Programmable ASIPs for Multimode MIMO Transceiver

Shahriar Shahabuddin, Olli Silvén, and Markku Juntti

***Abstract*—Application specific instruction-set processors (ASIP) are a programmable and flexible alternative of traditional finite state machine (FSM) controlled register-transfer level (RTL) designs for multimode basedband systems. In this paper, we present two ASIPs for small scale multiple-input multiple-output (MIMO) wireless communication systems that demonstrate the soundness and effectiveness of ASIPs for this type of applications. The first ASIP is programmed with multiple MIMO symbol detection algorithms for 4×4 systems. The supported detection algorithms are minimum mean-square error (MMSE), two variants of the selective spanning with fast enumeration (SSFE) and K-best list sphere detection (LSD). The second ASIP supports MMSE and zero-forcing dirty paper coding (ZF-DPC) algorithms for a base station (BS) with 4 antennas and for 4 users. Both ASIPs are based on transport triggered architecture (TTA) and are programmed with a retargetable compiler with high level language to meet the time-to-market requirements. The detection and precoding algorithms can be switched in the respective ASIPs based on the error-rate requirements. Depending on the algorithms, MIMO detection ASIP delivers 6.16 - 66.66 Mbps throughput at a clock frequency of 200 MHz on 90 nm technology. The precoder ASIP provides a throughput of 52.17 and 51.95 Mbps for MMSE and ZF-DPC precoding respectively at a clock frequency of 210 MHz on 90 nm technology.**

***Index Terms*—MIMO, OFDM, ASIP, TTA, VLSI.**

I. Introduction

The demand for multimode physical layer algorithm implementations has been growing recently for cognitive and adaptive wireless systems. The traditional finite state machine (FSM) controlled register-transfer level (RTL) or fixed hardware designs provide high data rate, use less logic gates and consume less power. The drawback of the fixed hardware implementation is that it operates on a fixed set of parameters and it is very difficult to modify the design in the future. Therefore, the non-programmable designs may not be the best choice for a multimode wireless systems where flexibility is a key requirement. It is possible to have different hardware accelerators to support different modes and parameters, but the whole system can become too large to accommodate all the accelerators. Moreover, the whole design and verification process can be very costly due to longer design time. Another solution is to implement the algorithms as software to program the digital signal processor (DSP) or microcontroller units (MCU). At first sight, the software implementation is ideal for flexible wireless systems, where the parameters and the algorithms are changed by software. However, it is very difficult to achieve the required rate for high speed applications with DSPs, as their architectures not tailored for any particular algorithm.

Application specific instruction-set processors (ASIP) that are customized for a small set of algorithms can be an ideal choice for multimode wireless systems. The software-hardware codesign method, which is typically used for an ASIP design, can be used to exploit the flexibility of software or high speed of the hardware whenever necessary. An ASIP customized for a small set of algorithms is an attractive solution in terms of cost, silicon area and high throughput. Most importantly, an ASIP reduces the design risk where the instruction memory can be loaded with new programs or control instructions. The control instructions can be easily obtained by a retargetable compiler for that particular customized architecture. It should be noted that the motivation for designing an ASIP is not to outperform pure monolithic hardware accelerators in terms of throughput. The motivation behind designing an ASIP is to demonstrate the price of programmability and flexibility does not exceed their benefits. In other words, a reasonable throughput/area efficiency can be obtained without compromising programmability.

In this paper, we present two ASIPs that supports two key functionalities of the baseband signal processing chain, namely, multiple-input multiple-output (MIMO) symbol detection and precoding. The processors are based on the transport triggered architecture (TTA). TTA is an exposed datapath processor design philosophy where the programs directly control the internal transport buses for a processor and computation happens as a side effect of the data transfer [3], [4]. The TTA ASIP typically consists of multiple data buses and function units (FU) that provides the opportunity of instruction level parallelism (ILP). The TTA ASIPs are thus particularly suited for high speed digital signal processing. The TTA based codesign environment (TCE) tool enables the designer to write an application with a high level language and design the target processor in a graphical user interface at the same time [5]. The ASIPs presented in this paper are designed with the help of TCE.

The first ASIP supports at least three MIMO symbol detection algorithms for a baseband receiver. A plethora of application specific integrated circuit (ASIC) and ASIP implementations supporting a single MIMO detection algorithm can be found in the literature [6]–[10]. A few digital very

S. Shahabuddin and M. Juntti is with Centre for Wireless Communications, University of Oulu, Oulu - 90570, Finland, e-mail: firstname.lastname@oulu.fi

O. Silvén is with Department of Computer Science and Engineering, University of Oulu, Oulu - 90570, Finland, e-mail: firstname.lastname@oulu.fi

The research is supported by Academy of Finland and 5G Communication with a Heterogeneous, Agile Mobile network in the Pyeongchang wInter Olympic competitioN (5G CHAMPION) project.

The detector implemented in this paper builds upon the ASIP presented at International Conference on Cognitive Radio Oriented Wireless Networks [1]. The precoder implemented in this paper builds upon the ASIP to be presented at European Conference on Networks and Communications [2].

large scale integration (VLSI) implementations for multimode MIMO detection can also be found in [11]–[14] We present a MIMO detector ASIP that supports minimum mean-square error (MMSE), 8-best list sphere detection (LSD) and two variants of selective spanning for fast enumeration (SSFE) detection algorithms. The ASIP delivers 6.16 - 66.66 Mbps throughput at a clock frequency of 200 MHz on 90 nm technology.

Our second ASIP supports multiuser MIMO (MU-MIMO) broadcast precoding for a base station (BS) transmitter. The dual-mode ASIP is programmed with MMSE and zero- forcing dirty paper coding (ZF-DPC) algorithms. The ASIP also supports norm-based scheduling and QR decomposition. A norm-based scheduler is used in this work that selects 4 user indices out of a total 20 users. The precoder ASIP provides a throughput of 52.17 and 51.95 Mbps for MMSE and ZF-DPC precoding respectively at a clock frequency of 210 MHz on 90 nm technology.

A. Outline

The paper is organized in the following way: The system models for detection and precoding are presented in Section II. The detector and precoder algorithms are presented in Section III and III-B respectively. The error-rate performance is presented in Section IV. The processor design methodology and the ASIP architectures are presented in Section V. In Section VI, the performance of the processors is discussed and the conclusion is drawn in Section VII.

B. Notation

Boldface lowercase and boldface uppercase letters stand for column vectors and matrices, respectively. For a matrix, $\mathbf{A}$, we denote its Hermitian transpose by $\mathbf{A}^H$. We use $A_{k,l}$ for the entry in the kth row and lth column of the matrix $\mathbf{A}$. The real and imaginary part of a complex-valued matrix $\mathbf{A}$ are denoted by $\Re(\mathbf{A})$ and $\Im(\mathbf{A})$, respectively. The identity matrix is $\mathbf{I}$ and ℓ_2-norm of the vector $\mathbf{a}$ is $\|\mathbf{a}\|_2 = \sqrt{\sum_K |a_k|^2}$.

