# Implementation of MU-MIMO Schedulers on SoC

Ganesh Venkatraman, Janne Janhunen, and Markku Juntti
Centre for Wireless Communications (CWC), Department of Communication Engineering (DCE),
FI - 90014, University of Oulu, Finland

***Abstract*—To avail the benefits of a multi-antenna system, multi-user multiple-input multiple-output (MIMO) systems multiplex different user data streams spatially over time-frequency resources. However, serving dozens of users with spatial multiplexing increases the scheduler complexity significantly. In this work, we demonstrate the computational requirements of different state-of-the-art MU-MIMO scheduling algorithms and evaluate the number of users that can be supported in real-time. To provide a fair comparison, we initially analyze the achievable sum rate of various scheduling algorithms using MATLAB for different MIMO configurations. We also propose a modification to the existing scheduling algorithm to improve the sum rate. We then implement the scheduling algorithms on Xilinx ZYNQ-ZC702 System-on-Chip (SoC) to illustrate the advantages.**

## I. Introduction

The current wireless standards are moving towards the packet switched networks to improve the system performance and flexibility as compared to the circuit switched networks. The radio access technologies aim at achieving higher throughput and better system performance but still targeting to low energy budget. The use of multi-antenna transmission is *de-facto* in all upcoming wireless access standards, favoring the spatial multiplexing of user data by applying transmit precoders for multi-user multiple-input multiple-output (MIMO) transmission. In order to achieve the best possible benefit of the multi-user MIMO transmission, the multiplexed users should have channel vectors as linearly independent as possible. By selecting users with the uncorrelated channel vectors, efficient linear precoders can be designed to decouple the user data streams and to avail the benefits of MU-MIMO technique. The selection of users with such a constraint is carried out by the schedulers to utilize the wireless system resources efficiently.

The scheduling algorithms based on sum rate maximization objective for MU-MIMO were discussed thoroughly in the literature. The search based on successive projections (SP) scheme for single-antenna receiver was presented in [1] and its extension to the multi-antenna receivers was provided in [2]. In [2], the users were selected iteratively by choosing the channel vector with the maximum gain on to the orthogonal subspace. The orthogonal subspace was obtained by evaluating the null space of the stacked channel vectors of the already chosen users from the earlier iterations. Similar algorithms addressing a lower computational complexity were proposed in [3] and [4]. User selection based on the volume maximization metric was discussed in [5]. The performance of the volume based selection is identical to the SP or block diagonalization (BD) scheme, since both schemes use the Gram-Schmidt (GS) procedure.

We evaluate the performance of the scheduling algorithms by the achievable sum rate. The iterative precoder design based on weighted minimum mean squared error (WMMSE) reformulation for the weighted sum rate maximization (WSRM) objective provides an efficient transmit precoder design to achieve improved performance [6]. However, the computational complexity involved in the WMMSE design is large in comparison to the elementary zero-forcing (ZF) based precoder design proposed in [7]. Even though the ZF based precoder design is inferior to the WMMSE approach, it provides a closed form expression to evaluate the precoder design followed by a bisection search to identify the power for each data stream. The choice of precoder design also plays a major role in the achievable sum rate. In the current work, we rely on the ZF based precoder design as compared to the iterative WMMSE approach.

In this paper, we study the computational complexity of different state-of-the-art MU-MIMO scheduling algorithms and evaluate the number of users that can be served in the system. To enable a fair comparison, we first analyze the achievable sum rate performance of the different algorithms and then we discuss the implementation of the scheduling schemes on Xilinx ZYNQ 7000 ZC702 System-on-Chip (SoC) to show how the current systems can take advantage of MU-MIMO techniques. This work is a continuation to our previous implementation studies [8]–[10] aiming at finding low-power but high performance solutions for the advanced long term evolution (LTE-A) transceivers. Current work excludes precoding implementation, but we considered a two-step scheduling-precoding design approach, where the scheduler identifies a subset of users in the first step and then the precoders are designed for the chosen subset in the second step.

The paper is organized as follows. A single-cell downlink system model is described in Section II followed by a brief discussion on the scheduler algorithms in Section III. Section IV presents the performance comparison between selected scheduler schemes using link level simulations. In Section V, implementation results of different scheduling algorithms are presented. Section VI summarizes the conclusions and future work.

