A 2.4-mm^2 130-mW MMSE-Nonbinary LDPC Iterative Detector Decoder for 4 × 4 256-QAM MIMO in 65-nm CMOS

Wei Tang, *Member, IEEE*, Chia-Hsiang Chen, *Member, IEEE*, and Zhengya Zhang, *Senior Member, IEEE*

***Abstract*— Iterative detection and decoding (IDD) employs a soft-in soft-out (SISO) detector and an SISO forward error correction (FEC) decoder in an iterative loop to improve the receiver performance in multiple-input multiple-output (MIMO) wireless communications. This paper describes a 256-QAM 4 × 4 prototype IDD design made up of a minimum mean square error (MMSE) detector and a nonbinary low-density parity-check (NBLDPC) decoder with the symbol size of the NBLDPC code matched to the modulation to enhance performance. By directly translating between nonbinary symbols and constellation points, the detector–decoder interface is simplified. We present a Gb/s MMSE detector using a shortened tandem scheduling, a low-latency dual-lookup reciprocal unit, an optimized interleaved microarchitecture, and a Gb/s NBLDPC decoder with efficient internal skipping paths and memory allocation. The designs were demonstrated in a 0.7-mm^2 1.38-Gb/s MMSE detector and a 1.7-mm^2 1.02-Gb/s-NBLDPC decoder that are integrated in a 65-nm CMOS test chip. The chip is measured to achieve 19.2 pJ/b in detection and 20.1 pJ/b/iteration in decoding.**

***Index Terms*— Iterative detection and decoding (IDD), minimum mean square error (MMSE) detector, multiple-input multiple-output (MIMO) processor, nonbinary low-density parity-check (NBLDPC) decoder.**

I. Introduction

ADVANCED wireless communication standards, such as IEEE 802.11n/ac and 3GPP LTE Advanced Release 10/11 [1], rely on multiple-input multiple-output (MIMO) communication to increase spectral efficiency and data rate. For example, IEEE 802.11n uses up to 4 × 4 antenna configuration (four-transmit and four-receive antennas), IEEE 802.11ac uses up to 8 × 4 antenna configuration, and 3GPP LTE Advanced Release 10 [1] calls for up to 8 × 8 antenna configuration. The enhancement in spectral efficiency and higher data rates are obtained at a significant computational cost. Workload profiling indicates that MIMO detection at the receiver costs up to 42% of the computing cycles and high power consumption [2].

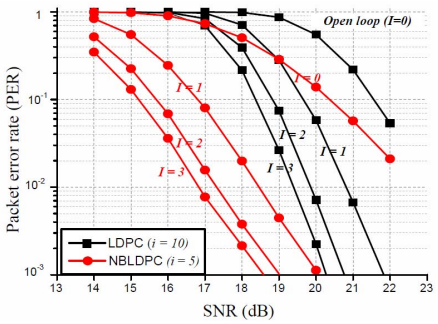


Fig. 1. PER comparison between MMSE-LDPC IDD design and MMSE-NBLDPC GF(16) IDD design under a 4 × 4 256-QAM MIMO system.

The latest MIMO wireless systems have adopted iterative detection and decoding (IDD) to reduce the signal-to-noise ratio (SNR) required for a reliable transmission. An IDD system consists of a soft-in soft-out (SISO) detector to cancel interference and an SISO forward error correction (FEC) decoder to remove errors. Detector and decoder exchange soft information to improve the error rate iteratively.

The state-of-the-art IDDs based on sphere decoding (SD) and binary low-density parity-check (LDPC) FEC have been demonstrated in [3] and [4] for up to 4 × 4 64-QAM systems, achieving up to 396 Mb/s in detection throughput [3] and 586 Mb/s in decoding throughput [4]. As antenna configuration continues to scale beyond 4 × 4 and modulation order increases above 64-QAM, the complexity of an SISO SD detector is expected to grow exponentially, making it impractical. An SISO minimum mean square error (MMSE) detector [5], [6] features a lower complexity and a higher throughput than an SISO SD detector. An MMSE detector can be more easily scaled to support a large antenna configuration and a high-order modulation. The drawback of an MMSE detector is its lower detection performance (measured in error rate). However, an IDD system can overcome this weakness by iteration.

Recent IDD designs have used LDPC codes for FEC [3], [4], [6], [7]. However, binary LDPC codes are not matched to high-order modulations, and a loss is expected. Compared to binary LDPC codes, nonbinary LDPC (NBLDPC) codes defined over the Galois field (GF) outperform binary LDPC codes of a comparable block

Manuscript received October 18, 2018; revised February 14, 2019; accepted February 28, 2019. Date of publication April 11, 2019; date of current version June 26, 2019. This paper was approved by Associate Editor Edith Beigne. This work was supported in part by NSF under Grant CCF-1054270 and in part by Intel. *(Corresponding author: Wei Tang.)*

W. Tang and Z. Zhang are with the Department of Electrical Engineering and Computer Science, University of Michigan, Ann Arbor, MI 48109-2122 USA (e-mail: weitang@umich.edu; zhengya@umich.edu).

C.-H. Chen was with the Department of Electrical Engineering and Computer Science, University of Michigan, Ann Arbor, MI 48109-2122 USA. He is now with the Apple Wireless Group, Cupertino, CA 95014 USA.

Color versions of one or more of the figures in this paper are available online at http://ieeexplore.ieee.org.

Digital Object Identifier 10.1109/JSSC.2019.2904876

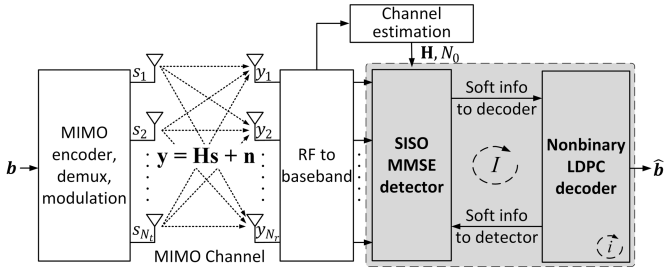


Fig. 2. Illustration of an $N_r \times N_t$ IDD MIMO system with an SISO MMSE detector and an NBLDPC decoder.

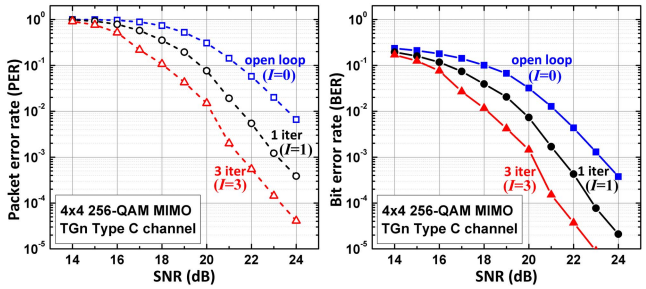


Fig. 3. Performance of the designed 256-QAM 4 × 4 MMSE-NBLDPC IDD chip (with five iterations of NBLDPC decoding).

length in coding gain [8]. Even at a moderate block length, an NBLDPC code offers a superior coding gain, and the coding gain improves with a larger GF size. Used in an IDD system, an NBLDPC code enhances the detection-decoding performance [9]. For example, a 1/2-rate 640-b GF(16) NBLDPC-based IDD system achieves over 2-dB gain over a 1/2-rate 640-b binary LDPC-based IDD system using half as many iterations as shown in Fig. 1.

Despite the good coding gain, the decoding of NBLDPC codes over a large GF(q) requires intensive computation and large memory. To reduce the decoding complexity, Declercq and Fossorier [10] and Voicila *et al.* [11] proposed the extended min-sum (EMS) decoding algorithm, using only n_m, where $n_m \ll q$, most reliable entries in a q-element log-likelihood ratio vector (LLRV) in belief-propagation decoding. The truncation reduces the complexity of elementary decoding operations from $O(q^2)$ to $O(n_m \log n_m)$, with only marginal bit error rate (BER) loss. Using the EMS algorithm, the work by Park *et al.* [12] demonstrated a Gb/s NBLDPC decoder.

In this paper, we present a high-speed 256-QAM 4 × 4 MMSE-NBLDPC IDD implementation. We match the GF size of the NBLDPC code to the QAM constellation size, thereby improving the performance and simplifying the detector–decoder interface. The superb error-correcting capability provided by the NBLDPC code allows us to implement the EMS decoding using only the top dozen entries out of a 256-entry LLRV to reduce the complexity of the decoder. Both the detector and the decoder designs are optimized through the algorithm, architecture, and circuit techniques to achieve higher throughput and lower power compared to the prior art.

The rest of this paper is organized as follows. In Section II, we present the background of the MMSE detection algorithm and the EMS decoding algorithm. Our unique nonbinary interface design is described in Section III. The circuit, architecture, and algorithm co-optimization for the MMSE detector and the NBLDPC decoder are described in Sections IV and V, respectively. Section VI provides the silicon measurement results, and conclusions are drawn in Section VII.