II. System models

A. System model for Data Detection

We consider a MIMO system that employs orthogonal frequency-division multiplexing (OFDM) with M_d transmit antennas, which are sending data over the channel, and N_d receive antennas such that $N_d \geq M_d$. A layered space-time architecture is applied at the transmitter and two streams of data bits are encoded horizontally according to the 3GPP Long Term Evolution (LTE) standard [15].

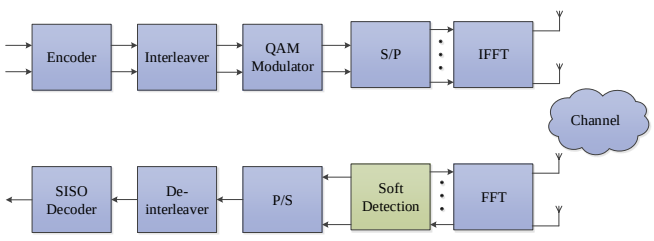

Fig. 1: System model for MIMO detection.

Each stream of the encoded data bits is interleaved, mapped to constellation points and multiplexed onto two different layers. The symbols are transferred to the time domain with an IFFT and sent over the channels. In the receiver end, the received symbols are first transferred to the frequency domain by a FFT. We assume perfect channel state information (CSI) and synchronization, as well as a sufficiently long cyclic prefix that can eliminate the inter-symbol interference. A soft output MIMO symbol detector is used that provides the log-likelihood ratio (LLR) for the decoder. A block diagram of the system model is presented in Fig. 1. By omitting the subcarrier index, the standard input-output relation per subcarrier can be written in the real domain as

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

where $\mathbf{y} \in \mathbb{R}^{2N_d}$ is the received signal vector, $\mathbf{x} \in \mathbb{R}^{2M_d}$ is the transmit symbol vector, $\mathbf{H} \in \mathbb{R}^{2N_d \times 2M_d}$ is the channel matrix, and $\mathbf{n} \in \mathbb{R}^{2N_d}$ is the circularly symmetric complex white Gaussian noise vector with zero mean and σ_d^2 variance.

B. System model for Precoding

We consider a single cell downlink channel with a M_p antenna BS serving a total N_p single antenna users. The set $\mathcal{U}$ consists of the integer indices of all users in the system. At any given instant, the BS transmits data for a subset $\mathcal{A} \subset \mathcal{U}$ where $|\mathcal{A}| = M_p$. $\mathcal{A}$ is the active user set that consists of the indices of the multiplexed users at a given scheduling instant. $\mathcal{A}$ is selected by a norm-based scheduler where the norm of all user channels are calculated and M_p user indices with the highest norm are selected [16].

The BS transmits to M_p different active users through M_p antennas at any time instant. However, the transmitted signals for different users interfere with each other and thus corrupt the signal designated to any particular user. Thus, the received signal for user k can be expressed as

$$r_k = \mathbf{b}_k^H \mathbf{d}_k + \sum_{j \neq k} \mathbf{b}_k^H \mathbf{d}_j + z_k, \tag{2}$$

where $\mathbf{b}_k \in \mathbb{C}^{M_p \times 1}$ is the channel vector between the BS and user k, $\mathbf{d}_k \in \mathbb{C}^{M_p \times 1}$ is the transmitted signal for user k and z_k is zero mean Gaussian noise.

The transmitted vector for user k is obtained by multiplying the precoding vector $\mathbf{w}_k$ and symbol u_k as

$$\mathbf{d}_k = \mathbf{w}_k u_k. \tag{3}$$

The precoding vector $\mathbf{w}_k$ is applied to avoid the interference caused by other transmitted signals.

We stack the channel vectors to form a channel matrix $\mathbf{B} \in \mathbb{C}^{M_p \times M_p}$ and precoding vectors to form the precoding matrix $\mathbf{W} \in \mathbb{C}^{M_p \times M_p}$ and thus the input-output relation can be written as

$$\mathbf{r} = \mathbf{B}\mathbf{W}\mathbf{u} + \mathbf{w}, \tag{4}$$

where $\mathbf{u}$ is a vector of the original symbols , $\mathbf{w}$ is the noise vector and $\mathbf{r}$ is the received signal vector.

Typically, precoders are designed with respect to a total power constraint of the form

$$E\|\mathbf{d}\|^2 = \text{Tr}\{\mathbf{W}\mathbf{W}^H\} \leq P, \tag{5}$$

where total power, $P > 0$. Total power constraint simplifies the design problem and leads to simple precoders.

III. Detection and Precoding Schemes

A. Detection Schemes

The function of the MIMO detector is to estimate the transmitted signal vector $\mathbf{x}$ and to feed the soft output to the decoder. We apply QR decomposition on the augmented channel matrix $\underline{\mathbf{H}}$ as

$$\underline{\mathbf{H}} = \begin{bmatrix} \mathbf{H} \\ \sigma_d \mathbf{I}_{2M_d} \end{bmatrix} = \underline{\mathbf{Q}_d \mathbf{R}_d} = \begin{bmatrix} \mathbf{Q}_{ad}\underline{\mathbf{R}_d} \\ \mathbf{Q}_{bd}\underline{\mathbf{R}_d} \end{bmatrix} \tag{6}$$

where $\underline{\mathbf{Q}_d} = [\mathbf{Q}_{ad}^T \ \mathbf{Q}_{bd}^T]^T$ is an orthogonal matrix and $\underline{\mathbf{R}_d}$ denotes an upper triangular matrix. The dimensions of matrices $\mathbf{Q}_{ad}$, $\mathbf{Q}_{bd}$ and $\underline{\mathbf{R}_d}$ are $2N_d \times 2M_d$, $2M_d \times 2M_d$ and $2M_d \times 2M_d$ respectively. The input-output relationship can be transformed into $\tilde{\mathbf{y}} = \underline{\mathbf{R}_d}\mathbf{x} + \tilde{\mathbf{n}}$ where $\tilde{\mathbf{y}} = \mathbf{Q}_{ad}^T\mathbf{y}$ and $\tilde{\mathbf{n}}$ contains noise $\mathbf{Q}_{ad}^T\mathbf{n}$ and additional self inteference. A modified MMSE has been discussed in [17] and [18] where the QR decomposition is used on augmented channel matrix to compute the MMSE equalization based detection. The MMSE detector is obtained from,

$$\tilde{\mathbf{x}}_{\text{MMSE}} = (1/\sigma)\mathbf{Q}_{bd}\mathbf{Q}_{ad}^T\mathbf{y} = (1/\sigma_d)\mathbf{Q}_{bd}\tilde{\mathbf{y}}. \tag{7}$$

The signal-to-interference-plus-noise ratio (SINR) vector $\boldsymbol{\rho}$ can be computed as $\rho_i = 1/(\mathbf{q}_i\mathbf{q}_i^T) - 1$ where $\mathbf{q}_i$ is the i-th column of $\mathbf{Q}_{bd}$. The max-log approximated LLR can be computed from the SINR and $\hat{\mathbf{x}}$ following [19].