## II. System Model

We consider a single-cell downlink MU-MIMO transmission with $N_T$ transmit antennas and $N_K$ users with $N_R$ antenna each. Let $\mathbf{H}_k \in \mathbb{C}^{N_R \times N_T}$ be the channel matrix seen by user $k$. Let $\kappa = \min(N_R, N_T)$ be the rank of the channel matrix $\mathbf{H}_k$. In order to perform scheduling over multiple spatial streams, the channel matrix $\mathbf{H}_k$ is decomposed into $N_R$ virtual single antenna receiver channels using singular value decomposition (SVD) as $\mathbf{H}_k = \mathbf{U}_k \mathbf{D}_k \mathbf{V}_k^{\mathrm{H}}$, where $\mathbf{U}_k \mathbf{U}_k^{\mathrm{H}} = \mathbf{I}_{N_R}$ and $\mathbf{V}_k \mathbf{V}_k^{\mathrm{H}} = \mathbf{I}_{N_T}$ are unitary matrices and $\mathbf{D}_k$ is a diagonal matrix. The virtual user index for the $i^{\text{th}}$ column vector of $\mathbf{U}_k \in \mathbb{C}^{N_R \times N_R}$ of the $k^{\text{th}}$ user is denoted by $\hat{k} = k\kappa + i$ and the equivalent single antenna receiver channel is $\mathbf{h}_{\hat{k}} = \mathbf{U}_k(i)^{\mathrm{H}} \mathbf{H}_k$. The total number of virtual single antenna users in the system is given by $K = \kappa N_K$. Let $\mathcal{U}$ be the set of indices of the users present in the system and let $\mathcal{A} \subset \mathcal{U}$ be the set of active users chosen by the scheduling algorithm for a transmission instant. For brevity, the virtual users are referred as users in the forthcoming discussion. Now, the equalized received symbol $\hat{d}_k$ of user $k \in \mathcal{A}$ is given by

$$\hat{d}_k = \mathbf{h}_k \mathbf{m}_k d_k + \sum_{i \in \mathcal{A} \setminus \{k\}} \mathbf{h}_k \mathbf{m}_i d_i + n_k, \qquad (1)$$

where $n_k \in \mathbb{C}$ is the equivalent zero mean complex Gaussian noise with variance $\mathcal{N}(0, N_0)$ and $\mathbf{m}_k \in \mathbb{C}^{N_T \times 1}$ is the transmit precoder corresponding to user $k$. The data symbol $d_k$ is assumed to be uncorrelated with $\mathrm{E}[|d_k|^2] = 1$.

## III. Scheduling Algorithms

The selection of a subset of users $\mathcal{A}$ from $\mathcal{U}$ is a combinatorial problem in general. Therefore it requires the complexity of $O(K^{N_T})$. Fig. 1 shows the outline of scheduling algorithms discussed in this section. Due to the simplicity of the greedy/norm based search, we omit the discussion on the greedy scheduling algorithm.

### A. Successive Projections

In the iterative SP selection algorithm, the compatibility of the user channel vectors are evaluated by projecting the channel vector on to the null space formed by stacking the existing channel vectors of the chosen users from the earlier iteration [1], [11]. Initially, a user with higher channel gain from the set $\mathcal{U}$ is selected for the transmission set $\mathcal{A}$. Then, the remaining users are selected from $\mathcal{U} \setminus \mathcal{A}$ by projecting the equivalent channel vectors $\mathbf{h}_{\hat{k}} = \mathbf{U}_k(i)^{\mathrm{H}} \mathbf{H}_k$ on to the null space of the user channel vectors in $\mathcal{A}$ as

$$\mathbf{F} = \left[\mathbf{h}^{\mathrm{T}}_{\mathcal{A}(1)}, \ldots, \mathbf{h}^{\mathrm{T}}_{\mathcal{A}(|\mathcal{A}|)}\right] \qquad (2a)$$

