# HARDWARE-ACCELERATED DESIGN SPACE EXPLORATION FRAMEWORK FOR COMMUNICATION SYSTEMS

Markus Kock, Sebastian Hesselbarth, Holger Blume (Institute of Microelectronic Systems, Leibniz Universität Hannover, Hanover, Germany; {kock, hesselbarth, blume}@ims.uni-hannover.de)

## ABSTRACT

We present a field programmable gate array (FPGA) based hybrid hardware-in-the-loop design space exploration (DSE) framework combining high-level tools (e.g. MATLAB/Simulink) and highly optimized hardware implementation. This enables derivation of comprehensive cost models using Monte-Carlo methods and quick development and optimization of signal processing hardware blocks. The achievable simulation speedup is a key enabling the characterization and optimization of complex communication systems using Monte-Carlo approaches which are infeasible for pure software simulation due to the large required stimuli sets. The framework supports a divide-and-conquer approach through a flexible partitioning of complex algorithms across the system resources on different layers of abstraction. This facilitates efficiently splitting the design process to different teams. The framework comprises a generic System-on-Chip (SoC) infrastructure template, module wrappers, the MATLAB-to-FPGA communication layer and design space navigation algorithms. The hardware template can be synthesized for a variety of emulator-like FPGA platforms. Results of a DSE for interference alignment algorithms are given as a case study.

## 1. INTRODUCTION

High throughput wireless communication systems including LTE, ECMA-368 (WiMedia) and IEEE 802.11ac are built around sophisticated digital signal processing algorithms. Among the research goals for future communication standards are higher spectral efficiency, higher energy efficiencies towards the Shannon limit and increased data rates. Naturally, these benefits come at the price of higher computational complexity. The demand for flexible realtime hardware platforms capable of delivering the required huge number of operations per second at a severely limited power and silicon area budget has led to the development of specialized hardware platforms for software defined radio (SDR) applications.

FPGA-based rapid prototyping systems are widely used in algorithm research and development. Combined with automatic and semi-automatic high-level description to hardware description language (HDL) code generation tools, they facilitate quick hardware deployment invaluable for proof-of-concept studies. However, the hardware efficiency achievable by this design flow is often limited and insufficient for real-world applications and the verification of application-specific integrated circuit (ASIC) designs. Techniques from FPGA-based ASIC verification and rapid prototyping are combined in this project for the design space exploration of highly optimized hardware architectures.

Design time is a limited resource, thus a high design efficiency is a further goal. In a typical implementation scenario for complex designs, high level reference models are used. The choice of optimization-blocks is often based on profiling results, with those blocks contributing significantly to the overall resource requirements being chosen for optimization. This leads to a hybrid design consisting of a mixture of high level blocks and highly optimized blocks, running on hardware ranging from general purpose processors (GPP), application-specific instruction-set processors, FPGA-based rapid prototyping systems and dedicated hardware accelerators. The presented design space exploration framework reflects this structure and allows the designer to freely move processing blocks between the different layers of optimization.

We present an FPGA-based design space exploration framework providing a state of the art System-on-Chip infrastructure template, debug facilities and a design flow for the joint optimization of hardware blocks and their design space exploration in Section 4. As a case study using the framework, the implementation and characterization of interference alignment (IA) algorithms has been chosen. Implementation results for 3-user 2x3 multiple-input multiple-output (MIMO) antenna selection IA are presented in Section 5.2. In Section 6, a parameterized hardware complexity estimation of K-user iterative minimum mean square error (MMSE) IA is presented, identifying it as a demanding candidate for an optimized hardware implementation and DSE.

