Matrix Decomposition Architecture for MIMO Systems: Design and Implementation Trade-offs

C. Studer*, P. Blösch, P. Friedli, and A. Burg*
*Integrated Systems Laboratory, ETH Zurich, Switzerland
email: {studer, apburg}@iis.ee.ethz.ch

Abstract— **The singular value decomposition (SVD) and the QR decomposition (QRD) are two prominent matrix decomposition algorithms used in various signal processing applications. In the field of multiple-input multiple-output (MIMO) communication systems, the SVD and the QRD are employed for beamforming and for channel-matrix preprocessing for MIMO detection, respectively. In this paper, we describe a minimum-area matrix decomposition architecture that is programmable to perform QRD *and* SVD with variable precision and we investigate the associated design and implementation trade-offs. Our reference implementation achieves a hardware efficiency of up to 325 k SVDs/s/mm^2 and 1.92 M QRDs/s/mm^2 for complex-valued 4×4-matrices in 0.18 µm CMOS technology.**

I. Introduction

Multiple-input multiple-output (MIMO) communication systems [1] constitute the basis for many upcoming wireless communication standards (e.g., IEEE 802.11n) and offer increased spectral efficiency (compared to single-antenna systems) by transmitting multiple data streams concurrently in the same frequency band. Matrix decomposition algorithms [2], such as the singular value decomposition (SVD) or the QR decomposition (QRD), have applications in various signal processing fields. The SVD, for example, is used in array processing or data compression, but can also be applied to MIMO systems in order to increase the system performance by the use of beamforming and power allocation. The QRD, for example, is a key prerequisite for many advanced MIMO detectors, such as the sphere decoder [3].

The SVD and the QRD mainly base on a specific sequence of Givens rotations [2]. CORDIC (coordinate rotation digital computer) algorithms have shown to be a suitable tool to efficiently perform Givens rotations in hardware [4]. Since the QRD only requires a subset of operations required for the SVD, an architecture which allows to compute the SVD would also provide all the functionality to compute the QRD. Hence, a programmable matrix decomposition architecture based on CORDIC arithmetic results in a single matrix decomposition unit (MDU) and is suitable for both decomposition algorithms.

Due to the relatively high computational complexity of the SVD, systolic arrays based on the Jacobi method have been proposed [4]–[6]. As illustrated in Fig. 1, systolic arrays lie on one end of the area/delay trade-off and are usually designed to achieve short computation time at the cost of large circuit area. However, in MIMO-OFDM systems [1] for example, multiple problems need to be solved concurrently, where the number of parallel tasks corresponds to the number of OFDM tones. The throughput of fast but large architectures (e.g., systolic arrays) is often difficult to match to an arbitrary number of problems, e.g., one systolic array might be insufficient in terms of throughput but two might exceed the available circuit area. Low-area architectures can be obtained by the use of time-sharing and lie on the other end of the area/delay trade-off (see Fig. 1). Ideally, time-shared architectures are equally efficient (in terms of area times the computation time) as systolic arrays and have the key advantage to be easily adaptable to individual throughput requirements by the use of parallel instantiation, i.e., the target throughput can be achieved by replication of a low-area instance. Additional area savings, while not reducing the decomposition throughput, can be obtained by iterative decomposition of each instance (see Fig. 1).

Contributions: We present reference VLSI implementation results of two MDUs optimzied for MIMO systems. The low-area implementation mainly bases on CORDIC arithmetic and is able to perform the QRD and the SVD of complex-valued 4×4 matrices. In order to improve the overall efficiency, the involved design and implementation trade-offs are investigated in detail. We present two architecture variants: one unit has been optimized for throughput and the other offers enhanced flexibility by controlling a precision/throughput trade-off.

Outline: The remainder of this paper is organized as follows. In Sec. II the SVD and QRD algorithms are described and all required operations are identified. The time-shared matrix decomposition archtecture is described in Sec. III and the associated design and implementation trade-offs are explored in Sec. IV. In Sec. V, we provide VLSI implementation results and conclude in Sec. VI.

