Conflict Resolution by Matrix Reordering for DVB-T2 LDPC Decoders

Cédric Marchand*†, Jean-Baptiste Doré*, Laura Conde-Canencia†, Emmanuel Boutillon†

*NXP Semiconductors, Campus Effisience, Colombelles BP20000 14906 Caen cedex 9, France.

†Université Européenne de Bretagne, Lab-STICC, CNRS, UBS, BP 92116 56321 Lorient, France.

Email: cedric.marchand@univ-ubs.fr

Abstract—**Layered decoding is known to provide efficient and high-throughput implementation of LDPC decoders. However, the implementation of the layered architecture is not always straightforward because of the memory access conflicts in the a-posteriori information memory. In this paper, we focus our attention on a particular type of conflict introduced by the existence of multiple diagonal matrices in the DVB-T2 parity check matrix structure. We illustrate how the reordering of the matrix reduces the number of conflicts, at the cost of limiting the level of parallelism. We then propose a parity extending process to solve the remaining conflicts. Fixed point simulation results show coherent performance without modifying the layered architecture.**

Index Terms—**Low-Density Parity-Check (LDPC) code, memory conflict, scheduling, VLSI implementation, layered decoder, DVB-T2.**

I. INTRODUCTION

Low Density Parity-Check (LDPC) codes [1] have gained a lot of attention due to their remarkable error correcting capabilities. Among all the published work on LDPC, the approach introduced in [2] led to the conception of structured codes which are now included in standards such as DVB-S2 and DVB-T2 [3] for digital video broadcasting, Wireless Local Area Networks (WiFi) (IEEE 802.11n), Wireless Metropolitan Area Networks(WiMAX) (802.16e)[4] and Wireless Regional Area Networks(WRAM) (IEEE 802.22) for wireless networks. These structured codes or architecture-aware codes (AA-LDPC [5]) can be efficiently implemented using a semi-parallel architecture [6], [7], [8], block-serial architecture [9], [10], [11], or layered decoder architecture [12], [13], [14].

Beside the implementation efficiency, the turbo message passing, introduced by Mansour [5], [15] and then referred to as layered decoding by Hocevar [14], decreases by two the number of iterations required to decode a code word compared to the traditional flooding schedule. Furthermore the use of a Soft-Output (SO) based Check Node Processor (CNP) [9], [12], [13], [14], [16] presents the advantage of a significant memory reduction.

Although DVB-S2 and DVB-T2 standards define structured parity check matrices, these matrices are not perfectly structured for layered decoder architecture, leading to conflicts in the SO memories. Throughput, silicon area and memory conflicts are bottlenecks that make the implementation of these standards a challenge. In [13] and [16] the authors present a solution to avoid conflicts based on the computation of the

variation (or delta) of the SO metrics to allow concurrent updates. The computation of this SO update needs either a costly memory access or an increase of the clock frequency by a factor of two.

In this paper, we use a layered decoder and we propose a solution based on the 'divide and conquer' strategy to overcome the memory conflicts. A reordering mechanism of the matrix called split algorithm creates a new structured matrix that reduces the parallelism, however with significant decreases in the number of conflicts. The remaining conflicts are avoided by an equivalent matrix using added punctured bits.

This paper is organized as follows: section II presents the layered decoder and the arising conflicts when using DVB-T2 matrices. In section III, we explain the splitting process which reduces the number of conflicts and we present results. In section IV, the remaining conflicts are removed by matrix transformation. Finally, an implementation overview is illustrated in section V.

II. MEMORY CONFLICTS IN THE DVB-T2 LDPC LAYERED DECODER

A LDPC decoder is defined by its parity check matrix $\mathbf{H}$ of M rows by N columns. Each column in $\mathbf{H}$ is associated with one bit of the codeword or Variable Node (VN), and each row corresponds to a parity check equation or Check Node (CN). A nonzero element in a row means that the corresponding bit contributes to this parity check equation. Fig. 1 shows the structure of the rate-$2/3$ short-frame DVB-T2 LDPC parity check matrix. The matrix is structured with shifted identity matrices showing the link between VN_n and CN_m. This structure is efficient for highly parallel decoders. In this section, after an overview of the layered decoder principle, we will focus on the DVB-T2 LDPC matrices in order to explain the arising memory conflicts.