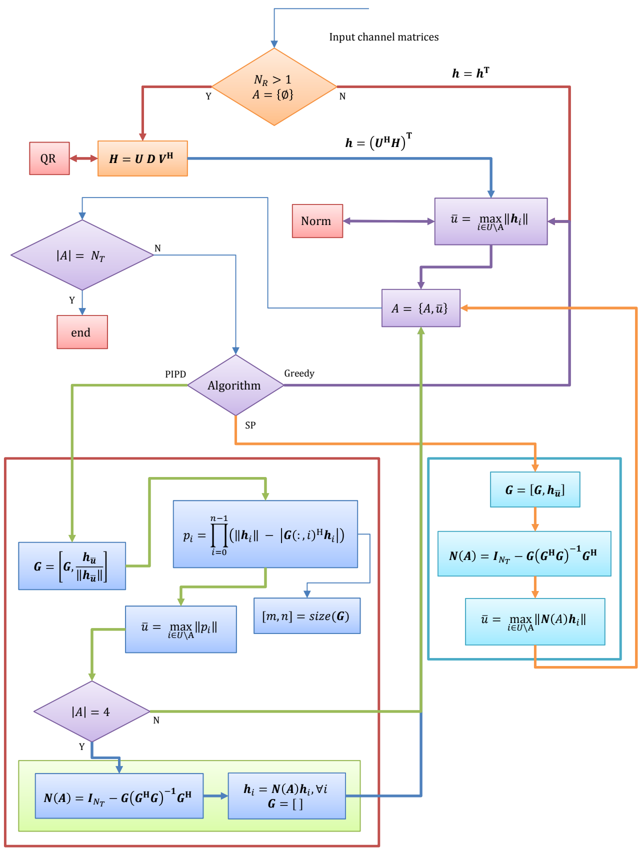
$$\mathbf{N}(\mathcal{A}) = \mathbf{I}_{N_T} - \mathbf{F}\left(\mathbf{F}^{\mathrm{H}}\mathbf{F}\right)^{-1}\mathbf{F}^{\mathrm{H}} \qquad (2b)$$

where $\mathbf{N}$ denotes the null space matrix and $\mathbf{I}_{N_T}$ is the identity matrix of size $N_T$. The metric used for the user selection is given by

$$m_i = \|\mathbf{N}(\mathcal{A})\, \mathbf{h}_i^{\mathrm{T}}\|, \ \forall\, i \in \mathcal{U} \setminus \mathcal{A} \qquad (3a)$$

$$\bar{u} = \arg\max_i m_i, \quad \mathcal{A} = \mathcal{A} \cup \{\bar{u}\}\,. \qquad (3b)$$

The above metric is evaluated for all users at each iteration to identify a user until the condition $|\mathcal{A}| = N_T$ is satisfied. Even though selecting $N_T$ users is not optimal in the lower signal-to-noise ratio (SNR) regime, the precoders can be recalculated by retaining the channel vectors with non-zero power.



Input channel matrices
$N_R > 1$ $A = \{\emptyset\}$
Y
N
$h = h^{\mathrm{T}}$
QR
$H = U\,D\,V^{\mathrm{H}}$
$h = (U^{\mathrm{H}}H)^{\mathrm{T}}$
Norm
$\bar{u} = \max_{i \in U \setminus A} \|h_i\|$
$|A| = N_T$
N
Y
end
$A = \{A, \bar{u}\}$
PIPD
Algorithm
Greedy
SP
$G = \left[G, \frac{h_{\bar{u}}}{\|h_{\bar{u}}\|}\right]$
$p_i = \prod_{i=0}^{n-1}(\|h_i\| - |G(:,i)^{\mathrm{H}} h_i|)$
$\bar{u} = \max_{i \in U \setminus A} \|p_i\|$
$[m,n] = size(G)$
$|A| = 4$
N
Y
$N(A) = I_{N_T} - G(G^{\mathrm{H}}G)^{-1}G^{\mathrm{H}}$
$h_i = N(A)h_i, \forall i$ $G = [\,]$
$G = [G, h_{\bar{u}}]$
$N(A) = I_{N_T} - G(G^{\mathrm{H}}G)^{-1}G^{\mathrm{H}}$
$\bar{u} = \max_{i \in U \setminus A} \|N(A)h_i\|$

Fig. 1. Block diagram of the scheduler algorithms: greedy, product of independent projections and successive projections.