## 2. RELATED WORK

FPGA-based emulation systems are commonly used for ASIC design verification, simulation acceleration and rapid

prototyping. While the underlying hardware platforms exhibit similar characteristics, the deployed design flow greatly varies depending on the application. A typical rapid prototyping design flow supported by the commercial BEE4 platform is shown in Fig. 1. The design flow allows the integration of hardware accelerators, e.g. signal processing blocks into a high-level MATLAB/Simulink system model. While facilitating a quick hardware deployment in algorithm research, proof-of-concept studies and first hardware demonstrators, the resulting design efficiency may not be suitable for production. The FPGA based framework used in the

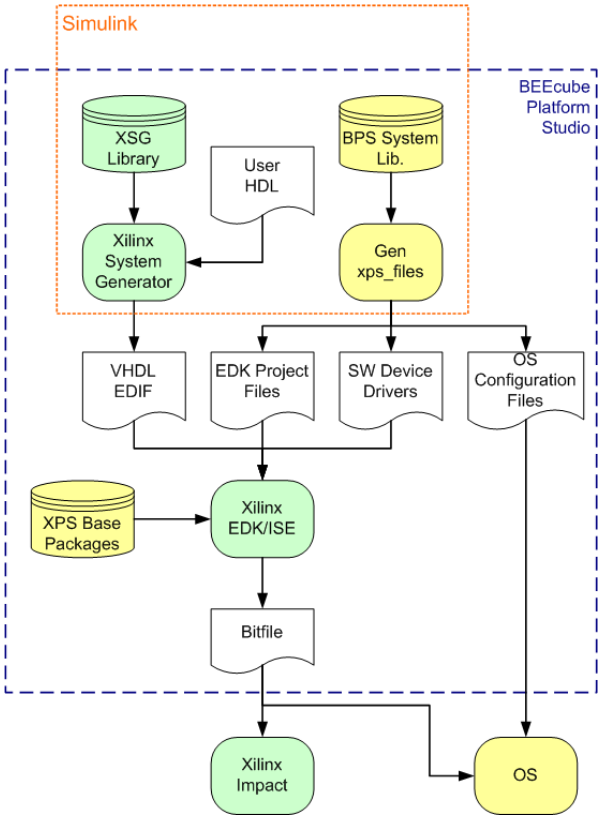


Fig. 1. BEEcube Platform Studio tool flow (source: [1])

design and verification process of an IEEE 802.11n MIMO baseband transceiver is presented in [2]. An experimental testbed for fixed antenna pattern interference alignment is presented in [3], but with a focus on algorithm exploration, while the focus of this work is on the design process of efficient hardware architectures.

An in-depth discussion of FPGA-based emulation systems in the design process of multiprocessor architectures for media applications can be found in [4]. The design framework used in the design of a soft-input soft-output MIMO sphere decoder is presented in [5].

## 3. WIRELESS COMMUNICATION SYSTEMS DESIGN SPACE EXPLORATION

The process of designing complex digital electronic circuits offers a large variety of options to the designer. There are many valid possible implementations that fulfill the specification, but they differ in certain properties, e.g. silicon area, power efficiency, flexibility, testability and design effort. These properties span the so-called design space. Typical properties include the silicon area, power efficiency, flexibility, testability and design effort. In general, these properties can not be chosen independent from each other. For example, flexibility and power efficiency are often contradictory requirements. At an early design stage, the designer has to choose important design parameters that will eventually determine the final implementation properties. Most of the parameters like the resulting power dissipation can not be directly chosen, but they are rather a result of many design choices: hardwired architectures tend to be more power efficient than programmable architectures. A design space exploration establishes relations between possible points in the design space, ultimately leading to cost functions modeling the relation between the design properties and parameters. These models serve as a quantitative basis to make important design decisions in an early design phase.

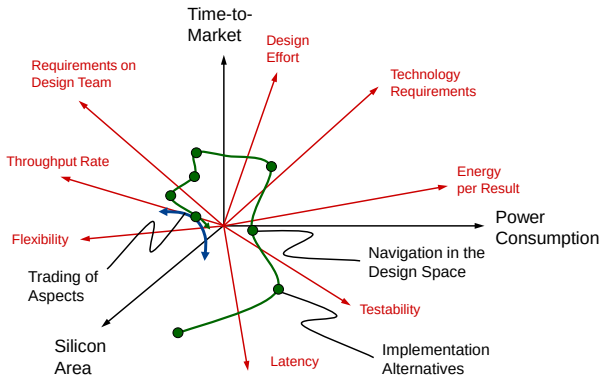


Fig. 2. Multi-dimensional design space (source: [6])

Certain parameters are of special interest in the domain of wireless communication platforms. The limited power budget in mobile devices puts hard constraints on the power efficiency, requiring power optimization across all layers of algorithm development, design implementation and semiconductor technology. This often conflicts with the demand for flexibility, which is another important requirement due to rapidly evolving communication standards. Flexibility and programmability is also required to keep pace with shorter product lifecycles to enable the re-use of the same hardware platform across multiple product generations. Also motivated by economical reasons, the total design time is of great importance.

