AN AREA AND ENERGY EFFICIENT HALF-ROW-PARALLELED LAYER LDPC DECODER FOR THE 802.11AD STANDARD

Meng Li, Frederik Naessens, Peter Debacker, Praveen Raghavan, Claude Desset,
Min Li, Antoine Dejonghe, Liesbet Van der Perre [*]

IMEC, Leuven, Belgium

ABSTRACT

Multi-gigabit LDPC decoders are demanded by standards such as IEEE 802.11ad and IEEE 802.15.3c. In order to achieve high throughput, most published multi-gigabit designs use row-paralleled architecture. In this paper, we proposed a half-row paralleled LDPC decoder with half layer level pipeline and single permutation network for the 802.11ad standard, which reduces the hardware resources almost by half compared to the state-of-the-art row-paralleled LDPC decoder, achieving a good trade-off between energy efficiency and area efficiency. The decoder achieves a throughput of 5.6 Gbps and consumes only 99 mW for the highest coding rate 13/16 at 5 iterations, working at 500 MHz by using 40nm G technology, yielding an energy efficiency of 3.53 pJ/bit/iteration and area efficiency of 35 Gbps/sqmm.

***Index Terms*—** 802.11ad, LDPC, multi-gigabit communication, layer decoding

1. INTRODUCTION

Since Low-Density Parity-Check (LDPC) codes [1] were adopted by the DVB-S2 standard in the year of 2003, the LDPC codes play an increasingly important role in the wireless communication domain due to their powerful error correction capability. Recently, multi-gigabit data rate standards working at the 60 GHz band, such as IEEE 802.11ad for the gigabit-wireless local area networks (WLAN) and IEEE 802.15.3c for the wireless personal area networks (WPAN), select LDPC codes as forward error correction (FEC). However, designing a LDPC decoder with multi-gigabit throughput while maintaining low power consumption and low area which is suitable for mobile device is a big challenge.

There exists multi-dimensional design space for multi-gigabit LDPC decoder, such as decoding scheduling, optimization of check node simplification, parallelism extension, pipeline stage optimization, frame level pipeline and etc. To design a high throughput decoder, the designer should consider multi-dimensional optimization and the optimization heavily depends on the constraints (performance, power or area) which is dedicated to certain application. Among all the aspects of design space, the parallelism of the decoder and the decoding scheduling highly impact the throughput.

A fully paralleled LDPC decoder[2] can reach the highest throughput. However as the parallelism increases, the routing congestion problem prohibits an efficient design and results in a large decoder with low area utilization, poor timing and power due to the presence of long interconnections. There are other high throughput LDPC decoder designs [3], which opens a door of partial paralleled architecture. A row-based partial paralleled architecture is a good alternative to reduce the routing congestion problem while increasing the throughput when compared to a block serial decoding [4] in which the check node and bit node parallelism is equal to the size of quasi-cyclic sub-matrix. The current multi-gigabit LDPC decoders [5],[6],[7] apply a row-based scheme [8] by extending the bit node parallelism to the code length while keeping the check node parallelism the same as the size of the sub-matrix, which achieves high throughput while reducing complexity compared to a fully paralleled architecture.

Apart from parallelism, the decoding scheduling also poses strong impact on the decoder's design. The state-of-the-art LDPC decoders use layer decoding, due to less requirement of memory and faster convergence speed by a factor of two when compared to the two-phase sum-product decoding. However, one drawback of the layer decoding is the data dependency between each layer. The message propagation of the current layer starts only after the corresponding posterior information is updated in previous layer, which prohibits a pipeline between two layers for a row-based architecture. In contrast to layer decoding, the two-phase decoding has less data dependency at the cost of performing double the amount of iterations for a similar BER performance. The authors of [5] introduce frame level pipeline to overcome the low convergence speed problem to speed up the throughput. The advantages and disadvantages of these two decoding scheduling are distinct, so the selection of the decoding schedule depends on design constraints.

