Abstract—This paper presents a high-throughput multi-mode Quasi-Cyclic Low-Density Parity-Check (QC-LDPC) decoder architecture for 60GHz wireless gigabit communication. A novel multi-mode, multi-frame, and multi-block parallel layered decoding architecture for IEEE 802.11ad standard is proposed. A novel local switch is proposed to minimize the hardware complexity and a $H$-matrix reformulation technique is proposed to eliminate the re-shuffling network for multi-rate LDPC decoders. A novel multi-frame pipeline technique suitable for a layered LDPC decoding is also proposed to meet the high throughput requirement of the 802.11ad standard. A three-stage pipeline and two block-parallel layered decoding technique is used to improve the clock speed and throughput. An efficient arbitrary-size Benes cyclic shifter, which nullifies the requirement of $2^n$ number of inputs, is proposed to perform $42 \times 42$ cyclic permutations. A 672-bit multi-mode decoder has been designed and implemented using 90-nm CMOS standard cell technology. Synthesis results show that the proposed QC-LDPC decoder requires 898K gate and can operate at 337 MHz to achieve a maximum required throughput of 6.1 Gbps with a maximum of 9 iterations; which meets the requirement of 60 GHz wireless gigabit communications (802.11ad).

Index Terms—Low density parity check (LDPC), forward error correction (FEC), multi-mode, gigabit, communication.

I. INTRODUCTION

Low density parity check (LDPC) codes, first proposed by Gallager in 1962 [1] have attracted much attention because of their excellent error correction performance, inherent parallelism and high throughput potentials. Therefore, they are being widely used in communication standards such as the DVB-S2, IEEE 802.16e and IEEE 802.11n. In addition, millimeter wave (mmWave) wireless gigabit communication described by the IEEE 802.11ad Working Group [2] is considering LDPC codes as the preferred choice for forward error correction (FEC).

In this paper, a novel multi-mode, multi-frame and multi block-parallel layered QC-LDPC decoder architecture for IEEE 802.11ad standard is proposed. For rates 3/4 and 13/16 the proposed architecture can process the single block layer within one clock cycle. The architecture is also capable of concurrent processing of two block rows for rate 1/2 and 5/8 within one clock cycle. Our previous work [4] is further extended and a novel Quasi-Cyclic (QC) $H$-matrix reformulation technique is introduced which enables complete elimination of a re-shuffling network used in existing LDPC decoders [5][6][8]. An arbitrary-size Benes network [15] is explored to counter the limitation of $2^n$ number of inputs for a cyclic permutation network. The proposed arbitrary-size Benes cyclic shifter can be designed and implemented for arbitrary number of inputs nullifying the $2^n$ limitation. Furthermore, a multi-frame pipelining specific to a layered decoding is also introduced. A three-stage pipelining is employed to achieve high clock speed and high data throughput.

The rest of this paper is organized as follows. Section II describes the design issues related to the multi-mode, multi-block parallel layered QC-LDPC decoder. Section III describes the proposed $H$-matrix reformulation technique to eliminate the re-shuffling requirement. In Section IV, single-rate decoder architecture and novel multi-mode, multi-block parallel layered decoder architecture with multi-frame pipelining are proposed. In Section V, a cyclic shifter based on arbitrary Benes network is presented. Section VI presents the implementation and comparison results. Finally, a conclusion is presented in Section VII.

II. DESIGN ISSUES RELATED TO MULTI-MODE, MULTI-BLOCK LAYERED QC-LDPC DECODER

The layered decoding algorithm [3] which is a modified version of the Belief-Propagation (BP) algorithm can reduce the average number of iterations using intermediate check-node (or variable-node) message values by almost 50% compared with the BP algorithm. Therefore, it offers $2 \times$ throughputs without performance degradation. However, due to the data dependency between consecutive rows in the layered decoding, the multi-row parallel processing and pipelining techniques cannot be applied directly. QC-LDPC codes, which are composed of sub-matrices, enable to implement the block parallel layered decoding architecture. In the QC-LDPC codes,
the consecutive rows in a block row of $H$-matrix (a layer) operate independently because the column weight of each sub-matrix is at most one. In other words, instead of one row, one block row which is composed of $z$ (i.e., $z = 42$ for 802.11ad) rows can be updated simultaneously in one clock cycle. However, the layered decoder needs to be extended to provide higher parallelism, which is necessary for high throughput applications. Our previous work [5] presents a layered decoder architecture that can update two or more block rows simultaneously within one clock cycle. But the same technique is not directly applicable in the case of IEEE 802.11ad. Also the architecture in [5] can only handle single rate and for multi-mode architecture. There is a need to develop some way to accommodate all four code rates while applying maximum available parallelism.

