A Flexible MCMC Detector ASIC

Dominik Auras, Sebastian Birke, Tobias Piwczyk, Rainer Leupers, and Gerd Ascheid
Institute for Communication Technologies and Embedded Systems
RWTH Aachen University, Aachen, Germany

Abstract— **This paper presents a flexible Markov Chain Monte Carlo based MIMO (multi-antenna) detector ASIC that efficiently uses the available computing resources by leveraging inherent chain-level and symbol-level parallelism offered by the underlying detection algorithm. Compared to a reference architecture from literature, we roughly double the throughput on average, while achieving a speedup of up to 4.5x for certain cases. The implementation layouted for a 90nm CMOS technology requires 1.3mm² and runs at 397 MHz.**

MIMO detection, Markov Chain Monte Carlo; Multi-antenna

I. INTRODUCTION

The trend in wireless broadband communications goes to very high transmission rates beyond 1 Gbit/s. Recent standards, such as Wi-Fi or LTE, adopt amongst others multi-antenna (MIMO) technologies to achieve this objective. They use multiple spatial data streams to convey more information in the same radio channel bandwidth, known as spatial multiplexing. These standards employ bit-interleaved coded modulation (BICM). Iterative detection and decoding is another promising technique to increase the data rates, also known as BICM-ID. It improves the receiver systems' spectral efficiency, i.e., how many bits per channel use we can transmit.

We present a Markov Chain Monte Carlo (MCMC) based MIMO detector, which recovers the information bits from the received signals, for BICM-ID receiver systems. It is an extension of [1,2] in prospect to mitigate the relatively low throughput. We double the number of processing units (GS/M circuits) and support the simultaneous processing of up to four inputs. Furthermore, the new flexible architecture allows a more fine-grained control of the algorithm's run-time parameters.

II. BACKGROUND

Markov Chain Monte Carlo based MIMO detection performs random sampling of the code bit's posterior likelihood function in order to compute the extrinsic log-likelihood ratios (LLRs) according to

$$\lambda_i^e = \ln \frac{P(c_i = 1)}{P(c_i = 0)} - \lambda_i^a \qquad (1)$$

given the received signal, channel transfer function observation, noise density etc. The likelihood function is unknown, but can be evaluated at distinct points. The samples are used to estimate the function.

Basically, the detector first draws several random code bit vectors as starting points for the MCMC chains. Then, it walks around by randomly accepting or rejecting transitions to close neighbors (one bit flipped). The code bit likelihood function is evaluated at every point. All values are finally used in (1) as estimate of the true likelihood function.

The transition probability depends on the current state of the MCMC chains, which summarizes all previously evaluated samples, and converges towards the desired likelihood function, which means that the chains step by step approach regions of interest with high likelihood values.

III. ARCHITECTURE

The proposed architecture, depicted in Fig. 1, bases on [1,2]. We process up to four inputs at once. The data distribution stage assigns an input to an idle FEP/Cluster combination. There are 16 GS/M circuits, executing the MCMC detection, organized into four clusters. Each cluster runs four parallel MCMC chains. Four parallel front-end processing (FEP) units preprocess the input data. Each FEP feeds only one cluster. This avoids layout-level signal congestion at the expense of energy consumption due to redundant computations. The LLR circuits support three cases: i) four clusters work on the same input, i.e., we combine 16 parallel chains to produce the output LLRs (use L16 in Fig. 1), ii) four clusters work on two inputs, then L16 and L8 produce the output, and finally iii) with four inputs, we use all four LLR circuits, one per cluster.

The detector implementation supports the two symmetric MIMO antenna configurations 2x2 and 4x4. Its hand-shaking based interface accepts one symbol per clock cycle and stalls appropriately if busy. The output interface shifts out one extrinsic LLR value per clock cycle.

The new flexibility of the architecture is to support multiples of four chains per symbol (finer granularity, [1] supports eight) while making use of all available GS/M circuits, by processing up to four symbols in parallel ([1] uses no symbol-level parallelism). Also, we doubled the number of GS/M circuits ([1] has eight).

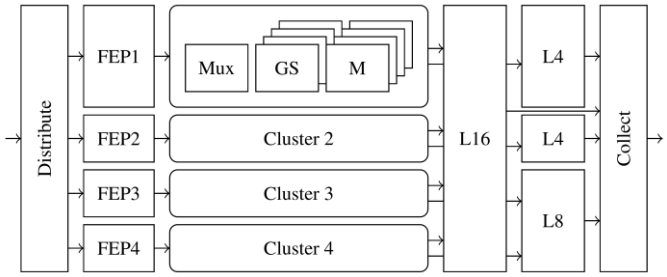

Figure 1: Architecture diagram

TABLE I. IMPLEMENTATION RESULTS

	This Work	[1]
Area	1.3 mm²	0.47 mm²
Gates	367 kGE[a]	149.5 kGE
Frequency	397 MHz	479 MHz
Chip density	88,6%	n/a
Code bit throughput	87.4[b] Mbit/s	52.0 [b] Mbit/s

a. One gate equivalent (GE) corresponds to the area of one 2-input drive-1 NAND gate.
b. Throughput for 4x4 64-QAM mode and 8 chains with 8 samples per chain.