II. Background

The block diagram of an IDD MIMO system is shown in Fig. 2. At the MIMO transmitter, the source bits are encoded by an FEC encoder into a code word. The code word is mapped to QAM symbols $\mathbf{s}[k]$, where $k = 1, \ldots, N_c$, corresponding to the kth OFDM tone (a total of N_c tones used). The symbol vectors are subsequently sent over N_t parallel transmit antennas. The signals travel through a wireless channel that introduces fading, interference, and noise. At the MIMO receiver, N_r antennas pick up the received symbols in every OFDM tone $\mathbf{y}[k], k = 1, \ldots, N_c$. After dropping the index k for the convenience, the per-tone received signal vector $\mathbf{y}$ can be modeled as

$$\mathbf{y} = \mathbf{H}\mathbf{s} + \mathbf{n} \tag{1}$$

where $\mathbf{y} \in \mathbb{C}^{Nr \times 1}$, channel matrix $\mathbf{H} \in \mathbb{C}^{Nr \times Nt}$, transmitted symbols $\mathbf{s} \in \mathbb{C}^{Nt \times 1}$, and the complex Gaussian noise $\mathbf{n}$ is modeled as $\mathcal{CN}(0, N_0)$.

An SISO MMSE detector performs MMSE filtering to cancel the interference and outputs the soft symbols and variances that represent the estimated symbols in a signal constellation and the likelihoods of the symbols, respectively. The soft symbols and variances are converted to a prior log-likelihood ratio (LLR) to be used in an SISO FEC decoder. An SISO FEC decoder performs error correction and outputs the posterior LLRs. The posterior LLRs are converted to soft symbols and variances and fed back to the SISO detector for the next IDD iteration. IDD iterations improve the quality of detection and decoding. A successful convergence is indicated by the convergence of soft symbols and narrowing of variances.

Fig. 3 shows the BER and frame error rate (FER) curves of the proposed 256-QAM 4 × 4 IDD system. The channel model is a 4 × 4 TGn Type-C channel [13]. The error rates improve with IDD iterations: a 3-dB gain is achieved from zero iteration ($I = 0$, also called open loop) to three iterations ($I = 3$). The performance gain is at the cost of higher latency and energy of receiver processing. In Sections II-A and II-B, we provide a brief introduction to MMSE detection and NBLDPC decoding.

A. MMSE Detection

In this paper, we use the MMSE parallel interference cancellation (MMSE-PIC) algorithm based on [5] as described in the following. In an IDD system, step 1 (pre-processing) is only done in the first iteration.

1) *Pre-Processing:* Compute Gram matrix $\mathbf{G} = \mathbf{H}^{\mathrm{H}}\mathbf{H}$ and perform match filtering $\mathbf{y}^{MF} = \mathbf{H}^{\mathrm{H}}\mathbf{y}$.
2) *Initialization:* Compute soft symbols s_t and variances σ_t^2, $t = 1, \ldots, N_t$, using the decoder's LLRs in the previous IDD iteration (in the first iteration, s_t and σ_t^2

are initialized to zero and average symbol energy E_s, respectively) and obtain MMSE filter matrix $\mathbf{A}$

$$\mathbf{A} = \mathbf{G}\Lambda + N_0\mathbf{I} \tag{2}$$

where $\Lambda = \text{diag}(\sigma_1^2, \ldots, \sigma_{N_t}^2)$.

3) *Matrix Inversion:* Compute $\mathbf{A}^{-1}$ by performing lower-upper decomposition (LUD) to get $\mathbf{A} = \mathbf{LU}$, followed by forward substitution (f-sub) to get $\mathbf{L}^{-1}$ and backward substitution (b-sub) to get $\mathbf{A}^{-1}$.
4) *Interference Cancellation:*

$$\mathbf{y}_t^{IC} = \mathbf{y}^{MF} - \sum_{j \neq t} \mathbf{g}_j s_j, \quad t = 1 \ldots N_t \tag{3}$$

where $\mathbf{g}_j$ is the jth column of $\mathbf{G}$.
5) *MMSE Filtering:*

$$\begin{aligned} \hat{s}_t &= \mu_t^{-1} \mathbf{a}_t^H \mathbf{y}_t^{IC}, \quad t = 1 \ldots N_t \\ \hat{\sigma}_t^2 &= \mu_t^{-1} - \sigma_t^2 \end{aligned} \tag{4}$$

where $\mathbf{a}_t^H$ denotes the tth row of $\mathbf{A}^{-1}$ and $\mu_t = \mathbf{a}_t^H \mathbf{g}_t$.

The outputs of the detector are the estimated soft symbols $\hat{s}_t$ and variances $\hat{\sigma}_t^2$. In an IDD system, the soft symbols and variances are converted to LLRs for FEC decoding.

B. Nonbinary LDPC Code and EMS Decoding

An $N \times M$ regular-(d_v, d_c) NBLDPC code over GF(q) can be depicted in a bipartite graph that consists of N VNs and M CNs. Each VN is connected to d_v CNs and each CN is connected to d_c VNs. The weight of the connection between VN j and CN i is $\alpha_{j,i}$, $\alpha_{j,i} \in$ GF(q).

An NBLDPC code is decoded by passing the messages between VNs and CNs. A message passed between a VN and a CN is a vector of q LLRs, called an LLRV, containing one LLR per GF(q) symbol. The EMS decoding algorithm keeps only n_m, $n_m \ll q$, most reliable LLRs, and uses a GF index vector (GFIV) to keep track of the n_m symbols. The LLRs in an LLRV are sorted and normalized: the LLR value of the most reliable GF symbol is set to 0 and the remaining LLR values are normalized to it. The steps of EMS decoding are described in the following. In iterative decoding, the first step (initialization) is done only in the first iteration.

1) *Initialization:* VNs are initialized with prior LLRVs $\mathbf{x}$.
2) *VN to CN Propagation:* Each VN sends a V2C message to each of the d_v-connected CNs. The message from VN j to CN i is denoted $\mathbf{u}_{j,i}$. The GFIV of each message is GF multiplied by $\alpha_{j,i}$, a permutation operation.
3) *CN Processing:* Each CN receives d_c V2C messages and computes C2V messages using (5). The message from CN i to VN j is denoted $\mathbf{v}_{i,j}$

$$\mathbf{v}_{i,j} = \bigoplus_{k \in \mathcal{M}(i), k \neq j} \mathbf{u}_{k,i} \tag{5}$$

where $\mathcal{M}(i)$ is the set of VNs connected to CN i, and $\bigoplus$ is performed using the forward–backward algorithm [14].

TABLE I
COMPARISON BETWEEN STANDARD CONVERSION METHOD AND DIRECT CONVERSION METHOD FROM SOFT SYMBOL TO LLRV

	Standard Conversion	Direct Conversion
Soft symbol to LLRs	$2n+1$ adds, $2n+2$ mults (bit-LLRs)	$2\lceil\sqrt{n_m}\rceil$ adds (symbol-LLRs)
LLRV construct	$n_m \times (n-1)$ adds	n_m adds
Total	$2n+2$ mults, $n_m(n-1)+2n+1$ adds	$2\lceil\sqrt{n_m}\rceil + n_m$ adds

4) *CN to VN Propagation:* Each CN sends C2V messages to the d_c-connected VNs. The GFIV of each C2V message is GF divided by $\alpha_{j,i}$, an inverse permutation.
5) *VN Processing:* Each VN receives d_v C2V messages and computes V2C messages $\mathbf{u}_{j,i}$ and posterior LLRVs $\mathbf{z}_j$ using the following equation:

$$\begin{aligned} \mathbf{u}_{j,i} &= \mathbf{x}_j + \sum_{k \in \mathcal{N}(j), k \neq i} \mathbf{v}_{k,j} \\ \mathbf{z}_j &= \mathbf{x}_j + \sum_{k \in \mathcal{N}(j)} \mathbf{v}_{k,j} \end{aligned} \tag{6}$$

where $\mathcal{N}(j)$ is the set of CNs connected to VN j, and Σ is performed using the elementary processing steps [11].

In an IDD system, the decoder's output LLRs are converted to soft symbols s and variances σ^2 for detection.

III. DETECTOR–DECODER INTERFACE AND OPTIMIZATION

An MMSE detector processes soft symbols, while an NBLDPC decoder processes LLRVs. Translations between soft symbols and LLRVs are required to implement an IDD system. Assume a 2^n-QAM constellation that is widely used in wireless communication systems. In this paper, we propose to match the QAM constellation and the GF size to enable the direct and simplified translations between soft symbols and LLRVs without any information loss. Matching the constellation and GF size provides the highest performance [15].