After observing the $H$-matrix [2] for rates 1/2 and 5/8, it is clear that two consecutive rows depict a column weight of one after interchanging block rows two and three as well as block rows five and six for the rate of 1/2. That enables simultaneous processing of two consecutive block rows. Similarly for the rate of 5/8, block rows four and five are exchanged to enable the concurrent processing of block rows three and four as well as block rows five and six. But, block rows one and two in rate 5/8 still cannot be processed in parallel. To overcome this problem, a grouping scheme is described below. The scheme exploits the maximum available parallelism and provides a multi-mode option at the same time.

Rate 1/2 $H$-matrix [2] depicts that a maximum of two block rows of rate 1/2 can be processed in parallel; that makes a total of four groups of layers in which each group consists of two block rows. All the block rows within each group are processed concurrently. However for rate 5/8, block rows one and two cannot be paralleled; but still a total of four groups by taking only single block row in groups one and two. For rates 13/16 and 3/4 the formation of groups is straightforward. Four groups are formed for rate 3/4 and three groups are formed for rate 13/16. Groups are shown as G1, G2, G3, and G4 in Fig. 2. Each group is processed in one clock cycle. In the case of rate 13/16, there is a one clock cycle stall due to a total of three groups instead of four. One important property, that is noticeable after combining two blocks rows in a group, is that there isn’t any check node degree more than eight for each combined block row. If there is just one block row in a group, then the check node degree is somewhere between eight and sixteen. This property is exploited to develop a transition from single rate architecture to multi-rate architecture with the little increase in complexity; and an almost negligible increase in critical path. The proposed multi-rate transformation scheme is shown in Fig. 5 and is described in Section IV. A similar multi-block processing grouping method is also applicable for other IEEE standards like 802.15.3.c [7].

III. $H$-MATRIX REFORMULATION FOR RE-SHUFFLING NETWORK ELIMINATION

A conventional LDPC decoding scheme [4] is shown in Fig. 1(a) where two shuffle networks and a check node based processor is required. The previous work [4] showed an algorithm for re-shuffle elimination. Here another reformulation technique suitable for multi-rate architectures is proposed. The reformulated $H$-matrix from this method is applicable for different types of decoder architectures.

$$X = (Z - (X_{\text{PREV}} - X_{\text{NEW}})) \bmod \mathbb{Z}$$  

$$X = (42 - (0 - 35)) \bmod 42 = 35$$  

$$X = (42 - (35 - 29)) \bmod 42 = 36$$  

$$X = (42 - (25 - 35)) \bmod 42 = 10$$  

The conventional shuffling scheme and the reduced shuffling scheme are shown in Fig. 1(a) and (b), respectively. The reduction or elimination equation is given in (1). Let’s take an example with the rate 3/4 to describe the whole process. Fig. 1(c) shows the standard procedure for shuffling and re-shuffling for first block column of rate 3/4. Fig. 1(d) shows the reduced shuffling procedure. 35 is the first cyclic shifting value in first block column of rate 3/4. At the start of iteration one, there was no cyclic shift before that iteration or equivalently there was a Zero cyclic shift. Thus, $X_{\text{PREV}}$ in (1) is Zero while $X_{\text{NEW}}$ is 35. By applying (1), 35 is retrieved as shown in (2). Then conventionally reshuffling by 42-35 = 7 is required but it can be eliminated by merging this re-shuffling with the shuffling of the next step. Conventionally a shuffle by 29 is required but after incorporating the re-shuffle by seven with the shuffle by 29, a new shuffle value is obtained as shown...
Min-Sum algorithm [17] can be expressed as follows: iteration, again there is net shuffle of 25 which again results in iteration, net previous cyclic shift was 25. So, the first shuffle shift at the start of iteration one was zero). But at the start of 2nd iteration, 35 cannot be considered as a first shuffle value, values remain same as in 1st iteration. At the end of the 2nd iteration, the log-likelihood ratio (LLR) message from layer l to next layer for variable node v is represented by P. In (6) N(c)/v denote the set of variable nodes connected to check node c excluding v variable node. In the case that all soft messages corresponding to the 1-components in an entire block row of parity check matrix is processed in a clock period, the computations shown in (5)–(7) are sequentially performed. It is clear that basically an adder, a subtractor and the unit to find 1st and 2nd minimum is needed.

