A new Architecture for High Speed, Low Latency NB-LDPC Check Node Processing

P Schläfer, V Rybalkin, N Wehn, M Alles, T Lehnigk-Emden, E Boutillon

A new Architecture for High Speed, Low Latency NB-LDPC Check Node Processing

P. Schläfer and V. Rybalkin and N. Wehn
Microelectronic Systems Design Research Group
University of Kaiserslautern, Germany
{schlaefer,rybalkin,wehn}@eit.uni-kl.de

M. Alles and T. Lehnigk-Emden
Creonic GmbH
Kaiserslautern, Germany
info@creonic.com

E. Boutillon
Lab-STICC
Université de Bretagne Sud, France
emmanuel.boutillon@univ-ubs.fr

***Abstract*—Non-binary low-density parity-check codes have superior communications performance compared to their binary counterparts. However, to be an option for future standards, efficient hardware architectures must be developed. State-of-the-art decoding algorithms lead to architectures suffering from low throughput and high latency. The check node function accounts for the largest part of the decoders overall complexity. In this paper a new hardware aware check node algorithm and its architecture is proposed. It has state-of-the-art communications performance while reducing the decoding complexity. The presented architecture has a 14 times higher area efficiency, increases the energy efficiency by factor 2.5 and reduces the latency by factor of 3.5 compared to a state-of-the-art architecture.**

I. Introduction

Upcoming standards like 5G will increase the demands on throughput and latency of communication systems. Especially applications like the Tactile Internet [1] require a significantly reduced latency. At the same time, error free transmission has to be guaranteed, which requires forward error correction schemes. Low-Density Parity-Check (LDPC) codes where first proposed by R.G. Gallager in 1963 [2] and rediscovered by D. Mackay and others in 1996 [3]. In the following almost two decades a lot of research has been carried out in this field. Today many commercial standards (WiMAX, WiFi, DVB-C2, DVB-S2, DVB-T2) make use of LDPC codes. Very long binary LDPC codes have been proven to perform close to the Shannon limit. However when considering short blocks of only some hundred bit length for low latency applications, they suffer under degradation in communications performance of more than 0.5 dB. The extension of binary LDPC codes to Galois Fields (GF(q)s) with $q > 2$ is a promising approach to solve this problem. Moreover the symbols of high-order modulation schemes can be directly mapped to the decoders input symbols. Thus an additional gain in communications performance is observed for systems combining high-order modulation and Non-Binary Low-Density Parity-Check (NB-LDPC) codes [4]. The performance gain of NB-LDPC codes comes at the cost of significantly increased decoding complexity. The decoding can be performed by message passing algorithms like Belief Propagation (BP), however the complexity increases with the size of the GF(q). A straightforward implementation of the BP algorithm has a complexity of $\mathcal{O}(q^2)$ [5]. In the last years several approaches have been proposed to reduce the decoding complexity without sacrificing the communications performance. Algorithms working in the Fourier domain [6] [7] have an excellent communications performance, but are still too complex for efficient hardware architectures. Symbol flipping algorithms [8] [9] in general have low complexity but suffer from heavily degraded communications performance. Approaches based on stochastic decoding [10] [11] have been presented as an alternative decoding method but introduce very high decoding latency. An extension of the well known binary Min-Sum algorithm to the non-binary domain, called Extended Min-Sum (EMS) algorithm [12] [13] [14] gives the best compromise between hardware complexity and communications performance. Therefore in this paper we will focus on the EMS algorithm which is the most promising starting point for efficient architectures.

To achieve the required throughput of today's applications, executing the algorithms in software is not sufficient. Dedicated hardware architectures become mandatory. The largest complexity in the EMS algorithm is the computation of the Check Node (CN). State-of-the-art architectures apply a so called Forward-Backward (FWBW) scheme [15] to process the CN. A serial calculation is carried out to reduce the hardware cost and to allow for reuse of intermediate results during the computation. However this scheme introduces high latency and degrades the throughput. This effect increases significantly when the size of the GF(q) or the CN degree grows.

The Syndrome-Based (SYN) CN algorithm, previously presented in [16] is the first approach allowing for efficient parallel computation of the CN function for higher-order Galois fields ($q \geq 16$). In this paper we present modifications of the original algorithm leading to the first hardware implementation of the SYN CN algorithm. For a GF(64) we observed an increase in area efficiency of factor 14 and a latency reduction of factor 5 compared to state-of-the-art CN architectures.

The paper is structured as follows. In Section II we review the decoding of NB-LDPC codes making use of the EMS algorithm. The SYN CN algorithm is presented in Section III and further optimizations for the hardware implementation are discussed in Section IV. The novel architecture is described in Section V. Finally Section VI presents a comparison with other CN architectures in means of area and energy efficiency, latency and communications performance. Section VII concludes the paper.

II. EMS Decoding

This section reviews the EMS algorithm to give an overview of the complete decoding process.

Let us consider an (N, K) NB-LDPC code over GF(q) where N is the codeword length and K the number of

information symbols. The code is defined by a sparse parity check matrix H with N columns, $M = N - K$ rows and elements $h_{m,n}$. The transmitted codeword consists of the codeword symbols $c = (c_1, c_2, \ldots, c_N), c_i \in \text{GF}(q)$. The decoder receives the noisy representation of the codeword symbols, $y = (y_1, y_2, \ldots, y_N)$.The decoding process can be partitioned in four main parts, the initialisation, the Variable Node (VN) update, the CN update and the permutations in the Permutation Nodes (PNs). It is important to mention that in contrast to a binary LDPC decoder instead of a single Logarithmic Likelihood Ratio (LLR) a set of q LLRs is exchanged between the nodes on each edge. This fact accounts for the significant increase in complexity compared to binary LDPC decoding.

