HARDWARE IMPLEMENTATION OF A LOW-COMPLEXITY DETECTOR FOR LARGE MIMO

B. Cerato[*]

Thyia Tehnologjie, 1000 Ljubljana, Slovenia

E. Viterbo,[†]

Università della Calabria, 87036 Rende (CS), Italy

ABSTRACT

Large MIMO systems represents an effective way to transmit reliably at very high data-rate, but their complexity still represents a problem for practical realization. This paper addresses the hardware implementation of a low-complexity and high-performance detector for a 32 $\times$ 32 MIMO. It allows to reach very high data rate, up to more than 170 Mbit/s with a 64 QAM with BER $10^{-1.5}$-10^{-2} and constitutes a cost effective improvement over basic detection schemes.

Index Terms— Large MIMO, MIMO detector, hardware implementation

1. INTRODUCTION

MIMO with a large number of Tx/Rx antennas raised a lot of interest in the last years due to the fact that they enable very high data rate, while maintaining reliable communications. In this paper we address an hardware implementation of the *likelihood ascent search* (LAS) receiver proposed in [1], for large MIMO systems, with tens of transmit and receive antennas. Large MIMO systems naturally appear in a multi-user environment, where there are many transmitting users and tens of receiving antennas, in the base-station.

The LAS detector concatenated with turbo codes was shown to achieve, in large MIMO settings, near AWGN-SISO performance and to achieve performances within 4.6 dB from theoretical capacity limits. Figure 1 shows the performance using a LAS detector on a 32 $\times$ 32 MIMO after the zero forcing (ZF) filter and compares them to ZF filter alone and the ZF filter followed by Lenstra-Lenstra-Lovasz (LLL) reduction. LLL reduction has been shown to allow a full diversity detection at the expense of a very high decoding complexity [2].

The paper is organized as follow: Section 2 presents a short overview of the LAS algorithm, useful to introduce its hardware implementation. A more detailed description can be found in [1]. Section 3 presents the fixed-point analysis and the architecture of the hardware implementation, while in Section 4 synthesis results and throughput for both ASIC and FPGA implementations are presented.

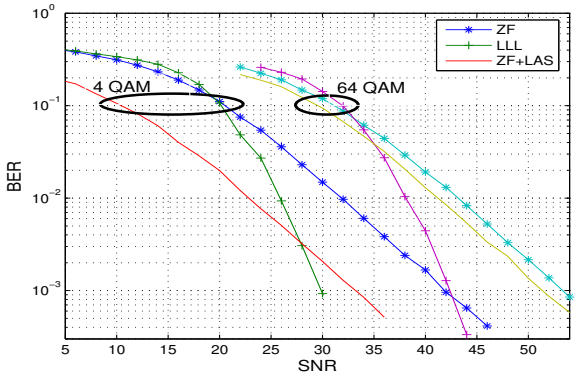


Fig. 1. Comparison of different ZF based detectors in a 32 $\times$ 32 MIMO

2. SYSTEM MODEL AND LAS DETECTOR

Practical MIMO systems with n transmitters and n receive antennas can be modeled through the "Block Fading" channel model, where each transmitted word $\mathbf{x}_c$ with entries from a M^2-QAM will be affected by an independently varying $n \times n$ complex channel matrix $\mathbf{H}_c$. Then, the n dimensional received vector can be expressed as:

$$\mathbf{y}_c = \mathbf{H}_c\mathbf{x}_c + \mathbf{n}_c$$

where $\mathbf{n}$ is the additive white Gaussian noise matrix with entries $\sim \mathcal{N}_c(0, N_0)$. Using a standard technique [3], we can transform our system into a double-dimensional real one

$$\mathbf{y} = \mathbf{H}\mathbf{x} + \mathbf{n}$$

where $\mathbf{y}$, $\mathbf{x}$, $\mathbf{n}$ are real vectors with $2n$ entries, $\mathbf{x}$ entries belong to M-PAM, i.e., $\mathbf{x} \in (M\text{-PAM})^{2n}$, and $\mathbf{H}$ is a $2n \times 2n$ real channel matrix.

The LAS algorithm receives an initial estimate $\mathbf{d}^{(\mathbf{0})}$ of $\mathbf{x}$, which can be obtained by means of a ZF or a minimum mean

[*]The first author performed the work while at Università della Calabria

[†]This work was supported by the STREP project No. IST-026905 (MASCOT) within the 6th framework program of the European Commission.

square error (MMSE) filter. Then it generates a sequence of search iterations with a single symbol update, where the likelihood increases monotonically until an absolute or a local minimum is reached.