IV. MULTI BLOCK LAYERED DECODER ARCHITECTURE

A. Min Sum Layered Decoding Algorithm

The layered decoding algorithm [3][9] with the modified Min-Sum algorithm [17] can be expressed as follows:

\[
\hat{L}^t[k] = P^t[k] - R^t[k-1] 
\]  

(5)

\[
R^t[k] = \alpha \times \prod_{n \in N(c)/v} \text{sign}(E^n[k]) \times \text{Min} \left[ E^t[k] \right] 
\]  

(6)

\[
P^t[k] = \hat{L}^t[k] + R^t[k] 
\]  

(7)

in (3). Thus, in that case X_{PREV} is 35 and X_{NEW} is 29 and reshuffle incorporated with shuffle results in net shuffling of 36. Similarly, other two reduced shuffle values (8 and 30) are derived as shown in Fig. 1(d). The procedure works smoothly until the complete 1st iteration but after entering into the 2nd iteration, 35 cannot be considered as a first shuffle value, because 35 was obtained using the initial condition (i.e. cyclic shift at the start of iteration one was zero). But at the start of 2nd iteration, net previous cyclic shift was 25. So, the first shuffle value for 1st and 2nd iteration shown as ‘35 / 10’ in Fig. 1(d) is actually different. Shuffle by 10 is the result of the calculation shown in (4). Now, though the shuffle by 10 is performed but still net shuffle value is 35. Note that the net shuffle value was also 35 after first shuffling in 1st iteration. So, the rest of the values remain same as in 1st iteration. At the end of the 2nd iteration, again there is net shuffle of 25 which again results in initial shuffle by 10 in 3rd iteration and the procedure continues until the completion of all the iterations.

Completion of final iteration results is a net shuffle of 25. But it is highly undesirable to provide permuted or shuffled version as decoded output. So, now another shuffle by 42-35 = 17 is required. For single rate case this final output shuffle can be achieved with the fixed wired shuffled network without any switches that don’t constitute complexity. But for multi-rate architecture this fixed wired shuffling is not possible because some code rates potentially offer different net shuffle values as in case of 802.11ad. The way to overcome this problem is described in the following.

Final shuffling is performed by the same shuffle network which is responsible for doing all the cyclic shifting from iteration one until the final iteration. But it adds 1 clock cycle delay in the final decoded output. Also no parity check operation is performed until the completion of the final iteration.

Let us assume for a total number of nine iterations, four group scheme takes four clock cycles to finish whole iteration. So after 9 × 4= 36 clock cycles another clock cycle is required to do the final shuffle which results in the net shuffle of Zero as an output. This is why another shuffle is introduced and three shuffle values are shown as ‘35/10/17’ in Fig. 2(b). Parity check can be employed at the end of final iteration if required. A similar H-matrix reformulation approach is applied on all block columns of rates 5/8, 3/4 and 1/2. Consequently Fig. 2(a), (b) and (c) show the reformulated H-matrices with reduced shuffle values.

IV. MULTI BLOCK LAYERED DECODER ARCHITECTURE

A. Min Sum Layered Decoding Algorithm

The layered decoding algorithm [3][9] with the modified Min-Sum algorithm [17] can be expressed as follows:

\[
\hat{L}^t[k] = P^t[k] - R^t[k-1] 
\]  

(5)

\[
R^t[k] = \alpha \times \prod_{n \in N(c)/v} \text{sign}(E^n[k]) \times \text{Min} \left[ E^t[k] \right] 
\]  

(6)

\[
P^t[k] = \hat{L}^t[k] + R^t[k] 
\]  

(7)

Here l denotes the layer and k is the iteration number. R denotes the check to variable message conveyed from check c of lth layer to variable node v. L represents the variable to check message from variable node v to check node c. In the kth iteration, the log-likelihood ratio (LLR) message from layer l to next layer for variable node v is represented by P. In (6) N(c)/v denote the set of variable nodes connected to check node c excluding v variable node. In the case that all soft messages corresponding to the 1-components in an entire block row of parity check matrix is processed in a clock period, the computations shown in (5)–(7) are sequentially performed. It is clear that basically an adder, a subtractor and the unit to find 1st and 2nd minimum is needed.

