The Power Cost of Over-Designing Codes

Karthik Ganesan, Pulkit Grover, and Jan Rabaey  
{karthikg, pulkit, jan}@eecs.berkeley.edu

Abstract—In modern coding theory, the code-design-goal is often to maximize the code girth, which yields better code performance (low error probabilities). In this paper, we experimentally demonstrate that over-designing codes can hurt: codes with larger girth can consume significantly larger decoding power. The effect is a consequence of the increased length of decoder interconnects that are required with increased girth. Concretely, we show that for (3, 4)-regular LDPC codes of girth 6 and 8 decoded using Gallager-A decoders, the decoders for girth 6 codes can consume up to 25% less power than those for girth 8 codes at high decoding throughputs. Existing results in the literature demonstrate that this effect will be greatly exaggerated at larger degrees and girths.

Index Terms—LDPC Decoders, Low Power Digital Circuits, Code Girth, Interconnects, Tradeoffs.

I. INTRODUCTION

Code and decoder design today is often done in isolation. In coding theory, a code is considered good if it performs well according to the traditional information-theoretic metrics of error-probability and gap from capacity. On the other hand, a decoder implementation is considered good when it consumes low power and has small area. This disconnect has been acknowledged in both theoretical and experimental communities. Consequently, in theory, the issues of implementation have been abstracted as issues of “complexity,” related in various ways to computational complexity of decoding. This acknowledgment, together with the fact that modern codes such as turbo codes [1], low-density parity-check (LDPC) codes [2] etc. approach capacity, led to the revolution that begot modern coding theory [3]. The proof of the success lies in the pudding: it is now possible to have codes and decoders that, at short distances, can work at tens of Gbps (e.g. [4]) while operating at low error probabilities. In absence of low complexity algorithms, realizing such high data-rates would have been extremely hard, if not impossible. This short-distance high data-rate technology is allowing extremely useful applications to be conceived and implemented.

But is the design of codes and decoders in isolation yielding the most efficient systems? While the core focus of theory is on the power required in transmissions, the “balance of power” has gradually shifted towards the decoder: the required decoding power at short distances easily exceeds transmit power at high data-rates in many cases [5], [6]. This shift has been reflected in the choice of codes for which decoders have been implemented. For example, Zhang et al. [4] design a power-optimized decoder for a regular LDPC code (within the 10-GBASE-T standard) that achieves a throughput of 47 GB/s with a power dissipation of 2.8 W. Why do the authors, and indeed also the standard, use regular LDPC codes? After all, theory tells us that regular LDPCs require larger transmit power than their irregular counterparts[1]. A wireless analogy can help. Suppose we want to communicate at a distance of 2 meters. How much is the required transmit power? Depending on the available bandwidth and the choice of code, it can be as small as a few tens of milliwatts! If a state-of-the-art decoder requires power that is on the order of hundred times larger than transmit power, why use a decoder?

This observation prompted many implementations (e.g. Marcu et al. [6]) to do away with the use of forward error correction and simply using low power transceivers with uncoded transmission. The rule-of-thumb gained by traditional theoretical analysis had to be rescinded. On the flipside, this new rule-of-thumb, namely that of using uncoded transmission, increases transmit power, which can cause interference to nearby links. What if we want many links to operate in the same geographical area [7]? Even this new rule-of-thumb breaks! Attempts to build a new theory that combines transmit and decoding power, and proposes joint design of code and decoder, are still in the works (see, e.g. [8], [9], [10], [5], [11] etc.).

This paper brings out another rule-of-thumb that breaks down in this modern era. Simultaneously, it also proposes a short-term “bandaid solution” to the problem while the “surgery” of a new theory is developed. As explained in [12], in constructing LDPC codes that achieve low bit-error probabilities, it is desirable that the codes are designed to have large girth (defined as the shortest cycle in the Tanner Graph representation of the code [13]). Traditionally, large girth codes are preferred because they provide a lower error floor and require a smaller number of decoding iterations for information bits to achieve their optimal decoded values [12]. The twist in the new era is the following: required wire-lengths in decoder implementations increase exponentially with the girth of the code [10]! Longer wires in turn could increase the decoding power. The question of interest in this paper is: by how much? Should we care about this power increase? If not, then the rule-of-thumb of designing codes of largest girth (the approach adopted

1Irregular LDPC codes can operate close to the channel capacity, while the regular codes cannot [3].
in [12], [14] etc.) is still valid!

