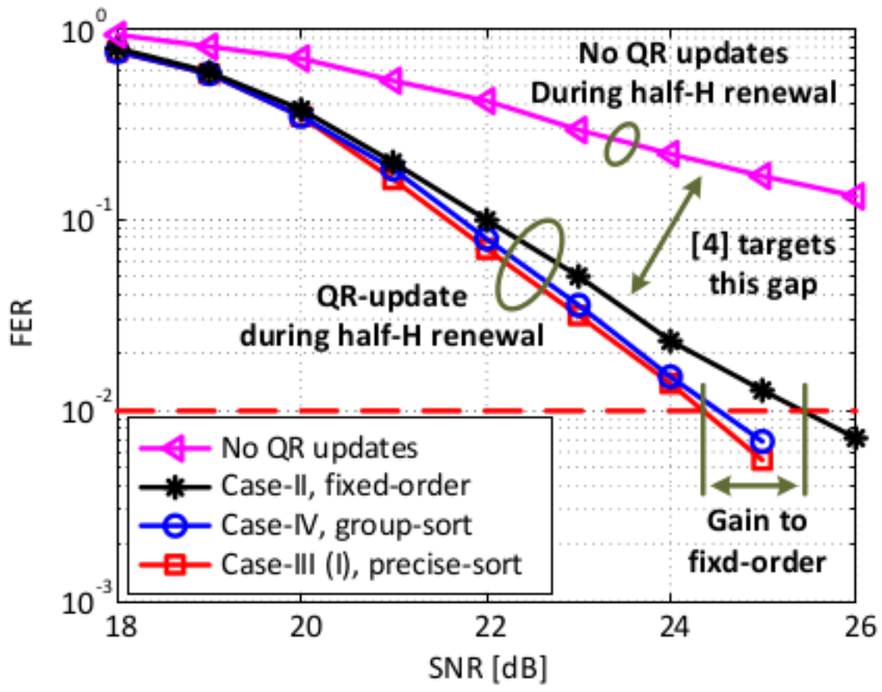


Fig. 2. Simulated FERs in a 4×4 MIMO LTE-A downlink using 3GPP EVA-70 channel model with 64-QAM modulation.

TABLE II
COMPLEXITY OF COMPUTATIONS IN THE HYBRID SQRD ALGORITHM

Complexity	Computation	Multiplication	DIV/SQRT/CORDIC*
$\mathcal{C}_1$	QRD (3)	$N^3 + 2N^2$	$2N$
$\mathcal{C}_2$	$\boldsymbol{Q}_{\text{old}}^H \tilde{\boldsymbol{H}}_{\text{new}}$ (4)	$\frac{1}{2}N^3$	0
$\mathcal{C}_3$	Triangularization (5)	$N^2 + 2N$	3
$\mathcal{C}_4$	Sorting (2)(10)	$4N$	0

* Division, square-root, and CORDIC, where the latter one is often used for generating GR matrices.

of $\mathcal{C}_1$. Computations required for both (4) and (5) have a total complexity of $\mathcal{C}_2 + \mathcal{C}_3$, which is significantly lower than $\mathcal{C}_1$, e.g., by about 42% for $N = 4$. Note that the product of $\boldsymbol{Q}_{\text{old}}^H \tilde{\boldsymbol{H}}_{\text{new}}$ in (4) requires only half of the matrix computations during QR updates, since only two columns change in $\tilde{\boldsymbol{H}}_{\text{new}}$. The complexity of sorting in both precise- and group-sort approaches is denoted as $\mathcal{C}_4$. Based on this analysis and in reference to Case-I, Table III shows the complexity reduction versus performance degradation of Case-II–IV for a 4×4 system. It shows that a 50% complexity reduction is obtained for Case-II. Moreover, combining the group-sort and the QR-update schemes results in more palatable trade-offs, i.e., 18% complexity reduction for only 0.2 dB performance degradation.