### B. Product of Independent Projections (PIPD)

The PIPD algorithm requires minimal computational complexity compared to the SP scheduling scheme as discussed in [12]. The complexity involved in selecting the first user remains the same, since the user with the highest channel norm is considered for the set $\mathcal{A}$ in both schemes. The complexity involved in selecting the remaining users for set $\mathcal{A}$ is significantly reduced in the PIPD scheduling scheme by virtue of using the product of independent vector projection displacements metric. A detailed description of the algorithm can be found in [12]. However, for a system configuration of $N_T = 8$ antennas, the performance of the PIPD scheme is significantly inferior to that of the SP scheduling algorithm. To overcome this problem, we modified the PIPD algorithm as highlighted in Fig. 1 by projecting user channel vectors on to the null space of existing users channel only when $|\mathcal{A}| = 4$.

Fig. 1 illustrates the operations involved in the metric calculations for greedy, SP and PIPD scheduling schemes. The blocks mentioned separately for the PIPD and the SP schemes are executed based on the selected algorithm. Greedy algorithm requires only the norm calculation of the virtual channel vectors with which the sorting operation can be used to find the leading $N_T$ users for set $\mathcal{A}$. In case of the SP selection scheme, the users are identified by projecting the corresponding channel vectors onto the null space of the existing users, channel vectors and selecting the one with the

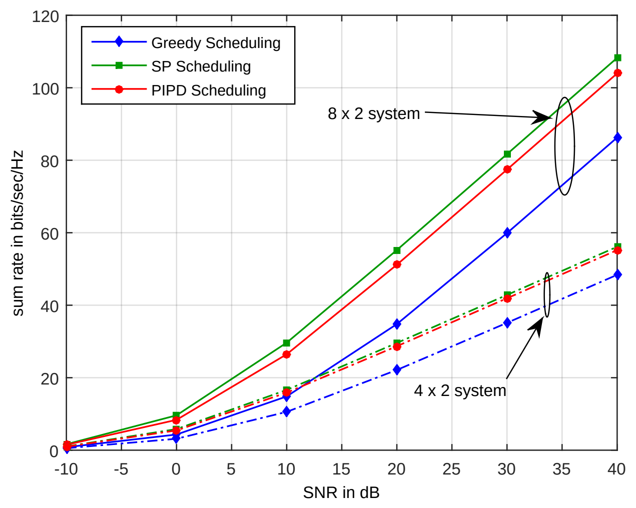


Fig. 2. Comparison of scheduler algorithms for $N_K = 50$ users.

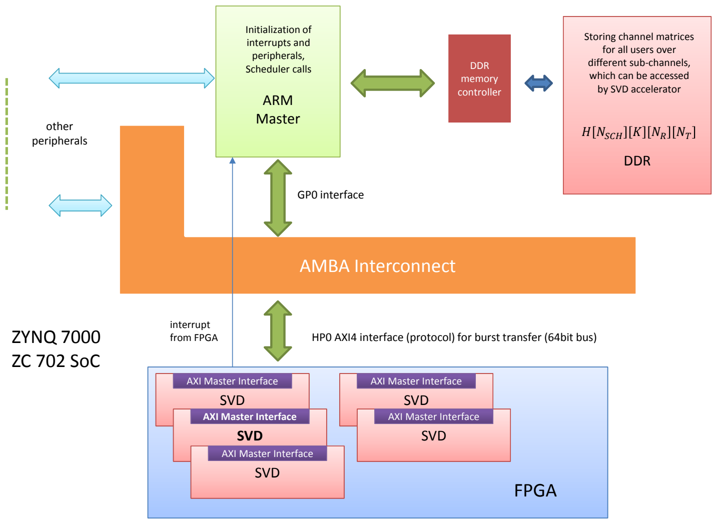

Initialization of interrupts and peripherals, Scheduler calls
ARM Master
DDR memory controller
Storing channel matrices for all users over different sub-channels, which can be accessed by SVD accelerator
$H[N_{SCH}][K][N_R][N_T]$
DDR
other peripherals
GP0 interface
AMBA Interconnect
ZYNQ 7000
ZC 702 SoC
interrupt from FPGA
HP0 AXI4 interface (protocol) for burst transfer (64bit bus)
AXI Master Interface
SVD
AXI Master Interface
SVD
AXI Master Interface
SVD
AXI Master Interface
SVD
AXI Master Interface
SVD
FPGA

Fig. 3. Software and hardware partitioning on ZYNQ 7000.

maximum projection gain. In order to evaluate the null space, a matrix inversion is needed, which requires significant amount of complexity as compared to the vector multiplications involved in the PIPD algorithm in Fig. 1.

