Energy Optimization of LDPC Decoder Circuits with Timing Violations

François Leduc-Primeau*, Frank R. Kschischang† and Warren J. Gross*
*Dept. of Electrical & Computer Engineering, McGill University, Montréal, Canada
†Dept. of Electrical & Computer Engineering, University of Toronto, Toronto, Canada

Abstract—This paper presents a quasi-synchronous design approach for signal processing circuits, in which timing violations are permitted, but without the need for a hardware compensation mechanism. A quasi-synchronous low-density parity-check decoder processing circuit based on the offset min-sum algorithm is designed, achieving the same performance and occupying the same area as a conventional synchronous circuit, but using up to 28% less energy.

I. INTRODUCTION

The time required for a signal to propagate through a CMOS circuit varies depending on several factors. Some of the variation results from physical limitations: the delay depends on the initial and final charge state of the circuit. Other variations are due to the difficulty (or impossibility) of controlling the fabrication process and the operating conditions of the circuit [1]. This variation in propagation delay is a source of energy inefficiency for synchronous circuits, and new design approaches have been proposed to tackle this. In better than worst-case (BTWC) [2] or voltage over-scaled (VOS) circuits, efficiency is improved by allowing some timing violations to occur, while including a mechanism to compensate or recover from these faults.

In the case of signal processing circuits, the average quality of the output is often an appropriate measure and this creates more possibilities for dealing with timing violations. A seminal contribution in this area was the algorithmic noise tolerance (ANT) approach [3, 4], which is to allow timing violations to occur in the main processing block, while adding a separate reliable processing block with reduced precision that is used to bound the error of the main block, and provide algorithmic performance guarantees. The downside of the ANT approach is that it relies on the assumption that timing violations will first occur in the most significant bits. If that is not the case, the precision of the circuit could degrade to the precision of the auxiliary block, limiting the scheme’s usefulness. For many circuits, including some adder circuits [5], this assumption does not hold. Furthermore, the addition of the reduced precision block and of a comparison circuit increases the area requirement.

We propose a general design methodology for digital circuits with a relaxed synchronicity requirement that does not rely on any hardware compensation mechanism, resulting in what we call quasi-synchronous circuits. With this approach, performance guarantees are provided by re-analyzing the algorithm while taking into account the effect of timing violations.

We use the quasi-synchronous approach to design energy-optimized low-density parity-check (LDPC) decoder circuits based on a state-of-the-art decoding algorithm and architecture. The topic of unreliable LDPC decoders has been discussed in a number of contributions. Varshney studied the Gallager-A and the Sum-Product decoding algorithms when the computations and the message exchanges are “noisy,” and showed that the density evolution analysis still applies [6]. The Gallager-B algorithm was also analyzed under various scenarios [7]–[9]. Finally, Ngassa et al. proposed a model for an unreliable quantized Min-Sum decoder, and performed a numerical evaluation of the density evolution equations as well as simulations of a finite-length decoder [10]. None of these contributions make an explicit link between the reliability of the computation and the energy consumed by the decoder.

In this paper, we present a circuit design workflow that allows modeling the effect of timing violations caused by a reduced supply voltage and increased clock frequency, while simultaneously capturing the energy consumption. We find that the density evolution analysis still applies to the decoder affected by timing faults, and use it to evaluate the decoder’s channel threshold and iterative performance. Finally, we show that under mild assumptions, the energy minimization of a complete LDPC decoder can be simplified to the energy minimization of a small test circuit, and present an optimization method based on Gear-Shift Decoding [11] that finds sequences of quasi-synchronous decoders that minimize decoding energy.

II. DESIGN FRAMEWORK

Optimizing the energy consumption of quasi-synchronous circuits requires an accurate modeling of the impact of timing violations and of the energy consumption. We propose to achieve this by characterizing one or several test circuits that are subsets of the complete circuit implementation.

A. Modeling Deviations

The term deviation refers to the effect of circuit faults on the result of a computation, and the deviation model is the bridge between the circuit characterization and the analysis of the algorithm. We are interested in modeling deviations occurring in a synchronous circuit, and therefore the computation can be modeled as a discrete-time system. In a deviation-free context, the circuit accepts an input \(X[t]\) at time \(t \in \{0, 1, 2, \ldots\}\) and outputs a result \(Y[t]\). Note that the circuit could require one
or several clock cycles to generate $Y[t]$, but this is irrelevant to the characterization of the computation. When operated in a quasi-synchronous manner, the circuit instead outputs $Z[t]$, which is a corrupted version of $Y[t]$. When analyzing the algorithm it is convenient to model the deviations on $Y[t]$ as a transmission through a memoryless communication channel, where the deviation $D[t]$ corresponds to additive noise, such that $Z[t] = Y[t] + D[t]$. However, it is challenging to generate simple and accurate models of the deviation $D[t]$.