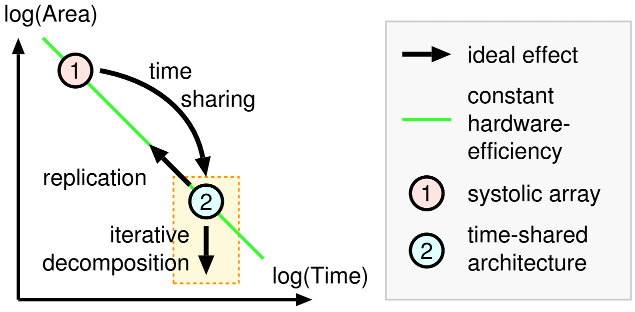

Fig. 1. Ideal impact of architectural transformations [7] on systolic arrays and time-shared architectures. The dashed box corresponds to the investigated design-space exploration for both MDUs by the use of iterative decomposition.

This work was supported by the STREP project No. IST-026905 (MASCOT) within the Sixth Framework Programme (FP6) of the European Commission.

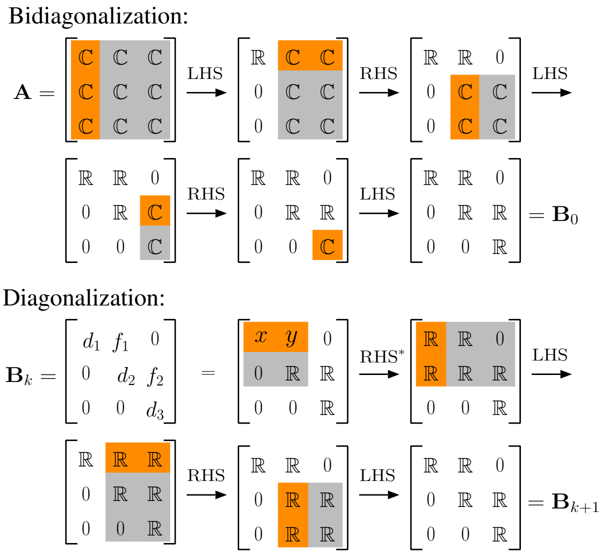

Fig. 2. Illustration of the bidiagonalization and diagonalization phases of the SVD according to [2] for a complex-valued 3×3 matrix. The entries affected in the corresponding update have been highlighted.

II. Matrix Decomposition Algorithms

To identify all required arithmetic operations and the underlying computational sequences, the SVD and QRD algorithms are briefly summarized below. More details for both matrix decomposition algorithms can be found in [2].

A. Singular Value Decomposition

The SVD of a complex-valued $M \times N$-dimensional matrix $\mathbf{A}$ is defined[1] as [8]

$$\mathbf{A} = \mathbf{U}\mathbf{\Sigma}\mathbf{V}^H \tag{1}$$

where $\mathbf{U}$ and $\mathbf{V}$ are complex-valued unitary matrices of dimension $M \times M$ and $N \times N$, respectively. The $M \times N$-matrix $\mathbf{\Sigma}$ contains the real-valued *singular values* on its main diagonal, i.e., $\mathrm{diag}(\mathbf{\Sigma}) = \{\sigma_1, \sigma_2, \ldots, \sigma_r\}$, where $r = \min\{N, M\}$ and $\sigma_1 \geq \sigma_2 \geq \ldots \geq \sigma_r$. The SVD procedure under consideration bases on the Golub-Kahan algorithm described in [2] and mainly performs the SVD in two phases:

1) Bidiagonalization: First, a memory is initialized with $\mathcal{M} = \{\mathbf{I}_M, \mathbf{A}, \mathbf{I}_N\}$ where $\mathbf{I}_L$ stands for a $L \times L$ identity matrix. During the bidiagonalization phase, Givens rotations are successively applied to $\mathbf{A}$ from the left-hand side (LHS) and from the right-hand side (RHS), such that the $M \times N$-dimensional inner matrix $\mathbf{A}$ gets bidiagonal and real-valued (denoted by $\mathbf{B}_0$) as illustrated in Fig. 2. All Givens rotations applied to $\mathbf{A}$ from the LHS and RHS are applied to the corresponding identity matrices. The resulting unitary matrices are denoted by $\tilde{\mathbf{U}}$ and $\tilde{\mathbf{V}}^H$ and the memory content after the bidiagonalization phase corresponds to $\mathcal{M} = \{\tilde{\mathbf{U}}, \mathbf{B}_0, \tilde{\mathbf{V}}^H\}$ where $\mathbf{A} = \tilde{\mathbf{U}}\mathbf{B}_0\tilde{\mathbf{V}}^H$.