In Section III we show that large-girth codes are indeed associated with an increased decoding power. Comparing two codes of small block-length, we observe that at high throughputs, this increase can be as large as 25% for just a 6 to 8 increase in girth. The exponential increase of wire-length in code girth ensures that at larger girths, block-lengths, and degrees of LDPC codes, the increase could be significantly larger! This partly explains why high-performance, high-throughput decoders (such as those in [4]) consume such large amounts of power. This effect is compounded by the associated wiring complexity due to large block-lengths that are often utilized to achieve high throughput in such high-performance chips. These results thus exemplify the danger in only constructing codes for the lowest error probabilities and/or only using decoders that are designed for such high-performance codes. Instead, we advocate that code and decoder design should be done in a joint manner so as to achieve the desired code performance, power consumption, area, and throughput for a specific application.

We claim that our results also simultaneously provide a band-aid solution. The problem is that while the high power associated with state-of-the-art decoders [4] suggests that decoding is too expensive, coding is necessitated when many links are operating simultaneously in close proximity. By building tiny decoders (block-length 180) and using simple decoding algorithms (Gallager’s bit-flipping decoders [2]) that have relatively low power consumption (see comparison in Table III), we suggest that use of a decoder may not necessarily be that expensive after all. What is the catch? The performance of our decoders is likely much worse than that of comparable decoders of larger block-lengths and high-complexity decoding. The results of this paper should therefore be seen as suggestive of a much richer set of ideas. In particular, at small block-lengths, one can envisage bringing back codes from classical coding theory (e.g. Reed-Solomon codes, BCH codes, etc.), and comparing their power with those of LDPC decoders.

II. MODEL AND ASSUMPTIONS

A. Decoder Model

To compare two codes of different girths, we choose (3, 4)-regular LDPC as the basic code construction that specifies the left and right degrees of the code. We use the guess-and-test algorithm in [15] and [14] to obtain a girth-6 code of block-length 20 and girth-8 code of block-length 36. The MATLAB code for this code generation algorithm can be found on [16]. The Tanner graphs for our chosen codes are shown in Figure 1. The end-to-end channel is modeled as a BSC.

Then, a simulation flow using CAD tools is carried out in order to obtain decoding power consumption for the chosen codes. The STMicroelectronics 90 nm Standard CMOS Process with 7 metal layers is used in the design of our decoders for simulation. Decoders are designed in a heirarchical manner, where behavioral verilog descriptions of check-nodes and variable-nodes (described in [2]) are first mapped to standard cells using logic synthesis and then layed out using place & route. These decoders use the Gallager A iterative message passing algorithm [2] with 1-bit messages. Then, these check-node and variable-node cells are connected according to their respective parity-check matrices to create block level layouts for girth-6 and girth-8 decoders (see Figure 2). Since the varying block-length of codes of different girth directly affects the decoder area and the wiring complexity, we fix the information density per unit area when creating the decoder layouts. Post-layout simulation is then performed, using extracted RC parasitics from the layout and typical corners for the ST 90 nm CMOS process. Various components of the total power consumption (including computation + glitches, interconnects, and clock tree) are isolated by means of post-layout netlist modification. Starting from a nominal supply voltage of 1.2 V, power reduction steps are taken by means of supply voltage scaling, and a range of clock frequencies at which the decoders can be run is determined. During this process, no changes to the circuit topology or transistor sizing in the decoder cells are made. Then, we perform a comparison of the power consumption for girth-6 and girth-

![Fig. 1: Tanner Graphs for (3,4)-regular LDPC Codes of block-length 20, girth-6 (bottom) and block-length 36, girth-8 (top).](image1)