A. Horizontal layered decoder

In the horizontal layered decoding algorithm, a VN is represented by a SO value (SO_v). This value is first initialized by the Channel Log Likelihood Ratio ($LLR = log(P(v = 0)/P(v = 1))$). Then the decoding proceeds iteratively until all the parity checks are verified or a maximum number of iterations is reached. For layered decoding, one iteration is split into sub-iterations, one for each layer. A layer can be
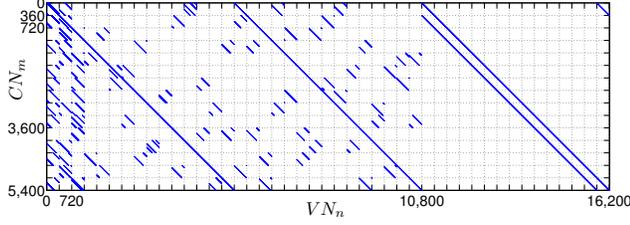
Fig. 1. Overall view of block-structured DVB-T2 Matrix of a rate-2/3 with N=16200

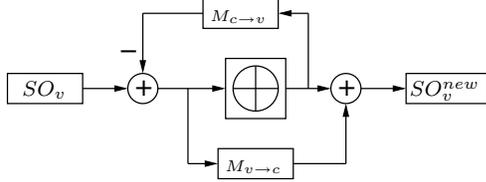Fig. 3. Zoom of a rate-$^2/_3$ DVB-T2 Matrix with N=16200

Fig. 2. SO based CNP

made of one or several CNs and the sub-iteration consists in updating all the VNs connected to the CNs of the layer. The update of the VNs connected to one CN is done serially in three steps. First, the message from VN to CN ($M_{v \to c}$) is calculated using the equation (1).

$$M_{v \to c} = SO_v - M_{c \to v} \qquad (1)$$

The second step is the serial $M_{c \to v}$ update, where $M_{c \to v}$ is a message from CN to VN. Let v_c be the set of all the VNs connected to the CN c and v_c/v be v_c without v. For implementation convenience, the sign (2) and the absolute value (3) of the messages are updated separately

$$sign(M_{c \to v}^{new}) = \prod_{v' \in v_c/v} sign(M_{v' \to c}) \qquad (2)$$

$$|M_{c \to v}^{new}| = f\left(\sum_{v' \in v_c/v} f(|M_{v' \to c}|) \right) \qquad (3)$$

where $f(x) = \ln \tanh\left(\frac{x}{2}\right)$. Equation (3) can be implemented using a sub-optimal algorithm such as the Min-Sum algorithm [17], the normalized Min-Sum algorithm or the λ-min algorithm [18].

The third step is the calculation of the SO_{new} value using (4). The updated SO_{new} value can be used in the same iteration by another sub-iteration leading to a two times faster convergence, compared to the flooding schedule.

$$SO_v^{new} = M_{v \to c} + M_{c \to v}^{new} \qquad (4)$$

From these equations, the CNP architecture in Fig. 2 can be derived. The left adder of the architecture performs equation (1) and the right adder performs equation (4). The central part is in charge of the serial $M_{c \to v}$ update.

Several CN may be grouped together to form a layer, whenever the column weights in the layer does not exceed

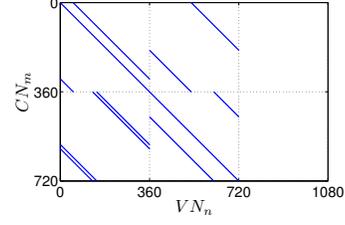one. The structured matrices made of identity matrices of size P allow us to compute layers made of CN Groups (CG) of P CNs. The layered decoder architecture is mainly based on P CNPs that first read serially the Groups of P VNs (VGs) linked to one CG and then the P CNPs write back the result to the VGs in the same order. The described architecture is efficient for structured codes. However, memory conflicts arise when using this architecture for DVB-T2 matrices.

B. Conflicts due to the DVB-T2 matrix structure

Fig. 3 shows a zoom in view on the first 720 VNs and CNs of the DVB-T2 LDPC matrix illustrated in Fig. 1. We can see that the first group of 360 CNs is linked twice to the first group of 360 VNs by two diagonals. The sub-matrix with a double diagonal in it will be called Double Diagonal Sub Matrix (DDSM).