2) Diagonalization: The diagonalization phase consists of multiple diagonalization steps (indicated with k) and is illustrated in Fig. 2. Givens rotations are subsequently applied from the LHS and from the RHS to the bidiagonal matrix $\mathbf{B}_k$ such that all off-diagonal entries f_i (for $i = 1, 2, \ldots, r-1$) of $\mathbf{B}_k$ become zero. The diagonalization phase is stopped whenever all f_i are considered to be zero and all d_i (for $i = 1, 2, \ldots, r$) correspond to the *unordered* singular values. In order to ensure convergence of the diagonalization phase and to reduce the overall computation time of the SVD, the first Givens rotation of each diagonalization step (indicated with RHS* in Fig. 2) is performed with a modified input vector $[x\ y]^T$, where $y = t_{12}$ and $x = t_{11} - \mu$ uses the Wilkinson shift [2]

$$\mu = a_n + c - \mathrm{sign}(c)\sqrt{a^2 + b_{n-1}^2} \tag{2}$$

with $c = \frac{1}{2}(a_{n-1} - a_n)$, $\mathbf{T} = \mathbf{B}_k^H\mathbf{B}_k$, and the trailing non-zero sub-matrix of $\mathbf{T}$ corresponds to

$$\mathbf{T}(n-1:n, n-1:n) = \begin{pmatrix} a_{n-1} & b_{n-1} \\ b_{n-1}^* & a_n \end{pmatrix}. \tag{3}$$

Analogous to the bidiagonalization phase, all Givens rotations are also applied to the corresponding unitary matrices such that finally, $\mathcal{M} = \{\mathbf{U}, \mathbf{\Sigma}, \mathbf{V}^H\}$ is the SVD in (1).

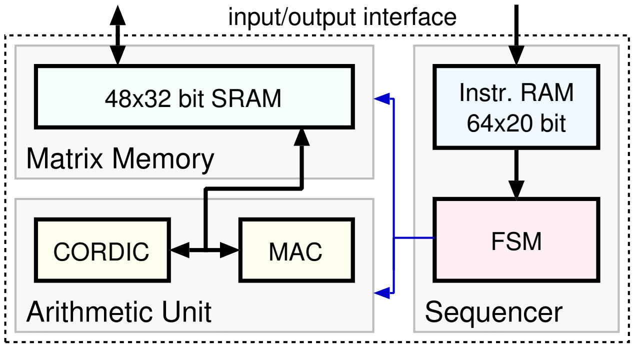

Fig. 3. Matrix decomposition architecture overview: the instruction-based sequencer controls the arithmetic unit and the matrix memory, in order to perform the decomposition sequence stored in the instruction RAM.

B. QR Decomposition

The QR decomposition of a $M \times N$-dimensional complex-valued matrix $\mathbf{A}$ is defined as [8]

$$\mathbf{A} = \mathbf{Q}\mathbf{R} \tag{4}$$

where $\mathbf{Q}$ is a complex-valued $M \times N$-matrix with orthonormal colums and the upper-triangular $N \times N$-matrix $\mathbf{R}$ has real-valued entries on its main diagonal. The QRD is performed in a similar fashion as the bidiagonalization phase of the SVD (cf. Fig. 2) where only the LHS Givens rotations are applied. This minor modification of the SVD algorthm results in an upper-triangular matrix $\mathbf{R}$ with real-valued entries on its main diagonal. All Givens rotations applied from the LHS to $\mathbf{A}$ are also applied to the unitary matrix $\mathbf{Q}$ such that $\mathcal{M} = \{\mathbf{Q}, \mathbf{R}, \mathbf{I}_N\}$ corresponds to the QRD in (4).