![Fig. 2: Floorplan views (to scale) of (3,4)-regular LDPC Decoders of block-length 20, girth-6 (left) and block-length 36, girth-8 (right). The layout dimensions are 146.25 μm x 146.25 μm for girth-6 and 196 μm x 196 μm for girth-8. Individual check-node and variable-node cells consume 17.92 μm x 17.92 μm area each and the circuit topology and sizing of transistors in these cells was not changed between girth-6 and girth-8 decoders.](image2)
8 decoders by means of stacking decoders in fully parallel configurations (where the block-length of the resulting code is a multiple of the LCM of the two original codes). This ensures a fair comparison for codes of different lengths: the decoding throughput is the same for a given clock cycle.

The simulations are performed assuming that the received sequence is an all-zero sequence. An assumption of all-zero transmitted sequence (codeword) is easy to justify [17] due to the linearity of the code, the symmetry of the channel, and the symmetry of the decoder with respect to ones and zeros. The received sequence however has errors, and will therefore require more switching activity at the decoder than what our simulations indicate. However, our empirical tests suggest that at our error-probabilities of operation the required power does not increase substantially for our decoders. This is because the raw bit-error probability from the channel for our decoders is already pretty low, making decoding errors rarer. A more refined analysis will need to perform simulations for various error-probabilities.

B. Full System and Transmission Model

Once decoding power consumption is obtained through simulation, the required transmit power is calculated under the following assumptions:

- BPSK modulation scheme is used
- Channel is AWGN
- Frequency (f) of 60 GHz
- Bandwidth (W) of 7 GHz
- The decoder performs a hard-decision on the observed channel outputs before starting the decoding process.

Then, we consider total power budgets (\( P_{\text{transmit}} + P_{\text{decoding}} \)) in the range of 100 mW to 1 W. We first fix the decoding throughput to be 30 Gbps and consider running decoders for 1 iteration, 2 iterations, or 3 iterations in the case of girth-8. Then, we take the corresponding decoding power numbers obtained from simulation and consider those fixed (see Section II-A for explanation). We subtract this from the total power budget and obtain the maximum allowable value of \( P_{\text{transmit}} \) for a given decoder to run for a given number of iterations. Then, we convert this transmit power to the channel flip probability \( P_{ch} \) using the Frii’s transmission equation:

\[
P_{ch} = Q \left( \sqrt{\frac{P_T}{(\lambda)\alpha kTW}} \right),
\]

where \( Q \) is the tail probability function of the standard normal distribution:

\[
Q(x) = \frac{1}{\sqrt{2\pi}} \int_x^\infty e^{-u^2/2} du,
\]

\( \lambda \) is the wavelength of transmission at \( f = 60 \) GHz:

\[
\lambda = \frac{3 \times 10^8}{60 \times 10^9 m} = 5mm.
\]

In the above equation, \( k \) is the boltzmann constant (1.38 * \( 10^{-23} \) J/K) and \( T \) is the temperature (assumed to be 300K).

We vary \( r \) (transmit distance) and \( \alpha \) (path-loss exponent) to simulate the effect of varying transmit conditions and compare the BER performance of girth-6 and girth-8 codes under different configurations.

To analyze the BER-performance of our LDPC decoders we apply the formula for the evolution of bit-error probabilities with number of decoding iterations using Gallager A message passing algorithm (given in [2]):

\[
p_c^{(l)} = p_{ch} - p_{ch} \left( \frac{1 + (1 - 2p_{ch}^{l-1})^{d_v-1}}{2} \right)^{d_c-1} + (1 - p_{ch}) \left( \frac{1 - (1 - 2p_{ch}^{l-1})^{d_c-1}}{2} \right)^{d_v-1}
\]

(1)

Where the parameters are defined as:

- \( l \) - Number of Decoding Iterations (2-3)
- \( d_c \) - Check Node Degree (4)
- \( d_v \) - Variable Node Degree (3)
- \( p_{ch} \) - Channel Flip Probability

A plot of these bit-error probabilities as a function of the \( \frac{E_b}{N_0} \) for BPSK transmission over the AWGN channel is given in Figure 3.