The first step in the EMS algorithm is the calculation of the symbol LLRs for the received codeword. For each received symbol y_i, a set of q LLR values is calculated. Under the assumption that all GF(q) symbols are equiprobable, the initialization LLRs for VN v are calculated as follows:

$$\begin{aligned} \forall x \in \text{GF}(q) \left\{ L_v[x] = ln\left(\frac{P(y_v|c_v = \tilde{x}_v)}{P(y_v|c_v = x)}\right)\right\} \\ \text{with } \tilde{x}_v = \max_{\forall x \in \text{GF}(q)} \{P(y_v|c_v = x)\} \, . \end{aligned} \tag{1}$$

Given this definition it follows that $L_v[\tilde{x}_v] = 0$ and $L_v[x] \geq 0$ with an increasing LLR representing a decreasing symbol reliability. For simplified reading, in the following we use an array to represent the message sets. Messages from VN to PN are denoted as U_{vp}, messages from PN to CN as U_{pc}. Respectively messages from CN to PN are called V_{cp} and messages from PN to VN as V_{pv}.

In the first iteration the VNs simply forward the channel values (Eq. (1)) to the according CNs. The VN outputs are calculated as $U_{vp}[x] = L_v[x]$. In all other iterations the VN operation is to combine the channel values L_v with the d_v incoming message sets V_{pv}, where d_v is the VN degree. The updated extrinsic messages U_{vp} of VN v are calculated as follows:

$$U_{vp}[x] = L_v[x] + \sum_{t=1, t\neq p}^{d_v} V_{tv}[x], \forall x \in \text{GF}(q), p = 1 \ldots d_v \, . \tag{2}$$

The VN function is to sum up all values for a certain Galois field element x received from the connected CNs and the according channel information $L_v[x]$. This however has to be performed for all q elements of the GF(q). To achieve the same message structure as before (LLR $= 0$ for the most reliable symbol, increasing LLR values for less reliable symbols), a normalisation of the U_{vp} messages with respect to the most reliable symbol has to be applied at the output of the VN.

The next step in the decoding is the permutation according to the parity check matrix H. The permutation of the VN v outputs U_{vp} is defined as:

$$U_{pc}[x] = U_{vp}[h_{c,v}^{-1} \cdot x], \forall x \in \text{GF}(q), p = 1 \ldots d_v \, , \tag{3}$$

where $h_{c,v}$ is the Galois field element at row c and column v of H, U_{pc} represents the input for CN c.

In the CN update d_c edges from U_{pc} are processed. The outputs of CN c are calculated as follows:

$$\begin{aligned} V_{cp}[x] = \min_{\forall \text{ permutations of } x_t} \sum_{t=1; t\neq p}^{d_c} U_{tc}[x_t] \\ \text{with } \sum_{t=1; t\neq p}^{d_c} x_t = x, \\ \forall x \in \text{GF}(q), p = 1 \ldots d_c \, . \end{aligned} \tag{4}$$

For every Galois field element x all possible input permutations fulfilling the parity check constraint are evaluated. The parity check constraint is given by the sum of the according Galois field elements. From all valid combinations the one with the highest reliability (smallest LLR) is chosen. Again, this task has to be performed for all q elements of the Galois field. The CN computation is the most complex part of the decoding and has a complexity of $\mathcal{O}(q^2)$ when calculated straightforward with the FWBW scheme [15].

Before one iteration is completed the outputs of CN c must be reverse permuted.

$$V_{pv}[x] = V_{cp}[h_{c,v} \cdot x], \forall x \in \text{GF}(q), p = 1 \ldots d_c \, . \tag{5}$$

This closes the loop and another iteration starts. The processing of a block is stopped as soon as a valid codeword is detected. To make this decision the estimated symbols $\hat{x}$ need to be computed for each VN v:

$$\hat{x}_v = \min_{x \in \text{GF}(q)} \left(L_v[x] + \sum_{p=1}^{d_v} V_{pv}[x] \right) \, . \tag{6}$$

In state-of-the-art EMS decoding one important simplification is applied. It has been shown in [12], that the sets of q messages exchanged between VNs and CNs can be truncated to carry only the n_m most reliable values per edge without sacrificing the communications performance. This approach significantly reduces the implementation complexity and is used for the algorithms presented in the following sections.

III. Syndrome-Based CN Processing

For the sake of clarity we review the basic SYN CN algorithm, earlier presented in [16], before architecture specific modifications are introduced in Section IV.

The state-of-the-art way of CN processing with the FWBW scheme [17] [18] has several drawbacks. Architectures making use of the FWBW scheme suffer from low throughput and high latency. Todays approaches to solve these issues are limited to small GF(q)s with $q \leq 16$ which have only small gain in Frame Error Rate (FER) compared to their binary counterparts [14]. Only with Galois fields of high-order significantly higher communications gains can be achieved. Therefore in [16] we proposed a new algorithm, the so called SYN CN processing which can also be applied to Galois field sizes of practical interest ($q \geq 64$).

The basic structure of the SYN CN processing is depicted in Fig. 1. In the first step of the algorithm the syndromes are calculated. In contrast to the classical use of the term syndrome, we define a syndrome as the sum of one GF(q),

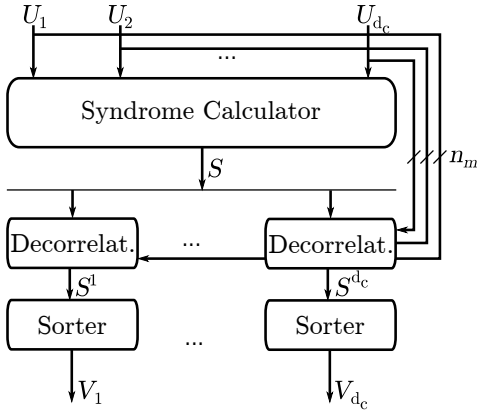

Fig. 1: Syndrome-based CN processing

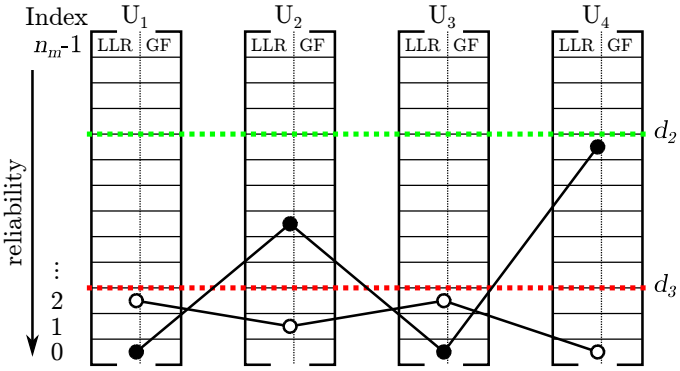

Fig. 2: Exemplary syndromes with two (filled circles) and three deviations (open circles) and maximum distances d_2 and d_3.