In this work, we propose a flexible half-row-based layer LDPC decoder with reduced routing network. In contrast to the row-based architecture, the parallelism of the bit node is only half of the code length. The proposed decoder achieves

[*] thanks to Andre Bourdoux for the work of matlab platform

978-1-4673-6238-2/13 $31.00 © 2013 IEEE

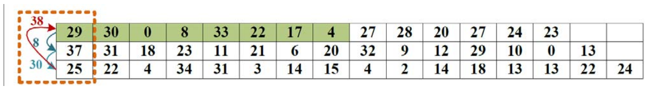
almost 3/4 throughput of the state-of-the-art row-based layer LDPC decoder by applying a half layer level pipeline, while reducing hardware resources almost by half. The permutation network taking charge of routing information from check nodes to bit nodes is eliminated by changing the shifting value for each block of information sent from bit nodes to check nodes. A 40G technology is used to verify the proposed architecture. To the best of knowledge, our work provides an optimal trade-off between the complexity and throughput, which results in a good result both in energy and area efficiency when compared to previous work.

The rest of the paper is organized as follows. In section 2, the 802.11ad standard and its LDPC codes are introduced, followed by a brief introduction of partial paralleled layer decoding algorithm. Section 3 details the proposed half-row-paralleled architecture with single permutation network. The implementation results are shown in section 4. Finally the conclusion of our work is drawn in section 5.

2. BACKGROUND

2.1. IEEE 802.11ad and its FEC

IEEE 802.11ad is an amendment to the 802.11 WLAN standard which enables up to 7 Gbps data rates in the unlicensed and globally available 60 GHz band. Primary 802.11ad applications will be removing wires between High-Definition multimedia, computer displays, I/O and peripheral, peer to peer data synchronization and higher speed LAN. 802.11ad adopts LDPC codes to provide a powerful error correction capacity.



29	30	0	8	33	22	17	4	27	28	20	27	24	23		
37	31	18	23	11	21	6	20	32	9	12	29	10	0	13	
25	22	4	34	31	3	14	15	4	2	14	18	13	13	22	24

Fig. 1. The base matrix of LDPC code for the 802.11ad standard, with coding rate $13/16$

A LDPC code is a linear block code defined by an $M \times N$ sparse parity check matrix H. The LDPC codes in IEEE 802.11ad are architecture-aware (AA) low-density parity-check codes. The parity check matrix is described as an $M_b \times N_b$ based matrix $\mathbf{H}_b$, with $M_b = M/z$ and $N_b = N/z$, where N is the code length, M is the number of parity checks and z is the size of sub-matrix which is either a zero matrix or a permutation of rows of an identity matrix. In our case, the length of the codes is equal to 672 and the size of sub-matrix is 42. Four coding rates, 1/2, 5/8, 3/4, 13/16 are supported. The base matrix of the coding rate $13/16$ is shown in Fig. 1. The number written in each block denotes the cyclic-right-permutation value for an identity sub-matrix. An empty block denotes a null sub-matrix of size 42×42.

3. THE LAYER DECODING SCHEDULE FOR PARTIAL PARALLELED DECODER

This type of AA-LDPC codes can be decoded by partial paralleled Sum-Product Algorithm (SPA) with layer scheduling, because a group of information can be calculated and pass between bit nodes and check nodes by passing through permutation network simultaneously. The decoding procedure is as follows: the *posterior* information vector $\vec{Q}_n$ with size of z is initialized with input llr_n and the *extrinsic* information vector $\vec{r}_{mn}$ is initialized with zeros. The *posterior* vector $\vec{Q}_n$ is stored based on column order while the *extrinsic* information vector $\vec{r}_{mn}$ is stored based on row order. Each iteration consists of M_b sub-iteration. During each sub-iteration, the *posterior* $\vec{Q}_n$ information passes through a permutation network with a shifting value $\epsilon_{m,n}$ equaling to $P(m, n)$, where $P(m, n)$ is the shifting value indicated in the base matrix. The check nodes utilize the *a priori* information, which is achieved by subtracting the *extrinsic* information from the *posterior* information, to calculate the updated *extrinsic* information by using SPA or simplified Min-sum [9] algorithm. Afterwards, the updated *extrinsic* information is added to the *a priori* information. Finally, the *posterior* information passes through the permutation network again with a shifting value $\tau_{m,n}$ equaling to $z - P(m, n)$ and saved back to memory.