![Fig. 3: Bit Error Probabilities for (3,4)-regular LDPC (2 and 3 iterations), repetition code of rate 1/4, and uncoded transmission as a function of \( \frac{E_b}{N_0} \). We assume BPSK transmission over an AWGN channel and compare our result to a repetition code which uses a majority function for decisions (with additional fair coin flip if necessary).](image)

III. SIMULATION RESULTS

A. Decoding Power - Code Performance Tradeoff

From our decoder simulations, we obtained a plot of Total Decoding Power Consumption vs. Throughput for block-length 180 (3,4)-regular LDPC Decoders of girth-6 and girth-8 (see Figure 4) and a plot of the Percentage increase in total decoding power consumption between girth-6 and girth-8 (see Figure 5). Our results show that decoding power consumption can increase upwards of 25% with only a 6 to 8 increase in girth.
B. Break-up of the power consumption: the cost of long interconnects

In order to determine which components of the total decoding power were contributing most to this increase, supply nodes in the decoder netlists were grouped based on whether they supplied current to the clock tree cells, the interconnects between variable-nodes and check-nodes, or the standard cells used for logic. Then, the power consumption for each of these three groups was simulated, giving separate estimates of the power consumed by interconnects between variable-nodes and check-nodes, the clock tree, and the rest of the decoder. A plot of the power consumed strictly by interconnects between variable-nodes and check-nodes is given in Figure 6. From these results, we noticed that the non-interconnect, non-clock-tree power consumed the majority of the total decoding power at all throughputs (72% on average for girth-6 decoders and 73.4% on average for girth-8 decoders). Since our decoding simulations were performed using the all-zero codeword, all parity checks and messages passed from variable-nodes to check-nodes were zeros, and therefore we expected the power consumed purely by computation to remain roughly the same when comparing our parallel decoders of block-length 180. However, we noticed that the non-interconnect, non-clock-tree power still increased by 10% on average from girth-6 to girth-8, even under this scaled comparison. At the highest throughputs, this percentage increase exceeded 25%, making it clear that this remaining power component was not purely computational.

Upon closer observation of the waveforms in our variable-node, we noticed that during the first decoding iteration, many glitches propagated through the critical path logic to the inputs of the message passing registers. Due to long wires of unequal lengths from the channel inputs to the variable-node logic, the resolution of these glitches was delayed. Due to proper timing of our decoder, these glitches were resolved before the first messages were passed and the decoders operated properly, however, large amounts of power were still consumed by these glitches. Table II shows this component of the power consumption separated across the two iterations of the simulation. The large disparity between the average power consumed in the two iterations is at least partly due to these glitches, as they did not occur during the second decoding iteration. From these results, we observe that interconnects between check-nodes and variable-nodes in these simple decoder implementations not
<table>
<thead>
<tr>
<th>Average Computation + Glitching Power by iteration</th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>Iteration</td>
<td>Girth-6</td>
</tr>
<tr>
<td>First Iteration</td>
<td>12.35 mW</td>
</tr>
<tr>
<td>Second Iteration</td>
<td>2.12 mW</td>
</tr>
</tbody>
</table>

TABLE I: Average (across all throughputs) Computation + Glitching power consumed in the first iteration and second iteration for block-length 180 (3,4)-regular LDPC Decoders of girth-6 and girth-8.

<table>
<thead>
<tr>
<th>Average Percentage of Total Decoding Power</th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>Component</td>
<td>Girth-6</td>
</tr>
<tr>
<td>Computation + Glitches</td>
<td>71.7%</td>
</tr>
<tr>
<td>Interconnects</td>
<td>16.2%</td>
</tr>
<tr>
<td>Clock Tree</td>
<td>12.2%</td>
</tr>
</tbody>
</table>

TABLE II: Average (across all throughputs) percentage breakdown for various components of total decoder power consumption for block-length 180 (3,4)-regular LDPC Decoders of girth-6 and girth-8.

only consume upwards of 16% of the total decoding power, but other long wires in the decoders (including those that compose the clock tree) are also responsible for a large part of the remainder of the power consumption. Hence, short wire lengths are optimal in decoder implementations for a number of reasons. A summary of this breakdown of power consumption is provided in Table II.