The minimizing criterion is the ML criterion and the cost function, which is minimized at each iteration, is

$$C^{(k)} = \|\mathbf{y} - \mathbf{H}\mathbf{d}^{(k)}\|^2 \quad (1)$$

$\mathbf{d}^{(k)}$ is the solution vector at iteration k. The minimization of (1) corresponds to the minimization of:

$$C'^{(k)} = \mathbf{d}^{(k)^T}\mathbf{H}^T\mathbf{H}\mathbf{d}^{(k)} - 2\mathbf{y}^T\mathbf{H}\mathbf{d}^{(k)} \quad (2)$$

Starting from the assumption that a one symbol update is only allowed, at iteration $k+1$, solution $\mathbf{d}^{(k+1)}$ can be obtained from $\mathbf{d}^{(k)}$ as:

$$\mathbf{d}^{(k+1)} = \mathbf{d}^{(k)} + \lambda_p^{(k)}\mathbf{e}_p \quad (3)$$

where $\mathbf{e}_p$ is a vector whose entries are all zeroes except the p-th one which is one and $\lambda_p^{(k)}$ is an integer set that makes the new solution belong to the solution space $(M\text{-PAM})^{2n}$. Defining the Gram matrix $\mathbf{G} \triangleq \mathbf{H}^T\mathbf{H}$ the difference in the cost function between two iteration can be evaluated through:

$$\mathcal{F}(\lambda_p^{(k)}) \triangleq C_p'^{(k+1)} - C_p'^{(k)} = \lambda_p^{(k)^2} G_{p,p} - 2\lambda_p^{(k)} z_p^{(k)} \quad (4)$$

where $z_p^{(k)}$ is the pth entry of $\mathbf{z}^{(k)} = \mathbf{H}^T(\mathbf{y} - \mathbf{H}\mathbf{d}^{(k)})$ and $G_{p,p}$ is the pth entry on the diagonal of $\mathbf{G}$. Equating the derivative of (4) to 0 we can obtain in closed form the optimum value of $\lambda_p^{(k)}$, which minimizes (4) for each p

$$\lambda_{p,opt}^{(k)} = 2\left\lfloor \frac{z_p^{(k)}}{2G_{p,p}} \right\rceil \quad (5)$$

where $\lfloor\cdot\rceil$ denotes the closest integer. The selected $\lambda_{p,opt}^{(k)}$ must guarantee that the new component of $\mathbf{d}^{(k+1)}$, $d_p^{(k+1)}$, still belongs to the M-PAM constellation.

In order to decide which entry must be updated, (4) is evaluated for each p and $\lambda_{p,opt}^{(k)}$ and the entry s given by:

$$s = \arg \min_{1<p<2n} \mathcal{F}(\lambda_p^{(k)}) \quad (6)$$

If $\mathcal{F}(\lambda_s^{(k)}) < 0$, $\mathbf{d}$ and $\mathbf{z}$ must be updated for the following iteration, otherwise it means that $\mathcal{F}(\lambda_s^{(k)}) = 0$, the algorithm has found a local minimum and terminates. The updating function for $\mathbf{d}$ is represented by (3), while $\mathbf{z}$ can be updated through:

$$\mathbf{z}^{(k+1)} = \mathbf{z}^{(k)} - \lambda_{s,opt}^{(k)}\boldsymbol{g}_s \quad (7)$$

where $\mathbf{g}_s$ represents the s-th column of matrix $\mathbf{G}$.

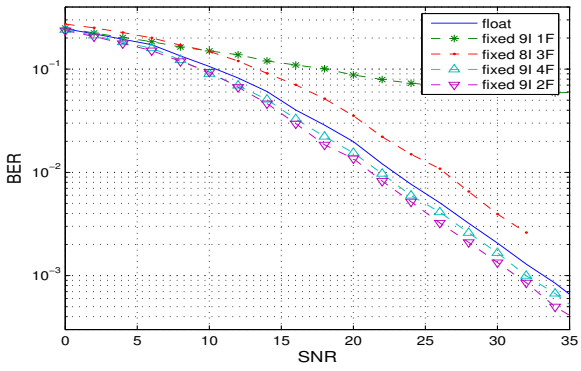


Fig. 2. Fixed-point analysis with 4-QAM and 32 antennas system

3. HARDWARE IMPLEMENTATION

This paper addresses an hardware implementation for a 32 × 32 MIMO which leads to a 64 × 64 real channel matrix and a received vector with 64 entries. Furthermore, this implementation is able to adaptively detect 4-/16-/64-QAM symbols. Extension to different dimensions cases is straightforward.