The K-best list-sphere detection (LSD) algorithm is a breadth-first suboptimal tree search MIMO symbol detection algorithm. The algorithm keeps a total K number of child nodes with the smallest accumulated Euclidean distances at each level. When going from $i+1$ to i, the K nodes at level $i+1$ expands to a total KC child nodes at level i. The child nodes are sorted according to their accumulated Euclidean distance and K child nodes are kept at level i. The rest of the nodes are deleted before spanning for next level. The K smallest accumulated Euclidean distances and corresponding symbol vectors are used as the final candidate list to compute the LLR.

SSFE is a breadth-first suboptimal tree-search detection algorithm that has a regular and deterministic dataflow [20]. SSFE can be characterized with a spanning vector $\mathbf{m} = [m_1, m_2, ..., m_{2N_d}]$. The spanning vector indicates the number of child nodes that span from the parent node on each level. Contrary to the K-best algorithms, the nodes are never deleted. Therefore, the sorting and deletion is not present in the SSFE algorithm, which reduces algorithm complexity.

B. Precoding Schemes

Transmitter precoding is an alternative of receiver optimization to combat multiple access interference in a synchronous multiuser channel. Transmitter precoding is a linear transformation of the transmitted signals that minimizes the mean squared errors at all the receivers. We use a regularization of the pseudo-inverse of the multiuser channel to compute the MMSE precoding matrix as

$$\mathbf{W}_{\text{MMSSE}} = \mathbf{B}^H(\mathbf{B}\mathbf{B}^H + \alpha^2\mathbf{I})^{-1}. \tag{8}$$

where α^2 is the regularization factor. A non-zero regularization factor can be used to allow a measured amount of multi-user interference. An extended channel matrix can be formed to solve the MMSE problem in the following way

$$\underline{\mathbf{B}} = \begin{bmatrix} \mathbf{B} & \alpha\mathbf{I}_{N_p} \end{bmatrix} \Leftrightarrow \underline{\mathbf{B}}^H = \begin{bmatrix} \mathbf{B}^H \\ \alpha\mathbf{I}_{N_p} \end{bmatrix}. \tag{9}$$

The right pseudo-inverse of the extended channel matrix takes the following form where the upper half of the equation is the same as MMSE precoder expression of (8).

$$\underline{\mathbf{W}} = \begin{bmatrix} \mathbf{W}_M \\ \alpha(\mathbf{B}\mathbf{B}^H + \alpha^2\mathbf{I})^{-1} \end{bmatrix}. \tag{10}$$

We apply QR decomposition on the Hermitian transpose of extended channel matrix $\underline{\mathbf{H}}$ as

$$\underline{\mathbf{B}}^H = \begin{bmatrix} \mathbf{B}^H \\ \alpha\mathbf{I}_{N_p} \end{bmatrix} = \underline{\mathbf{Q}_p\mathbf{R}_p} == \begin{bmatrix} \mathbf{Q}_{ap}\underline{\mathbf{R}_p} \\ \mathbf{Q}_{bp}\underline{\mathbf{R}_p} \end{bmatrix}. \tag{11}$$

From (11) and $\underline{\mathbf{R}_p}^{-1} = \frac{1}{\alpha}\mathbf{Q}_{bp}$ we get

$$\mathbf{W}_{\text{MMSE}} = \frac{1}{\alpha}\mathbf{Q}_{ap}\mathbf{Q}_{bp}^H. \tag{12}$$

A similar approach is taken in [21] to simplify the MMSE precoding applying QR on extended channel matrix. The regularization factor is traditionally calculated as

$$\alpha^2 = \frac{M\sigma_p^2}{P}, \tag{13}$$

where σ_p^2 is the noise variance and P is the power constraint.

Dirty paper coding (DPC) is a highly nonlinear technique and its implementation is a very challenging problem [22]. Zero-forcing dirty paper coding (ZF-DPC) is a reduced complexity suboptimal DPC scheme that was first proposed in [23]. The channel matrix is decomposed to a lower triangular matrix $\mathbf{L} \in \mathbb{C}^{M\times M}$ and a unitary matrix $\mathbf{Q} \in \mathbb{C}^{M\times M}$ to apply the ZF-DPC. It converts the symbol vector such a way that multiplying the symbol vector with $\mathbf{L}$ creates a diagonal matrix [24]. Afterwards, the modified symbol vector is multiplied by Hermitian transpose of the unitary matrix, $\mathbf{Q}^H$ and transmitted over the channel. A new symbol vector $\tilde{\mathbf{u}}$ to convert the non-diagonals of $\mathbf{L}$ to zero can be obtained as

$$\tilde{u}_i = u_i - \sum_{j=1}^{j=i-1} \frac{l_{ji}}{l_{ii}} u_j, \tag{14}$$

where $\mathbf{u}$ is the original symbol vector. ZF-DPC pre-cancels the interference without any loss of information.

IV. Error-rate Performance

A. MIMO Detection

We compare the performance of MMSE, 8-best and 16-best LSD and three variants of SSFE namely [11111111], [11111222] and [111112223] in a 3G LTE based MIMO-OFDM Matlab simulator. We assume 4×4 MIMO systems where 16-QAM and 64-QAM are applied. A 5 MHz bandwidth corresponding to 512 OFDM subcarriers is considered. One frame is equal to one OFDM symbol in the simulator. Thus, one frame consists of 512 subcarriers where 300 subcarriers are loaded with data and the rest are used as a guard interval. In the simulation, the mobile velocity is set at 3 kmph and

TABLE I: Simulation and Channel Model Parameters

Number of subcarriers	512 (300) active
Channel coding	Turbo Coding
Coding Rate	1/2
Symbol duration	71.39 μ s
Symbol time	66.7 μ s
Cyclic prefix duration	4.69 μ s
Modulation	16 QAM and 64 QAM
user velocity	3 kmph
Channel model	TU
Number of paths	6
path delays	[0 ... 2510] ns
path power	[0 ... -20] dB
BS azimuth spread	2° / 5°
MS azimuth spread	35°

the turbo decoder performs 6 iterations. We did not use any internal quantization for the turbo decoder. A 6-tap typical urban (TU) Vehicular A channel is assumed. The channel with base station (BS) azimuth spread of 5° is considered as moderately correlated channel, and with 2° as highly correlated channel. A list of the parameters for the simulations is presented in Table I. The bit error-rate (BER) performance of the detectors are shown in Figs. 2-4.

The detectors are simulated in an uncorrelated channel with 16-QAM and 64-QAM in Fig. 2. It can be observed that the MMSE and SSFE with a spanning vector of [11111222] exhibit performance similar to each other. Therefore, the detector with the lower complexity, i.e. MMSE, can be used instead of SSFE with [11111222] in this scenario. The SSFE with [11111111] can be used if the BER requirement is relaxed.

In Fig. 3, the detectors are simulated for a moderately correlated channel for 16-QAM and 64-QAM. For 64-QAM, LMMSE and SSFE with [11111111] requires very high SNR and thus not suitable for this scenario. The SSFE with [11111222] provides noticable performance gain over MMSE. In Fig. 4, the detectors are simulated for a highly correlated channel for 16-QAM. All the detectors presented here requires very high SNR for 64-QAM and not presented in Fig. 4. The LMMSE and SSFE with [11111111] exhibit very high SNR requirements even for 16-QAM.