Two major issues are that (1) the propagation delay through the circuit depends on its internal charge state and therefore on previous inputs, and (2) that delay variations on individual binary outputs exhibit a strong correlation due to shared signal paths within the circuit. To address these issues, we restrict the class of circuits that can be characterized to those computing memoryless functions, in the sense that $Y[t]$ is only a function of $X[t]$, and furthermore we assume that $X[t]$ can be modeled as a stationary random process. This allows us to use a Monte-Carlo simulation to measure the probability distribution of $Z[t]$ or $D[t]$. To alleviate the restriction imposed by the stationarity assumption, we can characterize the circuit for several distributions of $X[t]$.

We are interested in using the test circuit to characterize not only deviations, but also energy consumption. Both deviations and energy depend on the choice of some design parameters. Let $\Gamma$ be the set of design parameters that we are considering. For a probability distribution of $X[t]$ parametrized by $\rho$ and system parameters $\gamma \in \Gamma$, we define a function $f_\gamma(\rho)$ that provides a parametrization of the distribution of $Z[t]$, and a function $c_\gamma(\rho)$ that provides the average energy consumed by the circuit to produce one output.

### B. Engineering Workflow

To make the characterization as accurate as possible, we measure the deviations and the energy consumption directly on optimized circuit models generated by a commercial synthesis tool using TSMC’s 65 nm CMOS process and cell library. Synthesis is performed at $V_{dd} = 1.0\, \text{V}$, and the clock period is chosen as small as possible while keeping the area within 10% of the minimal area. We refer to the supply voltage and the clock period used for synthesis as the nominal operating condition. The choice of clock period gives a good compromise between speed and area when the circuit is operated at the nominal operating condition. However, the choices made by the synthesis tool are not necessarily good when the circuit is operated in a quasi-synchronous manner. In fact, it was shown in [12] that the power consumption of a circuit can be reduced by up to 32% when the synthesis optimization takes into account the frequency at which the clock constraint can be violated. We nonetheless use a standard synthesis method in this work, but we expect that the energy savings reported could be improved by using a more specialized synthesis algorithm.

Once the circuit is synthesized, we perform a static timing analysis of the gate-level model at various supply voltages. All timing analyses (including at the nominal supply) are performed using timing libraries generated by the Cadence Encounter Library Characterization tool. We then use this timing information to observe the dynamic effect of path delay variations as part of a functional simulation of the gate-level circuit. In this paper we focus on variations due to path activation, that is the variations in delay caused by the different propagation times required by different input transitions.

Switching activity data is collected during functional simulation, and the synthesis tool is then used to generate estimated dynamic power $P_{\text{dyn}}$ and leakage power $P_{\text{leak}}$. The tool performs the power estimation at the nominal clock frequency, and therefore the total energy consumed during one cycle by the quasi-synchronous circuit is given by $E_{\text{cycle}} = P_{\text{dyn}} T_{\text{clk,nom}} + P_{\text{leak}} T_{\text{clk}}$, where $T_{\text{clk,nom}}$ is the nominal clock period and $T_{\text{clk}}$ is the actual clock period used to run the quasi-synchronous circuit.

### III. LDPC Decoder

#### A. Algorithm

We consider that a sequence of information bits is encoded using an LDPC code of length $n$ and transmitted over the additive white Gaussian noise (AWGN) channel. The LDPC code is described by an $m \times n$ binary parity-check matrix $H$, or equivalently by a factor graph composed of variable nodes (VN) indexed from 1 to $n$ and of check nodes (CN) indexed from 1 to $m$. When the $H$ matrix and the factor graph represent the same code, the graph contains an edge between a variable node $i$ and a check node $j$ if and only if $H_{i,j} \neq 0$. The degrees of the variable and check nodes define a family of codes. When a code is regular, all the VNs have degree $d_v$ and all CNs have degree $d_c$, and the code family is specified by the pair $(d_v, d_c)$. Let us denote the transmitted codeword as $x = \{-1, 1\}^n$. The output of the AWGN channel is then given by $y = x + W$, where $W$ is a vector of $n$ independent and identically distributed zero-mean Gaussian random variables with variance $\sigma_w^2$. We define the belief channel output corresponding to the $i$-th codeword bit as