Let us consider the case where two CNs are computed in one layer and connected to the same VN. There are two updates of the same SO value. The calculation of the new SO (5) is deducted from equation (1) and (4). Assuming $\Delta M_{c_1 \to v} = M_{c_1 \to v}^{old} + M_{c_1 \to v}^{new}$ and using (5), we obtain the calculation of SO_v^{new1} and SO_v^{new2} in (6) and (7), respectively.

$$SO_v^{new} = SO_v^{old} - M_{c \to v}^{old} + M_{c \to v}^{new} \qquad (5)$$

$$SO_v^{new_1} = SO_v^{old} + \Delta M_{c_1 \to v} \qquad (6)$$

$$SO_v^{new_2} = SO_v^{old} + \Delta M_{c_2 \to v} \qquad (7)$$

Because the SO is updated serially in the layered architecture, the SO_v^{new2} will overwrite the SO_v^{new1} value. This conflict is equivalent to cut the $M_{c_1 \to v_1}$ message. This is usually called a cutting edge and will lead to performance degradation. Each DDSM will produce P cutting edges and thus has to be avoided in the structure of the matrix.

As an example, Fig. 4 illustrates the number of DDSMs in the rate-2/3 short-frame matrix. vg_i denotes the ith group of 360 VNs and cg_j denotes the jth group of 360 CNs . A square denotes a permuted identity matrix linking cg_j and vg_i. In this base matrix representation, each gray square corresponds to a DDSM. Note that there are 14 DDSMs in this example. In the next section we explain how to reduce the number of DDSMs.
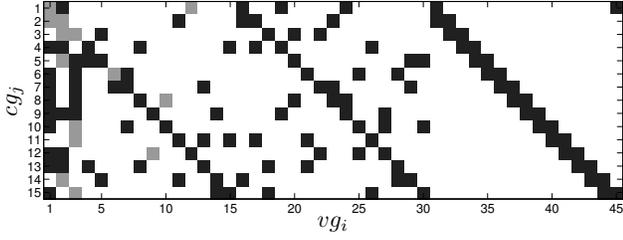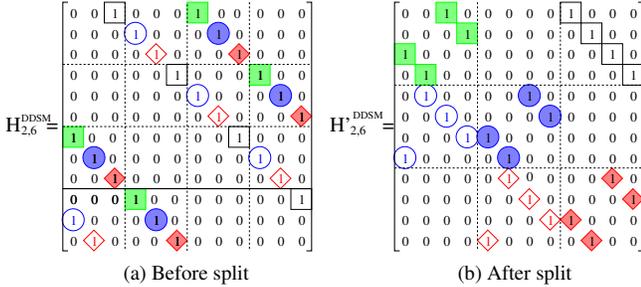
Fig. 4. Base DVB-T2 matrix representation

Fig. 6. Split base DVB-T2 matrix representation

(a) Before split   (b) After split

Fig. 5. Shifted identity matrix

III. CONFLICT RESOLUTION BY GROUP SPLITTING

To achieve the minimum required throughput of 90Mbps in the DVB-T2 standard, parallel processing of a fraction of the 360 CNs is enough [8] [9]. In [9], the authors have used 45 CNPs which lead to significant area reduction, therefore splitting the group of 360 CNs is considered. In [8] and [9], the splitting process has already been done implicitly through memory mapping. In the next subsection, we will show how to reorder the structured matrices initially designed for a parallelism of 360 to matrices designed for a parallelism of $360/S$, where S is the number of splits.

A. Construction of the sub-matrices

Let us define P_s as the number of CN working in parallel after a split. P, P_s and S are then linked by the equation:

$$S \times P_s = P$$

The construction process of the new matrix relies on the permutation of the rows and the columns defined as:

$$\sigma(i) = (i \mod S)P_s + \lfloor i/S \rfloor \tag{8}$$

where $\lfloor x \rfloor$ is the largest integer not greater than x. We first reorder the CN using (8) where i is the CN number. Then we reorder the VN in the same way.