The error-rate results are useful to select the detection algorithm for a specific channel condition and modulation order. We notice that SSFE with [11112223], 8-best and 16-best LSD provides similar performance in all the simulations. Therefore, the 8-best LSD with the lowest list size out of these

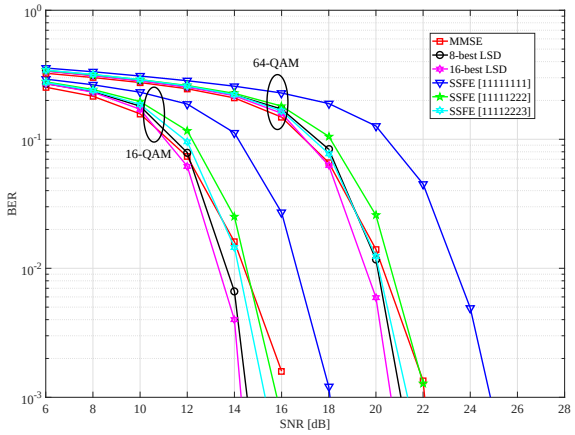

Fig. 2: Error-rate performance of the detectors in an uncorrelated channel.

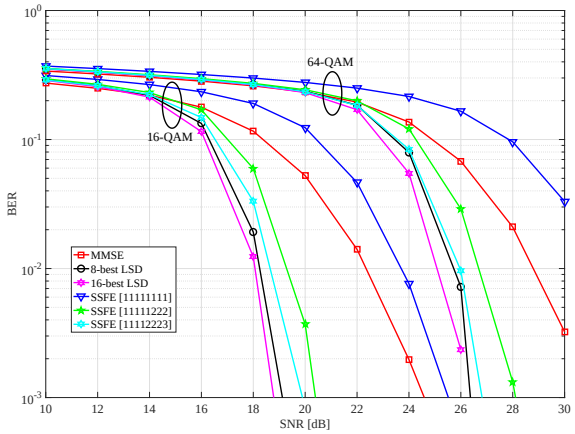

Fig. 3: Error-rate performance of the detectors in a moderately correlated channel.

three high performance detectors is used in this work. SSFE with [11111111] and [11111222] and MMSE are also selected as they operate on different SNR ranges. More simulation results can be found in [25].

B. Precoding

We present the BER performance of ZF, MMSE and ZF-DPC precoders for various SNR in Fig. 5. An additive white Gaussian noise (AWGN) channel is used and the BER is averaged over 100 000 Monte-Carlo trials. A norm-based scheduler is used to select four users out of $N = 20$ users. The modulation schemes are 16-QAM and 64-QAM. It can be seen from Fig. 5 that ZF-DPC performs better than MMSE for higher order modulations. ZF-DPC provides almost 3 dB gain over MMSE for 64-QAM in the high SNR region.

V. Application Specific Processors

A. Transport Triggered Architecture

TTA is a processor design philosophy where the program directly controls the internal data transport between different

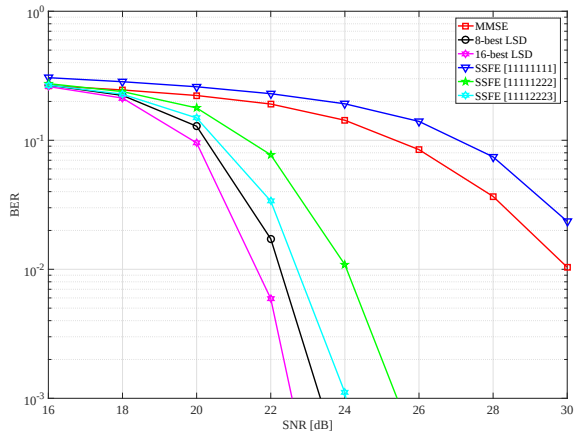

Fig. 4: Error-rate performance of the detectors in a highly correlated channel.

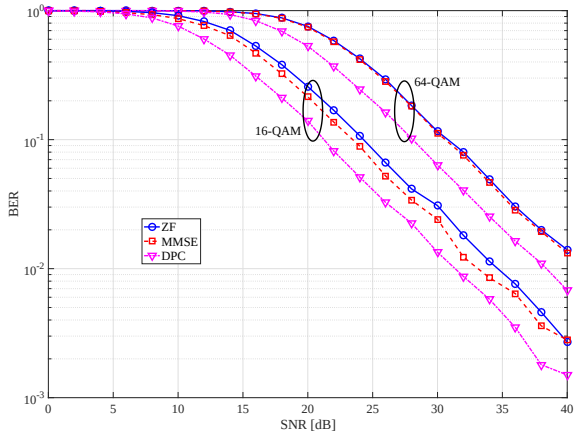

Fig. 5: Error-rate performance of the precoders in an uncorrelated channel.

function units (FU) of a processor. A typical processor consists of FUs to perform computation, register files (RF) to store the data and control unit for program flow control. Buses, ports and sockets make the interconnection network of the processor. The conventional reduced instruction set computer (RISC) or complex instruction set computer (CISC) exectues instructions sequentially and thus, they are not suitable for high speed digital signal processing. The advantage of ILP can be utilized with very long instruction word (VLIW) architecture that can execute several instructions in a single cycle. The TTA can be viewed as exposed datapath VLIW architecture that eliminates some of the bottlenecks of VLIW. The visibility of the interconnection network of TTA allows the designer to accurately control the operations. The TTA utilizes the concept of software bypassing, where operands can bypass the register files and move directly to the destination FU. The register pressure is low in TTA due to frequent data transport between the FUs. Due to the exposed datapath, the designer can reduce the load on the buses of the interconnection network and raise the maximum clock freuqency of the architecture.

In a TTA processor, one of the input ports of a FU is marked as a triggering port. A FU starts the computation when data is written on the triggering port. The TTA processor works with a single *move* instruction and does not require a complex instruction decoding unit. Most of the instructions scheduling decisions are taken at compile time which further reduces the processing logic of the TTA processor [3], [4].

A simple TTA processor is shown in Fig. 6. The processor includes three buses that are represented by three black horizantal lines. The vertical rectangular blocks going through the buses represent the sockets. The arrow above the socket shows whether a socket is an input or output. The processor of Fig. 6 consists of several FUs such as load-store unit (LSU), an adder (ADD), a multiplier (MUL) and a RF.

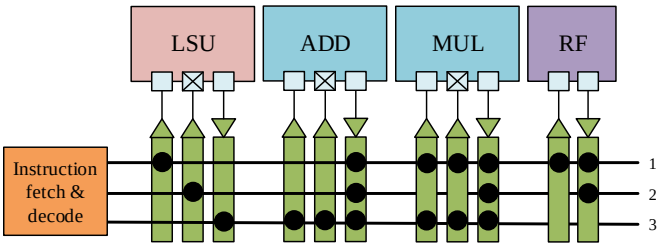

Fig. 6: Part of a TTA processor.

The smaller square with the cross inside the FUs indicates the triggering port. The connections between the FUs and the buses are illustrated by black dots in the sockets. If all the buses and FUs are connected, then the compiler has complete freedom in optimizing data moves. However, a fully connected processor may lead to high fan-out and low maximum clock frequency in synthesis.