A. Converting Soft Symbol to LLRV

In a conventional method, a soft symbol $\hat{s}_t$ in a 2^n-QAM constellation is converted to LLRV in two steps: 1) convert $\hat{s}_t$ to n bit-LLRs [16] and 2) assemble the bit-LLRs to symbol-LLRs [9] from n_m nearest neighbors and construct LLRV. The bit-by-bit conversion requires searching constellation points to find the nearest neighbors of $\hat{s}_t$ for each bit. The search can be done along the real and imaginary axes independently to narrow the search space, as shown in Fig. 4(a). The computational complexity of the bit-by-bit conversion is listed in Table I. Note that the search and bit LLR computation in Table I can be further simplified if the Gray mapping is used [17].

If the constellation and the GF size are matched, we propose a direct conversion method to bypass the heavy bit-LLR compute: 1) directly convert $\hat{s}_t$ to symbol-LLRs using (7) and

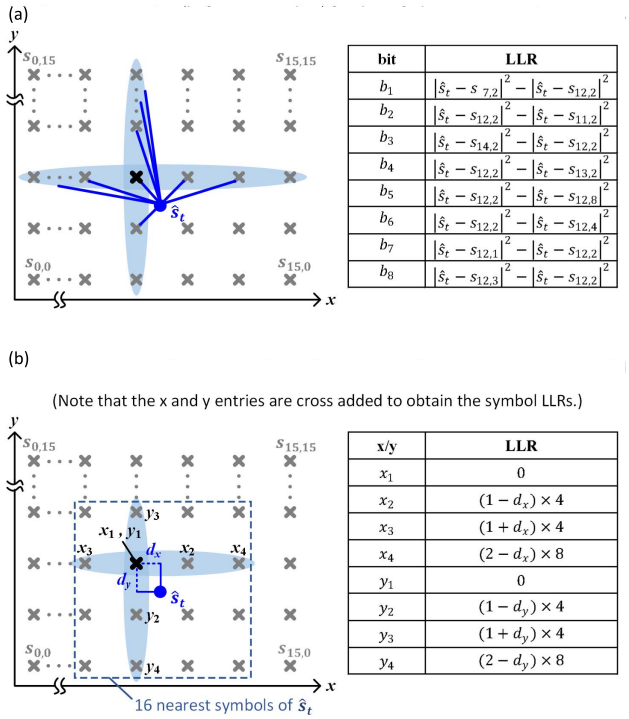


Fig. 4. Example of (a) bit-LLR and (b) symbol-LLR computations before SNR scaling for the soft detector output in 256-QAM. Note that the x and y entries are cross-added to obtain the symbol-LLRs.

2) construct LLRV from n_m symbol-LLRs

$$L_r = \ln \frac{P(s=r|\hat{s}_t)}{P(s=s_{x_1,y_1}|\hat{s}_t)} = \frac{1}{\hat{\sigma}_t^2}(|\hat{s}_t - r|^2 - |\hat{s}_t - s_{x_1,y_1}|^2) \quad (7)$$

where r is a GF symbol, representing a constellation point, and s_{x_1,y_1} represents the reference constellation point closest to $\hat{s}_t$, as shown in Fig. 4(b). Due to normalization, $L_{s_{x_1,y_1}} = 0$. In a QAM constellation, the real and imaginary parts of the LLR can be computed independently and summed.

An illustration of the direct conversion step 1) is shown in Fig. 4(b). The distance between the soft symbol $\hat{s}_t$ and its nearest neighbor s_{x_1,y_1} is d. The projection of d on the x- and y-axes are d_x and d_y, respectively. Without loss of generality, assume that the constellation points are spaced by 2. It follows that the distance from $\hat{s}_t$ to its second nearest constellation point along the x-axis, x_2, is $2-d_x$ and to the second nearest constellation point along the y-axis, y_2, is $2-d_y$. The real and imaginary parts of the LLR can be computed as follows:

$$\begin{aligned} L_{x_2} &= \frac{1}{\hat{\sigma}_t^2}(|2-d_x|^2 - |d_x|^2) = \frac{4}{\hat{\sigma}_t^2}(1-d_x), \\ L_{y_2} &= \frac{1}{\hat{\sigma}_t^2}(|2-d_y|^2 - |d_y|^2) = \frac{4}{\hat{\sigma}_t^2}(1-d_y). \end{aligned} \quad (8)$$

Notice that the square terms are canceled, and the calculation only requires ℓ_1 distance.

The direct conversion step 2) prepares LLRV from symbol-LLRs. In the EMS decoding of NBLDPC code, an LLRV consists of the LLRs of the n_m nearest symbols. In a QAM constellation, they are located within the dashed box, as shown in Fig. 4(b). Thus, the n_m nearest symbols' LLRs can be computed by cross-adding the distances to the $\lceil n_m{}^{(1/2)} \rceil$ nearest neighbors along the x- and y-axes, where $\lceil \cdot \rceil$ is the ceiling function.

TABLE II
COMPARISON BETWEEN STANDARD CONVERSION METHOD AND APPROXIMATE CONVERSION METHOD FROM LLRV TO SOFT SYMBOL

	Standard Conversion	Approximate Conversion
LLR → prob.	n_m table lookups	1 table lookup, 1 add
Soft symbol & variance compute	$3n_m+1$ mults, $2n_m+1$ adds	5 mults, 2 adds
Total	n_m table lookups, $3n_m+1$ mults, $2n_m+1$ adds	1 table lookup, 5 mults, 3 adds

The complexity of the direct conversion only depends on the choice of n_m, as shown in Table I. Since $n_m \ll 2^n$, the direct conversion is especially advantageous for large constellations. For instance, converting a 256-QAM soft symbol and its variance to a GF(256) LLRV ($n_m = 16$) requires 18 multiplies, 129 adds using the conventional bit-by-bit method, compared to only 24 adds using the direct conversion method.

B. Converting LLRV to Soft Symbol

In a conventional method, an LLRV is converted to a soft symbol in two steps: 1) convert symbol-LLRs in an LLRV to probabilities of the corresponding constellation points (the step is often done by table lookups) and 2) combine the positions of the constellation points weighted by their probabilities to compute the soft symbol and variance.

Following the EMS decoding of NBLDPC code, an LLRV consists of n_m most likely GF symbol-LLRs. To further simplify the conversion, we apply an approximation by choosing only the top two most likely GF symbol-LLRs. Suppose the two most likely GF symbols from the decoder are mapped to QAM symbols s_0 and s_1. Step 1 of the conversion is reduced to the following:

$$\begin{aligned} P_{\mathsf{s}_0} &= \frac{\exp(0.5L_{\mathsf{s}_0})}{\exp(0.5L_{\mathsf{s}_0}) + \exp(-0.5L_{\mathsf{s}_0})} \\ P_{\mathsf{s}_1} &= 1 - P_{\mathsf{s}_0}. \end{aligned} \quad (9)$$

The underlying assumption is that the two most likely symbols dominate. Thus, the probability of the second most likely symbol is approximated by $P_{\mathsf{s}_1} = 1 - P_{\mathsf{s}_0}$. With only two most likely symbols to consider, step 2 of the conversion is reduced to the following:

$$\begin{aligned} s_t &= P_{\mathsf{s}_0}\mathsf{s}_0 + P_{\mathsf{s}_1}\mathsf{s}_1 \\ \sigma_t^2 &= P_{\mathsf{s}_0}(\mathsf{s}_0 - s_t)^2 + P_{\mathsf{s}_1}(\mathsf{s}_1 - s_t)^2 \\ &= P_{\mathsf{s}_0}P_{\mathsf{s}_1}(\mathsf{s}_0 - \mathsf{s}_1)^2. \end{aligned} \quad (10)$$

The complexity of the approximate conversion is fixed regardless of the constellation size or choice of n_m, as shown in Table II. Using the approximate conversion, a GF(256) LLRV to a 256-QAM soft symbol and variance conversion requires only one table lookup, five multiplies, and three adds, a significant simplification over the conventional method

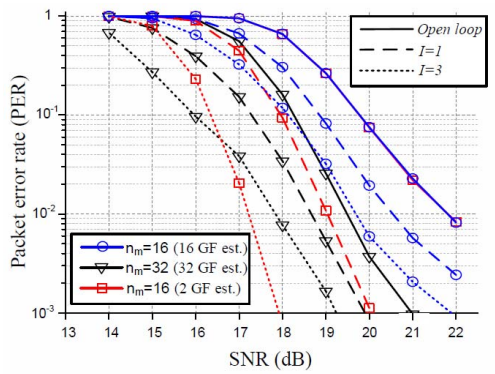


Fig. 5. Performance comparison among three setups for a 4 × 4 265-QAM IDD system using MMSE detection and a GF(256) NBLDPC code using: 1) $n_m = 16$ in NBLDPC decoding and the 16 symbol-LLRs for soft symbol estimation (standard conversion); 2) $n_m = 32$ in NBLDPC decoding and the 32 symbol-LLRs for soft symbol estimation (standard conversion); and 3) using $n_m = 16$ in NBLDPC decoding and only the two most likely symbol-LLRs for soft symbol estimation (approximate conversion).

that requires 16 table lookups ($n_m = 16$), 49 multiplies, and 33 adds.