Algorithm 1 layer decoding

```
Initialization:
$\vec{Q}_n = \vec{llr}_n \; \vec{r}_{mn} = \vec{0}$
Decoding:
for $t = 1, .., t_{max}$ {iteration }
  for $m = 1, .., M_b$ { sub-iteration}
    Check node processing
      $\vec{q}_{mn} = |\vec{Q}|_n^{\epsilon_{mn}} - \vec{r}_{mn}$
      $\vec{r}_{mn} = f\left(\vec{q}_{mk}\right), k \in N(m) \backslash n$
    Bit node processing
      $\vec{Q}_n = \vec{q}_{mn} + \vec{r}_{mn}, n \in N(m)$
      $|\vec{Q}|_n = \vec{Q}_n^{\tau_{mn}}$
Hard decision according to $|\vec{Q}|_n$
```

The LDPC decoding algorithm has been tested over a Matlab standard-compliant 802.11 ad physical layer simulator. The decoding performance of normalized min-sum layer decoding for the four coding rates, over indoor multipath channel model, QPSK, has been described in Fig. 2. Simulation results show that no significant BER and PER deviation is observed between the floating point decoding and fixed point decoding with 5 bits LLR quantization with maximum iteration number of 10.

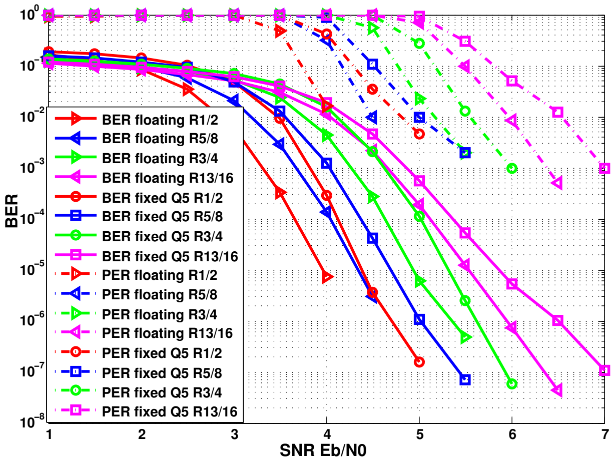


Fig. 2. The LDPC decoding BER/PER performance over indoor multipath channel for QPSK

4. ARCHITECTURE DESIGN

In this section we will detail a partial paralleled architecture with half-row-based hardware mapping with single permutation network which requires almost half of the hardware resources when compared to the conventional decoder with row-based hardware mapping [8] while achieving high throughput.

4.1. Row-based architecture

Previous work on the multi-gigabit LDPC decoder design focuses on partly paralleled hardware mapping, since it offers a good tradeoff between complexity and throughput. They fall into three categories: low-parallel, row-based and column-based. A low-paralleled decoder has $i \times z$ check node function units (CFU) work in parallel and $j \times z$ bit node function units (VFU) work in parallel, which instantiates small number of function units but needs a large number of cycles to process one iteration. When $i = 1$ and $j = 1$, it is a block serial decoder. A row-based decoder instantiates all N VFU while the parallelism of CFU is $i \times z$. The column-based decoder instantiates all CFU while the parallelism of VFU is a multiple of z, which is dedicated to special structured parity check matrix such as the codes in IEEE 802.15.3c.

A top level row-based architecture for 802.11ad layer decoder has been described in Fig. 4. The main blocks are VFN, APP RAM, CFU and permutation network. The APP RAM is divided into $N_b = 16$ blocks, all the 16 VFUs take charge of resorting and providing the *posterior* information simultaneously. The information saved in the APP RAM is in column order and follows a sequential order of the input LLR. Each set of the 42 *a priori* data passes through one permutation network and feed into different check node function units. There are $z = 42$ CFUs working in parallel. Each one has 16 inputs, which are connected to the 16 VFUs. For a min-sum layered LDPC decoder, the CFU calculates the minimum and second minimum value among valid inputs to update *extrinsic* and 16 *posterior* information. Each of the 16 outputs from the 42 CFU forms one set of the *posterior* information vector. There are 16 of them, which are in a row order. Each element in the *posterior* information vector goes through the permutation network with inverse shifting value and then is written back to the APP RAM.