III. VLSI Architecture

In contrast to a systolic array implementation, we describe a low-area matrix decomposition architecture, which is designed to operate on complex-valued 4×4-dimensional matrices. The time-shared architecture is depicted in Fig. 3 and consists of tree main components described in the following.

[1] In the following, the superscripts ^{H} and ^{T} stand for conjugate transposition and transposition, respectively.

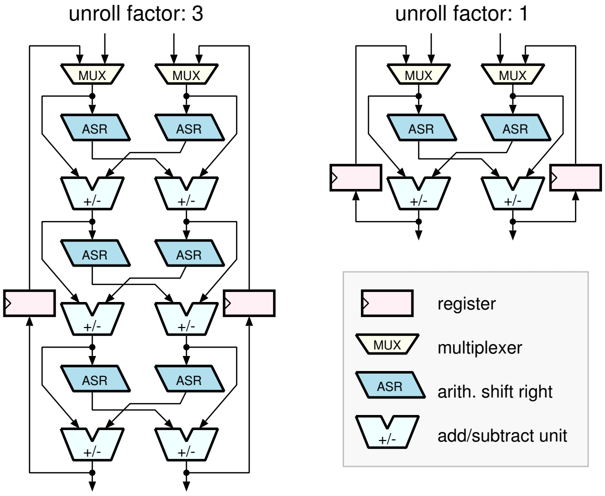

Fig. 4. CORDIC architecture overview: iterative decomposition can (ideally) reduce the area without affecting the computation time per CORDIC. The unroll factor corresponds to the number of micro-rotations per clock cycle.

A. Matrix Memory

The matrix memory provides storage for three complex-valued 4×4 matrices $\mathcal{M} = \{\mathbf{M}_1, \mathbf{M}_2, \mathbf{M}_3\}$, which is sufficient to store the result of an SVD and of a QRD (see Sec. II). A complex value in $\mathcal{M}$ is stored at a single memory adress, is 32 bits wide, and each real and imaginary part requires 16 bits. The matrix memory consists of a two-port 48×32 bit SRAM and requires $0.06\,\mathrm{mm}^2$ in 0.18 µm CMOS technology. The matrix memory interface allows to read or write two different real or imaginary parts in at most two clock cycles.

B. Arithmetic Unit

In order to design a high-level VLSI architecture of the arithmetic unit (AU), Givens rotations, square-roots, multiplications, and additions/subtractions are required to compute the SVD and the QRD (cf. Sec. II). Givens rotations and the square root can efficiently be computed by CORDIC, whereas multiplications and additions/subtractions are computed in a multiply-accumulate (MAC) unit.

CORDIC Unit: CORDICs can efficiently compute two-dimensional rotations [9] by performing a series of micro-rotations with the aid of shifts and additions/subtractions (cf. Fig. 4). To keep the circuit area low, a single CORDIC is used by the means of time sharing and has been designed to support vectoring and rotation. A complex-valued Givens rotation is performed by three real-valued *vectoring* CORDICs (denoted by C_1, C_2, and C_3), i.e.,

$$\begin{bmatrix} \mathbb{C} \\ \mathbb{C} \end{bmatrix} \overset{C_1}{\to} \begin{bmatrix} \mathbb{R} \\ \mathbb{C} \end{bmatrix} \overset{C_2}{\to} \begin{bmatrix} \mathbb{R} \\ \mathbb{R} \end{bmatrix} \overset{C_3}{\to} \begin{bmatrix} \mathbb{R} \\ 0 \end{bmatrix}.$$

In order to perform the corresponding rotation on a complex-valued 2-dimensional vector, four *rotation* CORDICs are required: the first two (C_1 and C_2) rotate each complex entry independently, whereas the third and fourth CORDICs rotates the real and imaginary part of both complex values by C_3.

We emphasize that the square-root required in (2) can efficiently be computed with the CORDIC in vectoring mode, since $\sqrt{d^2 + b_{n-1}^2}$ corresponds to the nonzero result of the CORDIC with $[d\ b_{n-1}]^T$ applied to the input.