The implementation cost of wireless communication al-

gorithms can be reduced by exploiting their numerical properties and finding appropriate approximations. These approximations trade accuracy for complexity and can often be assessed in terms of resulting error rate, throughput or related quantities. An approximation is often rated based on numerical simulations, as analytical bit error rate (BER) functions are not available across the whole communication chain.

Deriving comprehensive cost models using Monte-Carlo methods requires visiting a significantly larger number of points in the design space compared to existing heuristically driven parameter optimization approaches covered by existing FPGA-based simulation acceleration systems. The achievable simulation speedup is a key factor enabling the characterization and optimization of complex communication systems using Monte-Carlo approaches which are infeasible for pure software simulation due to the large required stimuli sets. Long simulation times in the order of days or even weeks heavily hinder the design process, as designing VLSI circuits is an iterative process and simulations typically have to be re-run several times during the design phase. Thus, increasing simulation speed by a few orders of magnitude can increase the designer's efficiency significantly.

## 4. DEVELOPMENT FRAMEWORK

The FPGA-based hybrid hardware-in-the-loop research and design space exploration framework created in this work combines high-level tools (e.g. MATLAB/Simulink) and optimized hardware blocks. Its application domain ranges from the design, optimization and verification of efficient and optimized signal processing blocks for computationally demanding next-generation wireless communication systems to system characterization and DSE.

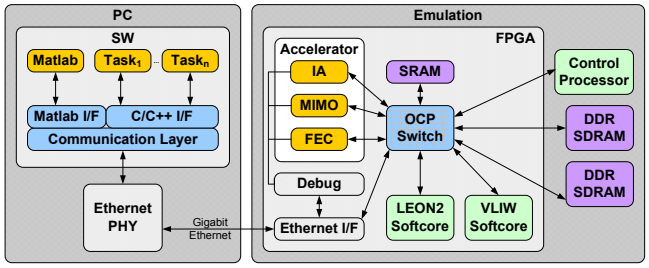


Fig. 3. Emulation framework block diagram

The framework consists of a host PC, FPGA-based emulation systems, a generic fully synthesizeable VHDL SoC infrastructure, dedicated processors, processor softcores and a software library providing a transparent communication application programming interface (API). This allows signal processing blocks to be split and run distributed on a highly heterogeneous signal processing system. Software API libraries provide unified transparent communication between MATLAB, C/C+, embedded software and the hardware on-chip multilayer bus system. The same resources are accessible from all components, enabling a flexible partitioning and migration of processing task between high-level software, embedded software and dedicated hardware modules. The framework block diagram is shown in Figure 3. The properties of the optimized on-chip infrastructure template make it suitable for usage in final ASIC targets and thus enable the test, debugging and characterization of signal processing blocks in their target environment. Using standard FPGA design flows, new computationally intensive processing cores are directly implemented as the optimized hardware target modules. Instrumentation is used to enable dynamic, software controlled parameter adjustment. The remaining blocks may continue to run as high-level models, enabling a divide-and-conquer implementation and verification approach. The framework provides transparent data transport between the substituted MATLAB modules and multiple parallel instances of their FPGA hardware counterparts. The same interfaces are available for hardware simulation via the ModelSim foreign language interface (FLI), effectively also providing a verification and debugging environment at minimal extra effort.

The PC is connected to the emulation systems via Gigabit Ethernet. The generic FPGA infrastructure template comprises an OCP multilayer bus, the Ethernet DMA interface, SDRAM controllers, on-chip memories and massively parallel parameterized softcore processors [7]. It has been adapted to and tested on a Xilinx Virtex-6 LX550T based BEE4 rapid prototyping system, the Xilinx Virtex-6 ML605 Evaluation Kit and the Virtex-5 LX220 based MCPA board [8] developed at IMS, see Fig. 4.