To further evaluate the hardware friendliness of the proposed algorithm, operations required in the four computations (Table II) are profiled. In Table IV, most operations are at vector level, representing a high degree of data level parallelism that can be exploited to improve throughput. In addition, most of them are shared among all computations, implying that extensive hardware reuse is possible.

IV. VLSI ARCHITECTURE AND IMPLEMENTATION RESULTS

Based on the operation analysis, we present a VLSI architecture for the proposed hybrid SQRD algorithm and analyze energy consumption and processing throughput. Considering the flexibility requirements in contemporary system designs for coping with algorithm evolutions, a reconfigurable architecture is proposed. Specifically, all required operations are mapped onto a vector processor, which are reconfigured on-the-fly to adopt an appropriate algorithm based on the run-time update condition.

A. Vector processor

Implementation of the vector processor is based on a reconfigurable array framework presented in [5]. Fig. 3 shows a microarchitecture of the processor, consisting of 6 processing (PE1–6) and 2 memory (ME1–2) elements interconnected via high-bandwidth low-latency

TABLE III
COMPLEXITY AND PERFORMANCE COMPARISONS OF CASE-I–IV

	Complexity	Complexity reduction	SNR degradation
Case-I	$\mathcal{C}_1 + \mathcal{C}_4$	– (ref.)	– (ref.)
Case-II	$\mathcal{C}_2 + \mathcal{C}_3$	50%	1.1 dB
Case-III	$\frac{5}{6}\mathcal{C}_1 + \frac{1}{6}(\mathcal{C}_2 + \mathcal{C}_3) + \mathcal{C}_4$	6%	0 dB
Case-IV	$\frac{1}{2}\mathcal{C}_1 + \frac{1}{2}(\mathcal{C}_2 + \mathcal{C}_3) + \mathcal{C}_4$	18%	0.2 dB

TABLE IV
OPERATION PROFILE OF THE HYBRID SQRD ALGORITHM FOR $N = 4$

	Vector operations			DIV/SQRT/CORDIC
	$\boldsymbol{A} \cdot \boldsymbol{B}$	$\boldsymbol{A} \odot \boldsymbol{B}^*$	$\boldsymbol{A} \pm \boldsymbol{B}$	
QRD (3)	17	4	6	4
$\boldsymbol{Q}_{\text{old}}^H \tilde{\boldsymbol{H}}_{\text{new}}$ (4)	8	0	0	0
Triangularization (5)	10	0	0	3
Sorting (2)(10)	4	0	0	0

* Element-wise vector multiplication.

links. According to the type of underlining operations, resource elements are partitioned into two parts. The upper half performs computationally intensive vector operations, while the lower part accelerates special operations like division/square-root and CORDIC. Operation modes of these elements are specified in embedded configuration memories, which are reloadable in every clock cycle. To ease run-time control of the whole processor, a master node (PE1) is responsible for tracking overall processing flow and controlling configuration memories based on instructions stored in ME1.

The vector block has 3 processing (PE2–4) and 1 memory (ME2) elements, functioning as a multi-stage computation path and a register bank, respectively. PE3 performs all vector operations in Table IV. To concurrently compute multiple data streams, it is constructed from 4 homogeneous parallel processing lanes, each having 4 complex-valued multiply-accumulate (CMAC) units. It can be seen from Table IV that vector dot product is the most often used operation. Thus, in order to reduce computation latency, a single-clock-cycle vector dot product is supported by each processing lane. This is accomplished by interconnecting adders in each row of CMACs to form an adder tree, which can add up 4 multiplication results in one clock cycle, achieving four concurrent vector operations in each clock cycle. To assist these vector computations, PE2 and PE4 pre- and post-process data to perform for example matrix Hermitian (4) and result sorting (2),(10). By combining these three processing elements, several consecutive data manipulations can be accomplished in one single instruction without storing and loading intermediate results. This execution scheme is similar to that of VLIW processors, but has additional flexibility for loading configurations into individual processing elements without affecting others, hence resulting in reduced control overhead.