B. Design Methodology

TCE provides an open source toolset for designing, implementing and simulating a TTA processor. The toolset includes a graphical processor design tool (ProDe) with an extensive library of function units [5]. TCE also includes a retargetable compiler called *tcecc* to compile high level language (C/C++, OpenCL) to low level TTA machine code for a particular TTA processor. A graphical and command line simulator is provided to analyze the execution of the program on the processor with detailed information about the resource usages and cycle counts during the execution.

If the designed processor does not meet the target performance, the designer can go back to the source code and design the processor again. In order to find the actual run time, the processor must be synthesized or mapped on a field programmable gate array (FPGA). A hardware database (HDB) is provided in TCE that includes VHDL description for some common FUs, RFs, sockets etc. TCE also allows the designer to build more complex SFU to reduce the processing latency. In that case, the designer has to write the VHDL description for the SFU and add it in the HDB. The VHDL description of the entire core can be generated by the processor generator (ProGe) tool. This VHDL can be simulated, tested and synthesized with third party tools. A diagram of the TTA processor design methodology is given in Fig. 7.

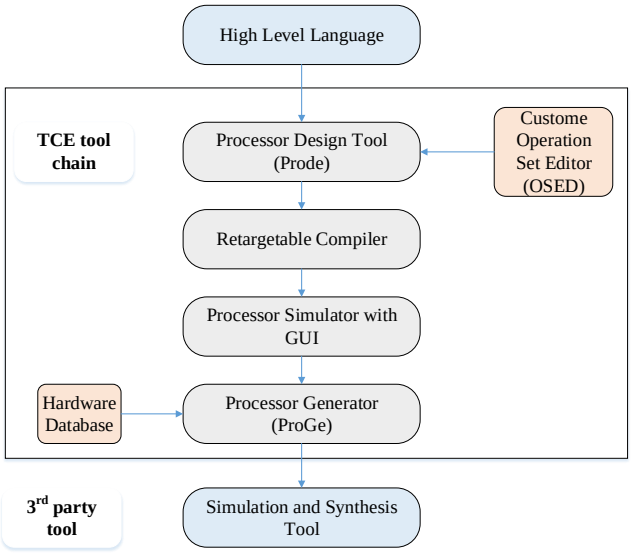

Fig. 7: TTA processor design methodology.

C. Used Case 1: Detector ASIP

A 16-bit fixed point TTA ASIP is designed to support the MMSE, 8-best LSD, SSFE with spanning vectors $[11111111]$ and $[11111222]$ for 4×4 MIMO systems. The 16-bit word length with 5-bit integer and 10-bit fraction has been used extensively in several implementations. We invite interested readers to go through [9] and [25] where the word length studies for these algorithms have been done. The detector ASIP includes LSU, arithmetic logic unit (ALU), global control unit (GCU) and RFs. The multimode detector takes $\underline{\mathbf{R}}_d$, $\tilde{\mathbf{y}}$ and $\mathbf{Q}_{bd}$ as inputs. Several LSU units are used to support memory accesses. LSU can be read the memory in three clock cycles and write in a single cycle. The ALU unit is used to perform basic arithmetic operations like addition, subtraction etc. Operations like shifting right or left are also included in the ALU. We also added several other arithmetic units to utilize the ILP supported by a TTA procoessor. The GCU is used to support jump and branching. Twenty eight buses are used in the design. Several RFs are used to save the intermediate results. A single Boolean register file is included in the processor design. MMSE detection only needs the conventional arithmetic units. Thus, we do not include any SFU to accelerate the MMSE.

A single cycle SFU called slicer is designed to accelerate the program execution of SSFE detection. The slicer unit selects a set of closest constellation points such that the partial Euclidean distance increment is minimized at each level. The slicer SFU takes two inputs where the first input defines the number of symbol candidates as outputs. The second input defines the value needed to be sliced. The slicer has three outputs that can deliver a maximum of three best symbol candidates. In the real valued signal model, 16-QAM and 64-QAM have four and eight symbol candidates respectively. However, due to the structure of the level update vector used in this work, three output is sufficient for the slicer. The rest of the SSFE calculation is done with the multipliers and adders of the ASIP.

A hardware sorter is designed as a SFU for the 8-best LSD algorithm. We use an insertion sorter that keeps the list in order all the time. A new value is compared to all the elements in parallel and the comparisons indicate where the new value should be stored or discarded. An example structure of a 4-value sorter is presented in Fig. 8. The earlier values are stored

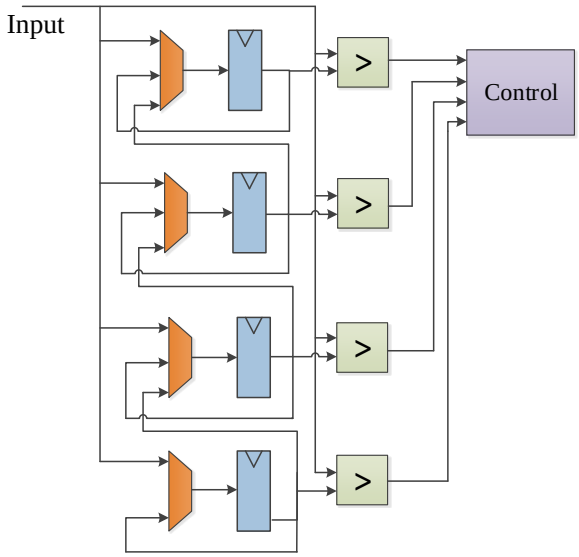

Fig. 8: Insertion sorter (ISORT) SFU.

in a register array such that the input and output of consecutive registers are connected. A simple combinational logic controls the multiplexers that selects the new inputs to be stored in the registers. The rest of the K-best algorithm is calculated with the general FUs.

D. Used Case 2: Precoder ASIP

The precoder ASIP is designed to support norm-based scheduler, QR decomposition, MMSE and ZF-DPC precoding for a BS with $M = 4$ antennas that serves M active users out of a total $N = 20$ users. A 32-bit fixed point TTA processor is designed for precoding. The ASIP includes traditional LSU, ALU, GCU and RFs. We use complex arithmetic units, such as, complex adders and multipliers where 16-bits are used for both real and imaginary parts.

Two SFUs are designed to accelerate the norm-based scheduling. A SFU called MGN is designed to calculate the absolute value of any complex number. Two real valued multipliers are used inside the MGN to compute the square of real and imaginary parts of the inputs. An insertion sorter SFU, very similar to the one described in Section V-C, is used for precoding. The insertion sorter takes the summation of absolute values and the corresponding indices as inputs at a time and keeps the indices of four highest values in sorted order.

The reciprocal of the norm of a vector is needed in QR decomposition. We design a three cycle inverse square root unit called ISQRT in this work. The architecture of the ISQRT unit is shown in Fig. 9.