BEEcube's BEE4 prototyping platform incorporates four Xilinx Virtex-6 LX550T FPGAs and 16 GB DDR3-1066 SDRAM; it also features 20 Gbps QSFP+ interfaces, FMC expansion slots and PCI Express slots. The Xilinx ML605 Evaluation Kit uses a Virtex-6 LX240T FPGA and DDR3 memory. The third supported emulation system has been developed at IMS [8]. It comprises a Xilinx Virtex-5 LX220T FPGA and an Intel IXP 460 ARM XScale RISC processor, see Fig. 4. The IXP 460 runs a Linux 2.6 kernel with the standard software development toolchains available. It is fully integrated as a control and signal processing component and controls the programming of the FPGA.

## 5. CASE STUDY: INTERFERENCE ALIGNMENT

### 5.1. Interference Alignment System Model

Interference alignment is a transmission technique applicable to the K-user interference channel that can be used to increase the sum capacity of a multiuser communication system. At a high signal-to-noise ratio (SNR), the sum capacity scales linearly with the number of users. A prominent manifestation is the combination of IA and multiuser MIMO orthogonal frequency-division multiplexing (OFDM)

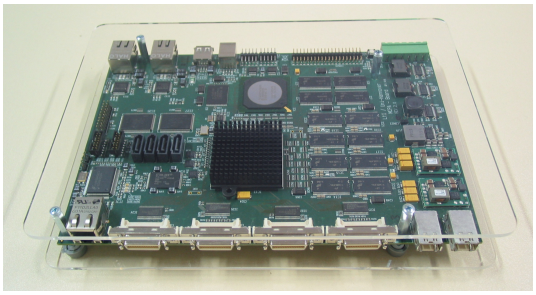

Fig. 4. FPGA-based emulation system developed at IMS

systems as shown in Fig. 5. Here, the transmitters apply linear precoding matrices $\boldsymbol{V}$ to their transmitted signals in order to align all interference at the receivers to a lower-dimensional subspace of the available receive space. The receivers extract the interference free subspace by multiplication with the decoding matrix $\boldsymbol{U}$. In general, the number of transmit antennas $N_t$ and receive antennas $N_r$ each has to be larger than the number of transmitted data streams $d$ for a solution to exist.

The received signal $\boldsymbol{y}_i$ at each receiver $i$ is

$$\boldsymbol{y}_i = \boldsymbol{H}_{ii}\boldsymbol{V}_i\boldsymbol{s}_i + \sum_{j\neq i}\boldsymbol{H}_{ij}\boldsymbol{V}_j\boldsymbol{s}_j + \boldsymbol{n}_i \tag{1}$$

with $\boldsymbol{s}_j$ being the transmit data at transmitter $j$ and $\boldsymbol{n}_i$ being the noise picked up by receiver $i$. $\boldsymbol{H}_{ii}$ are the desired channels, all $\boldsymbol{H}_{ij}$ convey interference. In the case of flat fading, IA can be applied to discrete-time signals.

For a sufficient number of antennas, perfect IA can be achieved for $\boldsymbol{V}$ and $\boldsymbol{U}$ satisfying:

$$\boldsymbol{U}_i^H\boldsymbol{H}_{ij}\boldsymbol{V}_j = 0 \quad \forall i \neq j \tag{2}$$

$$rank(\boldsymbol{U}_i^H\boldsymbol{H}_{ii}\boldsymbol{V}_i) \geq d \quad \forall i \tag{3}$$

where $d$ is the number of data streams per user.

### 5.2. 3-User Antenna Selection Interference Alignment

Closed-form solutions for the zero-forcing case in Eq. 2 are in general known for the $K = 3$ user case as [9]

$$\boldsymbol{E} = (\boldsymbol{H}_{31})^{-1}\boldsymbol{H}_{32}(\boldsymbol{H}_{12})^{-1}\boldsymbol{H}_{13}(\boldsymbol{H}_{23})^{-1}\boldsymbol{H}_{21} \tag{4}$$

$$\boldsymbol{V}_1 = \boldsymbol{v}(E) \tag{5}$$

$$\boldsymbol{V}_2 = (\boldsymbol{H}_{32})^{-1}\boldsymbol{H}_{31}\boldsymbol{V}_1 \tag{6}$$