The vector processor is parametrizable and in this work we have used 16 bits internal precision. The register bank (ME2) contains 16 general purpose vector registers and each configuration memory can buffer upto 16 hardware configurations. Moreover, the instruction memory (ME1) has a capacity of 4 Kbits.

B. Results and evaluation

Synthesized using a 65 nm CMOS standard cell library, the vector processor has a total core area of 0.71 mm^2 equivalent to 339 K two-input NAND gates (GE). Post-layout simulations show that maximum power consumption is 226 mW at 500 MHz with a nominal supply voltage of 1.2 V. Fig. 4 shows an area and power breakdown of the processor, wherein the vector block (PE1–4 and ME1–2) occupies 92% of the total area and consumes on average 85% of power. Among

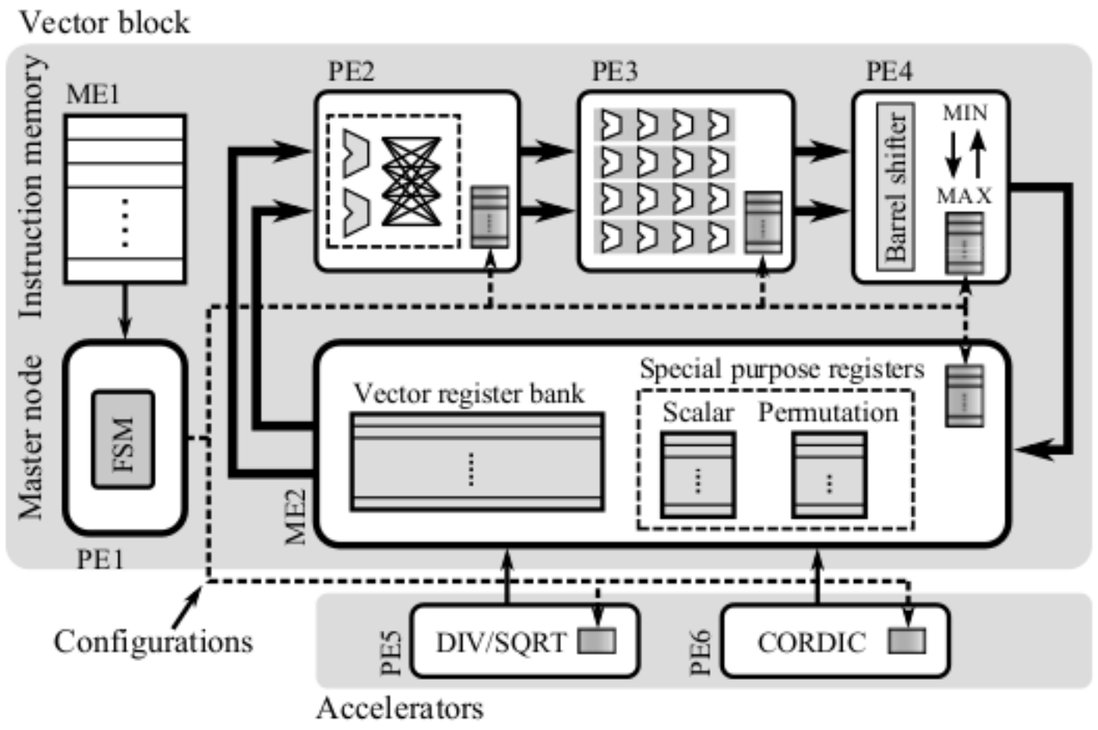
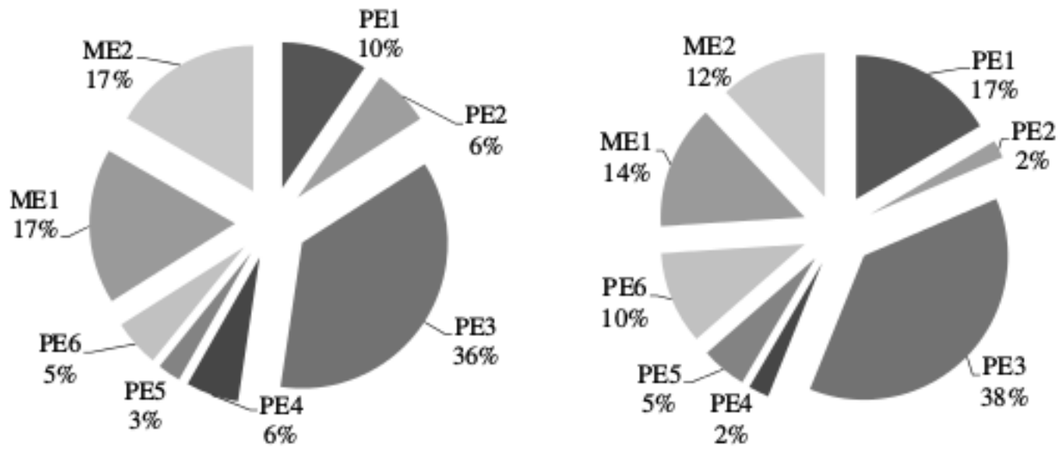