In Fig. 5, the error rate performances of the approximate conversion ($n_m = 32$, using only two symbols for conversion from LLRV to soft symbol) are compared with two standard conversions ($n_m = 16$ and $n_m = 32$, using all n_m symbols for conversion from LLRV to soft symbol). In the open-loop case, i.e., no iterations, the $n_m = 32$ standard conversion provides the best packet error rate (PER); the $n_m = 16$ standard conversion and the $n_m = 32$, two-symbol approximate conversion perform worse (note that these two curves overlap). Once iterations are turned ON, the $n_m = 32$, the two-symbol approximate conversion starts to outperform the two standard conversions. At three iterations and 10^{-2} PER, the approximate conversion provides the best performance.

IV. MMSE Detector Design

The MMSE detection is comprised of five functional steps: 1) initialization to compute soft symbols and MMSE filter matrix $\mathbf{A}$; 2) matrix inversion; 3) interference cancellation; 4) MMSE filtering; and 5) post-processing to compute updated soft symbols and variances. Note that step 2 and step 3 can be overlapped due to the lack of data dependence.

To shorten the latency, the functional steps need to be pipelined, and the latency of each step needs to be balanced. To improve the throughput, the cycle period also needs to be minimized. Among the five steps, matrix inversion and MMSE filtering cost the longest latency and the highest complexity. These two stages are the focus of our optimization.

A. Tandem Scheduling

Matrix inversion is done in three substeps: LUD, f-sub, and b-sub. A $N_t \times N_t$ matrix $\mathbf{A}$ is first decomposed to a lower and an upper triangular matrices, $\mathbf{L}$ and $\mathbf{U}$, using LUD. $\mathbf{L}^{-1}$ is then found by solving for $\mathbf{X}$ in $\mathbf{LX} = \mathbf{I}$ using f-sub. Finally, $\mathbf{A}^{-1}$ is found by solving for $\mathbf{Y}$ in $\mathbf{UY} = \mathbf{L}^{-1}$ using b-sub.

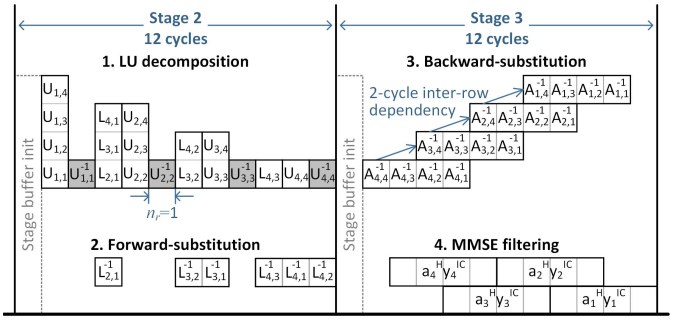


Fig. 6. Tandem scheduling of matrix inversion and MMSE filtering. Here, the element in the ith row and the jth column of the matrices $\mathbf{L}$, $\mathbf{U}$, $\mathbf{L}^{-1}$, $\mathbf{U}^{-1}$, and $\mathbf{A}^{-1}$ are indexed by the subscripts i and j.

LUD follows the Gaussian elimination that operates from the top row to the bottom row of matrix $\mathbf{A}$, obtaining $\mathbf{L}$ from the left to the right column and $\mathbf{U}$ from the top to the bottom row. In each step, LUD uses a reciprocal unit to compute the inverse of the diagonal element of $\mathbf{U}$. Assume $\mathbf{A}$ is 4 × 4 and suppose the reciprocal unit takes n_r cycles, a multiply-add (MAC) takes 1 cycle, and 16 parallel real-valued MACs are allocated, the critical path of LUD can be packed in $4n_r + 8$ cycles. Under this critical path, f-sub can be performed in tandem with LUD to hide its latency. As soon as the first element of $\mathbf{L}$ is available, f-sub can start. In this way, f-sub and LUD complete at the same time, as shown in Fig. 6. Once the last row of $\mathbf{L}^{-1}$ is found and buffered, b-sub starts from the bottom row to the top row of $\mathbf{L}^{-1}$ to compute $\mathbf{A}^{-1}$.

MMSE filtering is done by vector inner products: $\mathbf{a}_t^H \mathbf{y}_t^{IC}$, $t = 1, \ldots, N_t$. Recall that $\mathbf{a}_t^H$ denotes the tth row of $\mathbf{A}^{-1}$, and $\mathbf{y}_t^{IC}$ is the output of the interference cancellation step. We propose the tandem scheduling of b-sub and MMSE filtering. As soon as an element of $\mathbf{A}^{-1}$ is available, the corresponding product with $\mathbf{y}_t^{IC}$ can be performed. In this way, MMSE completes in only one cycle after f-sub is done.

With tandem scheduling, the matrix inversion and MMSE filtering are reduced from three coarse pipeline stages [5] to two stages, as shown in Fig. 6. Tandem scheduling also cuts the number of boundary registers between stages by 85%, as the output from the previous step is immediately consumed by the subsequent step.

B. Dual-Lookup Reciprocal Unit

Reciprocal is in the critical path of matrix inversion and dominates the latency. A popular reciprocal design is based on the Newton–Raphson division algorithm. Suppose we need to find $x = (1/d)$, the problem can be formulated as finding the root of $f(x) = (1/x) - d = 0$. Applying the Newton–Raphson method, the root can be found by iteration with an initial estimate x_0

$$x_{i+1} = x_i - f(x_i)/f'(x_i) = 2x_i - dx_i^2. \tag{11}$$

A baseline reciprocal unit is shown in Fig. 7(a) [5]. The initial estimate x_0 is retrieved from a lookup table (LUT). To reduce the LUT size, only the MSB bits are used to address the LUT. Two multiplies are needed to compute dx_i^2, which is

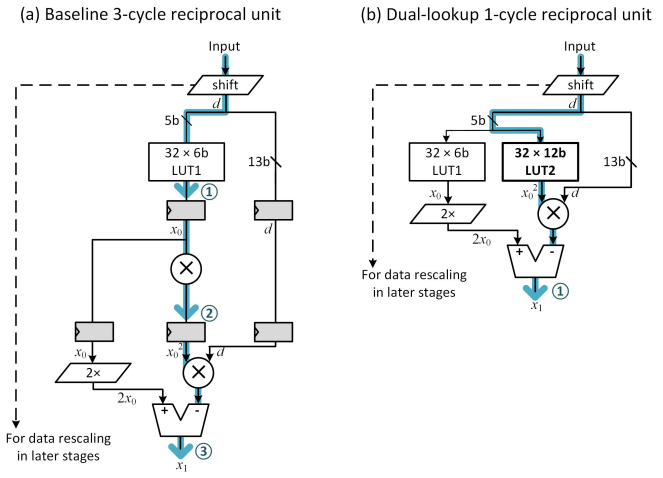


Fig. 7. Reciprocal unit designs. (a) Baseline three-cycle design from [5]. (b) Dual-lookup single-cycle design. The critical paths are highlighted in blue.

then subtracted from $2x_i$ to compute the reciprocal. A better approximation can be obtained by iterations. For an MMSE detector, it was shown that a 32 × 6 b LUT and one iteration are sufficient [5]. The baseline design is naturally divided into three pipeline stages, costing three cycles.

In the baseline design, the latency of the reciprocal unit is dominated by the two multiplies. To reduce latency, we design a dual-lookup reciprocal unit, as shown in Fig. 7(b), using two LUTs: a 32 × 6 b LUT for retrieving x_0 and a 32 × 12 b LUT for obtaining x_0^2. The addition of the 32 × 12 b LUT allows the redesigned reciprocal unit to have only one-cycle delay, which translates to six-cycle latency reduction of the matrix inversion stage. To make the best use of the limited LUT size, we apply dynamic scaling of the matrix **A** based on symbol variance, such that the input to the reciprocal unit falls in the range of $[1, 2)$. The designed reciprocal unit provides an average precision of 0.00044 and a maximum error of 0.0017 that are sufficient for an MMSE detector [18].

C. Relaxed Timing by Interleaving

The MMSE detector is pipelined to four stages, as shown in Fig. 8. Initialization is in stage 1. The tandem scheduling of LUD and f-sub allow them to be grouped in stage 2, and the tandem scheduling of b-sub and MMSE filtering allows them to be grouped in stage 3. Due to the lack of data dependence, interference cancellation is also done in stage 2. Post-processing is done in stage 4.