$$\boldsymbol{V}_3 = (\boldsymbol{H}_{23})\boldsymbol{H}_{21}\boldsymbol{V}_1 \tag{7}$$

$$\boldsymbol{U}_1 = \text{null}\left((\boldsymbol{H}_{13}\boldsymbol{V}_3)^H\right) \tag{8}$$

$$\boldsymbol{U}_2 = \text{null}\left((\boldsymbol{H}_{21}\boldsymbol{V}_1)^H\right) \tag{9}$$

$$\boldsymbol{U}_3 = \text{null}\left((\boldsymbol{H}_{31}\boldsymbol{V}_1)^H\right) \tag{10}$$

where $\boldsymbol{v}(E)$ denotes an arbitrary eigenvector of $E$, null$(\cdot)$ is the nullspace. Note that the choice of the eigenvector has an impact on the overall system performance.

Our implementation supports antenna selection based interference alignment. Here, the transmitters are equipped with 3 antennas, of which only 2 are to be actively used for data transmission. The interference alignment algorithm picks the best antenna combination as explained below. This leads to an increased channel orthogonality for the chosen channels at a reduced number of RF front ends.

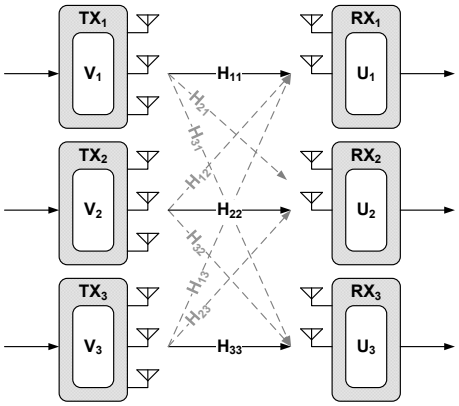


Fig. 5. Multi-user 2x3 MIMO system with transmitter-side antenna selection

The antenna selection can be done at either the transmitter, receiver or both ends. Basically this leads to a reduced number of required RF front ends and reduced power consumption compared to a system using all antennas. By leaving the worst modes unused, the resulting RF hardware complexity is reduced significantly at a moderate penalty in sum data rate.

The problem of finding the optimum antenna combination $\hat{i}$ from a set of $n$ combinations can be formulated as

$$\hat{i} = \underset{i=1\ldots n}{\arg\max} \sum_{k=1}^{K} C(\boldsymbol{V}_{k,i}, \boldsymbol{U}_{k,i}) \tag{11}$$

with $C(\boldsymbol{V},\boldsymbol{U})$ being a metric rating a set of matrices $\boldsymbol{U}$ and $\boldsymbol{V}$. $\boldsymbol{V}_{k,i}$ and $\boldsymbol{U}_{k,i}$ are the precoding and decoding matrices of user $k$ for a given antenna combination $i$. Equation 11 is solved by visiting all $n$ antenna combinations. The precoding and decoding matrices $\boldsymbol{V}$ and $\boldsymbol{U}$ are computed for every combination of antennas according to equations 4 to 10.

From an implementation point of view, the interference alignment algorithm constitutes the two-step process of

1) channel rate processing (CRP) and
2) symbol rate processing (SRP).

Based on a full set of given channel estimations, the CRP computes the best antenna combination and the corresponding precoding- and decoding matrices $\boldsymbol{V}$ and $\boldsymbol{U}$ for all devices. On the transmitter side, the SRP distributes each transmit symbol to both antennas by multiplication with $\boldsymbol{V}$. The two received antenna symbols are linearly combined into a single symbol by multiplication with $\boldsymbol{U}^H$ on the receiver side.

OFDM is chosen as the modulation scheme, resulting in separate $\boldsymbol{V}$ and $\boldsymbol{U}$ for each subcarrier.

### 5.3. Closed-Form IA Computational Complexity

The resource requirements of an optimized efficient integer implementation of the antenna-selection IA algorithm is presented in this section, based on FPGA implementation results. Target systems include SDR platforms, FPGAs and ASICs.