Multiply-Accumulate Unit: To compute the trailing submatrix of $\mathbf{T} = \mathbf{B}_k^H \mathbf{B}_k$ as described in (3), a real-valued multiply-accumulate (MAC) unit has been instantiated. The multiplier can be switched off in order to perform additions or subtractions required in (2).

C. Instruction-Based Sequencer

The programmable MDU contains a micro-code controlled sequencer. This sequencer consists of a 64×20 bit instruction RAM (of size $0.04\,\mathrm{mm}^2$ in 0.18 µm CMOS technology) that provides storage for 64 instructions. The finite state machine (FSM) decodes instructions, generatates memory adresses, and provides control signals for the AU.

Instruction Set: The SVD and QRD mainly base on a specific rotation sequence applied to the matrices in $\mathcal{M}$ (cf. Sec. III-A). To this end, a set of eight instructions has been defined. Four instructions are used to apply CORDICs from the left-hand side (LHS) or the right-hand side (RHS) to one complex-value or two real/imaginary-valued entries of $\mathbf{M}_2$ and to update all other affected entries in $\mathcal{M}$. One instruction is used to initialize the diagonalization phase of the SVD which subsequently performs all required diagonalization steps in a self-controlled manner. The remaining instructions are reserved to configure the number of CORDIC micro-rotations (see Sec. IV-A) or to control the program flow. The SVD of a complex-valued 4×4 matrix requires 27 instructions, whereas a QRD of equal size requires only 17 instructions.

SVD-Algorithm Modifications: To simplify the diagonalization phase of the SVD and to obtain a fixed throughput, the following modifications have been applied to [2]:

1) Off-diagonal entries of $\mathbf{B}_k$ (see Fig. 2) are considered to be zero, whenever $f_i < 2^{-\varepsilon}$ for $i = 1, 2, \ldots, r-1$.
2) Since the computational complexity of the diagonalization phase is data dependent, an early-stopping criterion has been introduced to obtain a fixed decomposition throughput. Whenever $k = K_{\max}$, the diagonalization phase is stopped and the current $\mathcal{M}$ is used as an estimate of (1).

Note that ε and $K_{\max}$ can be defined in the SVD-initialization instruction, which allows to reconfigure the arithmetic precision and the decomposition throughput at run time.

IV. Implementation Trade-offs

To reduce the area of the MDU and to improve the overall efficiency, implementation trade-offs associated with arithmetic precision, circuit area, and throughput are investigated in the following.

A. Fixed-Point Implementation Trade-offs

The arithmetic precision of a fixed-point implementation is assessed by the bit error rate (BER) of an IEEE 802.11n-based MIMO-OFDM system with coded beamforming [10]. The baseband input-output relation of the wireless channel is $\mathbf{y} = \mathbf{Hs} + \mathbf{n}$ where $\mathbf{H}$ corresponds to the $M_R \times M_T$-dimensional channel matrix, $\mathbf{s}$ denotes the M_T-dimensional

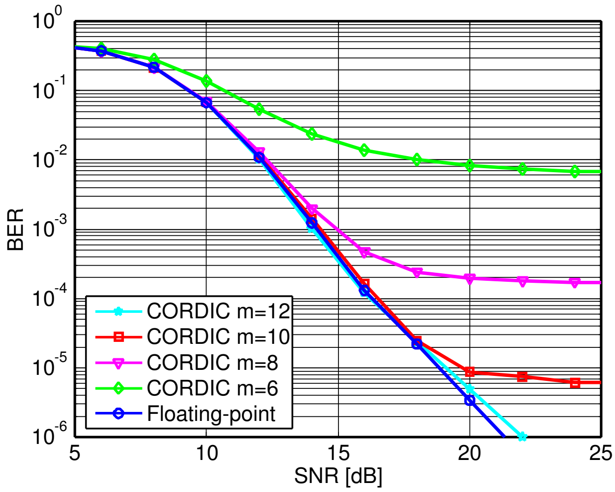

Fig. 5. Fixed-point SVD performance (16 fractional bits in the CORDIC, $\varepsilon = 7$, and $K_{\max} = 7$), measured in a beamformed MIMO-OFDM system. The number of micro-rotations in the CORDIC (denoted by m) has a significant impact on the BER.