C. Tradeoff between Transmit and Decoding Power

Following the method described in Section II-B, we considered transmission over short-distances with a path-loss exponent of $\alpha = 3$. At sufficiently large distances (those beyond 10m), there is no noticeable difference in the best attainable BER performance of girth-6 and girth-8 decoders.

However, if we decrease our transmit distance to 3.15m, we find that running decoders for 2 iterations becomes optimal within the range of considered total power budgets. Furthermore, we notice that running a girth-6 decoder for 2 iterations outperforms a girth-8 running for 2 iterations by close to two orders of magnitude in the error probability (see Figure 7).

Inbetween these two extreme cases, there are cases where running a girth-8 decoder for 3 iterations becomes optimal for some total power values. If we increase our transmit distance to 5.25m (see Figure 8), we notice that girth-6 codes still outperform girth-8 codes at low total power budgets, but the difference in error probability is decreased to roughly one order of magnitude. It becomes optimal to run a girth-8 decoder for 3 iterations only at around 900 mW of total power. Hence, the total power budget needs to be sufficiently large in order for an investment in large girth codes to be advantageous.

IV. DISCUSSIONS AND CONCLUSIONS

In this work, we have demonstrated that a small increase in code girth significantly increases the power consumption of short block-length LDPC decoders that implement 1-bit Gallager A message passing. We have also shown that interconnects in these simple decoder implementations are responsible for a large part of this power consumption. Our results also show that the power and area costs associated with a decoder can be much smaller than those for state-of-the-art decoders. Although such simple decoders sacrifice on BER performance, we have shown that they could provide a competitive alternative to uncoded transmission, especially since their low power consumption (at some throughputs comparable to ADCs [6]) and very small area allows for them to be easily accommodated in receiver implementations. Thus, through this example we suggest that low-performance codes/decoders may still be worth investing in and may be more suitable than their high-performance counterparts for specific applications.

However, the manner in which these tradeoffs scale for decoders with sophisticated message passing schemes (such as multi-bit and bit-serial messages) that are commonly used in high-performance decoders remains to be shown. In this work, we considered a 6 to 8 increase in code girth for (3,4)-regular LDPC codes. To achieve a girth of 10
for (3,4)-regular LDPC, a minimum code block-length of 156 is required [15]. Not only would this greatly increase the wiring complexity and area of our implementations, but from the results in this paper and those in [10], we can conjecture that the corresponding increase in decoding power will be much larger than what we have observed in Section III. An increase in the degree of our variable-nodes \( d_v \) and check-nodes \( d_c \) in the decoder can also lead to a similar increase in power: results in [15], [14] show that an increase in \( d_c \) from just 4 to 5 requires a block-length of 70 for girth-8 and 315 for girth-10! Clearly, the effect of wiring complexity on power consumption is an important design criterion for simple one-bit decoders. While methods for reducing wiring congestion for LDPC decoders of fixed girth exist and have proven effective [18], [19], the large increases in block-length that are required to increase the code girth will still increase the area of the decoder and hence, the overall lengths of wires required in the decoder. As we have shown in this paper and [10], this will lead to a large increase in decoding power consumption.

ACKNOWLEDGEMENTS

We acknowledge the support of NSF grant CCF-0917212. We thank Anant Sahai for stimulating discussions. We acknowledge the help of Jose Moura in providing code constructions that we started our simulations with. We thank the students, staff, faculty, and sponsors of the Berkeley Wireless Research Center and Wireless Foundations. In particular, Brian Richards assisted with the simulation flow. We also acknowledge STMicroelectronics for making available the technology information that was used in our simulations.

REFERENCES