B. Single-Rate Architecture

The proposed single rate (3/4 rate) architecture to perform all operations from (5) to (7) in sequence is shown in Fig. 3.
The proposed architecture computes the check to variable, variable to check and LLR messages corresponding to entire block row of $H$-matrix in one clock cycle. Single rate architecture for QC-LDPC decoder with eliminated re-shuffling network is shown in Fig. 3. The important parts of architecture are explained as follows.

1) **Switch Network**: It performs a cyclic shuffling. Cyclic re-shuffling network is eliminated and its elimination method is described in section III. IEEE 802.11ad requires a 42×42 cyclic shifter as a switch network. Details of its implementation are explained in section V.

2) **Control ROM**: It mainly consists of pre-computed layer wise control values for a cyclic shifter. Group wise skip control values are also stored in ROM. The skip values basically enable the decoder to mark the Zero sub-block in $H$-matrix. Skip values for the first group of rate 3/4 are 0000_0000_0000_0111. So, last three sub-blocks are bypassed. As the zero sub blocks are skipped or bypassed during processing, hence it is termed skip control.

3) **L-Adder**: CNBP (check node passed processor) starts from this particular stage. L-adder is responsible for the implementation of (5) (i.e. subtract check to variable message of previous iteration but same layer ($R^k[k-1]$) from LLR value from previous layer ($P^{c2v}[k]$)). But $c2v$ messages from previous iteration stored in RAM are in form of compressed signed magnitude format [10][11] and P messages are in 2's complement format. So, conventionally conversion from signed magnitude to 2's complement is required [5]. But it increases the critical path and the complexity. So, more efficient design from [11] is tailored according to the architecture requirement and presented in Fig. 4(a). Distributor [10] actually distributes the compressed $Min1$ and $Min2$ messages across all L-adders. Only one L-adder gets $Min2$ selected by index decoder and the $Min1$ is subtracted from the rest. Skip control is important here because if it is true, then output gets maximum value considering the quantized or fixed point precision as shown in Fig. 4(a). This in turn enables minimum unit to select correct minimum values. Skip control ensures that minimum unit gets maximum quantized value in case of Zero sub block. It is better to accommodate skip here because saturation to maximum value is required in the case of adder overflows.

4) **Minimum Unit**: It implements minimum part of (6). This is the major source of the critical path delay. So, the minimum unit needs a special design. A design from [12] is adopted. The design describes two basic units for finding minimum values. One is $2mVG$ and other is connection unit (CU). $2mVG$ performs compare and select operation, thus it is called C&S stage in Fig. 3. Once inputs are sorted from C&S stage, they are passed on to the next stage of connection units (CUs). For sorting of 1st minimum and 2nd minimum from 16 inputs (as in case of rate 3/4), total three stages of CUs after C&S stage are required.

5) **P-adder**: CNBP ends after this stage. P-adder performs the addition operation of (7). As in the case of L-adder, its efficient design is shown in Fig. 4(b). In the case of true skip value the L-adder saturates the output to maximum value for proper minimum operation, but that saturated value is not a next valid LLR value. So, in the case of P-adder, a skip control is required to ensure that the output depicts the same LLR value as input LLR value. The LLR value of the previous group or layer is replicated and passed on to the next group in case of zero sub-blocks.