## IV. Performance of MU-MIMO Scheduling

### A. Scheduling Resolution

The scheduling is performed over scheduling blocks (SBs) and the resolution of a SB significantly affects the performance of the overall throughput. For instance, if the SB resolution is equal to physical resource block (PRB), the scheduling complexity increases linearly with the number of SBs and the throughput performance depends on the number of users in the system. Thus, if the number of users is large, the performance will improve by utilizing the multi-user diversity with the help of a proper scheduling scheme. However, if the users are few in number, the performance degradation is quite significant due to the smaller coding length and lack of channel variations over the smaller SB to extract the channel diversity in the form of coding gain. In case of larger SB size with grouped PRBs, the scheduling complexity will be reduced significantly but the network throughput will also reduce noticeably, unless the channel fading over frequency domain is flat or nearly flat. In this case, the users with the transmission in such scheduling will attain higher throughput due to the longer coding length which needs to be quantified only by the system level study.

### B. Scheduling Performance

Fig. 2 compares the sum rate performance of three different scheduling schemes with $\mathrm{N}_T = 4, 8$ and $N_K = 50$ users. The sum rate of the PIPD scheme performs remarkably close to the SP scheme. Fig. 2 also shows a larger gap (approximately 10 bit/sec/Hz) between Norm/Greedy scheme which selects users based on the channel norm only and PIPD scheduling scheme. Note that the performance of the scheduling algorithms improves marginally by adding additional receive antennas, due to the channel hardening by the multi-user diversity. Increasing $N_T$ antennas in the system, improves the achievable sum rate of the system by multiplexing more number of users over the spatial dimension, which is evident from Fig. 2.

## V. Implementation Results

### A. ZYNQ ZC702 SoC Evaluation Platform

The MU-MIMO scheduler algorithms are implemented on the ZYNQ ZC702 SoC, which has dual core ARM processing subsystem (PS) and programming logic (PL) that are interconnected by Advanced Microcontroller Bus Architecture (AMBA) architecture bus [13]. In this study, we limit our scheduler implementation on to a single ARM PS only and utilize the maximum available resources of the PL in the board. The ARM PS is operating at 667 MHz and the PL operates at 100 MHz. The interconnect between the ARM PS and the PL is via Advanced Extensible Interface (AXI) AMBA interface bus. There are several high performance (HP) buses existing between the PS and the PL subsystems to reduce the transfer latency. We used HP0 interface to interconnect the accelerators implemented on PL to access the DDR memory, where the intended data are stored for the common access.

### B. Algorithm Partitioning between ARM PS and PL subsystem

In this work, we consider performing SVD processing on PL subsystem and scheduling algorithms implementation on ARM PS for the following reasons. Primarily, scheduling algorithms involve sequential processing. Even though metric calculations can be performed in parallel, however to select a candidate user, the maximum metric needs to be obtained over all parallel processing chains. Note that the NEON vector floating point (VFP) co-processor supports up to 8 lanes that can perform 8 computations and searches in parallel, and therefore it would be beneficial to consider PS implementation for less complex parallel implementations. Moreover, scheduling is required for every sub-frame, since it depends on the current number of backlogged packets. On the contrary, SVD processing is required only when the channel matrices change, *i.e.*, once in every $T_k$ sub-frames, where $T_k$ denotes the periodicity of uplink sounding pilots to measure channel that is based on Doppler frequency. Upon considering the above reasons, it would be beneficial to implement scheduling algorithms on ARM PS and SVD processing on PL subsystem.