Despite the optimizations done in the stages 2 and 3 of the pipeline, the long critical paths in the multipliers present a tight timing constraint. To loosen the constraint, we use a simple clock divider to create a 2× slow clock domain for stages 2 and 3 to allow the gates to be downsized and maintain the throughput across the two stages by duplicating the datapaths and interleaving between the two copies, as shown in Fig. 8. After gate downsizing, the duplication costs only 24% additional area over the baseline, as depicted in Fig. 9, but the throughput is increased by 38%, thanks to a higher clock frequency. The downsized gates also reduce the load capacitance, thus improving the energy efficiency.

V. NBLDPC Decoder Design

We choose a GF(256) NBLDPC code to match the 256-QAM constellation. To reduce the implementation cost of an NBLDPC decoder, we use a relatively short (52, 26) regular-(2, 4) NBLDPC code over GF(256) [19], [20] with a binary block length of 416 bits. In decoding, we adopt the EMS algorithm using $n_m = 12$.

A. Fully Parallel Architecture

To match the throughput and latency of the MMSE detector, the NBLDPC decoder is fully parallelized with 52 VNs and 26 CNs, as shown in Fig. 10. After the MMSE detection, the soft symbols and variances are translated to prior LLRVs for initializing VNs.

To start decoding, each VN passes V2C messages to the connected CNs through a routing network. Each CN generates C2V messages and sends them back to the connected VNs. The VNs use the C2V messages to update the V2C messages to send to the connected VNs for the next iteration. The decoding stops when the maximum iteration limit is reached. The fully parallel architecture achieves high throughput and low latency, but the data dependencies between CNs and VNs as well as within their internal stages cause inefficiency due to the pipeline stalls.

B. Low-Latency VN Design

A VN receives $d_v = 2$ C2V messages (LLRVs) and computes V2C messages (LLRVs) to start the next iteration. The VN processing is implemented by two elementary VNs (EVNs) [12] as well as a memory to store prior LLRV and two content addressable memories (CAMs) to store the two C2V LLRVs, one per EVN.

VN processing starts by loading C2V LLRVs to the two CAMs in $n_m = 12$ cycles. In the second step, 12 symbol-LLRs from the prior LLRV are read from memory one by one, from the most likely to the least, and sent to the two EVNs. Each EVN searches the symbol in its CAM. The matching symbol-LLR is read from the CAM and summed with the symbol's prior LLR. In the third step, the updated symbol-LLR is inserted to a sorter, and it takes 12 cycles to produce the complete V2C LLRV. The critical path of CN processing is 36 cycles, as shown in Fig. 11.

In the baseline design described earlier, VN processing cannot start until C2V LLRVs from the CNs are received and loaded to the CAMs. To cut the stall, we allow C2V LLRVs from the CNs to be directly forwarded to the EVNs instead of being stored in CAMs and relocate prior LLRV to a CAM. Since prior LLRV is updated only at the beginning of an iterative decoding and remains stationary, this relocation eliminates the between-iteration stall due to data loading.

With the C2V forwarding, incoming C2V LLRVs are streamed to the EVNs, and an EVN searches the CAM for a matching symbol in the prior LLRV. Only one CAM

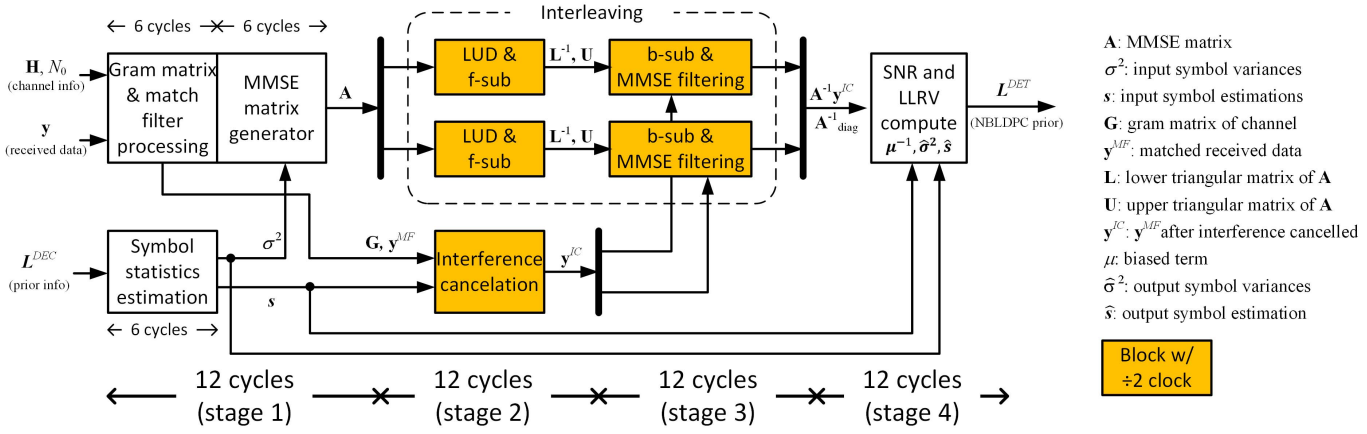


Fig. 8. Design of the MMSE detector in four task-based coarse pipeline stages. Stages 2 and 3 operate at a 2× slower clock frequency, and the remaining stages operate at the base clock frequency.

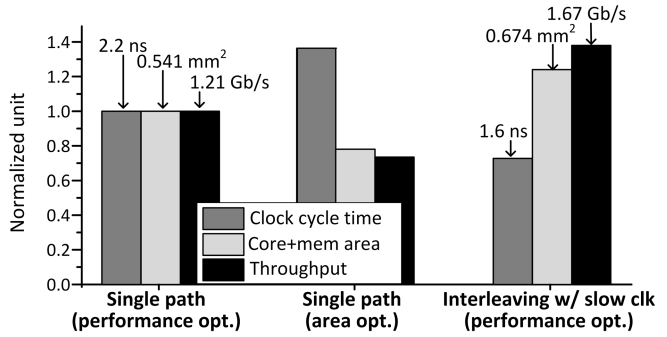


Fig. 9. Comparison between single-path performance/area optimized designs and dual-path interleaving with slow clock.

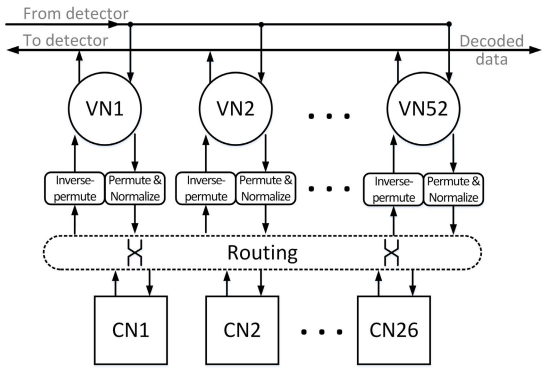


Fig. 10. Fully parallel architecture of NBLDPC decoder.

is required, but the CAM needs to provide two ports to support the independent reads by the two EVNs. The C2V loading latency is eliminated to reduce the critical path of CN processing to 25 cycles, as illustrated in Fig. 11.

Thanks to the simplified decoder–detector interface, the compute of posterior LLRV is reduced to finding the two most likely symbols by a small CAM and a simplified EVN. The posterior compute does not add to the critical path. In all, the

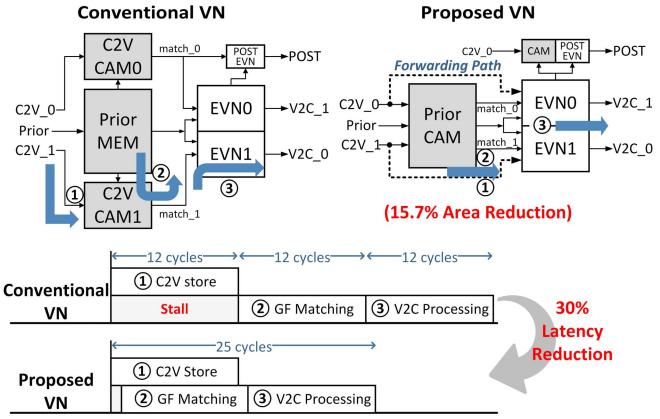


Fig. 11. Dataflow and latency of the conventional and the proposed VN designs.

proposed VN design uses 31% less storage, and the area is 15.7% smaller than the conventional VN design.

C. Low-Latency CN Design

A CN performs a parity check of $d_c = 4$ input V2C messages (LLRVs) and produces C2V messages (LLRVs) to send back to the connected VNs. The CN processing is implemented using the forward–backward algorithm [11] by six elementary CNs (ECNs), including a forward ECN, a backward ECN, and four merge ECNs, as well as six memory blocks to store the input LLRVs and the intermediate results [12], as shown in Fig. 12.