![Fig. 3. Single rate architecture for rate ¾ LDPC decoder.](image-url)

![Fig. 4. Efficient implementation of (a) L-adder, (b) P-adder.](image-url)
The core idea to convert from single-rate architecture to multi-rate architecture is shown in Fig. 5. It shows the overall idea to process rate 3/4 and rate 1/2 by making a slight modification in the single rate (rate 3/4) architecture shown in Fig. 3. The idea shown in Fig. 5 accommodates the maximum available parallelism of rate 1/2 because it is fully capable of processing two block layers or a group in one clock cycle. 8-input minimum stage in Fig. 5 consists of one C&S stage and two CU stages. In case of rate 1/2 as the check node degree for each block is eight or less than eight. So, two different block rows in a group are routed to two different 8-input minimum (8-Min) units. While for rate 3/4 Fig. 5 depicts a split in a block row because its check node degree is more than 8 for all block rows. Thus, one half of the sub-blocks are fed to a one 8-Min unit while other half is fed to the 2nd 8-Min unit. For 1/2 rate, four minimum values (two values for each block row) are required for one group which are ready after 8-Min stage. While for rate 3/4 just two minimum values are required. Thus another coupler stage is added which is actually 4-Min CU [12]. Finally multiplexers are added to select between Min1 and Min2 from 8-Min stage or from coupler stage. Multiplexers select coupler output for rate ¾; otherwise they select the output of 8-Min units. Based on the multi-rate transformation scheme in Fig. 5, a multi-rate architecture is proposed in Fig. 9 and the detailed description of multi-rate architecture is given below.

1) Local Switch: The ways inputs are distributed among 8-Min units all depends upon the local switch. For rate 3/4 and 13/16 there is no real need for a local switch network because any input can go to any 8-Min unit. There is no need for sorting or taking a Min1 and Min2 from some particular inputs; while the same case is not true for rates 5/8 and 1/2. For instance, the block row one in group one of 1/2 rate \(H\)-matrix needs minimum from block columns # 0, 2, 4, 6, 8 and block row two in group one of 1/2 rate \(H\)-matrix need minimum values from block columns # 1, 3, 5, 7, 9 and 10. So, block columns # 0, 2, 4, 6, 8 should be the inputs for one 8-Min unit whereas columns # 1, 3, 5, 7, 9 and 10 should be the inputs of the other 8-Min unit. Thus this local switch can be designed with 16-input Benes network [13][14] to correctly distribute the block columns among both 8-Min units. Also it critically affects the complexity and critical path of the whole architecture. Thus, it needs a special design method to reduce the complexity and critical path. The special design method of local switch is described as follows. A pattern of odd and even is observed between two block rows in each group. Considering group one of rate 1/2, block columns # 0, 2, 4, 6, 8 of block row one are all even but column # 1, 3, 5, 7, 9 and 10 of block row two consists of just one even number block column. Similar patterns are observed in the groups 2, 3 and 4 of rate 1/2 and groups 3 and 4 of rate 5/8. Let’s assume that one 8-Min unit is dedicated to odd columns while other is dedicated to even columns. But it is necessary to feed column number 10 of group one to the odd 8-Min unit rather than even 8-Min unit to get the correct Min1 and Min2 values of block rows 1 and 2. So, this is the basic function of a local switch, i.e. “if the block row is processed by odd 8-Min unit and there is an even entry then move this even entry to the nearest empty odd block column and vice versa”. Odd entries in
The shifting network are connected in odd and even placement to and shown in Fig. 6(c). Fig. 6(a) shows that outputs of Local extremely low complexity Local switch network is designed re-shifting Local switch networks are exactly same. The design and the control signals of these two shifting and another shifting network shown at the end of CNBP in Fig. 9. to their original odd or even positions. This is done through done, the messages again need a re-shifting which moves them columns and vice versa. Once the entire CNBP processing is shift from even block column to the nearest empty odd block shown with the dark background in Fig. 2 (a) and (c). The even block columns and even entries in odd block columns are shown with the dark background in Fig. 2 (a) and (c). The arrows in Fig. 2 (a) and (c) shows which of the entries need a shift from even block column to the nearest empty odd block columns and vice versa. Once the entire CNBP processing is done, the messages again need a re-shifting which moves them to their original odd or even positions. This is done through another shifting network shown at the end of CNBP in Fig. 9. The design and the control signals of these two shifting and re-shifting Local switch networks are exactly same.