## *C. SVD Implementation on PL*

The SVD is performed by repeated QR factorization to obtain left and right singular vectors of the channel matrices. The complex QR decomposition is performed using the Householder transformations [14]. Since the SVD incurs significant complexity and computational time, it is beneficial to implement on the PL. We use Vivado high-level synthesis (HLS) tool to implement SVD using C. To minimize the resource utilization and to optimize the speed through pipelined processing, respective directives are used in the HLS model.

The choice of fixed point implementation is considered over floating point design for the following reasons. At first, to implement a floating point multiply and accumulate (MAC) operation, LogiCORE IP requires three *DSP48* units, whereas fixed point implementation needs only one DSP48 unit by considering INT16 data format. Secondly, the precision required for real and imaginary part of channel entries can be limited to 16 bits, since the output resolution of analog-to-digital converter (ADC) is typically in the order of 14 bits. Therefore, considering the above two facts, we performed SVD implementation using fixed point representation. The fixed point version uses 18 bit signed representation for real and imaginary components with 15 bit fractional part and 2 bit for the integer. To improve the dynamic range of the accelerator, intermediate variables are stored in 25 bit extended format with 9 bit for the integer part and one bit for the sign without altering the fractional part share. It would be ideal, since one *DSP48* can perform $25 \times 18$ multiplication without compromising on the resource utilization and the dynamic range involved in the MAC operations [13].

The SVD accelerator is designed to perform multiple SVD operations with a single trigger from ARM PS. AXI-*Lite* interface is used to provide control information, including matrix size, number of matrices, and streams required, to PL accelerator from ARM PS. Once the control information is processed, matrices are fetched from DDR memory to local BRAM for SVD processing by the PL subsystem. To decompose the channel matrices of all users, we implement five SVD accelerators in parallel to improve the overall throughput of PL implementation. The restriction on total count of accelerators is due to resource limitations imposed by the ZC702 board as outlined in Table I. Each SVD accelerator is connected as AXI-*Master* to DDR memory via a 64 bit HP0 bus interface. Once configured from ARM PS, each SVD accelerator transfers all matrices from DDR memory with burst mode of transfer without processor intervention. Upon completing SVD for all channel matrices, equivalent virtual channel vectors corresponding to each user are written back via the same interface to a different address in DDR memory using an offset field configured by the ARM PS.

TABLE I
FPGA RESOURCE UTILIZATION FOR SVD

| Resource | Utilization % |
|---|---|
| Flip Flops | 39.4 |
| LUT | 72.6 |
| Memory LUT | 6.2 |
| BRAM | 60.7 |
| DSP48 | 86.4 (limiting resource) |
| BUFG | 3.1 |

Algorithmic optimizations are carried out on the SVD implementation by restricting the matrix multiplications only for the left singular vectors in each QR decomposition. Once the left singular matrix is computed, equivalent virtual single antenna channels are obtained as $\mathbf{h}_{\hat{k}} = \mathbf{U}_k^{\mathrm{H}}\mathbf{H}_k \in \mathbb{C}^{N_R \times N_T}$. Burst transfer is used to collectively write back all users equivalent virtual channel vectors to an output address in the DDR memory and an interrupt is given to the PS to notify the completion of the task. The burst mode transfer is opted to reduce the overhead involved in the memory transfer. Table I outlines the overall amount of PL resources consumed by 5 parallel SVD accelerators and its interconnects with the PS subsystem. Each SVD accelerator is implemented for $8 \times 8$ matrix. However, the actual operation is performed over the limited matrix size specified by the processor while initializing the accelerator. In this way, the computations are optimized for variable channel matrix sizes.

## *D. Scheduler Implementation on ARM*

The scheduling algorithms and the overall control is performed on the PS. Since we analyze only single scheduling block for the evaluation purposes, we require only a single ARM core to perform all the scheduler processing. To reduce the latency and the complexity, we resort to the fixed point implementation of the scheduler algorithms on the PS. The complex baseband channel is represented in a packed real and imaginary format with 16 bit representation by allocating 15 bits for the fractional part and one bit to represent the sign. Even though finding the null space of a matrix is computationally complex, the fixed point implementation is managed in the PS instead of the PL implementation to avoid the overhead involved in the transfer between PS and PL subsystems. The amount of overhead involved in the transfer would shadows the timing advantage obtained by the PL implementation of the null space matrix computation. The overall operation of the PS and PL subsystems are balanced since there is no starvation of resources by any subsystems after the initial SVD operation carried out in the PL. The complexities of various schemes are tabulated in Table II. Note that the overall complexity involved in the scheduler implementation can be reduced by limiting the stream search to the dominant streams only, since the singular values are sorted in the descending order of magnitude. The degradation in the performance is marginal due to the available multi-user diversity.