transmit signal, $\mathbf{n}$ the M_R-dimensional Gaussian noise vector, and $\mathbf{y}$ the M_R-dimensional receive signal. Beamforming is simulated by computing the fixed-point SVD of the channel matrix $\mathbf{H}$ and by transmitting $\mathbf{s} = \mathbf{V}\tilde{\mathbf{s}}$. The receiver converts the input-output relation into M_T single-input single-output channels

$$\tilde{y}_i = \sigma_i \tilde{s}_i + \tilde{n}_i \qquad \text{for} \qquad i = 1, 2, \ldots, M_T$$

where $\tilde{\mathbf{y}} = \mathbf{U}^H\mathbf{y}$, σ_i is the ith singular value, and the noise vector $\tilde{\mathbf{n}}$ has equal statistics as $\mathbf{n}$. Finally, a soft-output demapper produces reliability information for the subsequent soft-input channel decoder.

Fixed-Point Implementation: To convert the floating-point model in a fixed-point implementation, the floating-point SVD has been simulated to determine the threshold parameter ε and the required maximum number of diagonalization steps. Simulations have shown that setting $\varepsilon = 7$ and $K_{\max} = 7$ does not result in a significant BER performance loss. Note that reducing ε or increasing $K_{\max}$ only increases the computational complexity of the SVD and does not improve the error rate performance. Further simulations have shown that a near-optimal BER is achieved by using 16 fractional bits within the CORDIC. Finally, the required number of micro-rotations in the CORDIC has been evaluated. Figure Fig. 5 shows the impact of micro-rotations to the BER of a beamformed MIMO-OFDM simuation[2]. At least 12 micro-rotations are required to acheive a BER less than 10^{-6}. Reducing the number of micro-rotations results in a BER floor, which is not suitable for the scenario under consideration.

Precision/Throughput Trade-off: The number of micro rotations in the CORDIC directly influences the arithmetic precision of the MDU (see Fig. 5) and has also a significant impact on the throughput. A lower number of micro-rotations requires less clock cycles, which results in lowered computational complexity. We emphasize that the arithmetic precision requirements of a QRD for MIMO detection are usually lower than the precision for the SVD in the scenario considered above. As noted in [12], nine micro-rotations have found to be sufficient for a QRD and thus, it is beneficial to lower the precision in order to increase the decomposition speed. To this end, we propose a tunable arithmetic unit (denoted by AU II), where the number of micro-rotations is programmable, which allows to adjust the precision/throughput trade-off at run time. The maximum number of micro-rotations has been set to 12 to support sufficient precision for computing the SVD. An unroll factor of two has been chosen which allows to chose the number of micro-rotations from 12, 10, 8, and 6, depending on the application and precision requirements. Note that all BERs achievable by AU II are shown in Fig. 5.

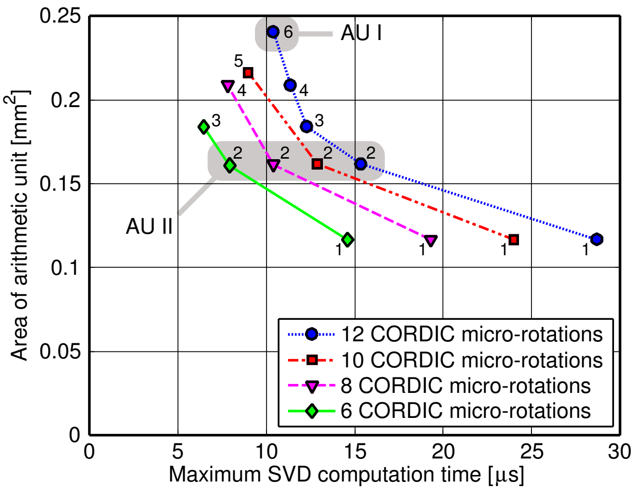

Fig. 6. Area/delay trade-off achievable by iterative decomposition of the CORDIC: the fast AU I computes the SVD with high precision (i.e., 12 micro-rotations). The precision of AU II can be adjusted at run time to 12, 10, 8, and 6 micro-rotations. The numbers next to the curves correspond to the CORDIC unroll factor.