As the SRP is straight forward, this section focuses on the costs of the 3-user 2x2 MIMO processing consisting of matrix inversions, matrix multiplications, eigenvector computation and normalization, see Eq. 4 to 10. The metric $C$ is computed for both eigenvectors. All intermediate matrices can be independently scaled by arbitrary scalars without affecting the antenna decision or $\boldsymbol{V}$ and $\boldsymbol{U}$. Exploiting this makes the cost of all involved 2x2 matrix inversions negligible and allows intermediate matrices to be block-normalized by shifting, i.e. extract a common power of 2 from all matrix elements. This results in reduced integer word lengths and thus reduced hardware costs. Table I summarizes the number of required real-valued mathematical base operations for antenna selection and the computation of $\boldsymbol{V}$ and $\boldsymbol{U}$ per antenna combination and subcarrier, without a final normalization step of $\boldsymbol{V}$ and $\boldsymbol{U}$. Complex multiplications are composed of three real multiplications, three additions and two subtractions, `INVSQRT` denotes the reciprocal square root [10].

TABLE I
OPERATION COUNTS FOR THE COMPUTATION OF $C$ PER ANTENNA COMBINATION $i$ AND SUBCARRIER

| OP | ADD | MUL | SQRT | INVSQRT |
|---|---|---|---|---|
| Matrix mult. | 696 | 348 | 0 | 0 |
| Eigenvectors | 15 | 8 | 3 | 0 |
| Metric score | 46 | 82 | 6 | 2 |
| #OPC | 757 | 438 | 9 | 2 |

To keep the total transmit power constant, the chosen antenna combination's precoding matrices $\boldsymbol{V}$ need to be normalized, resulting in 3 `ADD`, 8 `MUL` and 1 `INVSQRT` additional operations #OPN per transmitter and subcarrier. The above analysis implies that in general, the implementation cost is dominated by the multiplications in terms of silicon area and power consumption. As an example, a typical 128 subcarrier OFDM system with 2 antennas chosen out of 3 antennas per transmitter and 1 ms IA processing latency requires approx. 861 MMUL/s for realtime operation.

For the $K = 3$ users case with $N_{t,a} = 2$ active transmit antennas used out of $N_{t,p}$ physical antennas per transmitter and $N_{r,a} = 2$ receive antennas used out of $N_{r,p}$ antennas per receiver, the number of combinations to be visited per subcarrier is

$$n = \left( \binom{N_{t,p}}{N_{t,a}} \cdot \binom{N_{r,p}}{N_{r,a}} \right)^K = 27 \tag{12}$$

For realtime operation, the maximum allowable latency is defined to be $T_0$. Assigning relative operation costs $\alpha_i$ to each operation type $\mathrm{OP}_i$, the total computational cost for $M$ subcarriers becomes

$$C = \frac{M}{T_0} \cdot \left( n \cdot \sum_{i \in \mathrm{OP}} \alpha_i \cdot \#\mathrm{OPC_i} + K \cdot \sum_{i \in \mathrm{OP}} \alpha_i \cdot \#\mathrm{OPN_i} \right) \tag{13}$$

### 5.4. Hardware cost estimation

Using $\alpha$ as relative silicon area costs, the total silicon area implementation cost of an architecture without resource sharing can be estimated from Eq. 12 and 13. The relative area $\alpha$ of 16-bit arithmetic operations for an ASIC implementation based on [11] results in the values given in Table II. The relative costs $\alpha_{\mathrm{MUL}}$ of a multiplier are defined to be 1. For a system using antenna selection at the transmitter only

TABLE II
RELATIVE SILICON AREA COSTS OF 16-BIT ARITHMETIC OPERATIONS

| OP | ADD | MUL | SQRT | INVSQRT |
|---|---|---|---|---|
| $\alpha$ | 0.108 | 1 | 1.73 | 3 |

with $N_{t,p} = 3$ antennas, $M = 128$ subcarriers and $T_0 = 1\,\mathrm{ms}$, the total IA costs are estimated to be $C = 1.875\,\mathrm{GOPS}$.