Fig. 3. Microarchitecture of the vector processor, consisting of 6 PEs and 2 MEs. Solid and dashed lines depict data and control bus, respectively.



(a) Area (b) Power

Fig. 4. Area and power breakdown of the vector processor.

all, the homogeneous CMAC bank (PE3) consumes most of the area and power, and the master node (PE1) together with its instruction memory (ME1) take around 30% of area and power.

All computations in Table IV are manually mapped onto the processor with a focus on achieving high resource utilization and processing throughput. In case of the brute-force QRD, an interleaved processing scheme is adopted to utilize data awaiting time in sequential computations. Specifically, instead of computing one QRD at a time, decomposition of four $\boldsymbol{H}$s are handled concurrently, requiring in total 30 clock cycles. This is equivalent to having an execution time of 7.5 cycles per QRD. The product of $\boldsymbol{Q}_{\text{old}}^{H}\tilde{\boldsymbol{H}}_{\text{new}}$ (4) and triangularization (5) in Table IV require 2 and 3 execution cycles respectively, thanks to the parallel processing in PE3. Accordingly, the proposed QR-update scheme requires 5 cycles to compute. Table V summaries implementation results for the brute-force and the QR-update computations. Operating at 500 MHz, processing throughput of the QR-update is 100 MQRD/s and consumes 1.9 nJ per decomposition. This results in a 33% improvement compared to the brute-force counterpart.

Figure 5 presents design trade-offs between energy and performance for Case-I–IV of the hybrid SQRD algorithm. Taking the brute-force QRD (Case-I) as a reference, numbers on the horizontal axis measures the SNR degradation for reaching the target 10^{-2} FER, while the percentage of energy reduction is shown on the vertical axis. Accordingly, algorithms having their coordinates towards the bottom-left corner is desired. In Fig. 5, it clearly shows that the proposed group-sort QR-update scheme (Case-IV) achieves a good compromise, i.e., trading 0.2 dB performance for around 17% energy reduction. In case of energy-constrained systems, the fixed-

TABLE V
IMPLEMENTATION SUMMARY OF THE VECTOR PROCESSOR.

	Brute-force QRD	Proposed QR-update
Gate count [KGE]	339	
Max. freq. [MHz]	500	
Execution cycles	7.5	5
Throughput [MQRD/s]	67	100
Energy [nJ/QRD]	2.85	1.9