B. Example

Let us consider the following example of a double diagonal sub matrix $\mathbf{H}_{2,6}^{\mathbf{DDSM}}$ of size $P = 12$ in Fig. 5(a), where the values in subscript are shift parameters of the two diagonals i.e.: first diagonal is shifted by $\delta_1 = 2$ and the second diagonal is shifted by $\delta_2 = 6$. This double diagonals matrix will produces 12 cutting edges. After reordering the rows and
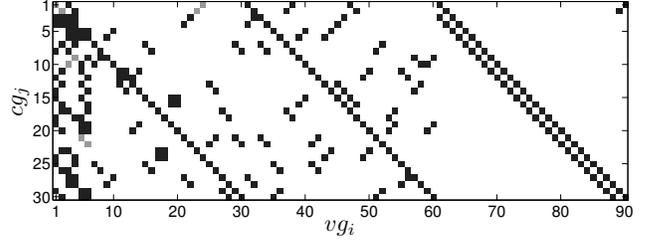
columns using equation (8) with $S = 3$ and $P_s = 4$, we obtain the new matrix $\mathbf{H}_{2,6}^{\prime\mathbf{DDSM}}$ in Fig. 5(b). Note that $\mathbf{H}_{2,6}^{\prime\mathbf{DDSM}}$ is composed of shifted identity matrices of size P_s and can be best described by its base matrix

$$\mathbf{H}_{\mathbf{base}}^{\prime\mathbf{DDSM}} = \begin{bmatrix} 2 & -1 & 0 \\ 1 & 2 & -1 \\ -1 & 1 & 2 \end{bmatrix}$$

where -1 is a null sub matrix. There is a convenient way to build the new base matrix layer by layer, using the shift value of the diagonal i before split δ_i^{old}. If we focus on the first layer of the new DDSM base matrix, the position of the sub shifted identity matrix is given by $vg_i = \delta_i^{old} \mod S$ and the shift value is given by $\delta_i^{new} = \lfloor \delta_i^{old}/S \rfloor$. Considering our example, the first diagonal shift $\delta_1^{old} = 2$ gives $vg_1 = 2$ and $\delta_1^{new} = 0$, and the second diagonal $\delta_2^{old} = 6$ gives $vg_2 = 0$ and $\delta_2^{new} = 2$. The next layer $l + 1$ is a copy of the previous layer with vg_i^l increased by one. However if $vg_i^l = S - 1$ then $vg^{l+1} = 0$ and $\delta_i^{l+1} = \delta_i^l + 1$. It is important to mention that the splitting of a DDSM does not always remove the double diagonals. If $(\delta_2 - \delta_1) \mod S = 0$ then $vg_1 = vg_2$ and there will remain DDSM in the sub-matrices.

C. DDSM in DVB T2 and simulation results

The rate-2/3 base DVB-T2 matrix (Fig. 4) is split by a factor of two and the obtained matrix is shown in Fig. 6. It can be observed that after the split, the number of grey squares are reduced from 14 to 8. In terms of cutting edges, this means a reduction from 14×360 cutting edges to 8×180. Tables I and II provides the equivalent number of DDSMs of size 360 as a function of the parallelism and the coding rate. An asterisk (*) in the table means that there are triple identity matrices among the counted DDSMs. Table I provides results for short frames and Table II provides results for long frames DVB-T2 LDPC codes. Significant reduction of the number of cutting edges can be observed by the proposed group splitting method.

Fig. 7 gives simulation results for a normalized Min-Sum fixed point layered decoder, with 30 iterations for short frames at a code rate of $2/3$ in Additive White Gaussian Noise (AWGN) channel. We simulate an architecture where the channel value is quantified on 5 bits, the SO on 7 bits and the normalization factor is 0.75. The curve denoted 'p45 (0 cuts)' shows the error performance with a parallelism of 45

		Short Frame						
S	P_s	1/4	1/2	3/5	2/3	3/4	4/5	5/6
1	360	4	8	0	14	9	9	20*
2	180	1	2	0	4	5	8	13*
3	120	1	1	0	3	3	2	11
4	90	0	1	0	2	2	7	5*
5	72	1	1	0	5	1	1	1
6	60	0	0	0	1	1	2	6
8	45	0	1	0	0	2	2	4*
9	40	1	0	0	1	2	0	3
10	36	0	0	0	1	0	1	1

