

Pilotless Frame Synchronization for LDPC-Coded Transmission Systems

Dong-U Lee, *Member, IEEE*, Hyungjin Kim, *Student Member, IEEE*, Christopher R. Jones, *Member, IEEE*, and John D. Villasenor, *Senior Member, IEEE*

Abstract—We present a pilotless frame synchronization approach that exploits feedback from a low-density parity-check (LDPC) code decoder. The synchronizer is based on syndrome checks using hard decisions from the channel observations. The bandwidth overhead associated with pilot symbols in conventional receiver architectures is eliminated while providing sufficient synchronization performance. An LDPC decoder coupled with our synchronizer exhibits negligible frame error rate degradation over a system with perfect synchronization. The complexity of the frame synchronizer is kept relatively low due to its XOR-based approach.

Index Terms—Block codes, channel coding, frame synchronization.

I. INTRODUCTION

RECENT advances in iteratively decoded channel codes such as low-density parity-check (LDPC) codes make it possible to operate at capacity-approaching signal-to-noise-ratios (SNRs). This in turn places more stringent requirements on the frame synchronization portions of receivers, which must successfully acquire frames at these lower SNRs. Conventional frame synchronizers rely on the insertion of pilot symbols. The correlation between the predefined pilot symbols and the received signal is examined to determine the correct frame boundary. Since similar patterns to the pilot symbols can appear in erroneous frame boundaries and the pilot symbols themselves are corrupted by noise, pilot symbols require a significant portion of the bandwidth and signal energy. For instance, the addition of a 78-bit pilot to a frame of length 1944 bits leads to a bandwidth efficiency loss of 0.17 dB.

In this paper, we show that the parity-checks inherent in LDPC codes can provide information that can direct the estimation of the frame boundary without the use of any pilot symbols. LDPC codes are commonly represented using a bipartite graph containing two sets of nodes. The variable nodes correspond to

Manuscript received July 12, 2007; revised December 6, 2007. The associate editor coordinating the review of this manuscript and approving it for publication was Dr. Mounir Ghogho. This work was supported in part by the Jet Propulsion Laboratory, the National Science Foundation under Grants CR-0120778 and CCF-0541453, and the Office of Naval Research under Contract N00014-06-1-0253.

D. Lee was with the Electrical Engineering Department, University of California, Los Angeles, CA 90095 USA. He is now with Mojix, Inc., Los Angeles, CA 90025 USA (e-mail: dongu@mojix.com).

H. Kim and J. D. Villasenor are with the Electrical Engineering Department, University of California, Los Angeles, CA 90095 USA (e-mail: hjkimnov@ee.ucla.edu; villa@icssl.ucla.edu).

C. R. Jones is with the Jet Propulsion Laboratory, Pasadena, CA 91109 USA (e-mail: christop@jpl.nasa.gov).

Digital Object Identifier 10.1109/TSP.2008.917348

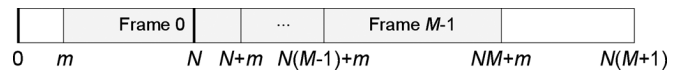


Fig. 1. Synchronizer buffer structure.

the codeword symbols and the constraint nodes represent the constraints that the code places on the variable nodes in order for them to form a valid codeword. The decoding procedure involves iterative computation of values associated with these nodes. After each decoding iteration, the metrics associated with each constraint node can be evaluated to determine the status of the associated parity-check. Normally, this information is utilized only within the LDPC decoding process to assess the convergence behavior of the iterative processing. A valid decoded codeword is obtained if all parity-check equations are satisfied. However, we show here that it has value in the frame synchronization as well. It will be shown that the behavior of the constraint nodes provides a useful metric to assess underlying accuracy of the frame offset estimates.

Throughout this paper, assume that the LDPC encoded frames are modulated via binary phase shift keying (BPSK) and perturbed by an additive white Gaussian noise (AWGN) channel with zero mean and variance of $N_0/2$ where N_0 is the one-sided noise spectral density. The received signal is buffered at the synchronizer as shown in Fig. 1, where N is the frame size, m is the true frame offset, and M is the number of consecutive frames used for synchronization. The task of the frame synchronizer is to find an estimate \hat{m} of m where $\hat{m} \in [0, N - 1]$. If $m = \hat{m}$, then frame synchronization is achieved.

The rest of this paper is organized as follows. Section II discusses previous work on frame synchronization. Section III examines the constraint node behavior of LDPC codes. Section IV describes the proposed pilotless frame synchronization methodology that exploits the constraint node behavior. Section V discusses hardware implementation considerations. Section VI provides experimental results and concluding remarks are given in Section VII.

II. RELATED WORK

Pilot-based synchronization is a well-established field. Massey [1] describes an optimal maximum-likelihood (ML) soft-decision correlation rule for locating periodically inserted pilot symbols within random data transmitted over an AWGN channel with BPSK modulation, which is later extended for M -ary symbol sets by Lui and Tan [2]. At the expense of increased complexity, performance gains of several decibels are observed over the conventional correlation rule. Reduced

complexity estimations of the optimal ML rule are also given in [1] for low and high SNRs. Nielsen [3] subsequently conducts simulation studies on Massey's correlation rules and demonstrates that the high SNR estimation rule performs as well as the optimal ML rule over a wide range of practical SNRs. Dolinar and Cheung [4] compare the synchronization performance of several variants of the standard correlation rule that operates on channel observations and on decoded bits. They show that correlators based on decoded bits are superior to those based on channel observations if the desired operating point utilizes a miss probability of many orders of magnitude higher than the false alarm probability.

Previous treatments addressing code-aided frame synchronization have focused on convolutional codes and turbo codes. Lorden *et al.* [5] propose a synchronization algorithm for a rate 1/2 Viterbi decoder. That algorithm works on hard quantized data in conjunction with a syndrome generator. Synchronization is judged by monitoring the pattern of the syndromes. The algorithm in [5] is later generalized by de Mateo [6] for rate 1/n codes. Mostofa *et al.* [7] embed pilot symbols within the encoded information sequence using a turbo decoder to assist the synchronization process. Their approach is based on the principle that although an error in trellis termination can cause errors in the decoded information sequence, the beginning portion occupied by the pilot symbols can be decoded error-free. Sun and Valenti [8] describe a maximum *a posteriori* frame synchronization method for turbo-coded transmission systems. The parity check equations of the turbo code are examined to identify the most probable start of the frame. Cassaro and Georghiadis [9] and Mielczarek and Svensson [10], [11] utilize the sum (or mean) of the magnitudes of log-likelihood ratios (LLRs) of the turbo decoder to assist frame synchronization. The rationale is that correctly aligned frames are likely have a higher sum than those that are unaligned.

Little work exists on LDPC code-aided frame synchronization. Matsumoto and Imai [12] use a similar approach to [9]–[11] in which the mean of the LLR magnitudes of the variable nodes is examined after an LDPC decoding iteration. While this approach provides good synchronization performance, it requires that a full LDPC decoding iteration to be performed for every possible frame offset candidate. Wymeersch *et al.* [13], [14] perform maximum-likelihood (ML) code-aided synchronization based on the expectation-maximum (EM) algorithm. In order to avoid the local maximum convergence problem of the EM algorithm, the authors propose a discrete EM method which locates the global maximum via an exhaustive search. The correct frame offset is found by correlating the received symbols (channel observations) against the LDPC decoded soft symbols. Similarly to [12], this approach has the downside of the need for LDPC decoding iterations. In contrast to the methods above, we exploit information available from the constraint nodes of the LDPC code. Thus, instead of doing a full LDPC iteration, we utilize hard decisions of the received symbols to compute the parity-check equations for each constraint node, thereby limiting the majority of the computations to simple XOR operations.