$$\mu_i = \alpha y_i / \sigma_w^2, \quad (1)$$

where $\alpha > 0$ is an input scaling constant. Given that $x_i = 1$ was transmitted, the conditional probability distribution of $\mu_i$ is a normal distribution with mean $\alpha / \sigma_w^2$ and variance $\alpha^2 / \sigma_w^2$, and therefore for a fixed $\alpha$ it can be described by a single parameter $\rho = \alpha / \sigma_w^2$. We refer to this distribution as a one-dimensional (1-D) normal distribution with parameter $\rho$. Equivalently, the conditional distribution of $\mu_i$ can be specified using its error probability $p_e$:

$$p_e = P(\mu_i < 0 | x_i = 1) = \frac{1}{2} \text{erfc}\left(\frac{\rho}{2 \alpha}\right), \quad (2)$$

where $\text{erfc}(\cdot)$ is the complementary error function.

The well-known Offset Min-Sum (OMS) algorithm is a simplified version of the Sum-Product algorithm that can usually achieve similar error-correction performance. It has been widely used in implementations of LDPC decoders [13]–[15]. To make our decoder implementation more realistic...
and show the flexibility of our design framework, we use a row-layered message-passing schedule. Decoder architectures optimized for this schedule have proven effective for achieving efficient implementations of LDPC decoders, e.g. [13]. Using a row-layered schedule also allows to pipeline the decoder to increase the circuit’s utilization. When using a row-layered schedule, the rows of the $H$ matrix are partitioned into $L$ sets called layers. We assume that all the columns in a given layer contain at most one non-zero element, which enables some architectural simplifications. The reader is advised to consult [13] and references therein for information on the Offset Min-Sum algorithm and the row-layered schedule.

Using the equivalent graph representation of the code, a layer with index $\ell$ can be defined as a set $L_\ell$ of check nodes, $\ell \in [1, L]$. To simplify the notation, we assume that the LDPC code is regular. Therefore each variable node receives exactly one message for each layer, and the inputs of a given VN can share the same index variable as the layers. Let us denote by $\lambda_i^{(t, \ell)}$ the belief messages received by VN $i$ from its neighboring CNs after evaluating layer $\ell$ during iteration $t$. Similarly, let $\mu_i^{(t, \ell)}$ be the message sent by a VN $i$ at the beginning of iteration $t$ and layer $\ell$. We denote the channel information corresponding to the $i$-th codeword bit by $\mu_i^{(0)}$, since it also corresponds to the first message sent by a variable node $i$ to all its neighboring check nodes. We also denote by $\Lambda_i^{(t, \ell)}$ the current sum of incoming messages at a VN $i$. We have

$$
\Lambda_i^{(t, \ell')} = \mu_i^{(0)} + \sum_{\ell=1}^{\ell'} \lambda_i^{(t, \ell)}, \quad 1 \leq \ell' \leq L.
$$

B. Decoder Architecture

The factor graph of the code can be used to represent the computations that are performed by the decoder. At each decoding iteration, one message is sent from variable to check nodes on every edge of the graph, and from check to variable nodes. A circuit block responsible for generating messages sent by variable (or check) nodes is called a variable (or check) node processor (VNP or CNP). A VNP sends and receives one message per clock cycle, while a CNP receives messages sent by variable (or check) nodes is called a variable decoding iteration, one message is sent from variable to check node processors. We use two pipeline stages for the examples, as shown in Figure 1.

Messages exchanged in the decoder are fixed-point numbers. The position of the binary point does not have an impact on the algorithm, and therefore the messages sent by VNs in the first iteration can be defined as rounding the result of (1) to the nearest integer, while choosing a suitable $\alpha$. The number of bits in the quantization, the scaling factor $\alpha$, and the OMS offset parameter are chosen based on a density evolution analysis of the reliable algorithm (described in Section IV). We quantize decoder messages on 6 bits, which allows a reliable decoder to have approximately the same channel threshold as a floating-point decoder.