3.1. Fixed-point analysis

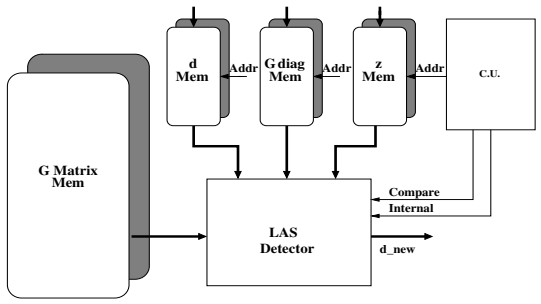
Fixed point analysis, realized through Monte-Carlo simulation, has evidenced (in particular in 4-QAM constellation) an interesting phenomenon. Figure 2, where #I indicates the number of bits used for integer part and #F those for fractional, shows that fixed-point implementation of this algorithm not only allows to obtain the same performance of floating-point one, but, with an appropriate choice of the number of bits for the fractional part it is possible to improve performance over the floating-point. This phenomenon can be explained by the fact that this discretization operation forces the algorithm to neglect some local minima, that may make it stop before the absolute minimum is reached. For 4-QAM for 2 bits of fractional part are optimal, while for 16-QAM 3 bits and for 64-QAM 5 bits are the optimum. A total number of 16 bits (11 for integer and 5 for fractional part) is enough to obtain the same performances of the floating-point implementation for 4-/16-/64-QAM.

3.2. Architecture

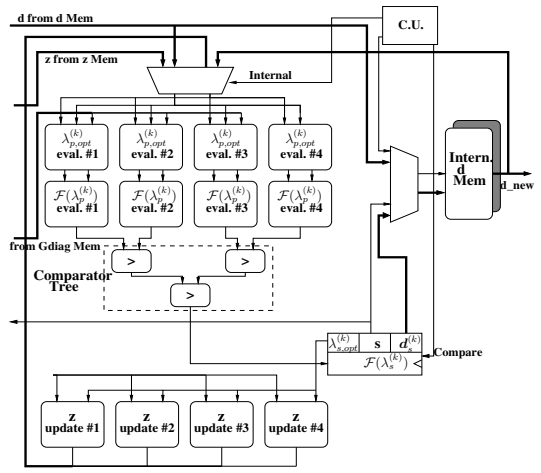
Figure 3(a) represents the overall architecture of the implemented detector. The system is composed of the memory subsystem and the LAS data-path, which realizes the algorithm described in Section (2), whose detailed architecture is depicted in Figure 3(b).

3.2.1. LAS Data-path

In this algorithm four computing intensive phases may be distinguished.



(a) System overview



(b) data-path

Fig. 3. Architecture of the implemented hardware block.

1. estimation of $\lambda_{p,opt}^{(k)}$ by (5);
2. computation of $\mathcal{F}(\lambda_p^{(k)})$ by means of (4);
3. search of the minimum, as in (6);
4. update of $\mathbf{z}$ vector through (7).

All these phases, except the third one, are easily parallelizable and this property made the algorithm very attractive, from a hardware point of view, in order to increase the throughput. Assuming that data are provided to the LAS block with the required rate, an almost complete trade-off between throughput and area can be achieved by using a different value of $\mathtt{par}$, i.e. the number of computing chains inside the block (thanks also to the very regular structure of the memory accesses). The only limit to the benefits of the parallelization is represented by the fact that the third of the above-mentioned phases can be effectively realized through the use of a comparator tree and requires for this reason a number of logic levels logarithmic with the number of computing chains. This imposes $2n/\mathtt{par} > \log_2(\mathtt{par})$. In our case $2n = 64$ and consequently $\mathtt{par} < 16$. As shown in Figure 3 in our implementation we choose $\mathtt{par} = 4$. Different trade-offs can be achieved by choosing a different value for $\mathtt{par}$.