Some earlier aspects of this work were presented in [15]. The present paper presents substantial new results on this topic, in-

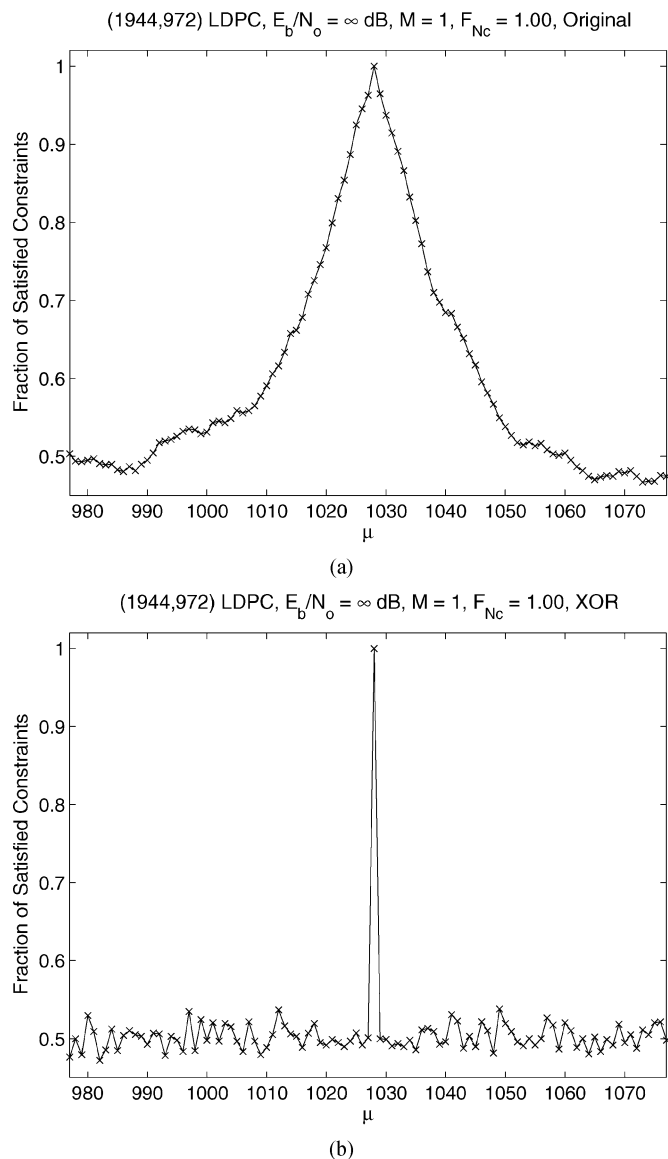


Fig. 2. Variation of fraction of satisfied constraints when using the original codeword and when the codeword is XORed with a PN sequence. (a) Original codeword. (b) Codeword XORed with a PN sequence.

cluding improved treatment for masking out undesirable properties of quasi-cyclic LDPC codes, the introduction of several new synchronization strategies, detailed analysis of constraint node behavior, examination of different LDPC codes, and a comprehensive set of results.

III. CONSTRAINT NODE BEHAVIOR

The (1944,972) quasi-cyclic irregular LDPC code proposed for the IEEE 802.11n standard [16] is used. The code has 810 degree-7 and 162 degree-8 constraint nodes. Fig. 2(a) shows the variation of the fraction of satisfied constraints with the frame offset μ . No noise is added, M is set to one, and F_{Nc} which determines the fraction of the constraints to be examined is set to one as well (i.e., all 972 constraints are examined). The true frame offset m is at $\mu = 1028$. The quasi-cyclic nature of the code yields ramps around m . The (1944,972) code examined in the figure has a quasi-cyclic periodicity (circulant size) of

81. The width of the ramp between offsets that yield a fraction of satisfied constraints near 50% is similar to the underlying circulant size of the code. This is clearly an undesirable property for a frame offset discriminator.

We are aware of at least two strategies that eliminate these ramps: 1) permute the intracirculant symbols of each codeword or 2) XOR the codeword with a frame-aligned pseudorandom noise (PN) sequence. Again, the (144,972) code considered in Fig. 2(a) uses size 81 circulant permutation matrices. Hence, the codeword can be permuted in groups of 81 bits to remove the ramps. An alternative and a more economical approach to implementing codeword permutation leverages a PN sequence. Many standards specify that symbols be multiplied by a random sequence prior to modulation so that dense symbol-phase transitions are likely to occur in the transmitted waveform. Given a PN sequence, z , that restarts at each codeword boundary, then $(c+z)H = zH = S$, the syndrome of the PN sequence, where c is the codeword and H is the parity-check matrix. At the receiver, constraint nodes should satisfy S rather than 0. Fig. 2(b) shows the variation of the fraction of satisfied constraints when the codeword is XORed with a PN sequence. Ramps are no longer observed since constraint S does not provide graceful degradation for shifts of size 81 or less (as is the case for unaltered codewords constrained against $S = 0$).

Fig. 3 shows the variation of the fraction of satisfied constraints with the frame offset μ at $E_b/N_0 = 3$ dB and $E_b/N_0 = 1$ dB respectively where E_b is the energy per bit. Both M and F_{N_c} are set to one. The true frame offset m in both plots is at $\mu = 1028$. Fig. 3(a) shows that at a high SNR, when $\mu \neq 1028$ (i.e., unsynchronized) around half of the constraints are satisfied, and when $\mu = 1028$ (i.e., synchronized) there is a clear peak. In a synchronization strategy that returns the offset which gives the largest number of satisfied constraints as the estimate $\hat{\mu}$, a true positive would be achieved. At a lower SNR however, as shown in Fig. 3(b), the variable nodes are heavily corrupted by noise, making it difficult to distinguish between correct and incorrect frame offsets. The synchronization strategy would fail, returning a false positive offset estimate of $\hat{\mu} = 1764$.

In order to characterize the performance of synchronization strategies such as the one discussed above, we need to obtain pdfs of the fraction of satisfied constraints for the cases when the frame is unsynchronized and when the frame is synchronized (i.e., when $\hat{\mu} \neq m$ and $\hat{\mu} = m$). The probability of a bit error for the binary symmetric channel is

$$p = Q\left(\sqrt{\frac{2E_s}{N_0}}\right) \quad (1)$$

where Q is the complementary cumulative normal distribution function and E_s is the energy per symbol. The probability that a constraint node of degree d_c is not satisfied is given by

$$P_{d_c} = \sum_{k=1}^{\lceil d_c/2 \rceil} \binom{d_c}{2k-1} p^{2k-1} (1-p)^{d_c-(2k-1)}. \quad (2)$$