A look-up table (LUT) is used to hold the precomputed inverse square root values of all possible integers of the fixed

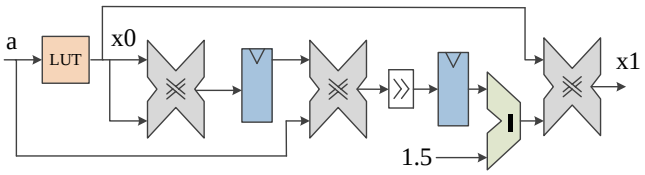

Fig. 9: Inverse square root (ISQRT) SFU.

point input. A 5-bit integer is used for the fixed-point input and thus, a LUT of size 2^5 is used for ISQRT. The output of the LUT x_0 is used as an initial guess for the Newton-Rhapson method. A single iteration of the Newton-Rhapson is used to find the square root of any input a as

$$x_1 = x_0(1.5 - .5 * a * (x_0)^2). \quad (15)$$

Three real multipliers are used in ISQRT and two registers are used in between to shorten the critical path. A similar approach is taken to design a real-valued division circuit that is needed for ZF-DPC precoding.

The channel vectors of $N = 20$ users are stored in a memory. LSU is used to read from memory and a first-in-first-out (FIFO) memory buffer is used to write the output of the processor. Due to the data dependency of QR algorithm, it is not possible to utilize more than four multipliers in an instant. Therefore, four complex-multipliers are included in the ASIP. The real division unit is used for the ZF-DPC calculations. Sixteen buses are used to support ILP and reduce the latency. Fifteen RFs are used to save the intermediate results in this work.

E. Challenges and Solutions of Detector-Precoder ASIP Design

The most challenging task of an ASIP design using TCE is the partitioning of software and hardware. It is not a simple task to determine which part of the algorithm should be executed by software and which part by hardware. The biggest bottleneck of high level software languages arises from loops and branches. A typical loop executes comparison, addition or subtraction and jump instruction for each step. Therefore, a large loop increases the number of instructions several times. The software execution of branches also might take a few cycles and it is difficult for the compiler parallelize such instructions.

The use of SFUs is also not straightforward because a designer needs to identify which part of the algorithm should be accelerated with a SFU and how big the SFU should be. The SFUs are absolutely necessary to increase the throughput of an ASIP. However, the algorithm specific SFUs reduce the programmability. For a multimode implementation, the programmability or flexibility is essential. Thus, a designer must keep the number of SFUs limited even though it can limit the throughput. The word length is also a drawback for the TCE based ASIP implementations. It is not simple to support FUs and buses with different word length. Besides, it is easier to use the commonly used data types rather than any arbitrary or user-defined data types.

We use only a few SFUs to accelerate the detection and precoding. Thus, our implementations are very flexible. We use 16-bit and 32-bit processors, and thus data transport between the FUs and buses are supported by common data type in C language. We conduct numerous simulations to find the necessary number of buses that can provide satisfactory utilization of FUs and SFUs. We unroll the software partially so that the computations associated with every step of loops can be reduced. Most of the branch instructions are replaced with SFUs that contain multiplexers.

VI. Results and Discussion

The ASIPs are synthesized using UMC 90-nm low-leakage standard cell library ($FSD0K\_A\ Core\ Cell$). Synopsys Design Compiler is used to estimate gate count and maximum achievable clock frequency. The operating conditions (temperature, operating voltage, manufacturing process quality) for synthesis are set to default value (TCCOM). The 16-bit detection ASIP takes an area of .293 mm^2 that is equivalent to 73 212 two-input drive-strength-one NAND gate equivalents. The maximum achievable clock frequency is 200 MHz. The critical path of the ASIP is located in the ISORT unit. It should be noted that the reported area of our implementation does not consider layout constraint. The latency and throughput of the different detection algorithms for 64-QAM is presented in Table III.

A comparison with different other implementations is presented in Table II. The detector ASIP provides low throughput but also consumes less area. For an OFDM based system, several of this ASIP can be used to process different OFDM tones. Therefore, we invite readers to focus on the hardware efficiency (throughput/area) in the comparison table. Chen $et\ al.$ proposed a reconfigurable ASIP (rASIP) that supports MMSE, MMSE successive interference cancellation (SIC) and Markov chain Monte Carlo (MCMC) detection [11]. The rASIP is constructed with a reconfigurable architecture coupled with a processor designed by the LISA toolset [26]. The ASIP provides superior hardware efficiency than our design in case of MMSE detection. However, the reconfigurable part of the rASIP consumes the majority of the logic gates. Thus, our design provides more flexibility with comparable performance.

Ahmed $et\ al$ presented an ASIP to support multi-tree selective spanning (MTSS) detection for different level update vectors [13]. Sheikh $et\ al$ presented an architecture to support different configurations of K-best list sphere detection (LSD) [12]. Even though the algorithms can be tuned to provide different performance, the implementations rely only on a single algorithm. We argue that such designs are best suited for an ASIC. Yan $et\ al$ presented a dual-mode architecture that supports MMSE and K-best LSD in [14]. The architecture is non-programmable and takes a large area. Our design provides a better compromise between flexibility and hardware efficiency. It should be noted that [11] and [14] also includes the preprocessing circuitry, so the comparison in Table II is not entirely fair. Instead of traditional ASIPs, a no instruction based computer (NISC) based approach was introduced in [27] for an iterative MIMO equalizer and demapper. A NISC approach eliminates the custom instruction set. The compiler is responsible for scheduling operations and decoding them into control words (CWs). The group of control signals are packed in the

TABLE II: Comparison of Mutimode Detectors

	[11]			[12]	[13]	[14]		Proposed			
Technology [nm]	65			22	40	65		90			
Clock freq. [MHz]	400			1000	600	550		200			
Core area^a [mm^2]	1.1 (2.1^a)			0.03 (0.5^a)	-	6.45 (12.53^a)		0.293			
Cell area^a [kGE]	525			125	-	3132.5		73			
Preprocessing	Included			Not included	Not included	Included		Not included			
Algorithm	MMSE	SIC	MCMC	K-best	MTMS	MMSE	K-best	SSFE [...1]	MMSE	SSFE [...2]	K-best
Throughput^a [Mb/s]	600	124.67	18.75	3200	2057	3300	2640	66.66	42.85	11.76	6.16
Scaled throughput^a [Mb/s]	433.33	90.03	13.54	782.22	914.22	2383	1920	-	-	-	-
Scaled throughput /area^a [Mb/(s×kGE)]	0.8248	0.1714	0.0258	6.256	-	0.7609	0.6130	0.9132	0.5870	0.1611	0.0844

^{a}Technology scaling to 90 nm assuming $A \sim 1/s^2$ and $t \sim 1/s$.