An ECN looks for the $n_m = 12$ most likely pairings of symbols from the two input V2C LLRVs, namely, LLRV1 and LLRV2. To support $n_m = 12$, an ECN uses a six-element insertion sorter [21]. The sorter queue is first loaded with the top six symbol-LLRs from LLRV1 in six cycles. After initialization, the top symbol-LLRs from LLRV2 are read from memory one by one and paired with the top entries from LLRV1 following the bubble check algorithm [22]. The paired symbols are summed to a combined symbol (GF addition).

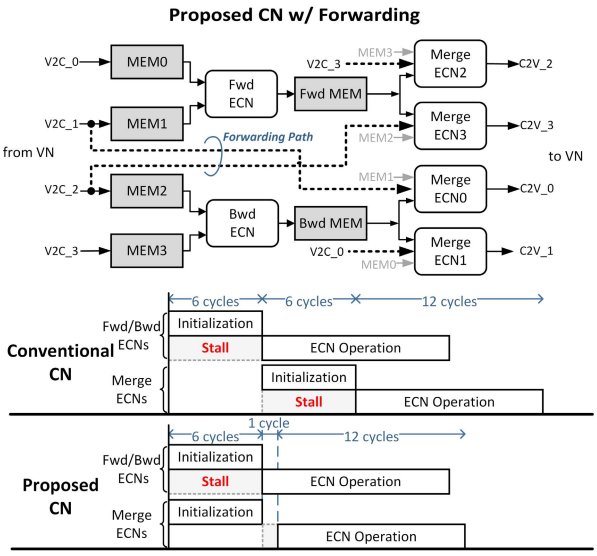


Fig. 12. Proposed CN design with V2C forwarding; the pipeline schedules of conventional and proposed CN design.

The LLR value of the combined symbol is computed and inserted to a sorter queue, and the top of the sorter queue is popped as the output of the ECN. It takes 12 cycles to produce the 12 symbol-LLRs to form an LLRV.

This baseline ECN design incurs a six-cycle stall in initialization. Though the forward/backward ECN latency can be hidden, as shown in Fig. 12, the merge ECN's initialization cannot be hidden because it requires reading from memory. We add data-forwarding paths to initialize merge ECNs concurrently with forward/backward ECNs, as shown in Fig. 12. The data forwarding allows the CN latency to be shortened from 24 to 19 cycles.

The baseline ECN sorter design uses shift registers to store symbol indices, memory indices, and LLR values [12]. We observe that the symbol indices are unused during sorting, wasting switching power to shift unused entries. We eliminate symbol index memory in ECN sorter to reduce its buffer size by 36%, which translates to 20% area reduction and 12% power reduction for one CN.

VI. Clock Gating Exploiting Regular Access

A total of 70.9-kb registers are used for buffering data in and between stages of the detector and the decoder. Registers are used in place of memory arrays to support high access bandwidth and the flexibility of placing small memory blocks. Registers are power hungry, but we recognize a power reduction opportunity, as most of the registers used in our design are regularly but infrequently updated due to the task-based pipeline stages, e.g., one update every 12 cycles for the 7.6-kb stage boundary registers in the detector and one update every 25 cycles for the 26.2-kb CN buffer registers in the decoder. A detailed tally of register usage and update frequency is shown in Fig. 13. We exploit the regular access to reduce power by enabling clock gating of the registers when they are idling, saving the detector power and the decoder power by 53% and 61%, respectively.

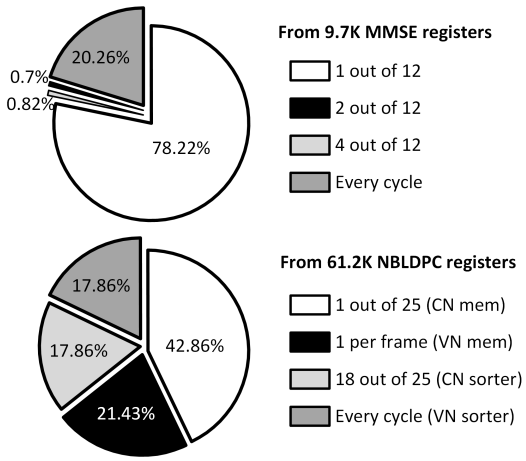


Fig. 13. Power breakdown and the activities of registers.

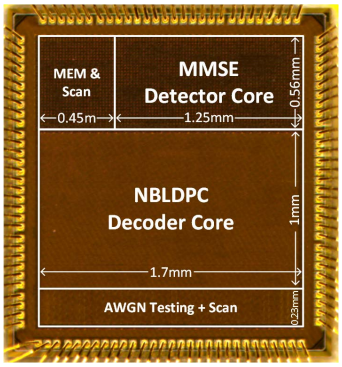


Fig. 14. Die photograph of the MMSE-NBLDPC IDD chip.

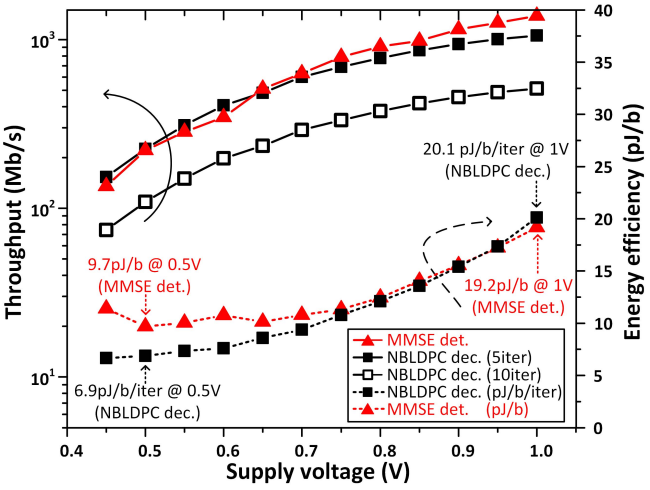


Fig. 15. Measured throughput and energy efficiency with voltage scaling.

VII. Chip Measurement Results

The MMSE-NBLDPC iterative detector–decoder test chip is fabricated in TSMC 65-nm technology. The die photograph is shown in Fig. 14. The chip dimension is 2.04 mm × 2.2 mm, and the MMSE detector core and the NBLDPC decoder core occupy 0.7 and 1.7 mm^2, respectively. At room temperature

TABLE III
COMPARISON WITH STATE-OF-THE-ART MIMO DETECTORS AND LDPC DECODERS

Detector	Noethen [3]	Borlenghi [4]	Winter [7]	Sun [6]	Studer [5]	This work
IDD design	Yes	Yes	No	Yes	Yes	Yes
Algorithm	SD SISO	SD SISO	SD SO	MMSE SISO	MMSE SISO	MMSE NB-SISO
MIMO system	$\leq 4 \times 4$	$\leq 4 \times 4$	$\leq 4 \times 4$	$\leq 4 \times 4$	4×4	4×4
Modulation	≤64	≤64	≤64	≤16	≤64	256
Technology [nm]	65	65	65	40	90	65
Core area [mm^2]	-	2.78	0.31	-	1.5	0.7
Preprocessing area [kGE]	383[a]	-[b]	-[b]	489	410	347[c]
Detection area [kGE]		872	215			
Frequency [MHz]	445	135	333	288	568	517
Power [mW]	87	-	38	-	189	26.5
Throughput [Mb/s]	396	194	296-807	2304	757	1379
Area efficiency [Mb/s/kGE]	1.03	0.22	1.37-3.75	4.96	1.85	3.68
Energy efficiency [pJ/b]	220	920	48	21.8	250	19.2

Decoder	Noethen [3]	Borlenghi [4]	Winter [7]	Sun [6]	Park [12]	This work
IDD design	Yes	Yes	No	Yes	No	Yes
Code	LDPC	LDPC	LDPC	LDPC	NBLDPC GF(64)	NBLDPC GF(256)
Block length	768	1944	768	1944	960	416
Technology [nm]	65	65	65	40	65	65
Core area [mm^2]	-	0.78	3.6	-	7.04	1.7
Decoding area [kGE]	-	-	-	509[e]	2780	935
Frequency [MHz]	500	299	267	288	700	307
Power [mW]	-	-	367	-	3866	103
Iterations	10	10	10	3[d]	10-30[d]	5-10
Throughput [Mb/s]	155	586	235.2	-[e]	1150	1024-512
Area efficiency [Mb/s/mm^2]	100.92	751	65.33	-[e]	163	602-301
Energy efficiency [pJ/b/iter]	232	21	170	-[e]	277	20.1

[a] : memory for data exchange is included.
[b] : data pre-processing block (QRD) is not included.
[c] : total area is 264 kGE if no interleaving processing.
[d] : with early termination.
[e] : not reported for decoder. The throughput, area efficiency, and energy efficiency of the entire IDD system are 794Mb/s, 0.79Mb/s/kGE, and 170pJ/bit respectively.

and 1.0-V supply, the MMSE detector runs at a maximum frequency of 517 MHz for a throughput of 1.38 Gb/s, and the NBLDPC decoder runs at 307 MHz for a throughput of 1.02 Gb/s (five iterations), as shown in Fig. 15.