LLR tuple from each input set. Each input U carries n_m messages, which allows for the calculation of a set of different syndromes called S. Individual syndromes are distinguished by the input elements which are chosen for the sum. Let K_{ic} be the set of n_m most reliable field elements from the ith edge of CN c. The syndrome reliability $R(x_1 \dots x_{d_c})$ and according Galois field element $G(x_1 \dots x_{d_c})$ for CN c are calculated as follows:

$$R(x_1 \dots x_{d_c}) = \sum_{t=1}^{d_c} U_{tc}[x_t], x_t \in K_{tc} \tag{7}$$

$$G(x_1 \dots x_{d_c}) = \sum_{t=1}^{d_c} x_t, x_t \in K_{tc} \tag{8}$$

One syndrome SYN is then defined as follows:

$$\text{SYN}(x_1 \dots x_{d_c}) = \left\{R(x_1 \dots x_{d_c}), G(x_1 \dots x_{d_c})\right\} \tag{9}$$

The syndrome set S contains all valid syndromes:

$$S = \left\{\text{SYN}(x_1 \dots x_{d_c}) : \forall x_1 \dots x_{d_c} \in K_{tc}\right\} \tag{10}$$

Calculating the syndromes in S as the sum of elements over all input edges (Eq. (7) and Eq. (8)), disregards one of the basic concepts of BP algorithms: In- and output of the same edge must not be correlated. Thus an additional step in the SYN CN processing is the decorrelation of in- and output:

$$R^i(x_1 \dots x_{d_c}) = R(x_1 \dots x_{d_c}) - U_{ic}[x_i], x_i \in K_{ic} \tag{11}$$

$$G^i(x_1 \dots x_{d_c}) = G(x_1 \dots x_{d_c}) - x_i, x_i \in K_{ic} \tag{12}$$

The result is a dedicated syndrome set S^i for every output i, which has no correlation with input i.

$$\text{SYN}^i(x_1 \dots x_{d_c}) = \left\{R^i(x_1 \dots x_{d_c}), G^i(x_1 \dots x_{d_c})\right\} \tag{13}$$

$$S^i = \left\{\text{SYN}^i(x_1 \dots x_{d_c}) : \forall x_1 \dots x_{d_c} \in K_{tc}\right\} \tag{14}$$

Once the S^i sets are computed, they are sorted by their syndrome reliability, represented by the LLRs. This gives direct access to the n_m most reliable syndromes which constitute the CN output sets V_{cp}.

The algorithm we proposed is an alternative to the conventional FWBW processing. It is the first approach for high-order Galois field decoding, allowing for massive parallel implementations and thus high throughput and low latency. However, without special treatment the calculation of the syndrome set S and the sorting of S^i introduce a high complexity. It has to be reduced to make the algorithm attractive for hardware implementations.

IV. Complexity Reduction of the Syndrome-Based CN Processing

In this section we are presenting an approach to reduce the complexity of the afore introduced SYN CN algorithm. The target is to allow the algorithm to be implemented efficiently in hardware. Therefore we discuss algorithmic modifications for simplifications of the syndrome set generation and the sorting while maintaining the communications performance.

A. Reducing the syndrome set cardinality:

The first point to optimize is the calculation of the syndrome set S. For the output computation only the most reliable values of S are used which makes the computation of all other syndromes superfluous. Thus a smart reduction of the cardinality of S, $|S|$ can significantly reduce the overall complexity of the algorithm without sacrificing the communications performance.

The first step for a reduction of $|S|$ is the separation of syndromes with high reliability from ones with low reliability. In the following, a concept similar to the configuration sets introduced in [19] [20] is applied to the computation of S. Therefore we define $d_c + 1$ deviation sets D_i with $i \in 0 \dots d_c$. This procedure is just a separation of the syndrome set in sub-sets:

$$S = \bigcup_{i=0}^{d_c} D_i \,. \tag{15}$$

Each set contains only syndromes deviating in exactly i elements from the most reliable element as shown in Fig. 2. The subset D_0 contains only one syndrome, which is the sum of the most reliable elements from all inputs. These sub-sets structure the data in a way that allows for easier access to syndromes with high reliability. Figure 3 shows the average LLR values of the syndromes in the sorted deviation sets D_i. One can observe, that the distribution of reliable LLRs depends

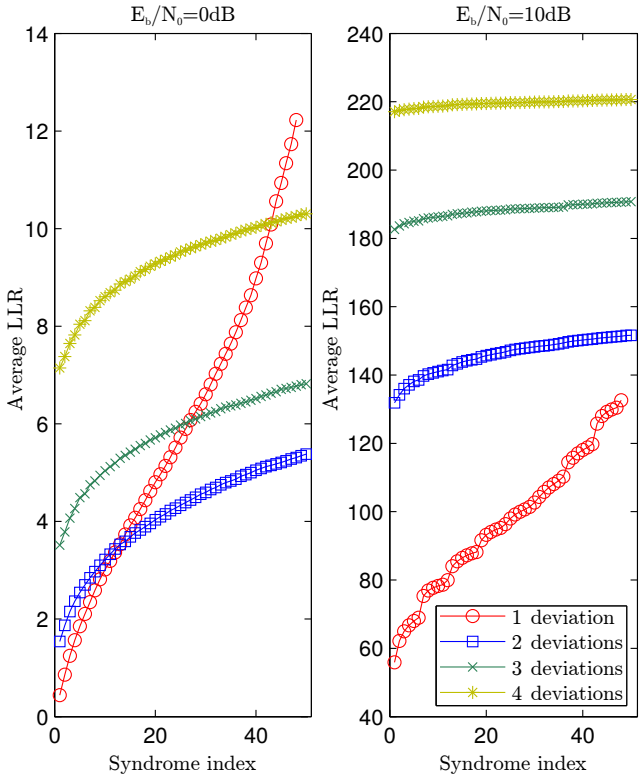

Fig. 3: Averaged LLR values of D_i for a GF(64), $d_c = 4$ code. D_i for $i > 1$ are truncated and show only the 48 most reliable syndromes.