IV. ALGORITHM ANALYSIS

A. Density Evolution on a Test Circuit

Density evolution (DE) is the most common tool for predicting the error-correction performance of an LDPC decoder. It relies on the assumption that messages passed in the factor graph are mutually independent, which holds as the code length goes to infinity. We use the semi-Gaussian density tracking method [16], an approximate DE method that is nonetheless accurate. The semi-Gaussian method tracks a one-dimensional parametrization of the message probability distribution, by assuming that VN-to-CN messages have a 1-D normal distribution, described in Section III-A. Note that this is an exact characterization of the VN-to-CN messages in the first iteration.

Let $p_e^{(0)}$ be the error probability of the communication channel, and $p_e^{(t)}$ be the error probability of a VN-to-CN message at the start of iteration $t$. Using a 1-D representation, the iterative progress of the decoder can be represented by an “Exit”
function \( f \) that evaluates the message error probability in the next iteration: \( p_e^{(t+1)} = f(p_e^{(t)}, p_c^{(0)}) \). The semi-Gaussian DE method evaluates points on this ExIT function by performing a Monte-Carlo simulation on the one-iteration computation tree corresponding to the chosen LDPC code family. Therefore, we build a test circuit that evaluates the one-iteration computation tree using the same processing blocks found in the complete decoder, and perform the Monte-Carlo simulation directly on the gate-level circuit model. We thus obtain ExIT functions \( f_\gamma(p_e^{(t)}, p_c^{(0)}) \) that depend on the choice of design parameter \( \gamma \), and simultaneously, the energy consumption \( c_\gamma(p_e^{(t)}, p_c^{(0)}) \) of the computation. The test circuit is illustrated in Fig. 2.

In order for the test circuit to be usable within the deviation model of Section II-A, we must show that it implements a memoryless function, and that its inputs can be modeled as a stationary process. It is clear that the function implemented is memoryless, since an iteration of the LDPC decoding algorithm only depends on the result of the previous iteration. To determine whether the test circuit’s inputs can be modeled as a stationary process, we must consider the way the circuit is used when instantiated in the final decoder. The input of a given VNP front is determined by the VN it is associated with, as well as the iteration and layer indices. If we assume that the code graph is cycle free, the messages sent by all VNs at iteration \( t \) and layer \( \ell \) are independent an identically distributed (i.i.d.). Therefore, when a VNP is assigned to process a sequence of distinct VNs belonging to the same \((t, \ell)\), its inputs can be represented by an i.i.d. process. If the number of CNPs instantiated in the decoder is significantly less than \( m_1/L \), then for most cycles the current and the next inputs of a VNP do belong to the same \((t, \ell)\), making this a reasonable approximation.

B. Validity of the DE Analysis

The validity of DE in predicting the performance of specific finite-length codes rests on two properties of the LDPC decoder. The first property is the conditional independence of errors, which states that the error-correction performance of the decoder is independent from the particular codeword that was transmitted. The second property states that the error-correction performance of a particular LDPC code concentrates around the performance measured on a cycle-free graph, as the code length goes to infinity. Both properties were shown to hold in the context of reliable implementations [17].

As shown in [6], the concentration property still holds for a decoder affected by deviations, as long as the effect of the deviations is local, that is as long as a deviation can only affect messages that are included in the directed neighborhood of the edge where it is applied, where the edge direction refers to the direction of message propagation. However, because our characterization of the deviations occurring in the decoder is obtained through Monte-Carlo simulation, we cannot determine analytically whether errors within the decoder remain independent from the choice of transmitted codeword. Instead, we perform two Monte-Carlo simulations on the circuit model: one using the all-one codeword, and one using randomly selected codewords. For all the cases that we considered, the difference in the bit error rates observed was not statistically significant.

C. Examples

To provide examples of ExIT and energy curves, we consider some regular families with a rate of \( r = \frac{4}{5} \) and \( d_e \in \{3, 4, 5\} \), and we vary the circuit’s supply voltage and clock period. The nominal operating conditions are \( V_{dd} = 1.0V \) and \( T_{clk} = 2ns \) for all test circuits, and therefore curves at \( 1.0V \) and \( T_{clk} = 2ns \) correspond to reliable circuits. The ExIT curves are generated using the semi-Gaussian method with measurements taken from the gate-level circuit model as described above. The probability distribution of the message \( \mu^{(t+1)} \) can be estimated using various parametrizations of the distribution. We perform the estimation using the measured error rate, which is a simple method that was shown to have a good accuracy [16]. At very low error rates, we also rely on extrapolated data to construct the ExIT curves. This allows us to perform energy minimization with very low residual error rate constraints, under the assumption that ExIT curves follow a log-linear trend at low message error rates. The energy curves are generated as described in Sect. II and give the energy in \( \text{J} \) required to perform one complete decoding iteration with the test circuit, that is for using the test circuit \( d_e \) times.