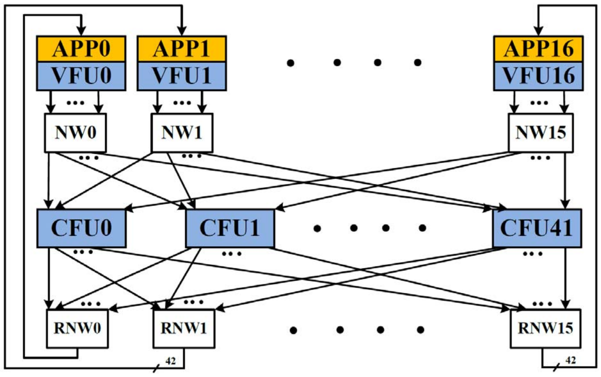


Fig. 4. Top level architecture of a row-based layer decoder

4.2. Half-row-based architecture

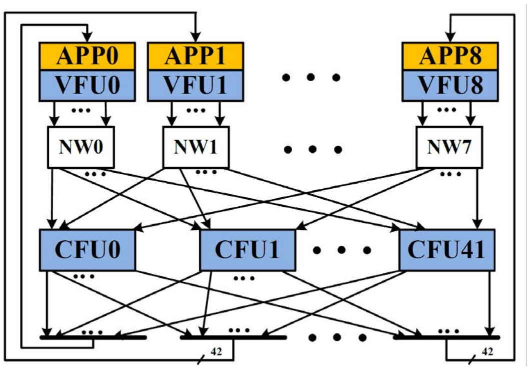


Fig. 5. Top level architecture of a half-row-based layer decoder

In order to meet a strict power and energy efficiency constraint of the LDPC decoder, a half-row-based architecture with a reduced permutation network is proposed, which is described in Fig. 5. Only 8 blocks of VFN and APP RAM are instantiated, hence the block number of permutation network is reduced to 8 as well. The parallelism of the bit node processor is reduced from 672 to 386 while the parallelism of the check node processor remains as 42. Since only $N_b/2 = 8$ groups of VFU are instantiated, we call this half-row-based or half-row-paralleled architecture.

With this kind of hardware mapping, both the check node

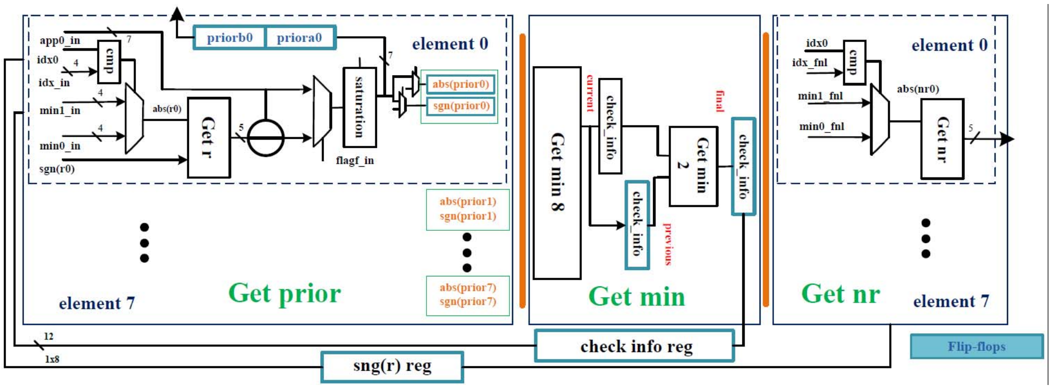


Fig. 3. Architecture of check node processor of the proposed half-row-parallel based layer decoder

and bit node process of one layer decoding is divided into two phases. During the first phase of the check node process, the *posterior* information vectors related to LLR0 to LLR385 pass through the permutation network are fed into each CFU. One CFU has eight *a priori* inputs. The one related to a null sub-matrix in the parity check matrix is set as a positive large number in order to avoid bringing wrong information to the check node processor. Then the CFU calculates the minimum, second minimum and the sign information, which is called $m1\_p1$, $m2\_p1$ and sg_p1, with the first 8 inputs, as shown in Equ. 3.

$$sg\_pi = sign(\vec{q}_{mk}); \quad k \in [0,8] + i*8 \tag{1}$$
$$m1\_pi = argmin1(\vec{q}_{mk}); \tag{2}$$
$$m2\_pi = argmin2(\vec{q}_{mk}); \tag{3}$$

During the second phase of the check node process, the *posterior* information vectors related to LLR386 to LLR762 pass through the network and feed into each CFU. One CFU calculates the minimum, second minimum and the sign information with the second 8 inputs, which is called $m1\_p2$, $m2\_p2$ and sg_p2. The final check node information can be achieved only in the second phase by combining the result from the previous phase and current phase, which is indicated in Equ. (4).

$$sg\_fnl = sign(sg\_p1, sg\_p2); \tag{4}$$
$$m1\_fnl = argmin1(m1\_p1, m1\_p2); \tag{5}$$
$$m2\_fnl = argmin2(m1\_p1, m1\_p2, m2\_p1, m2\_p2); \tag{6}$$

The bit node update is divided into two phases as well. The *posterior* information belongs to the first 8 LLR blocks is updated in the first phase and then follows the update of the *posterior* information belongs to the second 8 LLR blocks.