In the each computing chain the first stage is represented by the evaluation of $\lambda_{p,opt}^{(k)}$. As outlined in Section 2 the selected $\lambda_{p,opt}^{(k)}$ must guarantee that the corresponding $\mathbf{d}_p^{(k+1)}$ still belongs to the M-PAM constellation. To assure this, we propose to choose simultaneously between a $\lambda_{p,opt}^{\prime(k)}$ computed as in (5) and an alternative $\lambda_{p,opt}^{\prime\prime(k)}$.

In order to select $\lambda_{p,opt}^{\prime(k)}$, as stated in (5), we have to perform a division. Since this is an integer division whose result can be expressed on a small number of bits, we can use a simple fixed-point non-restoring binary division [4]. Our implementation, which enables the selection of $\lambda_{p,opt}^{\prime(k)}$ within up to a 8-PAM constellation, requires 4 clock cycles.

Concerning the selection of $\lambda_{p,opt}^{\prime\prime(k)}$, we can take into account some simplifying considerations. Since the terms $G_{p,p}$ on the diagonal of a Gram matrix are positive, from (5) we have $sign(\lambda_{p,opt}^{\prime(k)}) = sign(z_p^{(k)})$ and it is easy to anticipate which of the constellation boundary may be violated. In order to bring $d_p^{(k)}$ back into the constellation the following rule is used:

$$\begin{cases} d_p^{(k)} + \lambda_{p,opt}^{\prime\prime(k)} = M - 1 & \text{if } sign(z_p^{(k)}) > 0 \\ d_p^{(k)} + \lambda_{p,opt}^{\prime\prime(k)} = -M + 1 & \text{if } sign(z_p^{(k)}) < 0 \end{cases} \tag{8}$$

Thus:

$$\lambda_{p,opt}^{\prime\prime(k)} = (M-1)sign(z_p^{(k)}) - d_p^{(k)} \tag{9}$$

The selection between $\lambda_{p,opt}^{\prime(k)}$ and $\lambda_{p,opt}^{\prime\prime(k)}$ can be made choosing the minimum between the two in absolute value, which is very easy to decide since they have the same sign. The selection between the two requires one more clock cycle so the first stage takes a total of 5 clock cycles to perform its computation.

The second stage is represented by the evaluation of the metric by means of (4). This computation takes 3 clock cycles, one to square the value of $\lambda_{p,opt}^{(k)}$ and to multiply $\lambda_{p,opt}^{(k)}$ with $z_p^{(k)}$, the second to multiply $\lambda_{p,opt}^{\prime(k)^2}$ with $G_{p,p}$ and the last in order to produce the final metric with the subtraction in (4).

The metrics produced independently from each computing chain enter the *Comparator Tree*. This stage, after $\log_2(\mathtt{par})$ levels of comparators produces, the minimum between the $\mathtt{par}$ metrics.

The *control unit* (C.U.) enables the comparison and the storage between the $2n/\mathtt{par}$ metrics that come out from the *Comparator Tree*. After that all these metrics are compared and the minimum among them $\mathcal{F}(\lambda_s^{(k)})$ is selected, if $\mathcal{F}(\lambda_s^{(k)}) < 0$ the C.U. enables the updating of the proper $\mathbf{d}$ entry in the *"Internal* $\mathbf{d}$ *Mem"* and the algorithm continue using $\mathbf{d}$ entries coming from the *"Internal* $\mathbf{d}$ *Mem"* $\mathbf{z}$ entries that are produced at each clock cycle from the last stage of

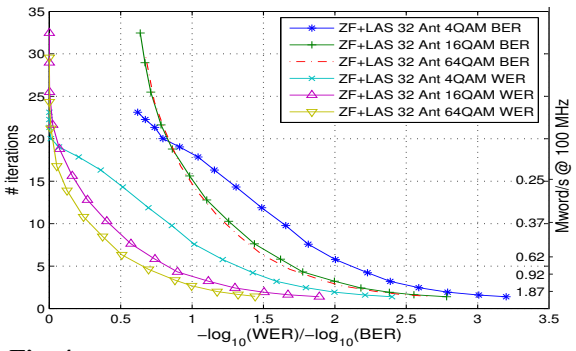


Fig. 4. Average # of iteration at varying of constellation size

Table 1. Synthesis Results

Xilinx Virtex 5 330 LX		**65 nm technology**	
LAS & CU			
# of Slice LUTs	2184 (1%)	Area[GE]	30,8K
# of DSP48Es	12 (6%)		
# of Slice Registers	3155 (1%)		
Register files			
# of fully used Bit Slices	1073 (25%)	Area[GE]	31,8K
G Matrix			
# of Block RAM/FIFO	4 (1%)	Area [mm^2]	1,72
Max. Freq. [MHz]	105.474	Max. Freq. [MHz]	300
Avg. throughput @ 100 MHz			
(4-QAM) 32 Mb/s	(16-QAM) 117 Mb/s		(64-QAM) 170 Mb/s

the computing chain in accordance with (7). If the selected metric instead do not improve the likelihood of the solution, the algorithm terminates.