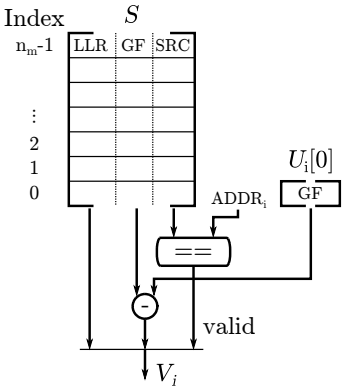

Fig. 4: Decorrelator for edge i.

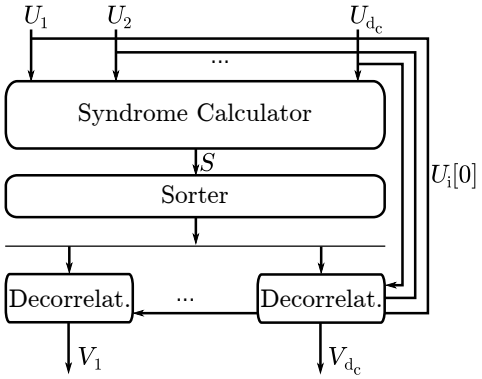

Fig. 5: Syndrome-based CN processing after algorithmic transformation

on the Signal-to-Noise Ratio (SNR). However, syndromes with more than two deviations e.g. D_i for $i > 2$ have such a low reliability that they rarely contribute to the generation of the outputs. Thus we can limit the calculation of sub-sets D_i to the ones with a low amount of deviations.

Another parameter for reduction of $|S|$ is the maximum allowed distance d_i of elements contributing to deviation D_i. The distance describes the position of the element in the input set relative to the most reliable element. The most reliable element has the index zero. Less reliable elements have higher indices which describe their rank in the sorted list of LLRs. For the calculation of D_i only elements with indices less or equal to d_i are considered. The maximum allowed distance for a certain deviation can be set dynamically based on the LLR value of the elements or it is fixed, as a predefined parameter. For each deviation a different maximum distance can be set, see Fig. 2, e.g. the higher the number of allowed deviations, the lower the maximum distance of the deviations, $d_1 \geq d_2 \geq \cdots \geq d_{d_c}$. Using this scheme implicitly keeps the best syndromes in each D_i and removes the less reliable ones. The cardinality of the subsets D_i can be calculated as follows:

$$|D_i| = \begin{cases} \binom{d_c}{i} \cdot (d_i)^i & \text{if } d_i \geq 1; i > 0 \\ 1 & \text{if } i = 0 \\ 0 & \text{else} \end{cases} \tag{16}$$

Combining both proposed techniques strictly reduces the cardinality of S and thus the computational complexity. The most reliable syndromes are calculated and only unreliable ones are removed. The parametrization for the number of deviations and their maximum distances is a critical step in the algorithm. Using for example only D_0, D_1 and D_2 with fixed distances $d_0 = 0, d_1 = n_m - 1, d_2 = 2$, $d_c = 4$ and $n_m = 13$, shrinks $|S|$ from 28561 to 73. For a code in GF(64) this is a very good trade-off between complexity and communications performance, see Section VI.

B. *Simplifying sorting:*

One big drawback of the original SYN CN algorithm presented in Section III is that every syndrome set S^i must be sorted separately to output the n_m most reliable syndromes. This is the case because of the decorrelation step applied before. To avoid the sorting of the decorrelated syndrome sets S^i, a simple but effective approach can be chosen. Instead of decorrelating every value, only syndromes using the most reliable element from the currently handled edge (LLR $= 0$) are considered. All other syndrome are not used for the current edge output. By this approach the order of the syndromes is not changed and it is sufficient to sort S instead of the d_c S^i sets. In addition, the LLR values are not modified in the decorrelation step which saves a real valued subtraction for every output message. Finally only the most reliable input element and not the complete input sets must be stored for the decorrelation. The proposed algorithmic modifications result in a slightly different data flow, see Fig. 5.

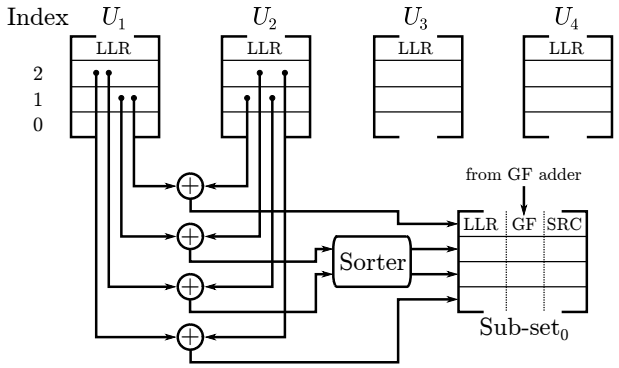

Fig. 6: Sub-set generation for D_2 with $d_2 = 2$, and SRC denoting the deviation positions.

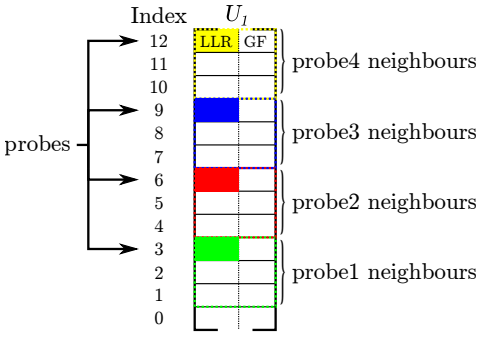

Fig. 7: Probes distribution for approximate sorting. The probes and related syndromes are highlighted in same color.

