Energy-Efficient High-Throughput Staircase Decoders

Downloaded from: https://research.chalmers.se, 2019-10-02 07:48 UTC

Citation for the original published paper (version of record):
Energy-Efficient High-Throughput Staircase Decoders
2018 Optical Fiber Communications Conference and Exposition, OFC 2018 - Proceedings: 1-3
http://dx.doi.org/10.1364/OFC.2018.TU3C.6

N.B. When citing this work, cite the original published paper.
Energy-Efficient High-Throughput Staircase Decoders

Christoffer Fougstedt and Per Larsson-Edefors
Dept. of Computer Science and Engineering,
Chalmers University of Technology, SE-412 96 Goteborg, Sweden
chrfou@chalmers.se

Abstract:
We introduce staircase decoder implementations achieving up to 1-Tb/s throughput with energy
dissipation of 1.2 pJ/information bit. The implementations are estimated to achieve >10.5 dB of net
coding gain depending on the configuration.

OCIS codes: (060.0060) Fiber optics and optical communication; (060.2330) Fiber optics communications

1. Introduction

Staircase codes [1] have attracted considerable interest in the research community. While staircase codes have been
considered at an algorithmic level [2, 3], to the best of our knowledge, no studies on application-specific integrated
circuit (ASIC) implementation aspects have been published in the open literature. In this paper, we will describe and
evaluate circuit implementations of a staircase decoder. Using a window to store staircase data blocks and a set of
Bose-Chaudhuri-Hocquenghem (BCH) decoders for the component codes, the staircase decoder design we propose
can support a wide range of implementations in response to different throughput needs.

The actual probability of performing correction of an error in a component code-word depends on the position
of the code-word in the window and the number of iterations performed. Since errors are successively corrected,
the component decoders are more active in the front-end of the window and during the first iterations. As power
dissipation depends on signal switching statistics, it is crucial to recognize the significant spatial and temporal variation
in switching activities when developing an energy-efficient decoder implementation. Here, the decoder design benefits
greatly from gating of the clock when circuits are idle, to reduce clock power and redundant logic signal switching.

Beside energy efficiency and coding gain, also throughput and latency are critical system parameters. The BCH
decoders that we employ in the staircase are an extension of our previous work [4]. They operate in a non-iterative
manner which simplifies the design of state machines for staircase control. Additionally, since the component codes
can be decoded with low-latency circuits, we are able to achieve very high staircase decoder throughput.

We will first introduce the staircase decoder architecture and the constituent parts, i.e., staircase window and BCH
decoders, with syndrome calculation, key-equation solver and Chien search. Then we will evaluate two implementa-
tions with different error correction capabilities and discuss the results that we obtain after synthesizing the implemen-
tations to a 28-nm process technology. Finally, we conclude the paper.

2. ASIC Implementation

Using the notation BCH(n,k,t), where n is the block length, k is the number of useful information bits, and t is the
number of bit errors that the code can correct, we here use BCH(511,484,3) and BCH(511,475,4) codes shortened
to 324 and 432 bits respectively, resulting in staircase codes with 20% overhead, with staircase code block lengths of
26,244 and 46,656, respectively.

2.1. Component Decoder Implementation

The implemented staircase decoders are based on the shortened BCH component codes above. The component codes
are decoded using a fully-parallel non-iterative direct-solution algorithm that has been modified to remove Galois field
(GF) inversions. While our previous paper [4] introduced such decoders for t = 1 and t = 2, here we use recently
developed BCH decoders with t = 3 and t = 4. The BCH decoders are pipelined between the syndrome computation
stage, the key-equation solver (KES), and the Chien search; the decoder thus decodes one component code-word in
three clock cycles. Since power dissipation depends on signal switching activities, the pipelining registers are clock
gated in sequence if a zero-syndrome is detected.

The number of found roots in the Chien search stage is compared to the expected number of roots from the error-
locator polynomial order. If they are not equal, the found roots are discarded. This reduces the miscorrection probabil-
ity, since a miscorrection can infer errors in the part removed when shortening the code, giving a discrepancy between
found roots and polynomial order.
2.2. Staircase Decoder Implementation

Power dissipation depends on the number of bits of a word that are switching, rather than the word length itself. One corrected error can cause at most \( \lceil \log_2(n) \rceil \) toggles in the syndrome computation tree, so the majority of gates in the XOR-tree remain static. Thus, our implementations use one syndrome computation block per row and per column, resulting in mostly idle logic gates during iterations, with the added benefit of reduced word length in the row-column muxes. The Peterson-based KES and Chien search are shared between one row and one column. Fig. 1a shows a block diagram of one row/column pair.

The implemented staircase decoders operate on a window of 5 blocks. The staircase memory is clock gated when not written to, and each block is clock gated if the row and column syndromes indicate no errors. The window is shifted after a specific number of iterations, which is a system parameter that we can change, and syndromes are then recomputed. While syndrome-recomputation may seem wasteful, it turns out that the syndrome computation dissipates less than 10% of the total power in the implemented staircase decoders.

Since there are no data dependencies between rows and no dependencies between columns, the implemented staircase decoders iterate between decoding, first, all rows and, then, all columns, instead of decoding the blocks in sequence. This design solution enables an increased throughput and was not found to incur any significant error-correction performance degradation in our 5-iteration MATLAB reference implementations.