The proposed detector accepts a new symbol every

$$n_{det} = \max\begin{pmatrix} KN_t, \\ ((N_s+1)KN_t+2)N_q/16, \\ (N_t^2(N_t+2)/2+5)N_q/16 \end{pmatrix} \quad (2)$$

cycles on average in the steady state, where N_s denotes the number of samples per chain, N_q is the number of chains per symbol, and KN_t is the number of code bits per symbol. This considers the execution times of the interface, the four FEP circuits, and the four clusters with four GS/M circuits each. The implementation can currently run in parallel four 4-chains, two 8-chains, or one 16-chains operation.

The original MCMC architecture requires

$$n_{old} = \max\begin{pmatrix} KN_t, \\ \lceil N_q/8 \rceil (N_s+1)KN_t+5, \\ N_t^2(N_t+2)/2+3 \end{pmatrix} \quad (3)$$

cycles per symbol. If N_q is not a multiple of 8, some GS/M circuits are disabled. For more than 8 chains, the detector runs them sequentially, e.g. a 16-chains detection is basically two sequentially chained 8-chains detections.

IV. IMPLEMENTATION RESULTS

The detector's RTL design has been synthesized with Synopsys Design Compiler in topographical mode, and a layout was obtained with the Cadence SoC Encounter 13.1 Foundation Flow, using a 1.0V standard-performance standard-cell library for the UMC 90nm SP-RVT LowK CMOS process. Tbl. I compares this work to [1]. The chip layout is shown in Fig. 2. One cluster is larger than a FEP or LLR circuit. Clusters and FEP/LLR-circuits each occupy roughly 50% of the total chip area. Similar to [1,2], the critical path is in the clusters.

We define the speedup over [1] as

$$S = \frac{n_{old}}{n_{det}} \times \frac{397\ \text{MHz}}{479\ \text{MHz}}. \quad (4)$$

Considering all possible combinations of $K = \{2,4,6\}$, $N_t = \{2,4\}$, $N_q = \{4,8,16\}$, $N_s = \{2,4,8,16\}$, we observe

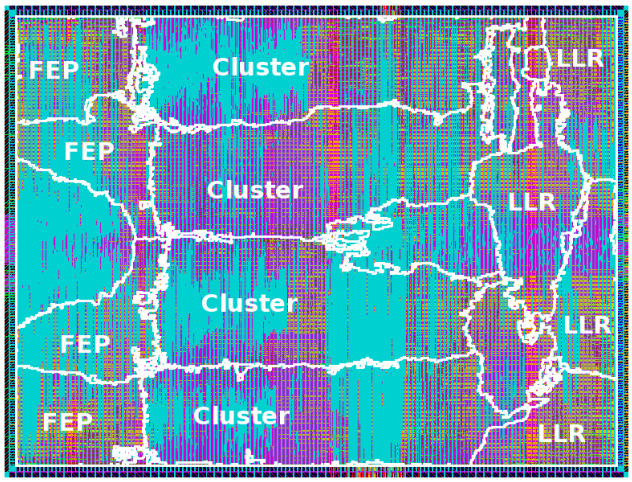

Figure 2: Chip Layout

that the maximum speedup is 4.5x. For one single case ($K = 2$, $N_t = 4$, $N_q = 16$, $N_s = 2$) the proposed ASIC is 20% slower than [1]. In all other cases, we are at least 30% faster, achieving 2.2x on average.

As expected, the architecture's additional flexibility leads to an increased area consumption and a lower operation frequency compared to [1]. This ultimately limits the implementation to achieve the full expected speedup given that we doubled the number of GS/M circuits. However, the more fine-grained run-time parameter control actually allows for a more efficient trade-off between communications performance and throughput, as analyzed in [2]. Due to the changed granularity, we are not directly comparable to [1,2]. A detailed communications performance is necessary to assess the benefit of the flexibility.

V. CONCLUSIONS & OUTLOOK

The proposed flexible architecture leverages several independent parallelism degrees in order to efficiently use the available resources: it computes independent MCMC chains in parallel, and process several input data vectors simultaneously. Its more fine-grained run-time parameter control compared to [1,2] allows a more efficient trade-off between communications performance and throughput.

As future work, we propose to share major parts of the LLR circuits, i.e., use the compare-select tree from the L16 circuit for all LLR computations. Furthermore, some chain samples can be processed simultaneously, which could possibly double the throughput, by combining neighboring GS/M circuits on-demand at run-time.

REFERENCES

[1] D. Auras, U. Deidersen, R. Leupers, and G. Ascheid, VLSI-SoC: Internet of Things Foundations: 22nd IFIP WG 10.5/IEEE International Conference on Very Large Scale Integration, VLSI-SoC 2014, Playa del Carmen, Mexico, October 6-8, 2014, Revised Selected Papers. Cham: Springer International Publishing, 2015, ch. A Parallel MCMC-Based MIMO Detector: VLSI Design and Algorithm, pp. 149–169.

[2] D. Auras, U. Deidersen, R. Leupers, and G. Ascheid, "VLSI design of a parallel MCMC based MIMO detector with multiplier-free gibbs samplers," in Very Large Scale Integration (VLSI-SoC), 2014 22nd International Conference on, Oct 2014, pp. 1–6.