TABLE II
COMPUTATIONAL COMPLEXITY FOR ONE SB WITH $N_K = 50$ USERS

| $N_T \times N_R$ | $\lambda$ streams | SVD Comp. msec | Greedy msec | SP msec | PIPD msec |
|---|---|---|---|---|---|
| $8 \times 4$ | 4 | 11.427 | 0.098 | 2.029 | 1.280 |
| $8 \times 4$ | 2 | 11.427 | 0.084 | 1.202 | 0.743 |
| $8 \times 2$ | 2 | 3.231 | 0.079 | 1.225 | 0.704 |
| $8 \times 2$ | 1 | 3.231 | 0.072 | 0.796 | 0.421 |
| $4 \times 4$ | 4 | 5.169 | 0.052 | 0.327 | 0.237 |
| $4 \times 4$ | 2 | 5.170 | 0.042 | 0.189 | 0.168 |
| $4 \times 2$ | 2 | 1.481 | 0.038 | 0.202 | 0.139 |
| $4 \times 2$ | 1 | 1.481 | 0.033 | 0.131 | 0.089 |

Table II compares the complexity of various algorithms using the time as the metric for $N_K = 50$ users. The legend $\lambda$ in Table II denotes the number of streams included in the scheduler algorithm search. It can take the maximum value of $\kappa$, which is the rank of the channel. As seen from Table II, significant amount of time is spent on the SVD computations. In total, the above implementation can perform $4,375$ SVD's with 16 QR decompositions per second for $8 \times 4$ system. On the other hand, it can perform $33,760$ SVD operations per second for a $4 \times 2$ channel matrix. The complexity of the PIPD algorithm is attributed to the norm evaluation which includes square root operation as well. The complexity of SP algorithm is due to the null space matrix evaluation and also due to the matrix-by-vector multiplication. Note that the scheduling algorithms are within the real-time constraint of 1 msec LTE sub-frame with exceptions on the $8 \times 4$ antenna configuration. The SVD operation can be performed over the radio frame of 10 msec. duration, which can be justified by the nomadic user assumption with the slowly changing channel fading.

The computational figures provided in Table II for SVD scales linearly with each additional user in the system and it increases with the number of users $N_K$ with each additional SB. The complexity of the scheduling algorithms scales linearly with the additional number of SBs while with the number of users, the is sublinear. Table II also provides information about the scalability of the system with the number of SBs that can be supported by the SoC implementation without compromising the real-time constraints. For example, assuming the channel is coherent for 10 ms, we can support 7 SBs with a $4 \times 2$ MIMO configuration for SVD computations. The current implementation can perform any one of the discussed scheduling scheme for 10 SBs at every 1 ms duration for a $4 \times 2$ system with $N_K = 50$ users by utilizing the additional PS core to operate in parallel.

## VI. CONCLUSIONS

We studied the computational complexity of different state-of-the-art MU-MIMO scheduling algorithms. We showed that the complexity is mainly attributed to the SVD decomposition of the channel matrices and the matrix multiplications involved in the metric calculations. We have demonstrated the performance of three different scheduling algorithms on a Xilinx ZYNQ ZC702 evaluation platform by partitioning the resources between PL subsystem and ARM processing core. We have shown that the current implementation can handle all scheduling algorithms with 50 users by considering $4 \times 4$ MIMO scenario. For a comparative study, we addressed the complexity and sum rate performance of different scheduling algorithms by implementing the same on TI Keystone II System-on-Chip (SoC) multi-core platform in [15].

## ACKNOWLEDGMENT

This research was done in BaSE (Baseband and System Technologies for Wireless Evolution) project supported by the Finnish Funding Agency for Innovation (Tekes), Nokia, Broadcom, Xilinx and Bittium Wireless.