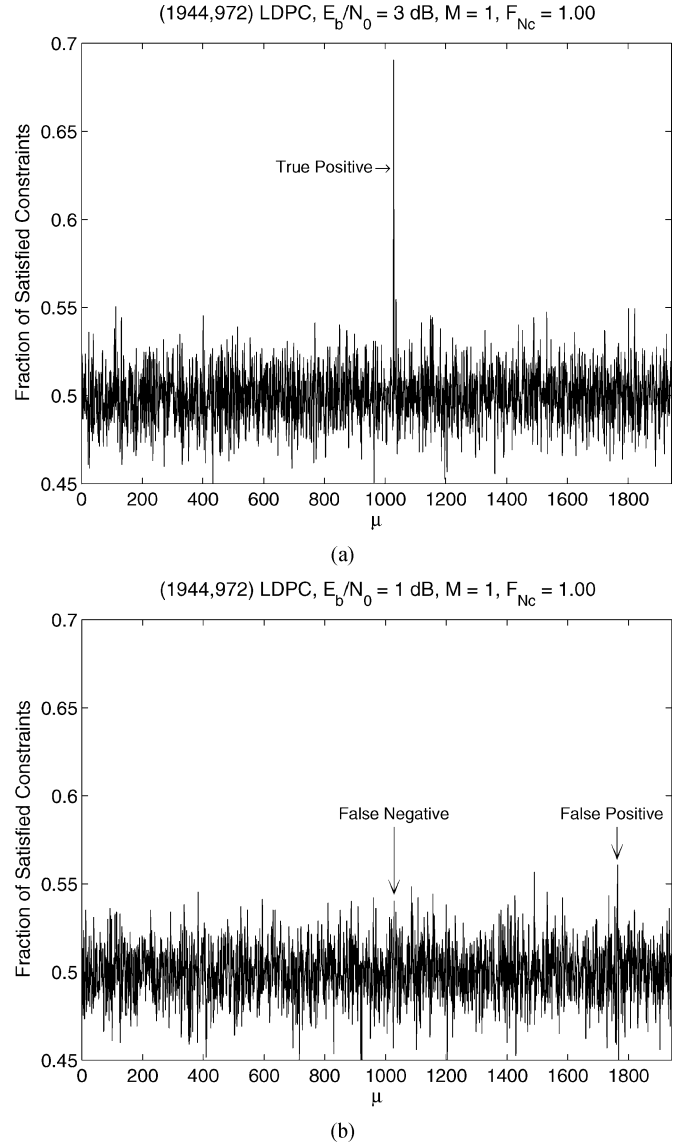


Fig. 3. Variation of fraction of satisfied constraints with μ at different SNRs. (a) $E_b/N_0 = 3$ dB. (b) $E_b/N_0 = 1$ dB.

The probability that a random constraint node is not satisfied is given by

$$\gamma = \sum_{j=d_{c,\min}}^{d_{c,\max}} \rho_j P_j \quad (3)$$

where $d_{c,\min}$ and $d_{c,\max}$ are the minimum and maximum constraint node degrees, and ρ_j is the constraint node degree distribution of the code. Using the equations above, the probability of k satisfied constraints from a set of N_c constraint nodes can be derived

$$P_{N_c}^k = \binom{N_c}{k} (1-\gamma)^k \gamma^{N_c-k}. \quad (4)$$

A potential problem with (4) is that it assumes that the variable nodes connected to the constraint nodes are independent of one another. This implies that no two constraints share the same variable node, i.e., all variable nodes are degree-1.

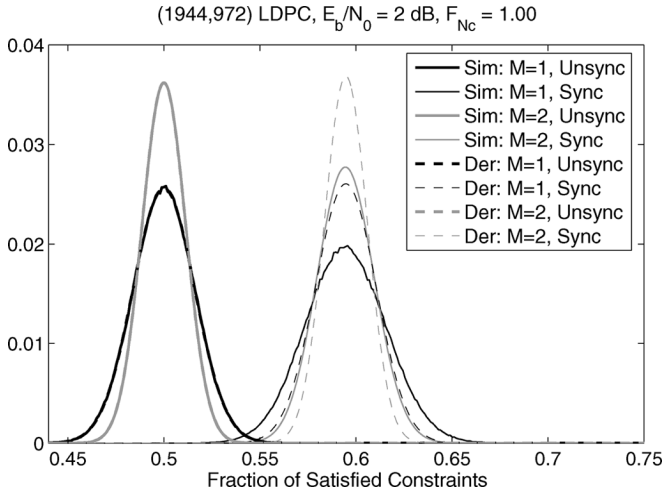


Fig. 4. Simulated and derived pdfs of fraction of satisfied constraints for unsynchronized and synchronized cases.

Fig. 4 compares the pdfs obtained through simulation and the pdf derived using (4) for $M = 1$ and $M = 2$ where as noted earlier M is the number of frames used for synchronization. SNRs are given in E_b/N_0 where $E_b/N_0 = E_s/N_0 - 10 \log_{10} R$, R being the code rate. When $M = 2$, there is a smaller common region between the unsynchronized and synchronized pdfs, potentially improving synchronization performance. For unsynchronized cases, the derived pdf is able to match the simulated pdf. When unsynchronized, the variable nodes are practically random bits [$\gamma = 0.5$ in (3)]; hence the degree of overlapping does not play a role.

In the synchronized cases, however, we see large disparities between the derived and simulated pdfs; they share the same mean, but the variances of the simulated pdfs are considerably larger. This can be explained by observing when an error in a single variable affects multiple constraints (as in the simulated pdf) larger variances naturally occur.

An additional and perhaps initially counterintuitive effect occurs with variation in channel SNR, in which decreasing SNRs yield smaller pdf variances. We explain this by noting that in the limit of very low SNR, synchronized and unsynchronized pdfs tend to the same (unsynchronized) distribution. The distribution of this pdf is narrow because most input blocks appear equally noisy, i.e., there is little variation in how different input blocks affect check equations. Higher channel SNRs, however, permit some cases where a large percentage of variables (and hence checks) are correct as well as abnormally noisy cases (relative to the operating SNR) that tend to drive to number of satisfied checks down toward the 50% level.

Interdependence of variables on a given set of checks is rather difficult to model due to code-specific irregularities; hence we use the simulated pdfs for the rest of this paper. Nevertheless, as a result of (4), we know that in order to achieve the best synchronization performance, one should: 1) minimize the overlap (reduces variance of synchronized pdf); 2) maximize the number of constraints N_c (reduces variance of unsynchronized and synchronized pdfs); and 3) use small constraint node degree d_c (increases mean of synchronized pdf where the mean is given by $N_c(1 - \gamma)$).

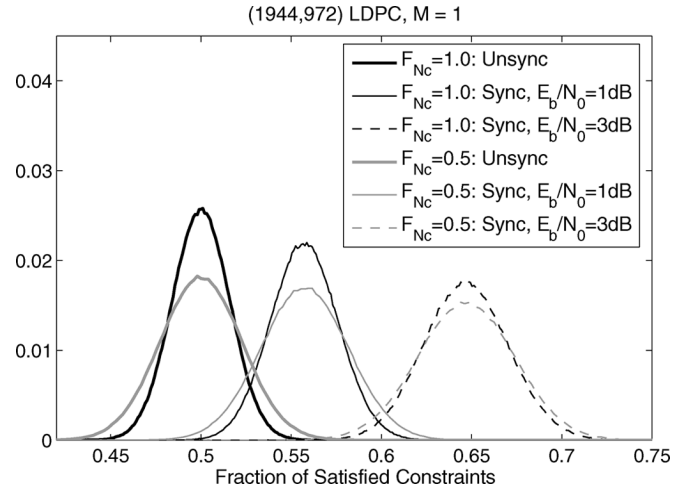


Fig. 5. PDFs of fraction of satisfied constraints for different F_{N_c} values and SNRs.