For the configuration above, the original MATLAB algorithm takes 3.63 s on an Intel Xeon 2.4 GHz CPU running MATLAB R2012a for the computation of the optimal antenna combination $\hat{i}$ and its corresponding precoding and decoding matrices $\boldsymbol{V}_k$ and $\boldsymbol{U}_k$ from a set of channel information $H$. The FPGA implementation created in this case study achieves realtime operation, requiring $380\,\mu\mathrm{s}$ at 100 MHz clock frequency on a Xilinx Virtex-6 LX550T FPGA in a BEE4 emulation system. Thus, the achieved speedup is 9553.

## 6. COST ESTIMATION FOR ITERATIVE K-USER IA

For the general $K > 3$ user case, there are no known closed-form solutions, but iterative algorithms exist. In this section, we present implementation complexity estimates of the minimum mean squared error interference alignment algorithm in [12].

The MMSE-IA algorithm starts with arbitrary precoding matrices $\boldsymbol{V}_k$, then iteratively updates the decoding and precoding matrices $\boldsymbol{U}_k$ and $\boldsymbol{V}_k$ according to Eq. 14 and 15 until convergence. The Lagrange multiplier $\lambda_k \geq 0$ is computed to satisfy $\|\boldsymbol{V}_k\|_2^2 \leq 1$ by Newton iteration.

$$\boldsymbol{U}_k = \left( \sum_{j=1}^{K} \boldsymbol{H}_{kj} \boldsymbol{V}_j \boldsymbol{V}_j^H \boldsymbol{H}_{kj}^H + \sigma^2 \boldsymbol{I} \right)^{-1} \boldsymbol{H}_{kk} \boldsymbol{V}_k \tag{14}$$

$$\boldsymbol{V}_k = \left( \sum_{j=1}^{K} \boldsymbol{H}_{jk}^H \boldsymbol{U}_j \boldsymbol{U}_j^H \boldsymbol{H}_{jk} + \lambda_k \boldsymbol{I} \right)^{-1} \boldsymbol{H}_{kk}^H \boldsymbol{U}_k \tag{15}$$

The number of required iterations is data-dependent. Each iteration step requires the following operations to be executed: matrix multiplication, pseudo-inverse, Newton iterations. Figure 6 summarizes the estimated number of

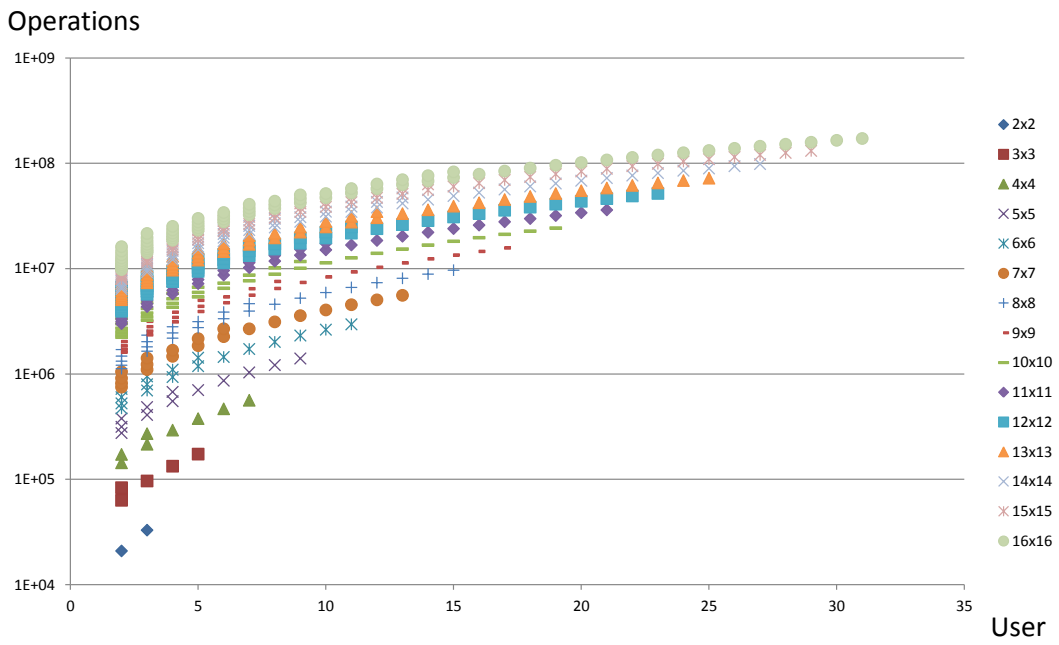


Fig. 6. Number of operations for iterative MMSE interference alignment (4 iterations)