The conventional layer LDPC decoder uses two permutation networks π which are implemented by a multi-level barrel shifter to route the *posterior* information vector between the APP RAM and check node function units, as shown in Fig. 4. The *posterior* information memorized in the APP RAM is in a sequential order. During each sub-iteration of layer decoding, the posterior information passes through one permutation network to permute in row order before it is fed into the CFU. Then the updated posterior information is routed by another permutation network in an inverse order. The objective of using two permutation networks is that the memorized posterior information vector can be read by the CFU in a proper order.

$$\epsilon_{m,n} = \begin{cases} P(m,n) & when \quad m{=}\gamma_n(0)\&(iter{=}0) \\ mod\,(P(m,n){-}P(\gamma_n(dv{-}1),n),42) & \\ & when \quad m{=}\gamma_n(0)\&(iter{>}0) \\ mod\,(P(m,n){-}P(\gamma_n(k{-}1),n),42) & \\ & when \quad m{=}\gamma_n(k)\&(k{>}0) \end{cases} \tag{7}$$

The number of permutation network can be reduced to one [10] as illustrated in Fig. 5. In this case, the posterior information vector memorized in the APP RAM is no longer following sequential order. When it is read by CFU, it passes through the permutation network with a shifting value equals to the difference of the two value in two consecutive layer in $\mathbf{H}_b$. In this case, the permutation value $\tau_{m,n}$ is equal to zero and the value of ϵ_{mn} is illustrated as equ.7, where $\gamma_n(k)$ means the k_{th} row index of non zero element in column n in the base matrix $\mathbf{H}_b$. The permutation values after one iteration for the first column of the parity check matrix of the coding rate $13/16$ are shown in Fig. 1.

4.3. Pipeline stage and half layer level pipeline of the half-row-based architecture

The most demanding block of the half-row-based decoder is the check node processor. The main block of the check node processor is illustrated in Fig.3. It includes three sub-blocks: **get prior**, **get min** and **get nr**. The blocks in blue mean the information need to be memorized, which is synthesized as flip-flops. Hence it is easy to observe that there are three stages of pipeline in the check node processing procedure. Considering the bit node process, each half layer decoding lasts four cycles, as indicated in Fig. 6.

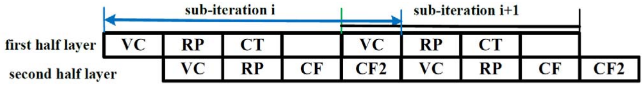


Fig. 6. The timing of half-row-based layer decoding

During the cycle **VC**, the *posterior* information read out from APP RAM passes through the permutation network. In the meanwhile, the sign of *extrinsic* information r and check node information : minimum, second minimum, index of minimum and the total sign, which is memorized in row order, is read out and fed into the CFU.

In the cycle **RP**, the CFU generates the *extrinsic* information and the *a priori* information. The sign and absolute value of the *a priori* information is registered.

In the cycle **CT**, the CFU calculates the minimum, second minimum value among 8 inputs, the corresponding index and the total sum of the sign of 8 *a priori* information, which is called current check node information and is registered.

In the cycle **CF**, the CFU gets the current check node information for the second 8 inputs, which belong to the same layer as the previous 8 inputs. Then the final check node information is calculated based on the registered value in cycle **CT** and the calculated value in current cycle. The *posterior* information update process is compressed into this cycle since it can be directly written back to the APP RAM, taking the benefit of one permutation architecture. Only the *posterior* information update and storage process in carried out in cycle **CF2**.