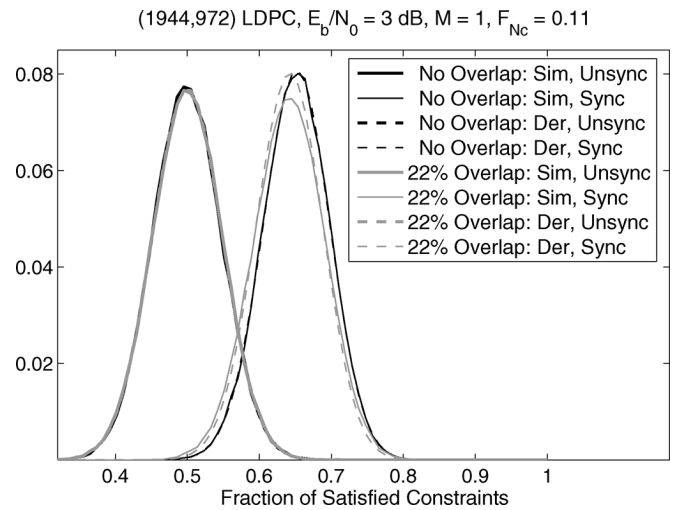


Fig. 6. PDFs of fraction of satisfied constraints with different degree of overlaps.

Fig. 5 provides pdfs for the fraction of satisfied constraints for different F_{N_c} values and SNRs. When $F_{N_c} = 0.5$, i.e., when half of the 972 constraint nodes are examined, the variances are notably larger, as predicted above. The unsynchronized case is independent of SNR and has a mean of 0.5 as expected. The mean as well as the variance of the synchronized cases increase with SNR, also as noted above.

When $F_{N_c} < 1.00$, we can use a greedy algorithm to find the set of constraint nodes with minimized overlaps and degrees. For instance for $F_{N_c} = 0.11$, i.e., when 107 constraint nodes are examined, the algorithm is able to find a set of constraint nodes that have no overlaps among them and all 107 constraints have a degree of seven. If, as part of a test, one randomly chooses 107 constraint nodes from the pool of 972, then according to the degree distribution of the code 22% of the attached variable nodes have degree higher than one (and hence overlap). In addition, there are 93 constraint nodes with degree-7 and 14 constraint nodes with degree-8.

Fig. 6 compares the pdfs of the n overlapped cases and 22% overlapped cases. When synchronized, if there are no overlaps,

the simulated pdfs match the derived pdfs. If there are overlaps, as seen earlier, there are differences in variances between the two. The mean of the pdfs with no overlaps is higher than the one with overlaps because the constraint nodes selected by the greedy algorithm exhibit lower degrees.

IV. PILOTLESS FRAME SYNCHRONIZATION

We describe three methods for pilotless frame synchronization based on the number of satisfied constraints. The first is the threshold method which sets a threshold for the number of satisfied constraints. If an offset equals or exceeds this threshold, synchronization is declared. The second is the maximum method which computes the number of satisfied constraints for each possible offset and picks the offset that results in the largest number of satisfied constraints. The third is a list-based synchronizer, a computationally efficient variation of the maximum method.

A. Threshold Method

For a given threshold θ which is the sum of the number of satisfied constraints over M frames, the threshold-based frame synchronizer computes frame offset estimate $\hat{\mu}$ as follows:

$$\hat{\mu} = \min_{\mu \in W} \mu \quad (5)$$

where

$$W = \left\{ \mu \mid \sum_{i=0}^{M-1} C_{\mu+iN} \geq \theta, \mu \in [0, N-1] \right\}.$$

C_j returns the number of satisfied constraints using the set of symbols $\{r_j, \dots, r_{j+N-1}\}$ as variable nodes where r_i is the i th symbol of the synchronizer buffer (Fig. 1).

In a threshold-based approach, a synchronization error occurs when all of the offsets are below θ or when a false positive ($\hat{\mu} \neq m$) is detected. With these two factors in mind, its frame synchronization error rate (FSER) is given by

$$\begin{aligned} \text{FSER} &= 1 - \frac{1}{N} \sum_{i=0}^{N-1} \left(1 - e_a^{(M)}\right)^i \left(1 - e_b^{(M)}\right) \\ &= 1 - \frac{1 - e_b^{(M)}}{N e_a^{(M)}} \left(1 - \left(1 - e_a^{(M)}\right)^N\right) \end{aligned} \quad (6)$$

where

$$\begin{aligned} e_a^{(M)} &= \sum_{i=\theta}^{MN_c} p_0^{(M)}(i) \\ e_b^{(M)} &= \sum_{i=0}^{\theta-1} p_1^{(M)}(i) \\ p_k^{(2)} &= p_k * p_k \\ p_k^{(M)} &= p_k^{(M-1)} * p_k. \end{aligned}$$

“*” is a convolution operator, $k \in [0, 1]$, p_0 is the discrete pdf of the unsynchronized case, p_1 is the discrete pdf of the synchronized case, $e_a^{(M)}$ is the probability of a false positive, $e_b^{(M)}$ is the

probability of all offsets being below θ , and i is the discrete pdf index. A small θ leads to a large $e_a^{(M)}$ and a small $e_b^{(M)}$, while the opposite is true for a large θ .

B. Maximum Method

The maximum method synchronizer computes the frame offset estimate $\hat{\mu}$ as follows:

$$\hat{\mu} = \arg \max_{\mu \in [0, N-1]} \sum_{i=0}^{M-1} C_{\mu+iN}. \quad (7)$$

Since the pdfs associated with the unsynchronized and synchronized cases have overlapping regions, $\hat{\mu}$ could result in a false positive ($\hat{\mu} \neq m$), which is the only factor that contributes to a synchronization error for the maximum method. The frame synchronization error rate (FSER) is given as follows:

$$\text{FSER} = 1 - \sum_{i=0}^{MN_c} p_1^{(M)}(i) \sum_{j=1}^N a(i, j) \quad (8)$$

where

$$a(i, j) = \frac{1}{j} \binom{N-1}{j-1} \left(p_0^{(M)}(i)\right)^{j-1} \left(\sum_{k=0}^{i-1} p_0^{(M)}(k)\right)^{N-j}.$$

C. List Method

A list-based frame synchronizer was first suggested by Robertson [17]. It is a strategy for constraining a high-complexity synchronizer to search over the Γ most probable candidates. In the first stage, the list-based synchronizer finds the number of satisfied constraints for all offsets and selects the best Γ offset candidates. In the second stage, a more thorough analysis is performed on the Γ offsets (by increasing F_{Nc} and/or M) and the best one is chosen. Good synchronization performance is obtained if the first stage is able to pull the true offset m into the list of candidates.

V. IMPLEMENTATION CONSIDERATIONS

Fig. 7 depicts the hardware architecture for the maximum method defined to (7). Architectures for the threshold and list methods can be devised with slight modifications to the one shown. The received bits are fed to a shift register from which multioperand XOR operations are performed for each constraint. Note that S , the syndrome of the PN sequence, discussed in Section III is fed to each XOR block. The results of each constraint are summed using multioperand addition to give the number of unsatisfied constraints U . The architecture in the figure computes the number of unsatisfied constraints rather than the number of satisfied constraints suggested in (7), because the multioperand XOR blocks output a zero when satisfied. The counter supplies the address for the RAM which holds the number of unsatisfied constraints to each offset. When $i < M - 1$ [(7)], the output of the multioperand adder is added to the corresponding location in the RAM and their sum is written back to the same location. By using a dual-port RAM, these two operations can be completed in a single clock cycle.