Considering the odd-even scheme discussed above, an extremely low complexity Local switch network is designed and shown in Fig. 6(c). Fig. 6(a) shows that outputs of Local shifting network are connected in odd and even placement to the L-adder, while connections from the cyclic switch network are the straight/one-on-one. Similarly, connections between P-adder and local re-shifting network are in odd and even placement as shown in Fig. 6(b). The control values for this local switch network are shown in Table I. “B” is a bar state for 2×2 switch in Fig. 6(c) and “C” is a cross state of a 2×2 switch. If B is considered as zero and C is considered as one then 8-bit decimal equivalent values are shown as Sw_Dec in Table I. Zero control value for a multiplexer means that upper input signal is passed; while one means that lower input signal is passed. Hence four bit equivalent decimal value is shown as Mux_Dec in Table I.

As it is clear from Table I, the local switch is active only for all groups of rate 1/2 and for group three and four of rate 5/8; while for group one and two of rate 5/8 and all groups of rates 3/4 and 13/16, it is completely transparent which means decimal zero control value. The complexity of this local switch network is just 20 multiplexers while critical path is just two multiplexer stages. So, these two local switches only add 40 multiplexers per CNBP along with four multiplexer delay in a critical path.

### Table I: Control Values For Local Switch

<table>
<thead>
<tr>
<th>GROUP-1</th>
<th>GROUP-2</th>
<th>GROUP-3</th>
<th>GROUP-4</th>
</tr>
</thead>
<tbody>
<tr>
<td>Sw0</td>
<td>B - B -</td>
<td>B B C C</td>
<td>B - B -</td>
</tr>
<tr>
<td>Sw1</td>
<td>B - C -</td>
<td>B B C C</td>
<td>B - C -</td>
</tr>
<tr>
<td>Sw2</td>
<td>B - B -</td>
<td>B B C C</td>
<td>B - B -</td>
</tr>
<tr>
<td>Sw3</td>
<td>B - C -</td>
<td>B B B B</td>
<td>B - C -</td>
</tr>
<tr>
<td>Sw4</td>
<td>C - B -</td>
<td>C B B B</td>
<td>C - B -</td>
</tr>
<tr>
<td>Sw5</td>
<td>B - B -</td>
<td>B B C C</td>
<td>B - B -</td>
</tr>
<tr>
<td>Sw6</td>
<td>B - B -</td>
<td>B B C C</td>
<td>B - B -</td>
</tr>
<tr>
<td>Sw7</td>
<td>B - B -</td>
<td>C B B B</td>
<td>B - B -</td>
</tr>
</tbody>
</table>

| Mux0 | 0 - 1 - 0 0 0 0 |
| Mux1 | 1 - 0 - 1 1 0 1 |
| Mux2 | 0 - 0 - 0 0 1 0 |
| Mux3 | 0 - 0 - 1 1 0 0 |
| Mux_Dec | 4 0 8 0 5 5 2 4 |

2) **Sign, index and Min1-Min2 Multiplexers:** Min1-Min2 Multiplexer is responsible for transferring correct Min1 and Min2 values. The coupler is previously described as only becoming active for groups one and two at a rate of 5/8; and all the other groups at rates 3/4 and 13/16. So, it selects the same coupler value at both even and odd sides, while if coupler is inactive as in case of all groups of rate 1/2 and group 3 and 4 of rate 5/8, then it selects different (output of 8-Min stage) values from both even and odd side as shown in Fig. 9. A similar scheme is applicable for sign and index multiplexers. As compared to the single rate architecture shown in Fig. 3, the multi-rate architecture needs two additional index decoders, two index multiplexers, two sign multiplexers and two Min1-Min2 multiplexers.

3) **Local Switch for Skip:** The use of skip was described in the single rate architecture. But for a multi-rate architecture, skip values also require a similar swap as other 16-inputs of CNBP, in order to correctly identify which particular input is skipped. This switch is implemented outside CNBP because the skip signals are the same for all CNBP’s. Thus it does not add an extra complexity to each CNBP.