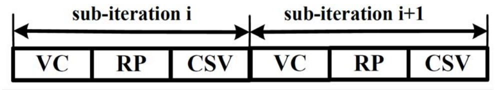


Fig. 7. The timing of row-based layer decoding

The similar pipeline stages can be applied for a row-based layer decoder. Since the 16 *posterior* elements are connected to one CFU, the check node information calculation might be finished in one cycle and only three pipeline stages might be enough, as shown in Fig. 7. However, in order to avoid message update conflicts, the next layer decoding can be triggered only after the previous layer finished the *posterior* information update. Hence for the coding rate 13/16, one iteration takes 9 cycles. The number of cycles required for given number of iterations can be calculated as Equ. 8.

$$N\_row = (3 \times Nlayers) \times Niter; \tag{8}$$

Different from row-based layer decoder, there is no data dependency between the first half layer and the second half layer, so a half layer level pipeline can be applied, as shown in Fig. 6, to achieve higher throughput. During each sub-iteration, the two phases of check node process is allocated in **CT** and **CF** and the two phases of bit node process is allocated in **CF** and **CF2**. Hence for the coding rate 13/16, one iteration takes 12 cycles. The number of cycles required for certain number of iterations can be calculated as Equ. 9.

$$N\_half\_row = (4 \times Nlayers) \times Niter + 1; \tag{9}$$

According to the previous explanation, reducing the bit node parallelism by half is an efficient way to reduce the hardware resource almost by half. When a pipeline between two half layers is applied, this kind of architecture achieves almost 75% of the throughput compared to a row-based paralleled layer decoder.

5. IMPLEMENTATION RESULTS

The proposed LDPC decoder with early stopping criteria is implemented by 40G CMOS technology, which is a performance oriented technology having low default voltage and fast speed. According to the measurement result, the decoder can achieve 5.6 Gbps throughput with 5 iterations working at 500 MHz for the highest coding rate. The decoder consumes 99mW (dynamic + leakage) during continuous decoding time with a supply voltage of 0.9 V. Power estimation is based on a physical synthesis netlist. Almost 68% of the power goes into the CFU. The permutation network and the control model takes 15% and 12% of the power, respectively. 5% of the power is consumed in the remaining components.

A comparison between our design and the state-of-the-art multi-gigabit LDPC decoder is listed in Table. 1. The decoder [5] published in ISCAS uses two-phase decoding to avoid data update conflict problem and uses a frame level pipeline to boost the throughput for two-phase decoding. It works at 150 MHz with the supply voltage as 0.8V. The throughput is given as 3.08 Gbps, but this value is achieved at which number of iterations is not indicated. Based on the information given in the paper, this throughput number is achieved at 4 number of iterations based on our calculation, which results in an energy efficiency of 7 pJ/bit/iteration. The decoder [6] published in ISIC uses layer decoding and row-based fully paralleled architecture. More aggressive pipeline compression is done, which results in an impressive number of required cycles per sub-iteration. Only two cycles are needed for one sub-iteration. To guarantee this, the decoder either works at very low frequency with a supply voltage of 1.0 V or works at high frequency with a high supply voltage.

	This work	ISCAS[5]	ISIC[6]	
Standard	802.11ad	802.11ad	802.15.3c	
Algorithm	Layer	Two-phase	Layer	
VFU parallelism	336	672	672	
CFU parallelism	42	42	21	
Frame pipeline	NO	Yes	Yes	
Quantization	5	5	6	
Technology	40nm G	65nm	65nm LP	
Area (mm^2)	0.16	1.33	1.3	
Area (mm^2)[1]	0.42	1.33	1.3	
Work frequency(MHZ)	500	150	400	200
Throughput (Gbps)	5.6 @ 5it	3.08	6.72 @ 10it	3.36 @ 10it
Throughput[2](Gbps)	3.45 @ 5it	3.08	6.72 @ 10it	3.36 @ 10it
VDD (V)	0.9	0.8	1.2	1.0
Power (mW)	99	84	537	210
Power (mW)[3]	198	213	373	210
Energy efficiency (pJ/bit/iter)	3.53	7	8	6.2
Energy efficiency[4] (pJ/bit/iter)	7.08	10.9	5.5	6.2
Area efficiency (Gbps/sqmm)	35	2.3	5.17	2.58
Area efficiency[5] (Gbps/sqmm)	8.2	2.3	5.17	2.58

Table 1. Comparisons with published multi-gigabit decoders

Although using different CMOS technology and supply voltage impact the final results, our design still shows a good performance in term of energy efficiency and area efficiency due to using the half-row based architecture to reduce the hardware resources and applying a half layer level pipeline to boost the throughput, when normalizing supply voltage and technology scaling.