TABLE III: Latency and throughput of different detection algorithms for 64-QAM

Algorithm	Clock Cycle	Throughput
SSFE [11111111]	72	66.66 Mbps
MMSE	112	42.85 Mbps
SSFE [11111222]	408	11.76 Mbps
8-best LSD	778	6.16 Mbps

CWs that are loaded to the datapath components at every cycle. However, the implementation in [27] is not a true multimode detector because only the MMSE detection is supported. The system model is also different as [27] assumes an iterative turbo equalizer, while this paper assumes a non-iterative MIMO detection; i.e. no feedback loop exists between detection and decoding. The MMSE equalization part of the NISC architecture achieves 162.8 and 12.2 mega symbols per second for block fading and fast fading respectively. The equalizer takes an area 0.126 mm2 and for a 16-QAM modulation scheme the scaled throughput for the NISC equalizer is 7.8226 Mbps. It should be noted that [12] and [27] achieves significantly higher scaled throughput than the multimode implementations. The architectures that support several algorithms ([11], [14] and proposed) achieves lower scaled throughput because the whole design cannot be optimized for a single algorithm. A homogeneous multiprocessor system consisting of several TTA processor proposed in this paper can increase the throughput. The TTA cores of the multiprocessor can process several OFDM subcarriers in parallel. For example, the scaled MMSE throughput of [11] can be reached with a multiprocessor consisting of ten proposed TTA cores. The ten cores take a total area of 2.93 mm^2 which is comparable to the 2.1 mm^2 of [11]. Similarly, 55 cores working on the same number of OFDM subcarriers can reach the scaled throughput of the ASIC presented in [14]. The total multiprocessor area is 16.11 mm^2 that is comparable to the 12.53 mm^2 of [14].

The 32-bit precoder ASIP takes an area of 0.44 mm^2 that is equivalent to 110 031 2-input NAND gate equivalents. The maximum achievable clock frequency is 210 MHz. The critical path of the ASIP is located in the complex multiplier. The latency associated with different parts of the precoder ASIP is provided in Table I. Memory access is a costly operation and can increase the latency significantly. For example, the scheduler needs to access the memory 80 times to read channel vectors of $N = 20$ users. QR decomposition also needs to access memory frequently and thus increase latency. The number of clock cycles needed for MMSE and DPC are nearly equal. However, DPC needs an extra division unit.

TABLE IV: Latency of different parts of the Precoder Chain

Algorithm	Clock Cycle
Norm-based Schduling	102
QR	187
QR (extended)	340
MMSE	92
DPC	97

A handful hardware implementation can be found for small scale MU-MIMO precoding. A DPC precoder based on nested trellis is implemented on FPGA in [28]. A Tomlinson-Harashima (TH) precoder is designed in [29] where the LQ decomposition is implemented in ASIP and the other parts are implemented as monolithic hardware. A fixed sphere encoder (FSE) based precoder is presented in [30]. The performance of the precoders are listed in Table V. The proposed ASIP provides higher throughput than the FPGA implementation presented in [29]. The FPGA implementation of [30] provides significantly higher throughput. However, the design is optimized for 6×6 MIMO configuration. In addition, our precoder ASIP is more realistic as it considers scheduling unlike the other implementations and more flexible as it is programmable. The existing small scale precoders are implemented for different algorithms, platforms and different MIMO configurations. Therefore, it is very difficult to compare our ASIP to the existing implementations fairly.

The focus for detection and precoding has been shifted to the large-scale MIMO systems lately. Prabhu $et\ al$ presented a baseband chip that supports massive MIMO detection and precoding in [31]. The precoder is designed for 128 transmit antennas and 8 users. The chip achieves 300 Mbps data rate on real measured channels. A massive MIMO ASIP for a similar configuration could be explored for a future work.

The system models and the architectures for ASIPs presented in this paper significantly deviates form each other. Therefore, we presented two separate multimode designs to demonstrate the soundness and effectiveness of a programmable acceleartor. Due to the commonalities between detection and precoding, it is possible to create a joint architecture. A single ASIP supporting

TABLE V: Performance of small scale precoders

Reference	Architecture	MIMO	Algorithm	Throughput
Proposed	TTA ASIP	4×4	MMSE	52.17 Mbps
Proposed	TTA ASIP	4×4	ZF-DPC	51.95 Mbps
[29]	ASIP & VLSI	4×2	TH	N/A
[28]	FPGA	-	DPC	51 Mbps
[30]	FPGA	6×6	FSE	559 Mbps
[31]	ASIC	128×8	MMSE	300 Mbps

data detection in the uplink and precoding in the downlink could be a natural direction for future research. A soft-core processor implementation on a FPGA platform that supports detection and precoding can be another interesting direction of research. The traditional soft-core processors are vendor specific and can only be used on the products provided by that vendor. On the other hand, the hard processor cores limited in numbers and can not be scaled to match the application. Also, the fixed hard cores can lead to difficulties in interconnection routing between the core and other logic. A TTA ASIP can work as a soft-core processor and can provide additional support other than detection and precoding. However, the current hardware database for common units provided with TCE is not targeted for recent FPGAs. Thus, the current implementations might lead to poor timing results for a FPGA.

This paper demonstrated the opportunities presented by ASIP designs for a multimode basedband signal processing system. The multimode designs are needed for adaptive or cognitive systems where the user want to switch the detection or precoding algorithms based on the channel condition. These implementations are also useful to system-on-chip (SoC) or FPGA vendors who can offer the developers with accelerators supporting a small set of algorithm. An ASIP is the natural choice for multimode systems and this paper demonstrated two such scenarios.

VII. Conclusions

We presented two ASIPs for small scale MIMO symbol detection and precoding. The first ASIP supports several MIMO detection algorithms namely MMSE, 8-best LSD and two variants of SSFE. The detector ASIP delivers 6.16 - 66.66 Mbps throughput at a clock frequency of 200 MHz on 90 nm technology. The hardware efficiency of our design provides more flexibility and comparable throughput to the existing multimode architectures. The precoder ASIP provides a throughput of 52.17 and 51.95 Mbps for MMSE and ZF-DPC precoding respectively. We have shown two examples of how an ASIP design can be a viable alternative for traditional FSM controlled RTL designs for an adaptive or multimode baseband signal processing system.

References

[1] S. Shahabuddin, J. Janhunen, E. Suikkanen, H. Steendam, and M. Juntti, "An adaptive detector implementation for mimo-ofdm downlink," in *Intl. Conf. Cog. Radio Wireless Net.*, June 2014, pp. 305–310.

[2] S. Shahabuddin, O. Silvén, and M. Juntti, "Asip design for multiuser mimo broadcast precoding," in *European Conf. Net. and Commun.*, June 2017.

[3] H. Corporaal, "Design of transport triggered architectures," in *Proc. Great Lake Symp. on VLSI*, Mar 1994, pp. 130–135.

[4] ——, *Microprocessor Architectures: From VLIW to Tta*. New York, NY, USA: John Wiley & Sons, Inc., 1997.

[5] O. Esko, P. Jääskeläinen, P. Huerta, C. S. de La Lama, J. Takala, and J. I. Martinez, "Customized exposed datapath soft-core design flow with compiler support," in *Proc. Intl. Conf. Field Prog. Logic App.*, ser. FPL '10. Washington, DC, USA: IEEE Computer Society, 2010, pp. 217–222. [Online]. Available: http://dx.doi.org/10.1109/FPL.2010.51

[6] A. Burg, M. Borgmann, M. Wenk, M. Zellweger, W. Fichtner, and H. Bolcskei, "Vlsi implementation of mimo detection using the sphere decoding algorithm," *IEEE J. Solid-State Circuits*, vol. 40, no. 7, pp. 1566–1577, July 2005.

[7] C. Studer, A. Burg, and H. Bölcskei, "Soft-output sphere decoding: Algorithms and VLSI implementation," *IEEE J. Sel. Areas Commun.*, vol. 26, no. 2, pp. 290–300, Feb. 2008.