B. Area/Delay Trade-off

Replication of a low-area unit can be used to achieve a given throughput. Lower area implies that the target throughput can be obtained more accurately. Hence, additional reduction in terms of area per decomposition unit, without a significant throughput decrease is highly desirable. To this end, iterative decomposition has been applied to the CORDIC unit to determine the optimum choice of the unroll factor. Since only the area of the AU is affected, the area/delay trade-off associated with the maximum number of micro-rotations and the CORDIC unroll factor is shown in Fig. 6. Two different MDUs have been designed: MDU I uses of the faster but less-flexible AU I (using 12 micro rotations and an unroll factor of six), whereas MDU II employs the slightly slower but configurable AU II (using an unroll factor of two with 12, 10, 8, and 6 micro rotations), which allows to control the precision/throughput trade-off at run time. Note that the critical path is not only determined by the CORDIC, but also by the MAC unit. Thus, to align the critical paths of both units, one pipelining register has been inserted in the multiplier if the CORDIC unroll factor is less than four.

[2] We consider a convolutionally encoded (rate 1/2, generator polynomials $[133_o\ 171_o]$, constraint length 7, random interleaving) MIMO-OFDM system with beamforming [10], four transmit and receive antennas, 16-QAM (using Gray mapping), 64 tones, and soft-input Viterbi decoding. One codeblock corresponds to 1024 bits, a TGn type C [11] channel model is used, and perfect channel state information at the transmitter and receiver is assumed.

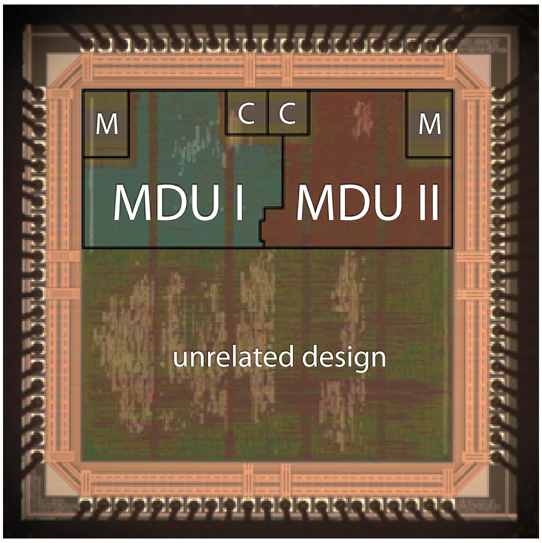

Fig. 7. MDU ASIC in 0.18 µm (1P/6M) CMOS technology. The top left corresponds to MDU I and the top right corresponds to MDU II. The matrix memories and instruction RAMs are denoted by M and C, respectively.

V. Implementation Results

The fabricated ASIC in 0.18 µm technology is depicted in Fig. 7 and contains both MDUs and an unrelated design.

1) SVD: The VLSI implementation results for the SVD for each MDU are given in Tbl. I. Note that for highest precision (i.e., 12 micro-rotations), the first unit achieves a slightly higher throughput than the MDU II and only requires 0.04 mm^2 more area. However, the second unit is able to achieve 55% higher efficiency (in terms of SVDs per second per mm^2) than MDU I by reducing the arithmetic precision down to six micro-rotations per CORDIC (see Fig. 5). At highest precision (i.e., using 12 micro rotations), MDU I and MDU II consume 160 mW and 106 mW, respectively.

2) QRD: The implementation results for a complex-valued QRD executed on each MDU are given in Tbl. II. At maximum precision (i.e., 12 micro-rotations which, however, is only desirable for the SVD computation), MDU I achieves the higher throughput than MDU II. Note that approximately six time more QRDs per second per mm^2 than SVDs are achievable. Reducing the precision of MDU II to six micro-rotations per CORDIC, allows to achieve 1.92 MQRDs/s/mm^2, which is 51% more efficient than the less-flexible MDU I. Hence, tuning the precision/throughput trade-off at run time can improve the overall efficiency of MDU II. Note that the power consumption increases by reducing the precision.