Our work is compared with the state-of-the-art MIMO IDD designs in Table III. The MMSE detector achieves higher reported throughput of an SISO MMSE detector [5]. The MMSE detector consumes only 19.2 pJ/b, an order of magnitude lower than previous SISO detector designs [3]–[5]. The NBLDPC decoder consumes 20.1 pJ/b/iteration, the lowest reported energy of an NBLDPC decoder [12], and it matches the efficiency of the binary LDPC decoder used in IDD [4]. Although our NBLDPC code is about half the size of [12], the one order of magnitude improvement in energy [12] is significant. The energy efficiency can be further improved by voltage and frequency scaling, as shown in Fig. 15. At 500-mV supply, the MMSE detector and the NBLDPC decoder consume 9.7 and 6.9 pJ/b/iteration, respectively, for throughputs above 200 Mb/s.

Our test chip is a proof of concept of an IDD system supporting a high-order modulation using a matching NBLDPC symbol. The matching provides the best performance and efficiency in conversions between soft symbols and LLRs. In a low-SNR case, a high-order modulation is not applicable. A possible solution is to pack multiple constellation symbols to a GF(256) code symbol. For example, in 16-QAM, two sets of I/Q symbol LLRVs are packed to a GF(256) symbol LLRV. Any mismatch will complicate the conversions and may lead to non-optimal performance. Further study is needed in the area of flexible NBLDPC decoder design to support the adjustable GF size and rate to match the modulation in order to achieve the best performance and efficiency.

VIII. CONCLUSION

We demonstrate an MMSE-NBLDPC IDD system for a 256-QAM 4×4 MIMO system to achieve an excellent error rate that improves with iterations. By matching the constellation and GF size of the nonbinary FEC code, soft symbols and symbol-LLRs between the detector and the decoder can be directly converted, simplifying the interface and making the IDD design practical.

To minimize latency over the iterative loop and improve throughput, tandem scheduling and a new dual-lookup reciprocal unit are employed to reduce the latency of the detector, and the critical paths of the detector are interleaved and placed in a slow clock domain to support a high throughput at a low cost. The resulting MMSE detector design achieves an 82% higher throughput and almost $3.5\times$ the throughput of the latest

SD detector. The NBLDPC decoder is fully parallelized to support the highest throughput. Internal data forwarding paths are created, and memory organization is optimized to reduce the decoding latency by 30% over the latest NBLDPC decoder design.

To lower the power consumption, automatic clock gating is applied to stage boundary and buffer registers to save 53% of the detector power and 61% of the decoder power. We demonstrate a 65-nm MMSE-NBLDPC iterative detector–decoder test chip that achieves 1.38 Gb/s in detection and 1.02 Gb/s in decoding (five iterations), consuming 26.5 and 103 mW, respectively.

REFERENCES

[1] 3GPP. *3GPP Standards Website*. [Online]. Available: http://www.3gpp.org

[2] F. Sheikh *et al.*, "3.2 Gbps channel-adaptive configurable MIMO detector for multi-mode wireless communication," in *Proc. IEEE Workshop Signal Process. Syst. (SiPS)*, Oct. 2014, pp. 1–6.

[3] B. Noethen *et al.*, "10.7 A 105 GOPS 36 mm^2 heterogeneous SDR MPSoC with energy-aware dynamic scheduling and iterative detection-decoding for 4G in 65 nm CMOS," in *IEEE ISSCC Dig. Tech. Papers*, Feb. 2014, pp. 188–189.

[4] F. Borlenghi, E. M. Witte, G. Ascheid, H. Meyr, and A. Burg, "A 2.78 mm^2 65 nm CMOS gigabit MIMO iterative detection and decoding receiver," in *Proc. ESSCIRC*, Sep. 2012, pp. 65–68.

[5] C. Studer, S. Fateh, and D. Seethaler, "ASIC implementation of soft-input soft-output MIMO detection using MMSE parallel interference cancellation," *IEEE J. Solid-State Circuits*, vol. 46, no. 7, pp. 1754–1765, Jul. 2011.

[6] W.-C. Sun, W.-H. Wu, C.-H. Yang, and Y.-L. Ueng, "An iterative detection and decoding receiver for LDPC-coded MIMO systems," *IEEE Trans. Circuits Syst. I, Reg. Papers*, vol. 62, no. 10, pp. 2512–2522, Oct. 2015.

[7] M. Winter *et al.*, "A 335 Mb/s 3.9 mm^2 65 nm CMOS flexible MIMO detection-decoding engine achieving 4G wireless data rates," in *IEEE ISSCC Dig. Tech. Papers*, Feb. 2012, pp. 216–218.

[8] M. C. Davey and D. MacKay, "Low-density parity check codes over GF(q)," *IEEE Commun. Lett.*, vol. 2, no. 6, pp. 165–167, Jun. 1998.

[9] S. Pfletschinger and D. Declercq, "Getting closer to MIMO capacity with non-binary codes and spatial multiplexing," in *Proc. IEEE Global Telecommun. Conf. (GLOBECOM)*, Dec. 2010, pp. 1–5.

[10] D. Declercq and M. Fossorier, "Decoding algorithms for nonbinary LDPC codes over GF(q)," *IEEE Trans. Commun.*, vol. 55, no. 4, pp. 633–643, Apr. 2007.

[11] A. Voicila, D. Declercq, F. Verdier, M. Fossorier, and P. Urard, "Low-complexity decoding for non-binary LDPC codes in high order fields," *IEEE Trans. Commun.*, vol. 58, no. 5, pp. 1365–1375, May 2010.

[12] Y. S. Park, Y. Tao, and Z. Zhang, "A fully parallel nonbinary LDPC decoder with fine-grained dynamic clock gating," *IEEE J. Solid-State Circuits*, vol. 50, no. 2, pp. 464–475, Feb. 2015.

[13] V. Erceg, L. Schumacher, and P. Kyritsi, *TGn Channel Models*, document IEEE 802.11-03/940r1, Garden Grove, CA, USA, 2004.

[14] H. Wymeersch, H. Steendam, and M. Moeneclaey, "Log-domain decoding of LDPC codes over GF(q)," in *Proc. IEEE Int. Conf. Commun.*, vol. 2, Jun. 2004, pp. 772–776.

[15] D. Declercq, M. Colas, and G. Gelle, "Regular GF (2q)-LDPC modulations for higher order QAM-AWGN channels," in *Proc. Int. Symp. Inf. Theory Appl. (ISITA)*, Parma, Italy, 2004, pp. 1–6.

[16] J. Erfanian, S. Pasupathy, and G. Gulak, "Reduced complexity symbol detectors with parallel structure for ISI channels," *IEEE Trans. Commun.*, vol. 42, no. 234, pp. 1661–1671, Apr. 1994.

[17] I. B. Collings, M. R. G. Butler, and M. McKay, "Low complexity receiver design for MIMO bit-interleaved coded modulation," in *Proc. 8th IEEE Int. Symp. Spread Spectr. Techn. Appl.*, Aug./Sep. 2004, pp. 12–16.

[18] D. Auras, R. Leupers, and G. H. Ascheid, "A novel reduced-complexity soft-input soft-output MMSE MIMO detector: Algorithm and efficient VLSI architecture," in *Proc. IEEE Int. Conf. Commun. (ICC)*, Jun. 2014, pp. 4722–4728.

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

[20] A. Venkiah, D. Declercq, and C. Poulliat, "Design of cages with a randomized progressive edge-growth algorithm," *IEEE Commun. Lett.*, vol. 12, no. 4, pp. 301–303, Apr. 2008.

[21] Y. Tao, Y. S. Park, and Z. Zhang, "High-throughput architecture and implementation of regular (2, dc) nonbinary LDPC decoders," in *Proc. IEEE Int. Symp. Circuits Syst.*, May 2012, pp. 2625–2628.

[22] E. Boutillon and L. C. Canencia, "Bubble check: A simplified algorithm for elementary check node processing in extended min-sum non-binary LDPC decoders," *Electron. Lett.*, vol. 46, no. 9, pp. 633–634, Apr. 2010.