TABLE I

NUMBER OF DDSM FOR SHORT FRAMES

		Long Frame					
S	P_s	1/2	3/5	2/3	3/4	4/5	5/6
1	360	8	32*	12	23*	31*	35*
2	180	4	19	5	10	13	21
3	120	2	16	4	8	15	12
4	90	2	8	2	3	6	13
5	72	0	8	2	3	9	11
6	60	1	6	1	3	5	3
8	45	0	2	0	3	3	5
9	40	2	4	1	3	4	2
10	36	0	4	1	2	2	5

TABLE II

NUMBER OF DDSM FOR LONG FRAMES

Fig. 8. Principle of a parity check matrix equivalent

which will produce no cutting edges. The parallelism of 60 and 90 (represented by 'p60' and 'p90' respectively) result in a performance loss of 1dB from the reference ('p45'). The significant performance loss motivates us to find a solution for the remaining DDSMs in the next section.

IV. PARITY CHECK MATRIX EQUIVALENT

In [19] a method to transform a parity check matrix by the introduction of VN of degree two is presented. With the help of this method, we build an equivalent matrix without DDSMs.
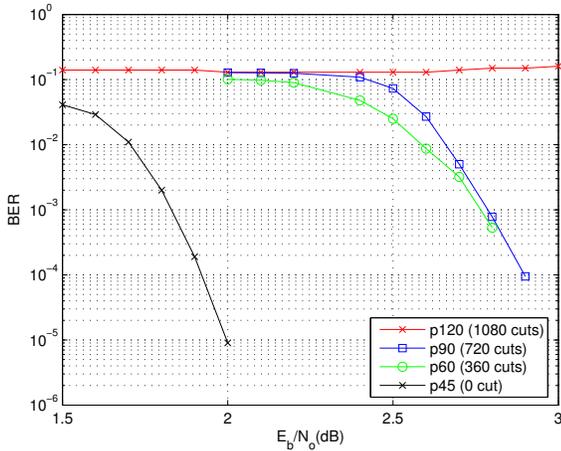
A. Principle

Taking into consideration one parity equation (9), it can be split into two equations (10) and (11) using a dummy VN p_0.

$$v_1 + v_2 + v_3 + v_4 + v_5 = 0 \bmod 2 \tag{9}$$

$$v_1 + v_2 + v_4 + v_5 + p_0 = 0 \bmod 2 \tag{10}$$

$$v_3 + p_0 = 0 \bmod 2 \tag{11}$$

Fig. 8(a) shows a layer with one DDSM build using equation (9). This layer is split into two layers without DDSM Fig. 8(b). For encoding, the matrix is theoretically extended and then the added VNs are punctured wich give back the original matrix. The encoding process remains unchanged. However, during the decoding process added VNs(dummy VN) are initialized with LLR values of 0. By using the BP algorithm, a flooding schedule and enough iterations, the performance of the extended matrix is equivalent to the original one. The equivalent matrix is less effective for a quantified layered decoder: The dummy VNs of degree two are used only for the communication between the split CNs. The messages going through the punctured VN are updated just one time during one iteration. This means one iteration delay is needed for the message to be sent from one split CN and received by the other. The next subsection presents the results of simulation on a fixed point normalized decoder.

B. Results

Fig. 9 shows a simulation result keeping the same conditions as in Fig. 7 but utilizing extended matrices. At a BER of 10^{-7}, we can observe that every added DDSM gives a performance loss of $0.1dB$ from the reference without conflict.