TABLE II

Implementation results of a complex-valued 4×4 QRD

MDU	I	II			
CORDIC rots.	12	12	10	8	6
QRD time [µs]	1.92	2.82	2.35	1.88	1.41
QRDs/s/mm^2	1.27 M	0.96 M	1.15 M	1.44 M	1.92 M
Power^a [mW]	155	105	112	118	128

VI. Conclusion

We described design and implementation trade-offs of two programmable matrix decomposition units (MDUs), able to compute the SVD and the QRD. Low area is achieved through extensive use of time sharing of a single CORDIC unit. The low-area MDUs have been shown to be suitable for MIMO-OFDM systems, since they can be easily adapted to individual throughput requirements by the use of replication. One unit has been optimized for throughput, where the throughput of the second unit is tunable by reducing the arithmetic precision at run time. The programability of both units allow to use the same architecture for different systems, which avoids the design of dedicated architectures for individual requirements.

References

[1] H. Bölcskei, D. Gesbert, C. Papadias, and A. J. van der Veen, Eds., *Space-Time Wireless Systems: From Array Processing to MIMO Communications*. Cambridge Univ. Press, 2006.

[2] G. H. Golub and C. F. van Loan, *Matrix Computations*, 3rd ed. The Johns Hopkins University Press, Baltimore and London, 1996.

[3] A. Burg, M. Borgmann, M. Wenk, M. Zellweger, W. Fichtner, and H. Bölcskei, "VLSI implementation of MIMO detection using the sphere decoder algorithm," *IEEE Journal of Solid-State Circuits*, vol. 40, no. 7, pp. 1566–1577, July 2005.

[4] J. R. Cavallaro and F. T. Luk, "CORDIC arithmetic for an SVD processor," *J. Parallel Distrib. Comput.*, vol. 5, no. 3, pp. 271–290, 1988.

[5] N. D. Hemkumar and J. R. Cavalaro, "A systolic VLSI architecture for complex SVD," in *Proceedings of the 1992 IEEE Intl. Symp. on Circuits and Systems*, May 1993.

[6] S.-F. Hsiao and J.-M. Delosme, "Parallel singular value decomposition of complex matrices using multidimensional CORDIC algorithms," *IEEE Trans. on Signal Processing*, vol. 44, no. 3, pp. 685–697, Mar. 1996.

[7] H. Kaeslin, *Digital Integrated Circuit Design: from VLSI Architectures to CMOS Fabrication*. Cambridge University Press, 2008.

[8] R. A. Horn and C. R. Johnson, *Matrix Analysis*. Cambridge University Press, 1999.

[9] B. Parhami, *Computer Arithmetic Algorithms and Hardware Designs*. Oxford University Press, New York, 2000.

[10] E. Akay, E. Sengul, and E. Ayanoglu, "Bit-interleaved coded multiple beamforming," *IEEE Transactions on Communications*, vol. 55, pp. 1805–1811, Sept. 2007.

[11] V. Erceg *et al.*, *TGn channel models*, May 2004, IEEE 802.11 document 03/940r4.

[12] 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 the IEEE Int. Symp. on Circuits and Systems*, May 2007.

TABLE I

Reference implementation results of a complex-valued 4×4 SVD for 0.18 µm (1P/6M) CMOS technology

MDU	I	II			
CORDIC micro-rotations	12	12	10	8	6
Core area [mm^2]	0.41	0.37			
Maximum clock frequency [MHz]	133	272			
Power Consumption^a [mW]	160	106	113	119	130
Maximum SVD time [µs]	11.57	15.83	13.29	10.75	8.2
Efficiency [SVDs/s/mm^2]	210 k	166 k	198 k	244 k	325 k
Bit error rate floor [BER]	$< 10^{-6}$	$< 10^{-6}$	$\approx 5 \cdot 10^{-6}$	$\approx 2 \cdot 10^{-4}$	$\approx 7 \cdot 10^{-3}$

^{a}Power consumption has been measured at the maximum clock frequency of the corresponding MDU with 1.8 V core supply.