The whole iteration requires 27 clock cycles.

3.2.2. Memory subsystem

The memory subsystem includes three register files containing respectively the diagonal of matrix $\mathbf{G}$, the first solution $\mathbf{d}$ and vector $\mathbf{z}$, and one big memory containing the $\mathbf{G}$ matrix. All these memories are actually composed of two identical one port memories used in a ping-pong way so that they can be written from the outside while the LAS detector works on another set of data. Furthermore each of them is organized as `par` smaller memories side by side in order to allow an easy access to read simultaneously `par` data.

Thus *"*$\mathbf{d}$ *Mem"* is composed of four register files of 16 entries with wordsize of 3 bit (the number of bits required to express symbols in 8-PAM), *"*$\mathbf{z}$ *Mem"* and *"*$\mathbf{G}$ *diag Mem"* are composed of four register files of 16 entries with wordsize of 16 bits (the data-path wordsize). The *"*$\mathbf{G}$ *matrix Mem"*, instead, is a big memory with four memories of 1096 entries with wordsize of 16 bits.

This big amount of memory, however could be shared with the memory already present on-board of the chip in order to realize the preprocessing phase which is all the same necessary for any kind of detection.

4. RESULTS AND CONCLUSIONS

In Figure 4 the average number of iteration performed by the LAS algorithm with different constellation sizes is shown. The right axis indicate the average number of detected word per second with a clock cycle of 100 MHz. In order to obtain the throughput in Mbit/s this number has to be multiplied for the number of bit in each transmitted word which is 64 (respectively 128 and 192) with 4-QAM (respectively 16 and 64-QAM).

Table 1 summarizes the occupation data onto an FPGA, a Xilinx Virtex 5 330 LX device and the synthesis results onto a 65 nm technology.

The area occupation for all the memory subsystem in Table 1 has been obtained by mapping the memory cells onto flip-flops since memory generator was not available. With a proper instantiation using a memory generator the area occupation can be reduced of about a factor 16.

This algorithm seems very well suited in large MIMO systems, in particular in conjunction with an outer channel decoder as turbo etc. These schemes in fact require that the inner system works in the region with a BER of $10^{-1.5}$-10^{-2} so that the outer decoder would be in the waterfall region where it shows its best performances. This detector allows to reach this BER region up to 10 dB before the ZF filter and this implementation shows that its complexity is feasible also in very big systems because it does not require so many resources besides the ones already required by a basic detection through a MMSE or ZF filter. Future work on this algorithm and its implementation will allow to obtain soft-out information to further improve the performances of an outer decoder. The proposed implementation furthermore allows to reach very high data rate, up to more than 170 Mbit/s with a 64 QAM with BER $10^{-1.5}$-10^{-2}.

5. REFERENCES

[1] K. Vishnu Vardhan, S. K. Mohammed, A. Chockalingam, and B. Sundar Rajan, "A Low-Complexity Detector for Large MIMO Systems and Multicarrier CDMA Systems," *IEEE JSAC*, vol. 26, no. 3, pp. 473–485, Apr. 2008.

[2] A. M. M. Taherzadeh and A. K. Khandani, "LLL lattice-basis reduction achieves maximum diversity in mimo systems," in *Proc. IEEE Intern. Symp. Inf. Theory (ISIT)*, Adelaide, Australia, Sept. 9–10, 2005.

[3] K. Vishnu Vardhan, S. K. Mohammed, A. Chockalingam, and B. Sundar Rajan, "A low-complexity near-ML performance achieving algorithm for large MIMO detection," in *Proc. IEEE Intern. Symp. Inf. Theory (ISIT)*, Toronto, Canada, July 6–11, 2008.

[4] Milos D. Ercegovac and Tomas Lang, *Digital Arithmetic*, Morgan Kaufmann Publishers, 2004.