Fig. 10 is a simulation for short frame at parallelism of 40 which is the minimum required parallelism for a 200 MHz pipelined layered decoder to reach the expected 90Mbps

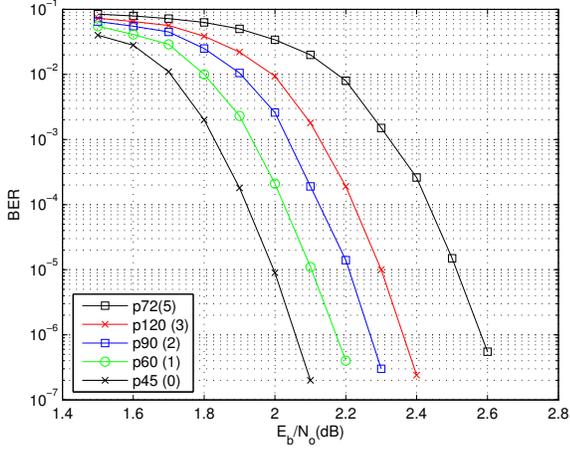Fig. 7. BER as a function of the parallelism on a fixed point simulation

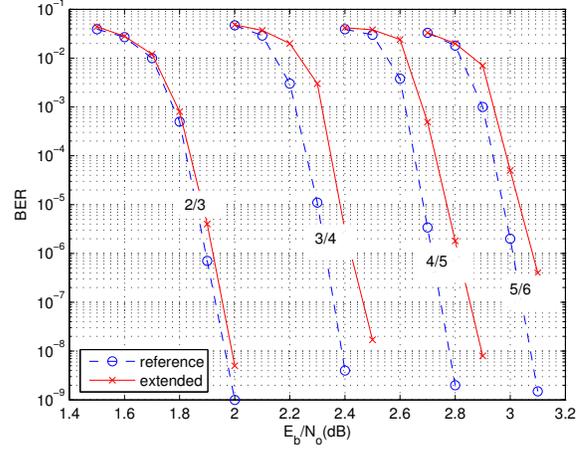Fig. 9. BER as a function of the parallelism with extended matrices

Fig. 11. BER for long frames with parallelism of 40 with extended matrices
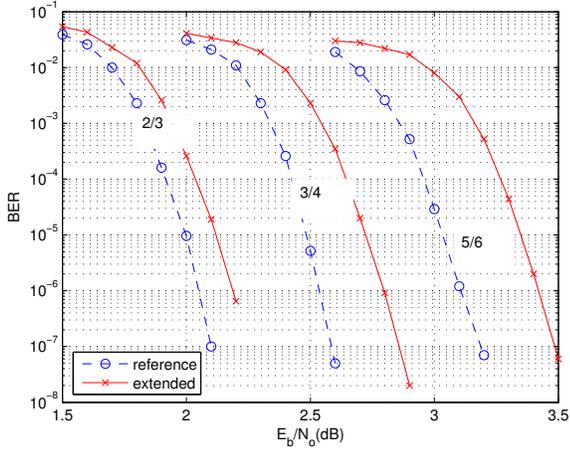

Fig. 10. BER for short frames with parallelism of 40 with extended matrices

throughput. The simulation results are presented at rates 2/3, 3/4 and 5/6. The curves represented by dashed lines are the references at a parallelism that gives no conflict. The curve represented by solid lines are the results with a parallelism of 40 and extended matrix. Despite the performance loss caused by the parity extension, the BER as a function of code rates is still coherent.

For long frames, the effect of extending the matrix is less severe as the density of DDSM over the number of identity matrices is sparser. Fig. 11 is a simulation for long frames in similar condition as in Fig. 10. At a BER of 10^{-7}, all code rates remains within 0.1dB from our reference.

Note that the throughput is reduced due to the two added identity matrices for each DDSMs. For example, in case of a rate-5/6 short frame with a parallelism of 40, the 3 DDSMs are removed by adding 6 identity matrices. Comparing with the 2055 identity matrices, this leads to a throughput reduction factor of 2055/(2055+6) which is negligible.

To summarize, the solution of extending the matrix for the remaining DDSMs gives coherent simulation results for short frames provided that the number of DDSMs does not exceed 3. Furthermore, in case of long frames results are within 0.1dB from the reference provided that the number of DDSM does not exceed 4. In next section, we present the architecture used for fixed point simulation.

V. ARCHITECTURE

The architecture proposed in Fig. 12 is mainly based on the architecture of a layered decoder. The counter counts to IM_{base} (i.e. the number of identity matrices in the base matrix). The ROM linked to the counter delivers the VG_i addresses and the associated shift value following the base matrix order. The size of the ROM dedicated to store VG_i and $Shift_i$ is $IM \times (log_2(S * N/360) + log_2(360/S))$.