### D. Multi-frame Pipelining

A famous pipeline technique for LDPC layered decoder is given in [11] and the similar technique is used in [5]. But it is based on the “approximate layered decoding approach” which causes bit-error rate (BER) degradation. LDPC decoder in [11] used two pipeline stages and its BER curve shows some degradation. Similarly three pipeline stages tend to cause more degradation. Hence this technique is not much efficient in terms of BER performance. On the other hand, the architecture in [6] used a multi-frame pipeline technique. But the decoding schedule in [6] is flooding and not layered. MCS (modulation and coding scheme) 1 to 12 [2] clearly shows that the increase in throughput actually follows the increase in number of frames (NCBPS). So, a novel multi-frame pipeline technique is required for a layered decoder. A concurrent processing using multi-frame pipeline is shown in Fig. 10, where a number followed by F denotes which frame is processed and a number followed by G denotes which group processed. Alphabets from A to D denote the sections divided by a pipeline. These sections are marked in the multi-rate architecture as shown in Fig. 9. A total of 16 clock cycles are required to complete an iteration of four different frames. Due to three pipeline stages, three more clock cycles are added at the beginning of first iteration. After the last iteration, an extra clock cycle is required to perform the final shuffle as explained in Section III. This final shuffle is shown in Fig. 10 as F(#)-S/N. Three pipeline stages after LLR registers are good enough to meet high frequency requirement without any BER degradation at all. Synthesis results show that section D is a critical path of the whole architecture.

### V. Arbitrary-Size Benes Cyclic Shifter

The Benes network is a type re-arrange able network [13][14]. A p×p Benes network is capable of performing any
The Benes network is a type re-arrange able network \([13][14]\). A \(p \times p\) Benes network is capable of performing any sort of permutation over \(p\) data units where \(p\) is a power of 2. For 802.11ad, the number of inputs is 42 which is not the power of 2. So, the nearest power of two (\(i.e.\) 64) is applicable. An efficient Benes based network is given in \([16]\) but it is still suffers from \(2^n\) problem. It is clear from \([15]\) that it is possible to create a Benes network for arbitrary number of inputs, which overcome the problem of \(2^n\) number of inputs. This particular network is more efficient complexity-wise as compared with network in \([16]\). To the best of our knowledge, arbitrary-size Benes (AS-Benes) network has never been used in any of the LDPC decoders before.

Based on the method given in \([15]\), a \(42 \times 42\) AS-Benes network design and its complete \(2 \times 2\) switch level implementation is shown in Fig. 7. A total of 194, \(2 \times 2\) switches are required to construct \(42 \times 42\) AS-Benes network. Control signals of AS Benes network are generated using the signal generation algorithm given in \([16]\). The first stage of all even sub-networks is set to the fixed bar state \([16]\). Hence the total of 37 switches is eliminated for a \(42 \times 42\) cyclic shifter, which are shown as dashed lines in Fig. 7. So, a total of 157 (=194-37) \(2 \times 2\) switches are required to construct \(42 \times 42\) AS-Benes cyclic shifter. While the closest network that can be constructed for \(42 \times 42\) is \(48 \times 48\) (=\(3 \times 2^5 \times 3 \times 2^2\)), which requires a total of 240, \(2 \times 2\) switches. So, it is clear that the AS-Benes network gives about 20\% saving in terms of hardware complexity. The control signals for cyclic shifting are pre-computed and stored in a cyclic switch network ROM as shown in Fig. 9.

VI. IMPLEMENTATION AND COMPARISON RESULTS

The proposed multi-mode, multi-block layered and multi-frame decoder was modeled in Verilog HDL and then simulated so as to verify the functionality using a test pattern generated from a C simulator. After complete verification of the design functionality, the proposed design was synthesized using appropriate time and area constraints. Both simulation and synthesis steps were carried out using a SYNOPSYS design tool and 90-nm CMOS technology optimized for 1.08V supply voltage.

\[
Throughput = \frac{frames \times freq \times TotalBits}{clkCycle + \text{pipeline} + 1}
\]  

(8)

\[
Throughput = \frac{4 \times 337 \times 10^7 \times 672}{(16 \times 9) + 4} = 6.1Gbps
\]  

(9)

\[
Throughput = \frac{1 \times 105 \times 10^6 \times 672}{(4 \times 9) + 1} = 1.9Gbps
\]  

(10)

\[
Throughput = \frac{2 \times 337 \times 10^7 \times 672}{(16 \times 9) + 4} = 3.0Gbps
\]  

(11)

Fig. 8. BER curves for multi-rate IEEE 802.11ad.
Fig. 8 depicts superior performance as compared with [6] even with lesser numbers of iterations because layered scheduling provides a faster convergence when compared to the flooding schedule.