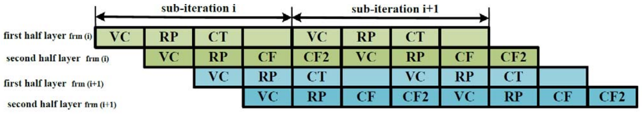

sub-iteration i
sub-iteration i+1
first half layer frm (i)
second half layer frm (i)
first half layer frm (i+1)
second half layer frm (i+1)
VC
RP
CT
CF
CF2

Fig. 8. The timing of half-row-based layer decoding with pipeline between 2 frames

6. CONCLUSION

A half-row-based partial paralleled layer LDPC decoder with single permutation network is proposed for the 802.11ad standard. The implementation using 40G technology occupies only 0.202 mm^2 and consumes 99 mW in total with 0.9 V supply voltage at a throughput of 5.6 Gbps @ 5 iterations for the highest coding rate, which results in an energy efficiency of 3.53 pJ/bit/iteration and area efficiency of 35 Gbps/sqmm. This shows to be both an area and energy efficient solution which utilizes layer decoding implemented a commercial technology at default supply voltage. The proposed architecture is flexible to extend to a row-based partial paralleled architecture and is flexible to apply a frame level pipeline, which is shown in Fig. 8 to achieve more than 10 or 15 Gbps throughput for the future communication scenario.

7. REFERENCES

[1] R.G. Gallager, “Low-density parity-check codes,” in *MA: MIT Press*, 1963.

[2] A. Darabiha, A.C. Carusone, and F.R. Kschischang, “A 3.3-gbps bit-serial block-interlaced min-sum ldpc decoder in 0.13um cmos,” in *Custom Integrated Circuits Conference, 2007. CICC ’07. IEEE*, sept. 2007, pp. 459 –462.

[3] Zhengya Zhang, V. Anantharam, M.J. Wainwright, and B. Nikolic, “An efficient 10gbase-t ethernet ldpc decoder design with low error floors,” 2010, vol. 45, pp. 843–855.

[4] M.M. Mansour, “A turbo-decoding message-passing algorithm for sparse parity-check matrix codes,” nov. 2006, vol. 54, pp. 4376 –4392.

[5] M. Weiner, B. Nikolic, and Zhengya Zhang, “Ldpc decoder architecture for high-data rate personal-area networks,” in *Circuits and Systems (ISCAS), 2011 IEEE International Symposium on*, may 2011, pp. 1784 –1787.

[6] Zhixiang Chen, Xiao Peng, Xiongxin Zhao, Qian Xie, L. Okamura, Dajiang Zhou, and S. Goto, “A 6.72-gb/s 8pj/bit/iteration ieee 802.15.3c ldpc decoder chip,” in *Integrated Circuits (ISIC), 2011 13th International Symposium on*, dec. 2011, pp. 7 –12.

[7] Shiang-Yu Hung, Shao-Wei Yen, Chih-Lung Chen, Hsie-Chia Chang, Shyh-Jye Jou, and Chen-Yi Lee, “A 5.7gbps row-based layered scheduling ldpc decoder for ieee 802.15.3c applications,” in *Solid State Circuits Conference (A-SSCC), 2010 IEEE Asian*, nov. 2010, pp. 1 –4.

[8] Philipp Schlafer, Christian Weis, Norbert Wehn, and Matthias Alles, “Ldpc decoder architecture for high-data rate personal-area networks,” in *Hindawi Publishing*, February 2012.

[9] J. Chen, A. Dholakia, E. Eleftheriou, M.P.C. Fossorier, and X.-Y. Hu, “Reduced-complexity decoding of ldpc codes,” aug. 2005, vol. 53, pp. 1288 – 1299.

[10] Sangmin Kim, G.E. Sobelman, and H. Lee, “A reduced-complexity architecture for ldpc layered decoding schemes,” june 2011, vol. 19, pp. 1099 –1103.

[1] Normalized area: area · $(65nm/technology)^2$

[2] Normalized throughput: throughput · $(technology/65nm)$

[3] Normalized power: power × $(65nm/technology) \cdot (1/Vdd_{tech})^2$

[4] Normalized energy efficiency: Normalized power / bit / iteration

[5] Normalized area efficiency: Normalized throughput / Normalized area