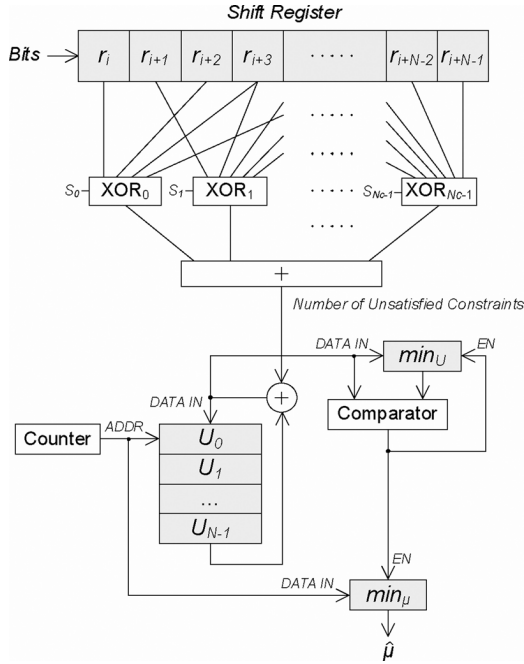


Fig. 7. Hardware architecture for the maximum method.

When $i = M - 1$, however, the output of the multioperand adder and the corresponding RAM location are added and fed to the comparator. If the value is less than the current minimum \min_U , \min_U and the offset index \min_μ are updated.

The computational burden of the synchronizer is dominated by the XOR blocks and the multioperand adder, and hence they will be used as a basis for complexity comparisons for the rest of this paper. Each n -bit XOR block comprises of a tree of $n - 1$ XOR gates with a depth of $\lceil \log_2 n \rceil$. The total number of XOR operations is given by

$$G_{\text{XOR}} = N_c \sum_{j=d_{c,\min}}^{d_{c,\max}} j \rho_j \quad (9)$$

where $d_{c,\min}$ and $d_{c,\max}$ are the minimum and maximum constraint node degrees, and ρ_j is the constraint node degree distribution. The multioperand adder consists of a tree of two-input adders, whose depth is $\lceil \log_2(N_c) \rceil$ levels with a total of $N_c - 1$ two-input additions. Though binary additions are sufficient for the first level, the subsequent levels require incrementally larger additions. Knowing that a full adder requires five gates and assuming ripple carry additions, approximately

$$G_{\text{ADD}} = \sum_{i=1}^{\lceil \log_2(N_c) \rceil} \left\lceil \frac{N_c}{2^i} \right\rceil \times 5i \quad (10)$$

gate-level operations are required. Since there are a total of N possible frame offsets and M frames are observed per offset, the total number of gate-level operations for the synchronizer is given by

$$G_{\text{TOTAL}} = NM(G_{\text{XOR}} + G_{\text{ADD}}). \quad (11)$$

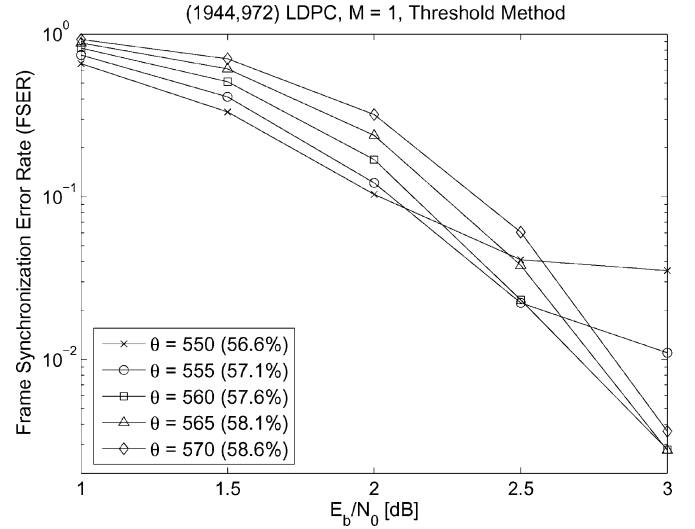
Fig. 8. FSERs of the threshold method at different values of the threshold θ . The percentages in the legend indicate $\theta/N_c \times 100$.

TABLE I
NUMBER OF FRAMES M THAT NEED TO BE ACQUIRED FOR A GIVEN FSER AND SNR WITH THE MAXIMUM METHOD AND THE (1944,972) CODE

FSER	E_b/N_0 [dB]				
	1.0	1.5	2.0	2.5	3.0
10^{-2}	4	2	2	1	1
10^{-4}	6	4	3	2	1
10^{-6}	8	5	3	2	2

For instance, when using the (1944,972) code with $M = 2$ and $F_{N_c} = 1.00$, $G_{\text{TOTAL}} = 1944 \times 2 \times (6966 + 9745) = 64,972,368$ gate-level operations.

VI. EXPERIMENTAL RESULTS

A. Threshold Method

Fig. 8 explores the FSER performance of the threshold method at different values of θ . The FSERs are plotted using (6). Simulations were also performed for these results, and the resulting curves precisely match the ones shown in the figure. The FSER performance improves with SNR, since higher SNRs separate the unsynchronized and synchronized pdfs further as demonstrated in Fig. 5 earlier. The best θ for each SNR varies due to the reasons discussed in Section IV-A. For the θ values considered in Fig. 8, at low SNR regions, increasing θ leads to a rapid increase in $e_b^{(M)}$, while $e_a^{(M)}$ reduces slowly, hence the increase in FSER. The opposite is true for high SNR regions.

B. Maximum Method

Table I shows the variation in the number of required frames M of the maximum method at various SNRs when the FSER is fixed. One requires more frames for low FSERs and low SNRs, which in effect separates the synchronized and unsynchronized pdfs by reducing their variances as shown in Fig. 4.

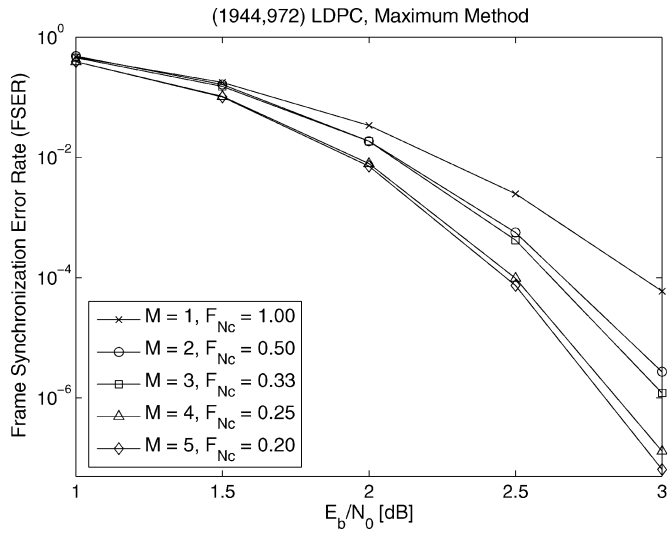


Fig. 9. FSERs of the maximum method at different values of M and F_{Nc} .

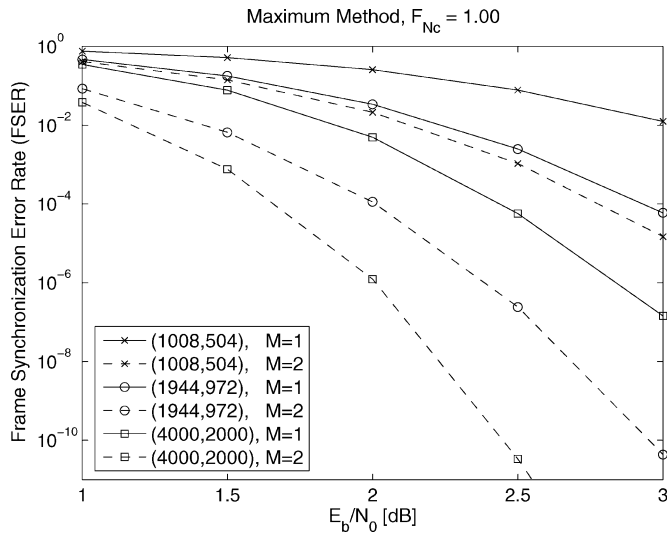


Fig. 10. Synchronization error rates of rate 1/2 codes of different lengths using the maximum method.