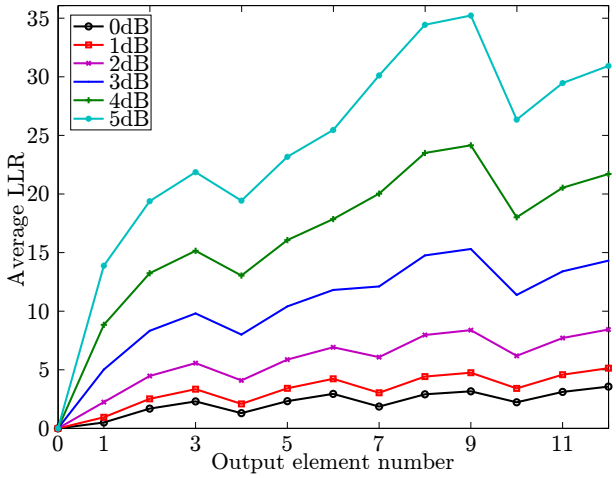

Fig. 8: Approximately sorted output LLRs for different SNR points.

Figure 4 shows the schematic operations of one decorrelator. Each syndrome is denoted with the additional information about which of the input edges contributed to the syndrome with a deviation. SRC in Fig. 4 stores the edges where deviations occurred and ADDR$_i$ represents the current edge. A simple comparison evaluates if a deviation from the current edge was involved in the syndrome calculation and thus if the syndrome is valid for the current edge or not. Only if no deviation occurred on the current edge, the decorrelated message is marked as valid and used for the output V_i.

Even though the sorting has been reduced to the syndrome set S, there is more potential for simplification. Sorting S can be divided into sorting the deviation sets D_i and merging them. Especially for D_1 the sorting can be further simplified. This is achieved due to the previous knowledge we have of the input data. We implicitly know that the sets U_{pc} are sorted according to their LLRs. The sorting of D_1 thus is limited to merging d_c sorted sets. For the higher-order deviations D_i for $i \geq 2$, the sorting can also be simplified because of the sorted input sets. Sorted sub-sets are generated with little effort which only have to be merged to achieve the final set. An example of the sub-set generation for D_2 with $d_2 = 2$ is given in Fig. 6 which can be extended easily to other deviations and distances. Once the sub-sets are sorted, the outputs can be generated by merging them iteratively. However, considering the NB-LDPC decoder as a whole, it can be observed, that an exact sorting of the CN outputs is not required. When the VN has calculated the a posteriori probability (APP) messages as the sum of the channel values and messages from the CNs (Eq. (2)), they have to be resorted anyway. Thus an approximately sorted CN output is sufficient and does not impair the decoder's communications performance. In the following we present a scheme which uses the robustness against approximately sorted CN outputs to further reduce the algorithms complexity.

To allow for this approximate sorting, so called probes are chosen and sorted according to their LLR. Figure 7 shows the distribution of probes within the input set U_1 used for D_1. In this paper we assume that the probes are equally distributed in the input sets which works well for the investigated configuration. The considered probe positions are 3, 6, 9 and 12. However, for other parameters of d_c and n_m other distributions are beneficial. Each probe is connected to a set of neighbouring syndromes, seee Fig. 7. The probes LLR value is considered to be representative for the whole group of syndromes. In the following, instead of iteratively merging the sub-sets syndrome by syndrome, the probes and their neighbouring syndromes are merged on set basis. This means that three neighbouring syndromes are chosen at a time, based on the reliability of the according probe. Only the LLR value of the probe has an impact on the sorting, the LLRs of the neighbouring syndromes are not considered. This scheme leads to some uncertainty in the merged set but is close enough to the exact solution not to degrade the decoders communications performance. Figure 8 shows the LLR values of the approximately sorted syndromes used for the CN output calculation. The approximation works well for low LLRs (high reliability) and only for the less reliable outputs there is a difference compared to the optimal sorting results.

Summarized, three notable benefits arise from the algorithmic transformations introduced in this section:

- Significant reduction of $|S|$.
- Approximated sorting of S instead of exact sorting of S^i.
- No LLR subtractions and no storage for U_i in the decorrelation step.

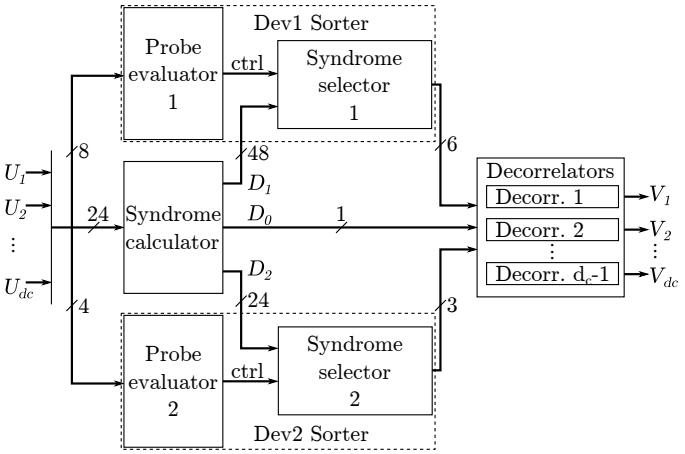

Fig. 9: Top-level architecture of the SYN CN.

V. Architecture

In this section we present a hardware implementation of the algorithm proposed before. The architecture is independent of the actual used NB-LDPC code, only the parameters $d_c = 4$, $q = 64$ and $n_m = 13$ are given. As we are working with truncated sets, we have to redefine $U_i(j), j \in 0...n_m - 1$ as the jth GF(q), LLR tuple from edge i. We use the LLR($\dots$) and GF($\dots$) notation to distinguish between the GF(q) and LLR part of one of these tuples respectively.

Figure 9 shows an overview of the CN hardware. To achieve low latency and high throughput, the input parallelism of the CN was chosen to be six GF(q), LLR tuples and an additional GF(q) input for the most reliable element. All input messages can thus be read within two clock cycles. The evaluation of the probes can be processed in parallel with the actual calculation of the syndrome set. The selection of the sub-sets is performed in dedicated units, the syndrome selectors, based on the result of the probe evaluators. Once S is calculated and the probes are evaluated, the most reliable sub-sets are used for the decorrelation. The parallelism with which the syndromes are process has significant impact on the overall throughput. It has been chosen to be three times three syndromes. In each clock cycle two sets of neighbouring syndromes from D_1 and one set from D_2, overall nine syndromes are processed. Thus, after a maximum of four clock cycles all output edges are filled with n_m valid messages. The output parallelism of the CN is chosen symmetrically to the inputs to be six GF(q), LLR tuples and one GF(q) message for D_0. The design is constructed from four main building blocks which will be explained in detail in the following sections.