In this architecture, the $\Delta shift$ value allows the use of one barrel shifter instead of two. As there is no barrel shifter in charge of shifting back the SOs values, at the next call of a VG, the SOs in this group are already shifted by the shift value of the previous call ($shift_{VG_n}^{old}$). The $\Delta shift$ value takes into account the shift value of the previous calls by doing the subtraction $\Delta shift = shift_{VG_n}^{new} - shift_{VG_n}^{old}$. $shift_{VG_n}^{old}$ is stored in a RAM of size $VG_N \times log_2(P_s)$. The SO value read in the RAM is written back in the RAM at the same address but with a delay of ϵ cycles where ϵ depends on the architecture. The $M_{v \to c}$ memory and the $M_{c \to v}$ memory are implemented with a FIFO of size dc and IM_{base} respectively. When a codeword is detected, the Most Significant Bit (MSB) of the SOs must be shifted back to generate the codeword. This can be done during the writing of the new SO value. While P_s new values are written in RAM SO, P_s MSBs value of RAM RO are read. These MSBs are shifted back using the barrel shifter with the shift value stored in RAM shift. This process takes M/P_s cycles.

The needed area to deal with one remaining DDSM using a parity check matrix equivalent would be 360 SO RAM. This extra cost is acceptable compared to the 64800 SO RAM
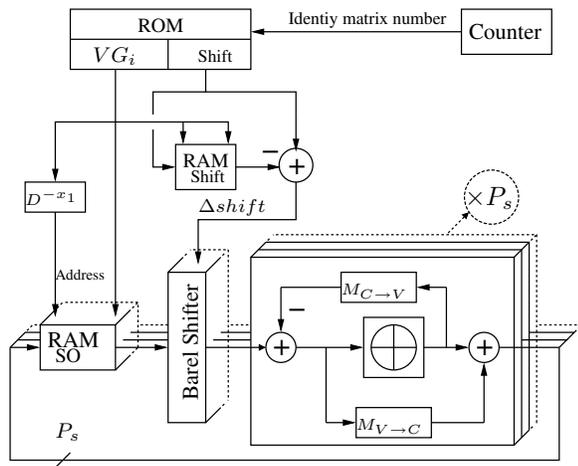
Fig. 12. Architecture of the proposed LDPC decoder

required for long frames. The area over-cost to deal with dummy VN (init with LLR=0) is negligible as the decoder must be able to deal with punctured code to match the standard requirement.

VI. Conclusion

In this paper we considered the problem of cutting edges in DVB-T2 matrices. These are due to the matrix structure and cause significant performance loss of the layered decoder. To deal with this problem, we proposed a solution based on the reordering of the matrices using a split process. This technique greatly reduces the number of DDSMs and the area of the CNP by a S factor, without any change in the layered decoder architecture. However the level of parallelism is also reduced. In the DVB-T2 standard, the number of splits are limited due to the required throughput and the problem of DDSMs remains for some of the matrices. For these cases we propose the use of an equivalent matrix, which is constructed by adding punctured parity bits. This technique efficiently removes the remaining DDSMs and the architecture remains unchanged. However, the proposed method results in a slight increase in memory area and a slight decrease in throughput. Performance loss due to the proposed method is well within the acceptable limits. The new structured matrix obtained by splitting can also provide solutions in the search of efficient layer scheduling to avoid conflicts due to pipelining. Future work is focused on hardware implementation of the proposed decoder architecture and the evaluation of its performance in terms of area and throughput at low FER.