Fig. 9 examines FSERs of the maximum method where M is varied from 1 to 5 and F_{Nc} is varied from 1.00 down to 0.20. The FSERs are plotted using (8). Like the results in Fig. 8, simulations were also performed for these results, and again, the curves precisely match the ones in the figure. The product of M and F_{Nc} is one for all curves, implying that the total number of constraints that need to be evaluated for each curve are the same. The figure indicates that FSER improvements are obtained by using a large M and a small F_{Nc} . This is because low F_{Nc} values can leverage a higher weighting of independent variable node observations through selection of constraint nodes possessing reduced overlap and degree (as discussed Section III). Although the low F_{Nc} cases provide better FSER performance with the same amount of computation, their buffering requirements are proportionally higher at a rate dictated by M .

Fig. 10 compares the FSER performance of various rate 1/2 codes of different lengths using the maximum method. The ir-

TABLE II
CONSTRAINT NODE DEGREE DISTRIBUTIONS OF LDPC CODES

Code	Constraint Node Degree							
	3	6	7	8	9	11	14	15
(1008,504), $R = 1/2$	-	9	436	59	-	-	-	-
(1944,972), $R = 1/2$	-	-	810	162	-	-	-	-
(4000,2000), $R = 1/2$	-	-	-	1995	5	-	-	-
(1944,1296), $R = 2/3$	-	-	-	-	-	648	-	-
(1944,1296), $R = 3/4$	-	-	-	-	-	-	405	81

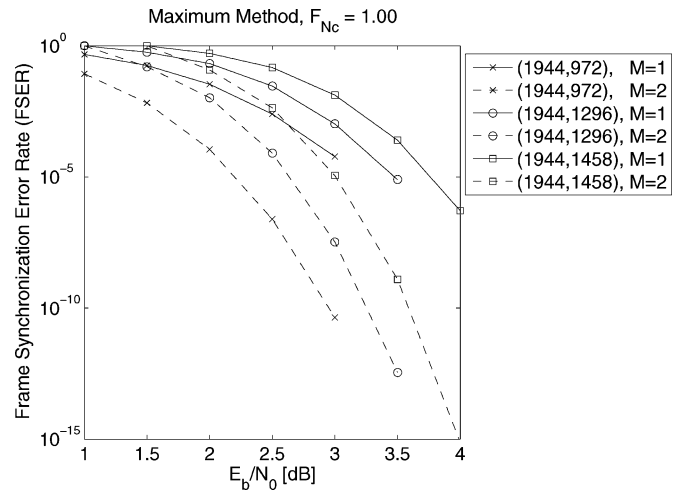


Fig. 11. Synchronization error rates of codes with rates using the maximum method.

regular (1008,504) and (4000,2000) codes are generated via the ACE algorithm in [18]. Their degree distributions are tabulated in Table II. Since the FSER performance is heavily dependent on the number of constraint nodes, the larger codes perform considerably better. Though (1944,972) at $M = 2$ and (4000,2000) at $M = 1$ require similar numbers of constraint node evaluations, (1944,972) at $M = 2$ achieves lower FSERs primarily due to its smaller constraint node degrees.

Fig. 11 examines the FSER performance of codes with different rates using the maximum method. All three codes are from the IEEE 802.11n draft standard [16]. Codes with lower rates have larger numbers of constraints and the constraints have lower degrees (Table II), hence the lower FSERs.

Fig. 12 shows the frame error rates (FERs) after 20 LDPC decoding iterations when the frame synchronizers in Fig. 11 are placed in front of an LDPC decoder. It is also important to note that the inherent FER of the code, given a certain SNR and number of iterations, constitutes a bound that limits the benefit of investing resources on synchronization methods that have much lower error rates (FSERs) than that of the code itself. The figure shows that for all three codes, $M = 2$, is required to approach the inherent FER performance (perfect synchronization) of the codes. However, the FSER gives us a good base for comparing the pure synchronization performance of different approaches, which is somewhat masked out if only the FER as a metric.

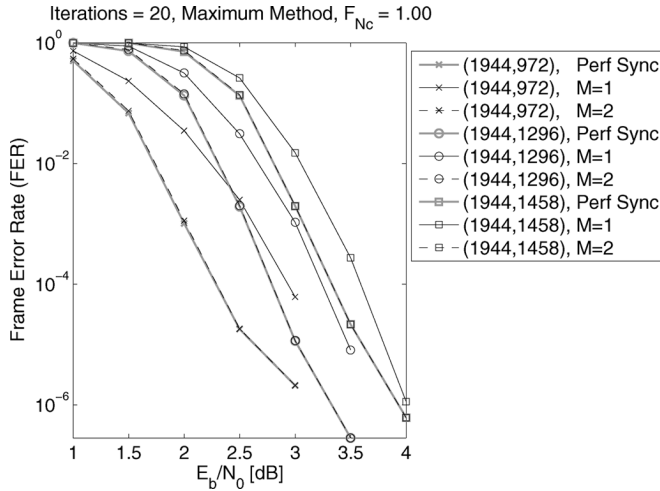


Fig. 12. Synchronization error rates of codes with rates using the maximum method.

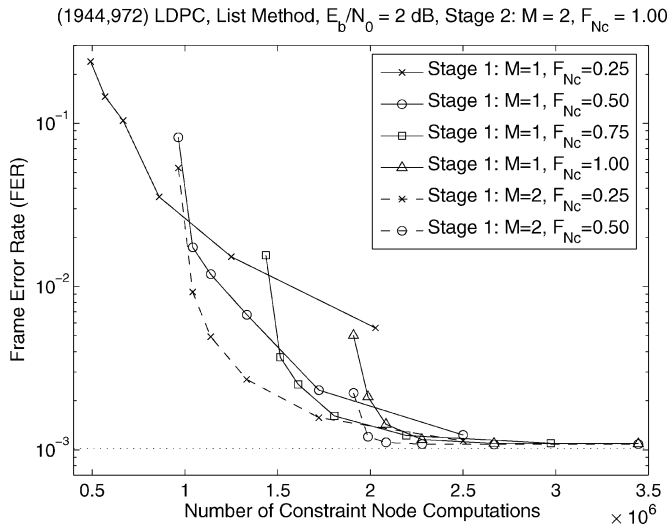


Fig. 13. Frame error rates when list frame synchronizers are combined with an LDPC decoder at 20 iterations.

C. List Method

Fig. 13 examines the FERs when list-based frame synchronizers are combined with a (1944,972) LDPC decoder at 20 iterations. The horizontal dotted line at around $\text{FER} = 10^{-3}$ is the inherent FER performance of the code with perfect frame synchronization. The second stage (Stage 2) is fixed at $M = 2$ and $F_{Nc} = 1.00$, because simulations indicated that a maximum-based synchronizer at $M = 2$ and $F_{Nc} = 1.00$ is able to reach the code performance. For each curve in the figure, the M and F_{Nc} values of the first stage (Stage 1) are varied. The six data points of each curve indicate $\Gamma = \{10, 50, 100, 200, 400, 800\}$. The x -axis represents the number of constraint node computations which is given by $N_c \times (N \times M_{\text{Stage1}} \times F_{Nc, \text{Stage1}} + \Gamma \times M_{\text{Stage2}} \times F_{Nc, \text{Stage2}})$. The figure shows that using $M = 2$, $F_{Nc} = 0.50$, and $\Gamma = 100$ in Stage 1 is able to approach the code performance with minimal complexity.