A. Probe Evaluator

The probe evaluator for D_1 processes two probes per edge in each clock cycle. For $d_c = 4$ overall eight LLRs have to be sorted, which is performed by a latency optimized sorting network. As each two messages are implicitly sorted already because they belong to the same edge, some simplifications on the network can be applied. The final sorting network is depicted in Fig. 10. The result of the sorting is not a sorted list of LLRs but rather the positions of the inputs where they come from. They are stored in a register and the same task is

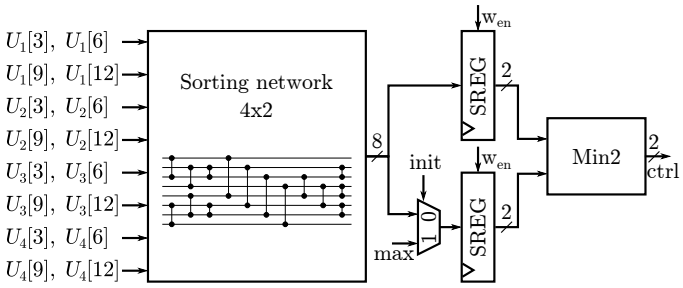

Fig. 10: Probe evaluator for D_1.

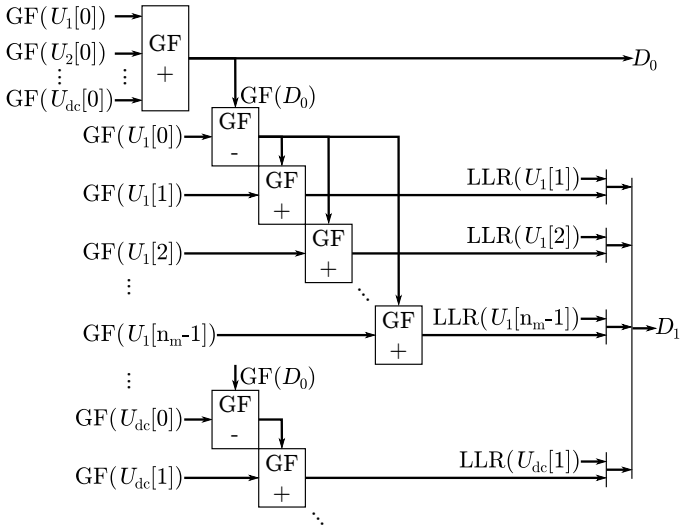

Fig. 11: Syndrome calculator for D_0 and D_1.

performed a second time for the second half of the input LLRs in the next clock cycle. Starting from the second clock cycle, every following clock cycle the positions of the two smallest probes are output. To perform this task, an additional sorter, selecting the two smallest probes from the registers is utilized. Once a probe is used for an output generation, it is removed by shifting the register content accordingly. The probe evaluator for D_2 is a simplified version of the one for D_1 as it considers only four inputs. Moreover it needs to generate only one output per clock cycle. The output of the probe evaluators is used as control signal for the syndrome selector components.

B. Syndrome Calculator

The syndrome calculation is carried out fully parallel. Due to the restrictions on the deviation distances ($d_0 = 0$, $d_1 = n_m - 1$, $d_2 = 2$), the required hardware is strictly limited. A sophisticated scheme for the calculation allows for further reduced hardware cost, see Fig. 11. Instead of calculating each syndrome as a sum of d_c inputs, intermediate results are used to minimize the number of explicit calculations. The calculation of D_0 involves $d_c - 1$ GF(q) additions, for D_1 overall $d_c \cdot n_m$ GF(q) additions are required. Thus for D_0 and D_1 only GF(q) calculations are required. For D_2 the processing scheme of D_1 can easily be extended and requires only one real valued addition per syndrome in addition to the GF(q) operations. Compared with state-of-the-art processing,

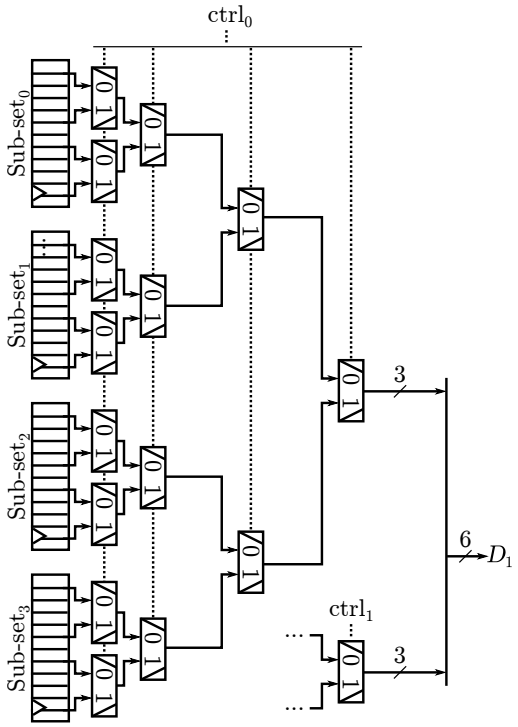

Fig. 12: Syndrome selector for D_1.

there is a significant saving in computational complexity. The output of the syndrome calculator is a number of sorted sub-sets used as input for the syndrome selectors.

C. Syndrome Selector

The syndrome selector receives the addresses of the most reliable syndrome sets from the probe evaluator and the syndromes D_i sorted in sub-sets from the syndrome generator. The complete hardware consists of a multiplexer tree choosing a set of the three most reliable syndromes from the sub-sets. To achieve six messages from D_1, the multiplexer tree is duplicated in this unit, see Fig. 12. For D_2 a single tree choosing between the six sub-sets of D_2 suffices.