[23] A. Tomasoni, M. Ferrari, D. Gatti, F. Osnato, and S. Bellini, "A low complexity turbo MMSE receiver for W-LAN MIMO systems," in *Proc. IEEE Int. Conf. Commun.*, vol. 9, Jun. 2006, pp. 4119–4124.

[24] E. Agrell, T. Eriksson, A. Vardy, and K. Zeger, "Closest point search in lattices," *IEEE Trans. Inf. Theory*, vol. 48, no. 8, pp. 2201–2214, Aug. 2002.

[25] M. O. Damen, H. El Gamal, and G. Caire, "On maximum-likelihood detection and the search for the closest lattice point," *IEEE Trans. Inf. Theory*, vol. 49, no. 10, pp. 2389–2402, Oct. 2003.

[26] A. Burg, M. Borgmann, M. Wenk, M. Zellweger, W. Fichtner, and H. Bolcskei, "VLSI implementation of MIMO detection using the sphere decoding algorithm," *IEEE J. Solid-State Circuits*, vol. 40, no. 7, pp. 1566–1577, Jul. 2005.

[27] E. M. Witte, F. Borlenghi, G. Ascheid, R. Leupers, and H. Meyr, "A scalable VLSI architecture for soft-input soft-output single tree-search sphere decoding," *IEEE Trans. Circuits Syst., II, Exp. Briefs*, vol. 57, no. 9, pp. 706–710, Sep. 2010.

[28] L. Liu, "Energy-efficient soft-input soft-output signal detector for iterative MIMO receivers," *IEEE Trans. Circuits Syst. I, Reg. Papers*, vol. 61, no. 8, pp. 2422–2432, Aug. 2014.

[29] S. A. Laraway and B. Farhang-Boroujeny, "Implementation of a Markov chain Monte Carlo based multiuser/MIMO detector," *IEEE Trans. Circuits Syst. I, Reg. Papers*, vol. 56, no. 1, pp. 246–255, Jan. 2009.

[30] X. Wang and H. V. Poor, "Iterative (turbo) soft interference cancellation and decoding for coded CDMA," *IEEE Trans. Commun.*, vol. 47, no. 7, pp. 1046–1061, Jul. 1999.

[31] Y.-T. Chen, C.-C. Cheng, T.-L. Tsai, W.-C. Sun, Y.-L. Ueng, and C.-H. Yang, "A 501 mW 7.6 lGb/s integrated message-passing detector and decoder for polar-coded massive MIMO systems," in *Proc. Symp. VLSI Circuits*, Jun. 2017, pp. C330–C331.

[32] W.-H. Wu, W.-C. Sun, C.-H. Yang, and Y.-L. Ueng, "A 794 Mbps 135 mW iterative detection and decoding receiver for 4×4 LDPC-coded MIMO systems in 40 nm," in *Proc. Symp. VLSI Circuits (VLSI Circuits)*, Jun. 2015, pp. C102–C103.

[33] S. Song, B. Zhou, S. Lin, and K. A. Abdel-Ghaffar, "A unified approach to the construction of binary and nonbinary quasi-cyclic LDPC codes based on finite fields," *IEEE Trans. Commun.*, vol. 57, no. 1, pp. 84–93, Jan. 2009.

[34] B. Zhou, J. Kang, S. W. Song, S. Lin, K. Abdel-Ghaffar, and M. Xu, "Construction of non-binary quasi-cyclic LDPC codes by arrays and array dispersions—[Transactions papers]," *IEEE Trans. Commun.*, vol. 57, no. 6, pp. 1652–1662, Jun. 2009.

[35] X. Zhang, F. Cai, and S. Lin, "Low-complexity reliability-based message-passing decoder architectures for non-binary LDPC codes," *IEEE Trans. Very Large Scale Integr. (VLSI) Syst.*, vol. 20, no. 11, pp. 1938–1950, Nov. 2012.

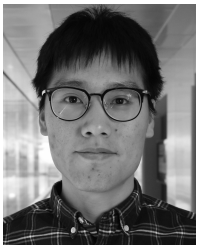
[36] X. Chen and C.-L. Wang, "High-throughput efficient non-binary LDPC decoder based on the simplified min-sum algorithm," *IEEE Trans. Circuits Syst. I, Reg. Papers*, vol. 59, no. 11, pp. 2784–2794, Nov. 2012.

[37] Y.-L. Ueng, K.-H. Liao, H.-C. Chou, and C.-J. Yang, "A high-throughput trellis-based layered decoding architecture for non-binary LDPC codes using max-log-QSPA," *IEEE Trans. Signal Process.*, vol. 61, no. 11, pp. 2940–2951, Jun. 2013.

[38] J. Lin and Z. Yan, "An efficient fully parallel decoder architecture for nonbinary LDPC codes," *IEEE Trans. Very Large Scale Integr. (VLSI) Syst.*, vol. 22, no. 12, pp. 2649–2660, Dec. 2014.

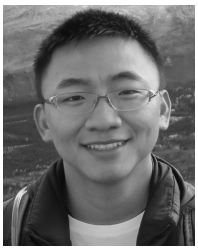
[39] C.-L. Lin, C.-L. Chen, H.-C. Chang, and C.-Y. Lee, "Jointly designed nonbinary LDPC convolutional codes and memory-based decoder architecture," *IEEE Trans. Circuits Syst. I, Reg. Papers*, vol. 62, no. 10, pp. 2523–2532, Oct. 2015.

[40] Y.-L. Ueng, C.-Y. Leong, C.-J. Yang, C.-C. Cheng, K.-H. Liao, and S.-W. Chen, "An efficient layered decoding architecture for nonbinary QC-LDPC codes," *IEEE Trans. Circuits Syst. I, Reg. Papers*, vol. 59, no. 2, pp. 385–398, Feb. 2012.
[41] R. Yazdani and M. Ardakani, "Efficient LLR calculation for non-binary modulations over fading channels," *IEEE Trans. Commun.*, vol. 59, no. 5, pp. 1236–1241, May 2011.
[42] H. Kaul *et al.*, "A 1.45 GHz 52-to-162 GFLOPS/W variable-precision floating-point fused multiply-add unit with certainty tracking in 32 nm CMOS," in *IEEE ISSCC Dig. Tech. Papers*, Feb. 2012, pp. 182–184.



Wei Tang (S'15–M'19) received the B.S. degree from National Chiao Tung University, Hsinchu, Taiwan, in 2011, and the M.S. and Ph.D. degrees in electrical engineering from the University of Michigan at Ann Arbor, Ann Arbor, MI, USA, in 2019.

He was a Visiting Ph.D. Student with Lund University, Lund, Sweden, and a Graduate Research Intern with Intel Labs, Santa Clara, CA, USA. He is currently a Post-Doctoral Research Fellow with the University of Michigan at Ann Arbor. His research interests are in high-speed, energy-efficient detector and forward error correction decoder designs for small-scale multiple-input multiple-output (MIMO) and massive MIMO systems.



Chia-Hsiang Chen (S'10–M'14) received the B.S. degree from National Chiao Tung University, Hsinchu, Taiwan, in 2008, and the M.S. and Ph.D. degrees in electrical engineering from the University of Michigan at Ann Arbor, Ann Arbor, MI, USA, in 2012 and 2014, respectively.

In 2015, he joined Intel Labs, Santa Clara, CA, USA, with a focus on architecture and system for low-power and wireless communication. He has been with the Apple Wireless Group, Cupertino, CA, USA, since 2017.

Zhengya Zhang (S'02–M'09–SM'17) received the B.A.Sc. degree in computer engineering from the University of Waterloo, Waterloo, ON, Canada, in 2003 and the M.S. and Ph.D. degrees in electrical engineering from the University of California at Berkeley (UC Berkeley), Berkeley, CA, USA, in 2005 and 2009, respectively.

Since 2009, he has been a Faculty Member with the University of Michigan at Ann Arbor, Ann Arbor, MI, USA, where he is currently an Associate Professor with the Department of Electrical Engineering and Computer Science. His current research interests include low-power and high-performance VLSI circuits and systems for computing, communications, and signal processing.

Dr. Zhang was a recipient of the David J. Sakrison Memorial Prize from UC Berkeley in 2009, the National Science Foundation CAREER Award in 2011, and the Intel Early Career Faculty Award in 2013. He serves on the Technical Program Committees of the Symposium on VLSI Circuits and the IEEE Custom Integrated Circuits Conference (CICC). He was an Associate Editor of the IEEE TRANSACTIONS ON CIRCUITS AND SYSTEMS—PART I: REGULAR PAPERS from 2013 to 2015 and the IEEE TRANSACTIONS ON CIRCUITS AND SYSTEMS—PART II: EXPRESS BRIEFS from 2014 to 2015. He has been an Associate Editor of the IEEE TRANSACTIONS ON VERY LARGE SCALE INTEGRATION (VLSI) SYSTEMS since 2015.