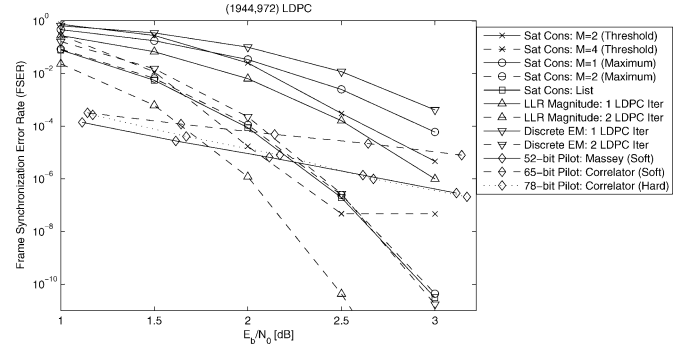


Fig. 14. Synchronization error rate comparisons of various synchronization approaches for the (1944,972) code.

D. Comparisons With Other Methods

The pilot sequences used in this section are constructed by concatenating a length-13 Barker code [1] with optimally searched concatenation polarities. The ‘‘Correlator’’ results use the standard correlation rule which finds the estimate $\hat{\mu}$ via

$$\hat{\mu} = \arg \max_{\mu \in [0, N-1]} \sum_{i=0}^{L-1} s_i r_{i+\mu} \quad (12)$$

where L is the length of the pilot sequence and s_i is the pilot symbol. The received symbol r_i can either be a soft or a hard decision. The ‘‘Massey’’ results use the high SNR estimation rule of Massey’s optimum synchronization method [1], which is known to provide similar synchronization performance to the more complex optimum rule in practical situations [3]. The correlator computes the estimate as follows:

$$\hat{\mu} = \arg \max_{\mu \in [0, N-1]} \left(\sum_{i=0}^{L-1} s_i r_{i+\mu} - \sum_{i=0}^{L-1} |r_{i+\mu}| \right) \quad (13)$$

where r_i is a soft value. Note that the hard decision correlator is often chosen for practical receivers due to its low implementation complexity. When pilots are used, there is a $10 \log_{10}(1 + L/N_v)$ dB loss where N_v is the number of variable nodes (code-word size) due to the pilot overhead during transmission. Hence, the use of a 52-bit, 65-bit, or 78-bit pilot for a 1944-bit frame leads to a bandwidth efficiency loss of 0.11, 0.15, or 0.17 dB, respectively.

Fig. 14 compares FSERs of different synchronization approaches for the (1944,972) code. LLR magnitude results employ the approach by Matsumoto and Imai [12], while the discrete EM results employ the approach by Wymeersch *et al.* [13], [14]. Although the original work conducted by Wymeersch *et al.* utilizes pilots as well as the LDPC decoder, we use only the information from the LDPC decoder to make a fair comparison against our approach and the LLR magnitude approach. In order to increase synchronization performance of the LLR magnitude and discrete EM methods, one can either increase the number of LDPC iterations and/or M . We fix M to one and vary the number of LDPC iterations, as this leads to the best performance/complexity tradeoff for both methods.

TABLE III
COMPLEXITY COMPARISONS BETWEEN DIFFERENT FRAME SYNCHRONIZATION METHODS FOR THE (1944,972) CODE

Method	Reference	Pilots [bits]	Memory [kilo bits]	Operations [million gates]
Correlator (Hard)	[1]	78	4	2
Massey (Soft)	[1]	52	63	18
LLR Magnitude: 2 LDPC Iter	[12]	0	110	900
Discrete EM: 2 LDPC Iter	[13], [14]	0	110	1000
Sat Cons: $M = 2$ (Maximum)	This Paper	0	27	65
Sat Cons: List	This Paper	0	27	34

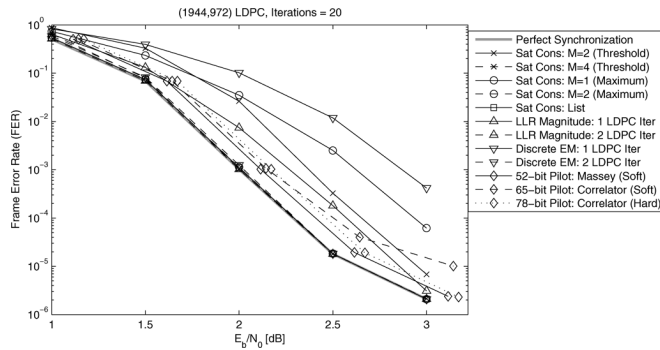


Fig. 15. Frame error rate comparisons of various synchronization approaches for the (1944,972) code.

For the threshold method results, the threshold that gives the best overall FER performance is chosen. With the threshold methods, the frame offset estimate $\hat{\mu}$ is found at $i = (N - 1)/2$ [(6)] on average. Thus, the threshold methods are set at $M = 2$ and $M = 4$, since their computational complexities are comparable to the maximum methods at $M = 1$ and $M = 2$, respectively. The list synchronizer uses $M = 2$, $F_{NC} = 0.50$, and $\Gamma = 100$ for the first stage, and $M = 2$, $F_{NC} = 1.00$ for the second stage as identified in Section VI-C. Interestingly, the list method performs marginally better than the maximum method at $M = 2$. This is because the probability of the true offset being included in the list of 100 candidates and being chosen from the list is higher than the probability of finding the true offset from a pool of 1944 offsets. The figure also shows the difference in trend between the LDPC-based and pilot-based methods. While the reduction in FSER is exponential for the LDPC-based curves analogous to the FER behavior of LDPC codes, the pilot-based curves show a linear reduction behavior. While piloting methods might also exploit multiple frames ($M > 1$) to achieve better performance, parameters for the curves in Fig. 14 are chosen to minimize complexity such that an operating point near $\text{FER} = 10^{-6}$.

Fig. 15 shows the frame error rates after LDPC decoding when the frame synchronizers in Fig. 14 are combined with an LDPC decoder at 20 iterations. Although the FSER characteristics of the 52-bit Massey's piloting method and the 78-bit hard correlator are better than the FER performance of the code, when coupled with an LDPC decoder, their FER performance is always inferior to the code performance due to the inherent decibel loss of pilot-based schemes as discussed above. The following methods are able to match the code

performance: the threshold method at $M = 4$, the maximum method at $M = 2$, the list method using the set of parameters identified in Section VI-C, and the LLR magnitude and the discrete EM methods with two LDPC decoding iterations. The figure indicates that the LDPC-based synchronization methods are able to save several tenths of a dB compared to the best piloting methods. Equivalently, while providing comparable FER performance they are able to save 2.6% of the bandwidth over the 52-bit Massey's method and 3.8% of the bandwidth over the 78-bit hard correlator.