D. Decorrelator

Decorrelation has to be performed individually for each edge of the CN. The output parallelism of the decorrelator is six messages per clock cycle. Two times three syndromes from D_1 and another three from D_2 are processed per clock cycle. By construction the messages of a set always have deviations on the same edges. Thus it is sufficient to check for one of the messages in a set if it is valid or not, which is indicated with a valid flag. If only a part of the received sets is valid, they are rearranged by multiplexers in such a way, that only valid messages are used for the output. In the best case all syndromes received in one clock cycles are valid. As the output parallelism is only six, the surplus syndromes are stored in an additional register and reused in the next clock cycle. Before the messages are sent to the VN, the actual decorrelation is applied which is a subtraction of the most reliable GF(q) value of the current edge.

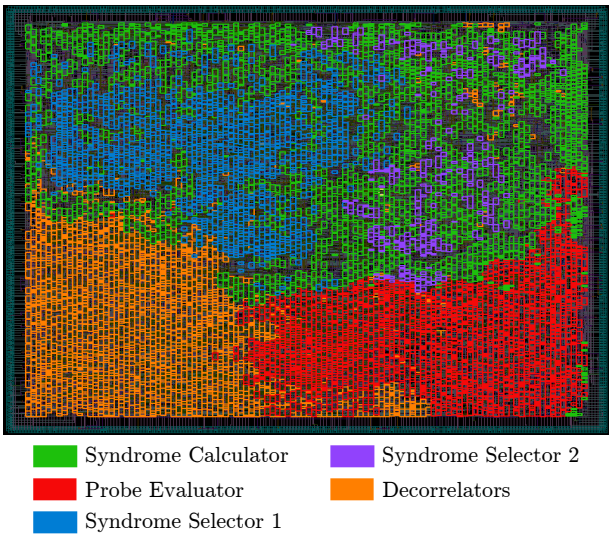

Fig. 13: SYN CN chip layout.

VI. Results

The architectures presented in this section are based on a code which has a code word length N of 16 GF(64) symbols (96 bits), a Code Rate (CR) of 0.5, $d_v = 2$ and $d_c = 4$. Code generation has been performed based on the approach presented in [21]. The SYN CN makes use of truncated vectors of size $n_m = 13$. The architecture is implemented on a 65nm low power bulk CMOS library from ST Microelectronics. We considered the following PVT parameters: Worst Case (WC, 1.1V, 125°C), Nominal Case (NOM, 1.2V, 25°C) and Best Case (BC, 1.3V, -40°C). Synthesis was performed with Synopsys Design Compiler in topographical mode, Placement & Routing (P&R) with Synopsys IC Compiler. Synthesis as well as P&R were performed with the Worst Case PVT settings of the 65nm library.

The final design reaches a Worst Case frequency of 400 MHz. After P&R approximately 0.07 mm^2 are required for the core. An area utilization (cell/core area ratio) of 84 % is achieved. We also evaluated the power consumption of the physical design using Synopsys PrimeTime-PX. The average power consumption at a clock frequency of 400 MHz is 30 mW for the Nominal Case. The physical layout of the CN can be seen in Fig. 13. Compared to a state-of-the-art CN based on the FWBW architecture presented in [18] and implemented in the same technology, the SYN CN achieves a 14 times higher area efficiency and 2.5 times higher energy efficiency. At the same time the latency (processing time for one CN update) is reduced from 21 clock cycles to only 6 clock cycles. A comparison of achieved communications performance shows the difference with respect to the state-of-the-art hardware aware FWBW decoding, see Fig. 14. The syndrome based CN computation has a superior performance compared to the FWBW implementation.

Our next target is the design of a high throughput decoder based on the presented CN. Future challenges are the extension of the presented design to higher edge degrees and even higher-order Galois fields ($q \geq 256$).

TABLE I: State-of-the-art NB-LDPC CN comparison, with parameters $d_c = 4$, $q = 64$ and $n_m = 13$.

Decoder	SYN CN	FWBW CN
CMOS Technology	65nm SVT	65nm SVT
Supply Voltage [V]	1.2	1.2
Frequency [MHz]	400	250
Quantization	8 bit	8 bit
Post P&R Area [mm^2]	0.067	0.112
Area Utilization	84%	70%
Throughput [Mel/s][a]	5200	620
Energy Eff. [pJ/el]	**5.7**	13.8
Area Eff. [Gel/s/mm^2]	**77.5**	5.5
Decoding Latency [CC][b]	**6**	21

[a]el = elements
[b]CC = Clock Cycles

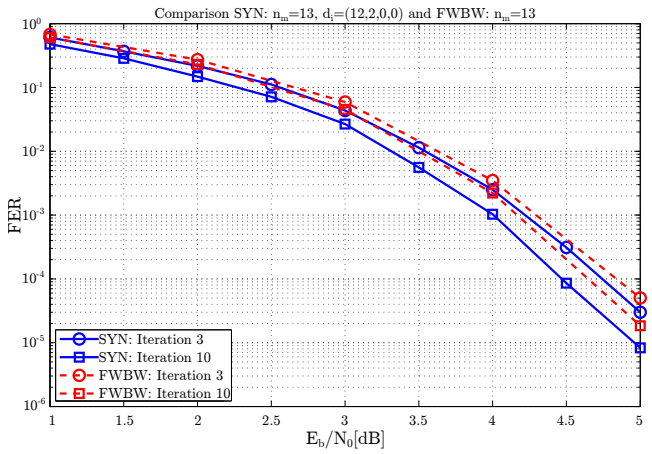

Fig. 14: Communications performance of the SYN CN compared to the state-of-the-art FWBW scheme.

VII. Conclusion

We have presented a new algorithm and architecture for the CN processing of NB-LDPC decoders. The SYN CN algorithm is the first hardware aware CN processing, allowing for low latency and high throughput decoder architectures using high-order Galois fields. A hardware implementation is presented which underlines the potential of the proposed algorithm. A 14 times higher area efficiency and 2.5 times higher energy efficiency is achieved compared to a state-of-the-art CN architecture while the latency is reduced by factor 3.5. This is a big step into the direction of efficient NB-LDPC decoders with sufficient throughput for future applications.

References

[1] G. Fettweis, "The Tactile Internet: Applications and Challenges," *IEEE Vehicular Technology Magazine*, vol. 9, no. 1, pp. 64–70, 2014. [Online]. Available: http://ieeexplore.ieee.org/stamp/stamp.jsp?arnumber=6755599

[2] R. G. Gallager, "Low-Density Parity-Check Codes," *IRE Transactions on Information Theory*, vol. 8, no. 1, pp. 21–28, Jan. 1962.