References

[1] R. Gallager, *Low-Density Parity-Check Codes*. PhD thesis, Cambridge, 1963.
[2] E. Boutillon, J. Castura, and F. Kschischang, "Decoder first code design," in *In Proc. 2nd International Symposium on Turbo Codes & Related Topics*, (Brest, France), pp. 459–462, Sept. 2000.
[3] "Frame structure channel coding and modulation for the second generation digital terrestrial television broadcasting system (DVB-T2)," *DVB Document A122*, 2008.
[4] "Air interface for fixed and mobile broadband wireless access systems," in *P802.16e/D12 Draft*, (Washington, DC, USA), pp. 100–105, IEEE, 2005.
[5] M. M. Mansour and N. R. Shanbhag, "High-throughput LDPC decoders," *IEEE Transactions on Very Large Scale Integration VLSI Systems*, vol. 11, pp. 976–996, Dec. 2003.
[6] Y. Sun, M. Karkooti, and J. Cavallaro, "High throughput, parallel, scalable LDPC encoder/decoder architecture for OFDM systems," in *Design, Applications, Integration and Software, 2006 IEEE Dallas/CAS Workshop on*, (Richarson, USA), pp. 39–42, Oct. 2006.
[7] F. Kienle, T. Brack, and N. Wehn, "A synthesizable IP core for DVB-S2 LDPC code decoding," in *DATE '05: Proceedings of the conference on Design, Automation and Test in Europe, 2006*, (Munich, Germany), pp. 100–105, IEEE Computer Society, Mar. 2005.
[8] M. Gomes, G. Falcao, V. Silva, V. Ferreira, A. Sengo, and M. Falcao, "Flexible parallel architecture for DVB-S2 LDPC decoders," in *Global Telecommunications Conference, 2007. GLOBECOM '07. IEEE*, (Washington, USA), pp. 3265–3269, Nov. 2007.
[9] J. Dielissen, A. Hekstra, and V. Berg, "Low cost LDPC decoder for DVB-S2," in *Design, Automation and Test in Europe, 2006. DATE '06. Proceedings*, vol. 2, (Munich, Germany), pp. 1–6, Mar. 2006.
[10] M. Rovini, G. Gentile, F. Rossi, and L. Fanucci, "A minimum-latency block-serial architecture of a decoder for IEEE 802.11n LDPC codes," in *Very Large Scale Integration, 2007. VLSI - SoC 2007. IFIP International Conference on*, (Atlanta, USA), pp. 236–241, Oct. 2007.
[11] T. Bhatt, V. Sundaramurthy, V. Stolpman, and D. McCain, "Pipelined block-serial decoder architecture for structured LDPC codes," in *Acoustics, Speech and Signal Processing, 2006. ICASSP 2006 Proceedings. 2006 IEEE International Conference on*, vol. 4, (Toulouse, France), p. IV, May 2006.
[12] T. Brack, M. Alles, F. Kienle, and N. Wehn, "A synthesizable IP core for WIMAX 802.16e LDPC code decoding," in *Personal, Indoor and Mobile Radio Communications, 2006 IEEE 17th International Symposium on*, (Helsinki, Finland), pp. 1–5, Sept. 2006.
[13] M. Rovini, F. Rossi, P. Ciao, N. L'Insalata, and L. Fanucci, "Layered decoding of non-layered LDPC codes," in *Digital System Design: Architectures, Methods and Tools, 2006. DSD 2006. 9th EUROMICRO Conference on*, (Dubrovnick, Croatia), pp. 537–544, Sept. 2006.
[14] D. Hocevar, "A reduced complexity decoder architecture via layered decoding of LDPC codes," in *Signal Processing Systems, 2004. SIPS 2004. IEEE Workshop on*, (Austin, USA), pp. 107–112, Oct. 2004.
[15] M. Mansour and N. Shanbhag, "Low-power VLSI decoder architectures for LDPC codes," in *Low Power Electronics and Design, 2002. ISLPED '02. Proceedings of the 2002 International Symposium on*, (Monterey, USA), pp. 284–289, Aug. 2002.
[16] E. Boutillon and F. Guilloud, "LDPC decoder, corresponding method, system and computer program," *US patent 7,174,495 B2*, Feb. 2007.
[17] M. Fossorier, M. Mihaljevic, and H. Imai, "Reduced complexity iterative decoding of low-density parity check codes based on belief propagation," *IEEE Transactions on communications*, vol. 47, pp. 673–680, May 1999.
[18] F. Guilloud, E. Boutillon, and J.-L. Danger, "lambda-min decoding algorithm of regular and irregular LDPC codes," *Proceedings of the 3rd International Symposium on Turbo Codes and Related Topics*, Sept. 2003.
[19] D. Divsalar, S. Dolinar, and C. Jones, "Construction of protrograph LDPC codes with linear minimum distance," in *Information Theory, 2006 IEEE International Symposium on*, (Washington, USA), pp. 664–668, July 2006.