The curves are shown in Fig. 3. The channel threshold when using a reliable implementation of the decoder architecture is \( p_e^{(0)} = 0.12 \) for the \( (3, 6) \) family, \( p_e^{(0)} = 0.11 \) for the \( (4, 8) \) family, and \( p_e^{(0)} = 0.09 \) for the \( (5, 10) \) family. For a fair comparison, \( p_e^{(0)} = 0.09 \) is used for all curves shown in Fig. 3. We see that the energy required to perform a decoding iteration with a \( (4, 8) \) or a \( (5, 10) \) decoder is much greater than for a \( (3, 6) \) decoder. Despite the fact that they also progress faster towards zero error probability, our energy optimization
results (discussed in Sect. V) show that the (3, 6) family is a better choice for minimizing energy.

V. ENERGY OPTIMIZATION

A. Objective

The performance of the LDPC code and of its decoder can be described by specifying a vector \( P = (p_e^{(0)}, p_{\text{res}}, T_{\text{dec}}) \), where \( p_e^{(0)} \) is the output error rate of the communication channel, \( p_{\text{res}} \) the residual error rate of VN-to-CN messages when the decoder terminates, and \( T_{\text{dec}} \) the expected decoding latency. Since the decoding is iterative, it is possible to change the decoder’s parameters as the decoding progresses. For example, this can be achieved by instantiating a pipelined sequence of decoder circuits, where each circuit is responsible for a subset of the decoding iterations. We use the notation \( \gamma = [\gamma_1, \gamma_2, \ldots] \) to denote a parameter sequence where a first parameter set \( \gamma_1 \) is used for \( N_1 \) iterations, then a second set \( \gamma_2 \) for \( N_2 \) iterations, and so on. The design parameters that we consider are the supply voltage \( V_{\text{dd}} \), clock period \( T_{\text{clk}} \) and message scaling factor \( \alpha \). A parameter set is denoted \( \gamma = [V_{\text{dd}}, T_{\text{clk}}, \alpha] \).

The decoder’s performance \( P \) and energy consumption \( E \) depend on \( \gamma \). The energy minimization problem can be stated as follows. Given a performance constraint \( P = (a, b, c) \), we wish to find the value of \( \gamma \) that minimizes \( E \), subject to \( p_e^{(0)} \geq a \), \( p_{\text{res}} \leq b \), \( T_{\text{dec}} \leq c \). Just like in the standard DE, we propose to use the code’s computation tree as a proxy for the entire decoder, and furthermore to use the energy consumption of the test circuit as the optimization objective. To be able to replace the energy minimization problem of the final decoder with an energy minimization problem involving the test circuit, we make the following assumptions. (A1) The ordering of the energy consumption is the same for the test circuit and for the final decoder, that is, for any \( \gamma_1 \) and \( \gamma_2 \), \( E_{\text{TEST}}(\gamma_1) \leq E_{\text{TEST}}(\gamma_2) \) implies \( E_{\text{DEC}}(\gamma_1) \leq E_{\text{DEC}}(\gamma_2) \), where \( E_{\text{TEST}}(\gamma) \) and \( E_{\text{DEC}}(\gamma) \) are respectively the energy consumption of the test circuit and of the complete decoder when using parameter \( \gamma \). This is reasonable because the test circuit used is very similar to a computation unit used in the final decoder, and the fact that this unit might be instantiated multiple times does not affect the ordering of energy consumption. (A2) The average message error rate in the full decoder is the same as measured in the computation tree. This is the standard DE assumption, which is reasonable for long LDPC codes. (A3) The latency of the complete decoder is proportional to the latency of the test circuit, that is, if \( T_{\text{dec}}(\gamma) \) is the decoding latency of the test circuit when using parameter sequence \( \gamma \), the latency of the complete decoder is given by \( \beta T_{\text{dec}}(\gamma) \), where \( \beta \) is a constant that does not depend on \( \gamma \). This is reasonable because even if the clock period of the complete decoder is increased to accommodate longer interconnects, such an increase is independent of the design parameters we are optimizing.