[3] D. MacKay and R. Neal, "Near Shannon limit performance of Low-Density Parity-Check Codes," *Electronic Letters*, vol. 32, pp. 1645–1646, 1996.

[4] S. Pfletschinger and D. Declercq, "Getting Closer to MIMO Capacity with Non-Binary Codes and Spatial Multiplexing," in *Global Telecommunications Conference (GLOBECOM 2010), 2010 IEEE*, 2010, pp. 1–5. [Online]. Available: http://ieeexplore.ieee.org/stamp/stamp.jsp?arnumber=5684077

[5] M. Davey and D. MacKay, "Low-Density Parity Check Codes over GF(q)," *IEEE Communications Letters*, vol. 2, no. 6, pp. 165–167, Jun. 1998.

[6] L. Barnault and D. Declercq, "Fast decoding algorithm for LDPC over GF(2q)," in *Proc. IEEE Information Theory Workshop*, Mar.–Apr. 2003, pp. 70–73.

[7] T. Lehnigk-Emden and N. Wehn, "Complexity Evaluation of Non-binary Galois Field LDPC Code Decoders," in *Proc. 6th International Symposium on Turbo Codes & Iterative Information Processing (ISTC'2010)*, Brest, France, Sep. 2010.

[8] Y. Lu, N. Qiu, Z. Chen, and S. Goto, "An efficient majority-logic based message-passing algorithm for non-binary LDPC decoding," in *Circuits and Systems (APCCAS), 2012 IEEE Asia Pacific Conference on*, 2012, pp. 479–482. [Online]. Available: http://ieeexplore.ieee.org/stamp/stamp.jsp?arnumber=6419076

[9] F. Garcia-Herrero, D. Declercq, and J. Valls, "Non-Binary LDPC Decoder Based on Symbol Flipping with Multiple Votes," *IEEE Communications Letters*, vol. 18, no. 5, pp. 749–752, 2014. [Online]. Available: http://ieeexplore.ieee.org/stamp/stamp.jsp?arnumber=6787156

[10] A. Ciobanu, S. Hemati, and W. Gross, "Adaptive Multiset Stochastic Decoding of Non-Binary LDPC Codes," *IEEE Transactions on Signal Processing*, vol. 61, no. 16, pp. 4100–4113, 2013. [Online]. Available: http://ieeexplore.ieee.org/stamp/stamp.jsp?arnumber=6519310

[11] C.-W. Yang, X.-R. Lee, C.-L. Chen, H.-C. Chang, and C.-Y. Lee, "Area-efficient TFM-based stochastic decoder design for non-binary LDPC codes," in *Circuits and Systems (ISCAS), 2014 IEEE International Symposium on*, 2014, pp. 409–412. [Online]. Available: http://ieeexplore.ieee.org/stamp/stamp.jsp?arnumber=6865152

[12] D. Declercq and M. Fossorier, "Decoding Algorithms for Nonbinary LDPC Codes Over GF," *IEEE Transactions on Communications*, vol. 55, no. 4, pp. 633–643, 2007. [Online]. Available: http://ieeexplore.ieee.org/stamp/stamp.jsp?arnumber=4155118

[13] A. Voicila, D. Declercq, F. Verdier, M. Fossorier, and P. Urard, "Low-complexity decoding for non-binary LDPC codes in high order fields," *IEEE Transactions on Communications*, vol. 58, no. 5, pp. 1365–1375, 2010. [Online]. Available: http://ieeexplore.ieee.org/stamp/stamp.jsp?arnumber=5464236

[14] E. Li, D. Declercq, and K. Gunnam, "Trellis-Based Extended Min-Sum Algorithm for Non-Binary LDPC Codes and its Hardware Structure," *IEEE Transactions on Communications*, vol. 61, no. 7, pp. 2600–2611, 2013. [Online]. Available: http://ieeexplore.ieee.org/stamp/stamp.jsp?arnumber=6516166

[15] H. Wymeersch, H. Steendam, and M. Moeneclaey, "Log-domain decoding of LDPC codes over GF(q)," in *Proc. IEEE International Conference on Communications*, vol. 2, Jun. 2004, pp. 772–776.

[16] P. Schläfer, N. Wehn, M. Alles, T. Lehnigk-Emden, and E. Boutillon, "Syndrome Based Check Node Processing of High Order NB-LDPC Decoders," in *International Conference on Telecommunications (ICT2015)*, submitted for publication.

[17] E. Boutillon and L. Conde-Canencia, "Bubble check: a simplified algorithm for elementary check node processing in extended min-sum non-binary LDPC decoders," *Electronics Letters*, vol. 46, no. 9, pp. 633–634, 2010. [Online]. Available: http://ieeexplore.ieee.org/stamp/stamp.jsp?arnumber=5457396

[18] E. Boutillon, L. Conde-Canencia, and A. Al Ghouwayel, "Design of a GF(64)-LDPC Decoder Based on the EMS Algorithm," *IEEE Transactions on Circuits and Systems I: Regular Papers*, vol. 60, no. 10, pp. 2644–2656, 2013. [Online]. Available: http://ieeexplore.ieee.org/stamp/stamp.jsp?arnumber=6595153

[19] D. Declercq and M. Fossorier, "Extended minsum algorithm for decoding LDPC codes over GF(q)," in *Proc. International Symposium on Information Theory ISIT 2005*, Sep. 2005, pp. 464–468.

[20] E. Li, K. Gunnam, and D. Declercq, "Trellis based Extended Min-Sum for decoding nonbinary LDPC codes," in *Wireless Communication Systems (ISWCS), 2011 8th International Symposium on*, 2011, pp. 46–50. [Online]. Available: http://ieeexplore.ieee.org/stamp/stamp.jsp?arnumber=6125307

[21] C. Poulliat, M. Fossorier, and D. Declercq, "Design of regular ($2,d_c$)-LDPC codes over GF(q) using their binary images," *IEEE Transactions on Communications*, vol. 56, no. 10, pp. 1626–1635, 2008.