Table II shows performance comparison results for the proposed architecture with and without pipelining. The proposed multi-mode pipelined decoder operates at a maximum...
frequency of 337 MHz and it can process four frames at a time. The throughput is calculated using (8). For pipelined decoders, (9) depicts maximum of 6.1 Gbps, while (10) shows that the maximum of 1.9 Gbps can be achieved without pipeline. The pipelined decoder has the capability to operate all the modes [2] (MCS 1 to 12). For modes with a lesser number of frames, the decoder is still capable of achieving the required throughput. For instance MCS 6 to 9 require 2 frames. Thus, 3rd and 4th frame acts as stalls. Despite these stalls, the decoder can still achieve the required throughput as shown in (11).

VII. CONCLUSIONS

A novel multi-mode, multi-frame, multi-block parallel layered LDPC decoder is designed for wireless gigabit communication. The decoder supports all single carrier modes (MSC 1 to 12) given in IEEE 802.11ad. An H-matrix reformulation technique is also introduced to completely eliminate the re-shuffling part in QC-LDPC decoder architecture, which gives a reduction in complexity and critical path. For multi-mode applications, a very low complexity local switch is designed specifically for the IEEE 802.11ad application that converts whole single rate architecture to multi-rate architecture. To achieve high throughput as per requirement of IEEE 802.11ad, a multi-frame pipelining technique is also introduced. With three pipeline stages (with one more stage of LLR registers), the decoder is fully capable of processing 4 frames concurrently. A Benes based low complexity arbitrary-size cyclic shifter is also introduced to perform 42×42 cyclic shifting. The proposed architecture is expected to be incorporated in next-generation wireless gigabit communications.

REFERENCES


TABLE II

IMPLEMENTATION AND PERFORMANCE COMPARISON OF MULTI-MODE LDPC DECODERS FOR 802.11AD.

<table>
<thead>
<tr>
<th></th>
<th></th>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>CMOS tech.</td>
<td>90-nm</td>
<td>90-nm</td>
<td>65-nm</td>
</tr>
<tr>
<td>Max. clock freq.</td>
<td>105 MHz</td>
<td>337 MHz</td>
<td>150 MHz</td>
</tr>
<tr>
<td>65-nm scaled freq. [18]</td>
<td>131 MHz</td>
<td>421 MHz</td>
<td>150 MHz</td>
</tr>
<tr>
<td>Total iterations</td>
<td>9</td>
<td>9</td>
<td>15</td>
</tr>
<tr>
<td>Max. throughput (Gbps)</td>
<td>1.9</td>
<td>6.1</td>
<td>3.0</td>
</tr>
<tr>
<td>Gate count (gates)</td>
<td>584K</td>
<td>898K</td>
<td>-</td>
</tr>
<tr>
<td>Area after synthesis</td>
<td>1.280 mm²</td>
<td>1.967 mm²</td>
<td>1.3 mm²</td>
</tr>
<tr>
<td>65-nm scaled area [18]</td>
<td>0.64 mm²</td>
<td>0.983 mm²</td>
<td>1.3 mm²</td>
</tr>
<tr>
<td>Supply voltage</td>
<td>1.08 V</td>
<td>1.08 V</td>
<td>0.8 V</td>
</tr>
<tr>
<td>Multi-rate</td>
<td>Yes</td>
<td>Yes</td>
<td>Yes</td>
</tr>
<tr>
<td>No. of frames processed concurrently</td>
<td>1</td>
<td>4</td>
<td>2</td>
</tr>
<tr>
<td>Scheduling</td>
<td>Layered</td>
<td>Layered</td>
<td>Flooding</td>
</tr>
<tr>
<td>Quantization</td>
<td>(9, 2)</td>
<td>(9, 2)</td>
<td>(5, 0)</td>
</tr>
<tr>
<td>Required RAM (bits)</td>
<td>12K</td>
<td>48K</td>
<td>-</td>
</tr>
</tbody>
</table>