[8] C. Studer, S. Fateh, and D. Seethaler, "ASIC implementation of soft-input soft-output MIMO detection using MMSE parallel interference cancellation," *IEEE J. Solid-State Circuits*, vol. 46, no. 7, pp. 1754–1765, May 2011.

[9] J. Janhunen, T. Pitkanen, O. Silven, and M. Juntti, "Fixed- and floating-point processor comparison for mimo-ofdm detector," *IEEE J. Sel. Topics Signal Process.*, vol. 5, no. 8, pp. 1588–1598, Dec 2011.

[10] J. Antikainen, P. Salmela, O. Silven, M. Juntti, J. Takala, and M. Myllyla, "Application-specific instruction set processor implementation of list sphere detector," in *Proc. Annual Asilomar Conf. Signals, Syst., Comp.*, Nov 2007, pp. 943–947.

[11] X. Chen, A. Minwegen, S. B. Hussain, A. Chattopadhyay, G. Ascheid, and R. Leupers, "Flexible, efficient multimode mimo detection by using reconfigurable asip," *IEEE Trans. VLSI Syst.*, vol. 23, no. 10, pp. 2173–2186, Oct 2015.

[12] F. Sheikh, C. H. Chen, D. Yoon, B. Alexandrov, K. Bowman, A. Chun, H. Alavi, and Z. Zhang, "3.2 gbps channel-adaptive configurable mimo detector for multi-mode wireless communication," *J. Sig. Processing Syst.*, vol. 84, no. 3, pp. 295–307, 2016.

[13] U. Ahmad, M. Li, A. Amin, L. V. Perre, R. Lauwereins, and S. Pollin, "An energy-efficient reconfigurable asip supporting multi-mode mimo detection," *J. Sig. Processing Syst.*, vol. 85, no. 1, pp. 5–21, Oct. 2016.

[14] Z. Yan, G. He, Y. Ren, W. He, J. Jiang, and Z. Mao, "Design and implementation of flexible dual-mode soft-output mimo detector with channel preprocessing," *IEEE Trans. Circuits Syst. I*, vol. 62, no. 11, pp. 2706–2717, Nov 2015.

[15] 3GPP, "Evolved Universal Terrestrial Radio Access (E-UTRA); Physical channels and modulation," 3rd Generation Partnership Project (3GPP), TS 36.211, Jan. 2016. [Online]. Available: http://www.3gpp.org/ftp/Specs/html-info/36211.htm

[16] S. Han, C. Yang, M. Bengtsson, and A. I. Perez-Neira, "Channel norm-based user scheduler in coordinated multi-point systems," in *Proc. IEEE Global Telecommun. Conf.*, Nov 2009, pp. 1–5.

[17] D. Wubben, R. Bohnke, V. Kuhn, and K. D. Kammeyer, "Mmse extension of v-blast based on sorted qr decomposition," in *Proc. IEEE Veh. Technol. Conf.*, vol. 1, Oct 2003, pp. 508–512 Vol.1.

[18] P. Luethi, C. Studer, S. Duetsch, E. Zgraggen, H. Kaeslin, N. Felber, and W. Fichtner, "Gram-schmidt-based qr decomposition for mimo detection: Vlsi implementation and comparison," in *Asia Pacific Conf. on Circuits and Syst.*, Nov 2008, pp. 830–833.

[19] I. B. Collings, M. R. G. Butler, and M. McKay, "Low complexity receiver design for mimo bit-interleaved coded modulation," in *Proc. IEEE Int'l Symp. Sprd Spec. Tech. and App.*, Aug. 2004, pp. 12–16.

[20] M. Li, B. Bougard, E. E. Lopez, A. Bourdoux, D. Novo, L. V. D. Perre, and F. Catthoor, "Selective spanning with fast enumeration: A near maximum-likelihood mimo detector designed for parallel programmable baseband architectures," in *Proc. IEEE Int'l Conf. Commun. (ICC)*, May 2008, pp. 737–741.

[21] C. W. Chen, H. W. Tsao, and P. Y. Tsai, "Equal-rate qr decomposition based on mmse technique for multi-user mimo precoding," in *Proc. IEEE Int'l Symp. Personal, Indoor and Mobile Radio Commun.*, Sept 2013, pp. 435–440.

[22] A. D. Dabbagh and D. J. Love, "Precoding for multiple antenna gaussian broadcast channels with successive zero-forcing," *IEEE Trans. Signal Process.*, vol. 55, no. 7, pp. 3837–3850, July 2007.

[23] G. Caire and S. Shamai, "On the achievable throughput of a multiantenna gaussian broadcast channel," *IEEE Trans. Inf. Theory*, vol. 49, no. 7, pp. 1691–1706, July 2003.

[24] L. N. Tran, M. Juntti, M. Bengtsson, and B. Ottersten, "Beamformer designs for miso broadcast channels with zero-forcing dirty paper coding," *IEEE Trans. Wireless Commun.*, vol. 12, no. 3, pp. 1173–1185, March 2013.

[25] E. Suikkanen, J. Janhunen, S. Shahabuddin, and M. Juntti, "Study of adaptive detection for mimo-ofdm systems," in *2013 International Symposium on System on Chip (SoC)*, Oct 2013, pp. 1–4.

[26] H. M. Anupam Chattopadhyay and R. Leupers, *LISA: A Uniform ADL for Embedded Processor Modelling, Implementation and Software Toolsuite Generation*. Morgan Kaufmann, jun 2008, ch. 5, pp. 95–130.

[27] M. Rizk, A. Baghdadi, M. Jezequel, Y. Mohanna, and Y. Atat, "Design and prototyping flow of flexible and efficient nisc-based architectures for mimo turbo equalization and demapping," *Electronics*, vol. 5, no. 3, 2016. [Online]. Available: http://www.mdpi.com/2079-9292/5/3/50

[28] P. Bhagawat, W. Wang, M. Uppal, G. Choi, Z. Xiong, M. Yeary, and A. Harris, "An fpga implementation of dirty paper precoder," in *Proc. IEEE Int'l Conf. Commun. (ICC)*, June 2007, pp. 2761–2766.

[29] K. Shimazaki, S. Yoshizawa, Y. Hatakawa, T. Matsumoto, S. Konishi, and Y. Miyanaga, "A vlsi design of an arrayed pipelined tomlinson-harashima precoder for mu-mimo systems," in *Asia-Pac. Sig. Inf. Proc. Assoc. Conf.*, Oct 2013, pp. 1–4.

[30] M. Barrenechea, L. Barbero, M. Mendicute, and J. Thompson, "Design and hardware implementation of a low-complexity multiuser vector precoder," in *Conf. on Design and Arch. for Sig. and Img. Processing*, Oct 2010, pp. 160–167.

[31] H. Prabhu, J. N. Rodrigues, L. Liu, and O. Edfors, "3.6 a 60pj/b 300mb/s 128x8 massive mimo precoder-detector in 28nm fd-soi," in *Proc. IEEE Int'l Solid State Circuits Conf. (ISSCC)*, Feb 2017, pp. 60–61.