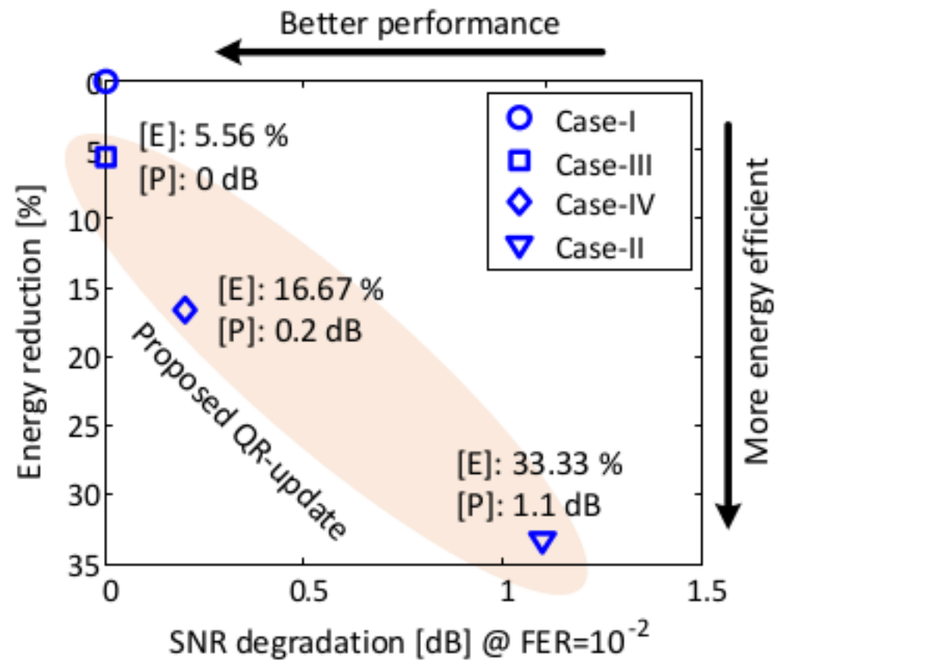


Fig. 5. Energy (E) and performance (P) trade-off for Case-I–IV of the hybrid SQRD algorithm.

order scheme (Case-II) can be adopted to further reduce the energy consumption, i.e., by 33% in total, whereas the precise-sort scheme (Case-III) is used when high performance is demanded.

V. Conclusion

This paper exploits the algorithm design and VLSI implementation of an energy efficient SQRD processor for LTE-A systems. At the algorithm level, a hybrid decomposition algorithm is proposed to reduce computational complexity by combining various QR-update schemes and the traditional brute-force SQRD method. Algorithmic analyses show that a complexity reduction of up to 50% is achieved. To leverage the flexible decomposition algorithm, a reconfigurable vector processor is developed which is able to dynamically switch between different QRD schemes based on energy distribution of spatial channels. Implementation results demonstrate a wide-range of energy-performance trade-offs using the proposed solution.

References

[1] P. Luethi *et al.*, "VLSI Implementation of a High-Speed Iterative Sorted MMSE QR Decomposition," in *IEEE International Symposium on Circuits and Systems*, May 2007, pp. 1421–1424.

[2] L. Liu, J. Löfgren, and P. Nilsson, "Area-Efficient Configurable High-Throughput Signal Detector Supporting Multiple MIMO Modes," *IEEE Transactions on Circuits and Systems I: Regular Papers*, vol. 59, no. 9, pp. 2085–2096, 2012.

[3] J.Y. Wang *et al.*, "A 2x2-8x8 sorted QR decomposition processor for MIMO detection," in *IEEE Asian Solid State Circuits Conference*, no. 1-4, 2010.

[4] L. Gor and M. Faulkner, "Power Reduction through Upper Triangular Matrix Tracking in QR Detection MIMO Receivers," in *IEEE 64th Vehicular Technology Conference (VTC)*, Sept. 2006.

[5] C. Zhang, L. Liu, and V. Öwall, "Mapping Channel Estimation and MIMO Detection in LTE-Advanced on a Reconfigurable Cell Array," in *IEEE International Symposium on Circuits and Systems*, May 2012.

[6] G. H. Golub and C. F. Van Loan, *Matrix Computations*, 1st ed. Johns Hopkins University Press, 1983.

[7] L. Barbero and J. Thompson, "Fixing the Complexity of the Sphere Decoder for MIMO Detection," *IEEE Transactions on Wireless Communications*, vol. 7, no. 6, pp. 2131–2142, June 2008.