operations for the computation of a set of $\boldsymbol{V}$ and $\boldsymbol{U}$ matrices, based on well-known optimized hardware implementations. Comparing the iterative approach computational complexity to the the closed-form 2x2 IA implementation presented in Section 5.2, the number of operations is increased by a factor of approximately 60.8. The high computational complexity clearly identifies iterative IA algorithms as candidates for dedicated hardware implementation using hardware accelerated approaches.

## 7. CONCLUSION

Using a unified infrastructure framework closely resembling a realistic target architecture for a joint implementation/verification and design space exploration helps to increase the design quality and accelerate the design process. Slim and clean interface definitions within the framework are the key to portability and support of a range of emulation platforms with varying complexity. Combined with the simplified design partitioning approach, parallel development and verification of distinct system blocks can be carried out on a number of basic platforms, resulting in a quick final system integration on a large emulator.

The presented implementation costs of a dedicated hardware architecture for a simple closed-form interference alignment algorithm and cost the estimates for iterative form IA imply a demand for further research into efficient IA hardware architectures.

## REFERENCES

[1] BEEcube Inc., *BEEcube Platform Studio Version 4.0 User Manual*, 2011.

[2] P. Greisen, S. Haene, and A. Burg, "Simulation and emulation of mimo wireless baseband transceivers," *EURASIP Journal on Wireless Communications and Networking*, vol. 2010, no. 1, 2010.

[3] O. González, D. Ramírez, I. Santamaría, J. García-Naya, and L. Castedo, "Experimental validation of interference alignment techniques using a multiuser MIMO testbed," in *Smart Antennas (WSA), 2011 International ITG Workshop on*, feb. 2011, pp. 1 –8.

[4] J. Martín-Langerwerf, *Emulationsbasierte Analyse von Multiprozessorsystemen für Multimedia-Anwendungen*. Shaker Verlag, 2011.

[5] F. Borlenghi, D. Auras, E. M. Witte, T. Kempf, G. Ascheid, R. Leupers, and H. Meyr, "An FPGA-accelerated testbed for hardware component development in mimo wireless communication systems," in *SAMOS XII: International Conference on Embedded Computer Systems: Architectures, Modeling and Simulation*, jul 2012.

[6] T. G. Noll, "Application domain specific embedded FPGAs for SoC platforms," in *Irish Signals and Systems Conference 2004 (ISSC'04)*, Belfast, Ireland, June-July 2004.

[7] G. Paya-Vaya and H. Blume, "TUKUTURI: A dynamically reconfigurable multimedia soft-processor for video processing applications," in *Poster presentation at the 7th International Conference on High-Performance and Embedded Architectures and Compilers, HiPEAC'12, Paris, France*, vol. USB Proceedings, 2012.

[8] C. Banz, S. Hesselbarth, H. Flatt, H. Blume, and P. Pirsch, "Real-Time Stereo Vision System using Semi-Global Matching Disparity Estimation: Architecture and FPGA-Implementation," *Transactions on High-Performance Embedded Architectures and Compilers, Springer*, 2012.

[9] V. Cadambe and S. Jafar, "Interference alignment and degrees of freedom of the K-user interference channel," *Information Theory, IEEE Transactions on*, vol. 54, no. 8, pp. 3425 –3441, aug. 2008.

[10] P. Salmela, A. Burian, T. Järvinen, A. Happonen, and J. H. Takala, "Low-complexity inverse square root approximation for baseband matrix operations," *ISRN Signal Processing*, vol. vol. 2011, 2011.

[11] P. Pirsch, *Architectures for Digital Signal Processing*. John Wiley & Sons, Inc., 1998.

[12] D. Schmidt, C. Shi, R. Berry, M. Honig, and W. Utschick, "Minimum mean squared error interference alignment," in *Signals, Systems and Computers, 2009 Conference Record of the Forty-Third Asilomar Conference on*, nov. 2009, pp. 1106 –1110.