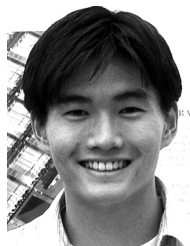
Table III compares the computation and memory requirements of the best schemes identified in Fig. 15. The threshold method at $M = 4$ is not included here, because the maximum method at $M = 2$ is able to meet the code performance with similar computational complexity while providing deterministic throughput and half of the buffering requirements. The number of gate-level operations are estimated using the methodology described in Section V. For the LLR magnitude method, the Offset-Min-BP [19] LDPC decoder implementation discussed in [20] is assumed. We assume that the soft symbols in 52-bit Massey, LLR magnitude, and discrete EM are represented with 16 bits. Though the LLR magnitude and discrete EM methods were found to meet the code performance in Fig. 15, their complexities are much higher than those of the methods presented here due to their needs for two full LDPC decoding iterations. Furthermore, they may not be practical due to the high delays associated with the LDPC decoder. While the list-based synchronizer presented here requires around 20 times the complexity of the 78-bit hard correlator and two times the complexity of the 52-bit Massey's method, it is able to eliminate the bandwidth occupied by pilots as discussed above. There will be many applications in which this tradeoff will be well worth making, particularly given that the cost of computations will continue to decrease as hardware technologies continue to evolve, while bandwidth will always remain a limited resource.

VII. CONCLUSION

We have presented a pilotless frame synchronization scheme that utilizes feedback from the LDPC code constraints. Experimental results show that at the expense of increased computation, the bandwidth occupied by traditional piloting methods can be eliminated while providing sufficient synchronization performance. Many environments and applications would benefit from the ability to gain increased noise resilience and/or increased bandwidth at the cost of some increased receiver processing complexity.

REFERENCES

- [1] J. Massey, "Optimum frame synchronization," *IEEE Trans. Commun.*, vol. COM-20, no. 2, pp. 115–119, Apr. 1972.
- [2] G. Lui and H. Tan, "Frame synchronization for Gaussian channels," *IEEE Trans. Commun.*, vol. COM-35, no. 8, pp. 818–829, Aug. 1987.
- [3] P. Nielsen, "Some optimum and suboptimum frame synchronizers for binary data in Gaussian noise," *IEEE Trans. Commun.*, vol. 21, no. 6, pp. 770–772, Jun. 1973.
- [4] S. Dolinar and K. Cheung, Frame synchronization methods based on channel symbol measurements Jet Propulsion Lab., Pasadena, CA, The Telecommunications and Data Acquisition Progress Rep. 42-98, 1989.
- [5] G. Lorden, R. McElice, and L. Swanson, "Node synchronization for the Viterbi decoder," *IEEE Trans. Commun.*, vol. COM-32, no. 5, pp. 524–531, May 1984.
- [6] M. de Mateo, "Node synchronization technique for any $1/n$ rate convolutional code," in *Proc. IEEE Int. Conf. Communications*, 1991, vol. 3, pp. 1681–1687.
- [7] M. Mostofa, K. Howlader, and B. Woerner, "Decoder-assisted frame synchronization for packet transmission," *IEEE J. Sel. Areas Commun.*, vol. 19, no. 12, pp. 2331–2345, Dec. 2001.
- [8] J. Sun and M. Valenti, "Optimum frame synchronization for preambleless packet transmission of turbo codes," in *Proc. IEEE Asilomar Conf. Signals, Systems and Computers*, 2004, vol. 1, pp. 1126–1130.
- [9] T. Cassaro and C. Georghiades, "Frame synchronization for coded systems over AWGN channels," *IEEE Trans. Commun.*, vol. 52, no. 3, pp. 484–489, Mar. 2004.
- [10] B. Mielczarek, "Synchronization in turbo coded systems," Licentiate Thesis 342L, Chalmers Univ. Technol., Goteborg, Sweden, 2000.
- [11] B. Mielczarek and A. Svensson, "Timing error recovery in turbo-coded systems on AWGN channels," *IEEE Trans. Commun.*, vol. 50, no. 10, pp. 1584–1592, Oct. 2002.
- [12] W. Matsumoto and H. Imai, "Blind synchronization with enhanced sum-product algorithm for low-density parity-check codes," in *Proc. IEEE Int. Symp. Wireless Personal Multimedia Communications*, 2002, vol. 3, pp. 966–970.
- [13] H. Wymeersch and M. Moeneclaey, "ML frame synchronization for turbo and LDPC codes," in *Proc. Int. Symp. DSP and Communications Systems*, 2003.
- [14] H. Wymeersch, H. Steendam, H. Bruneel, and M. Moeneclaey, "Code-aided frame synchronization and phase ambiguity resolution," *IEEE Trans. Signal Process.*, vol. 54, no. 7, pp. 2747–2757, Jul. 2006.
- [15] D. Lee, H. Kim, C. Jones, and J. Villasenor, "Pilotless frame synchronization via LDPC code constraint feedback," *IEEE Commun. Lett.*, vol. 11, no. 8, pp. 683–685, Aug. 2007.
- [16] Part 11: Wireless LAN Medium Access Control (MAC) and Physical Layer (PHY) Specifications 2007, IEEE P802.11n/D1.10.
- [17] P. Robertson, "A generalized frame synchronizer," in *Proc. IEEE Global Telecommunications Conf.*, 1992, vol. 1, pp. 365–369.
- [18] T. Tian, C. Jones, J. Villasenor, and R. Wesel, "Selective avoidance of cycles in irregular LDPC code construction," *IEEE Trans. Commun.*, vol. 52, no. 8, pp. 1242–1247, Aug. 2004.
- [19] J. Chen and M. Fossorier, "Density evolution for two improved BP-based decoding algorithms of LDPC codes," *IEEE Commun. Lett.*, vol. 6, no. 5, pp. 208–210, May 2002.
- [20] C. Jones, E. Vallés, M. Smith, and J. Villasenor, "Approximate-MIN* constraint node updating for LDPC code decoding," in *Proc. IEEE Military Communications Conf.*, 2003, vol. 1, pp. 157–162.



Dong-U Lee (A'07–M'07) received the B.Eng. degree in information systems engineering and the Ph.D. degree in computing, both from Imperial College London, London, U.K., in 2001 and 2004, respectively.

From 2005 to 2007, he was a Postdoctoral Researcher at the Electrical Engineering Department, University of California, Los Angeles (UCLA), where he developed high-performance hardware designs for wireless communications and mathematical function evaluations. He is currently a Research Scientist at Mojix, Inc., Los Angeles, CA, where he is specializing in hardware implementation aspects of RFID systems. His research interests include computer arithmetic, communications, design automation, reconfigurable computing, and video image processing.



Hyungjin Kim (S'06) received the B.S. degree and the M.S. both in electrical engineering from the Seoul National University, Seoul, Korea, in 1997 and 1999, respectively. He is currently pursuing the Ph.D. degree at the Electrical Engineering Department, University of California, Los Angeles.

His research interests include MIMO communications, channel coding, sensor networks, compression and encryption algorithms, video/image processing, and signal processing.



Christopher R. Jones (M'03) received the B.S., M.S., and Ph.D. degrees in electrical engineering from the University of California, Los Angeles (UCLA) in 1995, 1996, and 2003.

He is currently with the Jet Propulsion Laboratory, Pasadena, CA on problems related to low-density parity-check codes. Specific interests include photograph selection and lifting, encoder/decoder architectures, hybridization with ARQ protocols, and integration with timing/carrier recovery techniques.



John D. Villasenor (S'89–M'89–SM'97) received the B.S. degree in 1985 from the University of Virginia, Charlottesville, and the M.S. and Ph.D. degree in 1986 and 1989 from Stanford University, Stanford, CA, all in electrical engineering.

From 1990 to 1992, he was with the Radar Science and Engineering section of the Jet Propulsion Laboratory in Pasadena, CA, where he developed methods for imaging the Earth from space. He joined the Electrical Engineering Department at the University of California, Los Angeles (UCLA) in 1992, and is currently Professor. He served as Vice Chair of the Department from 1996 to 2002. At UCLA, his research efforts lie in communications, computing, imaging and video compression, and networking.