## REFERENCES

[1] T. Yoo and A. Goldsmith, "On the optimality of multiantenna broadcast scheduling using zero-forcing beamforming," in *IEEE Journal on Selected Areas in Communications*, vol. 24, no. 3. IEEE, march 2006, pp. 528 – 541.

[2] A. Tolli and M. Juntti, "Scheduling for Multiuser MIMO Downlink with Linear Processing," in *Personal, Indoor and Mobile Radio Communications, 2005. PIMRC 2005. IEEE 16th International Symposium on*, vol. 1, Sept 2005, pp. 156–160.

[3] Z. Shen, R. Chen, J. Andrews, R. Heath, and B. Evans, "Low complexity user selection algorithms for multi-user MIMO systems with block diagonalization," in *IEEE Transactions on Signal Processing*, vol. 54, no. 9. IEEE, 2006, pp. 3658–3663.

[4] Z. Youtuan, T. Zhihua, and Z. Jinkang, "An Improved Norm-Based User Selection Algorithm for Multiuser MIMO Systems with Block Diagonalization," in *IEEE 66th Vehicular Technology Conference, VTC-Fall*. IEEE, 2007, pp. 601–605.

[5] L. Jin, X. Gu, and Z. Hu, "A Novel Volume-Based Scheduling Scheme for Multiuser multiple-input multiple-output Downlink System," in *IEEE Conference on Radio and Wireless Symposium (RWS)*. IEEE, 2010, pp. 448–451.

[6] Q. Shi, M. Razaviyayn, Z.-Q. Luo, and C. He, "An Iteratively Weighted MMSE Approach to Distributed Sum-Utility Maximization for a MIMO Interfering Broadcast Channel," in *IEEE Transactions on Signal Processing*, vol. 59, no. 9. IEEE, 2011, pp. 4331–4340.

[7] Q. H. Spencer, A. L. Swindlehurst, and M. Haardt, "Zero-forcing methods for downlink spatial multiplexing in multiuser MIMO channels," in *IEEE Transactions on Signal Processing*, vol. 52, no. 2. IEEE, 2004, pp. 461–471.

[8] J. Janhunen, T. Pitkänen, O. Silvén, and M. Juntti, "Fixed- and Floating-Point Arithmetic Processor Comparison for MIMO-OFDM Detector."

[9] T. Hänninen, J. Janhunen, and M. Juntti, "Novel Detector Implementations for 3G LTE Downlink and Uplink," *Analog Integr. Circuits Signal Process.*, vol. 78, no. 3, pp. 645–655, Mar. 2014. [Online]. Available: http://dx.doi.org/10.1007/s10470-013-0128-5

[10] S. Shahabuddin, J. Janhunen, M. Juntti, A. Ghazi, and O. Silvén, "Design of a Transport Triggered Vector Processor for Turbo Decoding," *Analog Integr. Circuits Signal Process.*, vol. 78, no. 3, pp. 611–622, Mar. 2014. [Online]. Available: http://dx.doi.org/10.1007/s10470-013-0183-y

[11] A. Tolli and M. Juntti, "Scheduling for Multiuser MIMO Downlink with Linear Processing," in *IEEE 16th International Symposium on Personal, Indoor and Mobile Radio Communications, (PIMRC)*, vol. 1. IEEE, sept. 2005, pp. 156 –160.

[12] G. Venkatraman, A. Tolli, J. Janhunen, and M. Juntti, "Low Complexity Multi-User MIMO Scheduling for Weighted Sum Rate Maximization," in *Signal Processing Conference (EUSIPCO), 2013 Proceedings of the 22nd European*. IEEE, 2014, pp. 820–824.

[13] *Zynq-7000 - All Programmable SoC (UG585)*, Xilinx Inc., Feb 2015. [Online]. Available: http://www.xilinx.com/support/documentation/user_guides/ug585-Zynq-7000-TRM.pdf

[14] H. Simon, *Adaptive Filter Theory*. Prentice Hall, 2002, vol. 3.

[15] G. Venkatraman, J. Janhunen, and M. Juntti, "Exploiting Multi-Core SoC Architecture for MU-MIMO Schedulers," in *3rd IEEE Global Conference on Signal & Information Processing GlobalSIP*. IEEE, Dec. 2015, (to appear).