Clearly, if A1 holds and if the performance of the complete decoder is the same as that of the test circuit, then the solution of the respective energy minimization problems is also the same. Because of A2, the complete decoder can achieve the same residual error rate as the test circuit when the channel quality is the same. Finally, the decoding latency of the complete decoder might differ from the latency of the test circuit, but if A3 holds and \( \beta \) is known, we can minimize the energy of the decoder subject to constraint \( P = (p_e^{(0)}, p_{\text{res}}, T_{\text{dec}}) \) by instead minimizing the energy of the test circuit with constraints \( P' = (p_e^{(0)}, p_{\text{res}}, T_{\text{dec}}/\beta) \).

B. Solver

The optimization problem is solved by adapting the dynamic programming approach proposed in [11]. By quantizing the error probability domain, we can construct a trellis graph where each node corresponds to a pair \( (p_e, t) \), where \( p_e \) is the quantized error probability, and \( t \) is the decoding iteration index. We further establish a set \( S = \{\gamma_1, \gamma_2, \ldots, \gamma_K\} \) of design parameters that can be used throughout the decoding. Each parameter \( \gamma = [V_{\text{dd}}, T_{\text{clk}}, \alpha] \) has an associated ExIT function \( f_{\gamma} \), cost function \( c_{\gamma} \), and iteration latency \( T_{\gamma} \). Since the row-layered architecture updates one VN edge at a time, the iteration latency is given by \( T_{\gamma} = d_{\text{v}} \cdot T_{\text{clk}} \).

We denote the cost of a path \( P \) by \( C_P = (E_P, T_P) \), where \( E_P \) and \( T_P \) are respectively the energy and latency required by the path. Starting from a state \( (p_e, t) \), a rule \( \gamma \) brings the decoder to a state \( (f_{\gamma}(p_e, p_e^{(0)}), t + 1) \) and adds a cost \( (c_{\gamma}(p_e, p_e^{(0)}), T_{\gamma}) \) to the total decoding cost. The energy minimization then simplifies to finding the path \( P \) with lowest energy that starts from the initial state \( (p_e^{(0)}, 0) \) and reaches a state \( (p_e, t) \) such that \( p_e \leq p_{\text{res}} \) and \( T_P \leq T_{\text{dec}} \). The initial state is chosen as the lowest quantized error state such that \( p_e^{(0)} \geq p_e \).

Several properties of the optimal path can be used to limit the number of paths that must be explored. Paths that do not make progress in terms of error rate can be discarded,
and paths that achieve a state worst or equal to another path and that are known to be more expensive can also be discarded. The error probabilities were quantized using 1000 logarithmically spaced states within each decade, and all problems were solved in a few minutes at most.

C. Results

As an example, we consider the (3, 6) code family. The test circuit is synthesized as described in Section II-B, with a nominal $T_{\text{clk}} = 2\text{ ns}$, corresponding to a 6% area increase over the minimal area. We compare the energy consumption of our optimized quasi-synchronous circuit with a standard design that achieves the same performance $(p_{\text{e}}, \alpha, \beta, T_{\text{dec}})$. Recall that the channel threshold of a reliable (3, 6) decoder is approximately $p_{\text{e}}^{(0)} < 0.12$. Suppose that we choose the performance constraint to achieve this threshold, and set the residual error rate at $p_{\text{res}} = 10^{-8}$. We use the latency of the (reliable) standard test circuit as the latency constraint. By allowing to change the value of $\alpha$ during the decoding, with $\alpha \in \{4, 8\}$ the standard test circuit can achieve $p_{\text{res}} = 10^{-8}$ in 24 iterations, requiring 144 ns and 530 pJ.

An optimized quasi-synchronous circuit can instead satisfy the $(0.12, 10^{-8}, 144 \text{ ns})$ constraint using the same number of iterations and the same latency, but consuming only 381 pJ, a 28% reduction in energy consumption. By restricting the set $S$ of available parameters, we can obtain a simpler parameter sequence that nonetheless achieves a 25% reduction in energy.

Finally, we showed that when decoding regular LDPC codes of rate $\frac{1}{2}$, a pipelined sequence of quasi-synchronous decoders is expected to reduce the energy consumption by up to 28%, even when using a standard synthesis method.