2.3. Implementation Area and Power Dissipation

Fig. 1b shows the power dissipation and area for different circuit blocks of the staircase decoder, using the evaluation methodology outlined in Sec. 3. It is clear from the figure that the power dissipation of component decoders, for reasons outlined in Sec. 1, is not very significant. Also, the figure indicates that power dissipation and area are not strongly correlated, illustrating that algorithm complexity is not a very useful metric for power dissipation in circuits whose signals have very different and varying switching activities. The disconnect between abstract complexity metrics and power dissipation is exacerbated by the addition of a clock tree, which is required for synchronization but which has no direct correspondence in an algorithmic model.

3. Results and discussion

The decoders were synthesized using a 28-nm FD-SOI standard-cell flow using regular threshold voltages, slow process corner, 0.9 V, and 125°C in Cadence Genus using physical wire models, at a clock rate of 550 MHz. Using Cadence Incisive, the resulting netlists were simulated in a VHDL testbench generating uniformly-distributed staircase-encoded input data, which were transmitted over a binary-symmetric channel (BSC), with a bit-error rate (BER) of \( 10^{-2} \). Beside providing verification of functionality, this simulation generated internal switching activity statistics which were backannotated to the netlist in Cadence Genus to estimate power dissipation using the typical process corner at 25°C. Clock-tree estimation was performed in Cadence Genus. Synopsys PrimeTime was used to estimate the distribution of data and clock internal power in registers for a \( t = 3 \) decoder implementation using 5 iterations; approximately 70% of the internal register power was found to be caused by clocking. Static power dissipation accounts for less than 1% of total power.
Table 1: Evaluation Results

<table>
<thead>
<tr>
<th></th>
<th>$t = 3$</th>
<th></th>
<th>$t = 4$</th>
</tr>
</thead>
<tbody>
<tr>
<td>Iterations</td>
<td>3 4 5 6</td>
<td>Iterations</td>
<td>3 4 5 6</td>
</tr>
<tr>
<td>Cell area (mm$^2$)</td>
<td>7.37</td>
<td>Cell area</td>
<td>18.37</td>
</tr>
<tr>
<td>Throughput (Gb/s)</td>
<td>601 463 376 317</td>
<td>Throughput</td>
<td>1069 823 668 563</td>
</tr>
<tr>
<td>Power dissipation (W)</td>
<td>0.601 0.534 0.491 0.463</td>
<td>Power dissipation</td>
<td>1.298 1.132 1.028 0.955</td>
</tr>
<tr>
<td>Energy per information bit (pJ/bit)</td>
<td>1.00 1.15 1.31 1.46</td>
<td>Energy per information bit</td>
<td>1.21 1.38 1.54 1.70</td>
</tr>
<tr>
<td>Estimated net coding gain (dB)</td>
<td>10.0 10.1 10.3 10.3</td>
<td>Estimated net coding gain</td>
<td>10.4 10.5 &gt;10.5 &gt;10.5</td>
</tr>
<tr>
<td>Block decoding latency (ns)</td>
<td>181.8 236.3 290.9 345.4</td>
<td>Block decoding latency</td>
<td>181.8 236.3 290.9 345.4</td>
</tr>
</tbody>
</table>

BER simulations were performed on the VHDL staircase decoder implementation using a VHDL BSC testbench, in which the input BER was swept, providing both a functional verification and performance metrics. The resulting output BER was used to estimate coding gain by extrapolation down to $10^{-15}$ using the `berfit` function in MATLAB. We want to stress that these estimations should be considered as approximations, since excessive runtime limits accurate low-BER statistics. However, the results are consistent with [2, 3], taking into account algorithmic differences.

Table 1 presents the implementation data obtained for the two different decoders ($t = 3$ and $t = 4$). The number of iterations has an impact on throughput, power dissipation, energy per bit, net coding gain and latency, so we list data for 3–6 iterations.

Focusing on throughput, we can notice that the decoder with $t = 4$ can provide very high throughput (in excess of 1 Tb/s) at an energy efficiency of 1.21 pJ/bit. However, the area increase from $t = 3$ to $t = 4$ is substantial and indicates $t = 5$ may not be cost effective from an area utilization perspective. Thanks to extensive clock gating of idle decoder portions, the power dissipation of the 1-Tb/s implementation is limited to under 1.3 W.

As far as energy efficiency, we can reach as low as 1.0 pJ/bit for a 600-Gb/s implementation with a net coding gain of 10.0 dB. As the number of iterations is increased, the power dissipation is decreasing because the signal switching activity is going down for the later iterations. Even though the power dissipation is decreasing with iteration count, the energy efficiency is degrading. This is caused by a reduction in throughput due to an increasing number of iterations. In comparison to recent low-power soft-decision LDPC implementations [5], our staircase decoders achieve better energy-efficiency and higher throughput at the expense of slightly lower coding gain.

4. Conclusion

We presented energy-efficient staircase decoder ASIC implementations, which were evaluated in a 28-nm FD-SOI process technology. Depending on decoder configuration, the implementations can achieve up to 1-Tb/s throughput at a power dissipation of 1.3 W, resulting in an energy per information bit of 1.21 pJ/bit. The implemented decoders are estimated to achieve between 10 dB and >10.5 dB net coding gain, and energy per information bit ranges from 1.0 to 1.7 pJ/bit, depending on configuration.

Acknowledgement: The authors would like to thank Dr. Lars Svensson and Dr. Kevin Cushon for fruitful discussions. This work was financially supported by the Knut and Alice Wallenberg Foundation.

References