ACKNOWLEDGEMENTS

The authors wish to thank CMC Microsystems for providing access to the Cadence tools and TSMC 65nm CMOS technology, and Gilles Rust for advice on Cadence tools and cell library characterization. Warren Gross is a member of ReSMiQ and of SYTACom.

REFERENCES

[1] S. Ghosh and K. Roy, “Parameter variation tolerance and error resiliency: New design paradigm for the nanoscale era,” Proceedings of the IEEE, vol. 98, no. 10, pp. 1718–1751, Oct. 2010.
[2] T. Austin, V. Bertacco, D. Blauuw, and T. Mudge, “Opportunities and challenges for better than worst-case design,” in Proc. ASP-DAC, 2005.
[3] R. Hegde and N. R. Shanbhag, “Energy-efficient signal processing via algorithmic noise-tolerance,” in Proceedings of the 1999 international symposium on Low power electronics and design, ser. ISLPED ’99, New York, NY, USA: ACM, 1999, pp. 30–35.
[4] B. Shim, S. Sridhara, and N. Shanbhag, “Reliable low-power digital signal processing via reduced precision redundancy,” IEEE Transactions on Very Large Scale Integration (VLSI) Systems, vol. 12, no. 5, pp. 497–510, may 2004.
[5] Y. Liu, T. Zhang, and K. Parhi, “Computation error analysis in digital signal processing systems with overscaled supply voltage,” Very Large Scale Integration (VLSI) Systems, IEEE Transactions on, vol. 18, no. 4, pp. 517–526, april 2010.
[6] L. Varshney, “Performance of LDPC codes under faulty iterative decoding,” IEEE Trans. on Information Theory, vol. 57, no. 7, pp. 4427–4444, juillet 2011.
[7] F. Leduc-Primeau and W. J. Gross, “Faulty Gallager-B decoding with optimal message repetition,” in Proc. 50th Allerton Conf. on Communication, Control, and Computing, Oct. 2012.
[8] S. M. S. Tabatabaei Yazdi, H. Cho, and L. Dolecek, “Gallager B decoder on noisy hardware,” Communications, IEEE Transactions on, vol. 61, no. 5, pp. 1660–1673, May 2013.
[9] C.-H. Huang, Y. Li, and L. Dolecek, “Gallager B LDPC decoder with transient and permanent errors,” Communications, IEEE Transactions on, vol. 62, no. 1, pp. 15–28, January 2014.
[10] C. Ngassa, V. Savin, and D. Declercq, “Min-sum-based decoders running on noisy hardware,” in Global Communications Conference (GLOBECOM), 2013 IEEE, Dec. 2013, pp. 1879–1884.
[11] M. Ardakani and F. R. Kschischang, “Gear-shifting decoding,” IEEE Trans. on Communications, vol. 54, no. 6, Juillet 2006.
[12] A. Kahng, S. Kang, R. Kumar, and J. Sartori, “Slack redistribution for graceful degradation under voltage overscaling,” in Design Automation Conference (ASP-DAC), 2010 15th Asia and South Pacific, jan. 2010, pp. 825–831.
[13] C. Roth, P. Meinerzhagen, C. Studer, and A. Burg, “A 15.8 pb/ibit/iter quasi-cyclic LDPC decoder for IEEE 802.11n in 90 nm cmos,” in Solid State Circuits Conference (A-SSCC), 2010 IEEE Asian, Nov 2010, pp. 1–4.
[14] K. Cushon, C. Leroux, S. Hemati, S. Mannor, and W. Gross, “A min-sum iterative decoder based on pulsewidth message encoding,” IEEE Trans. on Circuits and Systems II: Express Briefs, vol. 57, no. 11, pp. 893–897, nov. 2010.
[15] Y. Sun, G. Wang, and J. Cavallaro, “Multi-layer parallel decoding algorithm and VLSI architecture for quasi-cyclic LDPC codes architecture for quasi-cyclic LDPC codes,” in Circuits and Systems (ISCAS), 2011 IEEE International Symposium on, May 2011, pp. 1776–1779.
[16] M. Ardakani and F. Kschischang, “A more accurate one-dimensional analysis and design of irregular LDPC codes,” Communications, IEEE Transactions on, vol. 52, no. 12, pp. 2106–2114, Dec. 2004.
[17] T. J. Richardson and R. L. Urbanke, “The capacity of low-density parity-check codes under message-passing decoding,” IEEE Trans. on Information Theory, vol. 47, no. 2, 2001.