A 600-Mb/s Encoder and Decoder for Low-Density Parity-Check Convolutional Codes

Tyler Brandon, John C. Koob, Leendert van den Berg, Zhengang Chen, Amirhossein Alimohammad, Ramkrishna Swamy, Jason Klaus, Stephen Bates, Vincent C. Gaudet, Bruce F. Cockburn and Duncan G. Elliott
Department of Electrical and Computer Engineering
University of Alberta, Edmonton, Alberta, Canada
Email: {brandonyjkoob|bergzgchen|amir|swamy|klaus|sbates|gaudet|cockburn|elliott}@ece.ualberta.ca

Abstract—A 600-Mb/s rate-1/2 (128,3,6) LDPC convolutional code encoder and decoder was implemented in a 90-nm CMOS process. The encoder operates at 1.1 GHz and includes built-in all-phase termination. The decoder design maximizes throughput while minimizing the number of memory banks and delivering an information throughput of 1 bit per clock cycle. The size of the decoder controller is minimized by sharing it among an arbitrary number of decoder processors. The decoder dissipates 0.61 nJ of energy per decoded information bit at an SNR of 2.0 and a throughput of 600 Mb/s. An integrated test system enables accurate power measurements for various SNR settings.

I. INTRODUCTION

In the early 1960’s, Robert Gallager discovered that low-density parity-check (LDPC) codes had promising error correction properties [1]. However, due to large computational resource requirements, LDPC codes were not pursued until it became technologically practical to exploit their full potential [2]. LDPC convolutional codes (LDPC-CCs) may be more suitable than LDPC block codes (LDPC-BCs) for applications where the data size is unbounded, such as streaming video, or variable-length, such as packet-switching networks [3]. As well, LDPC-CCs enable low-power encoder implementations with low hardware cost, which are suitable for distributed sensor networks. We investigated the advantages of LDPC-CC encoders and decoders in previous implementations for FPGAs and ASICs in [4]–[6]. In this paper, we present a 600-Mb/s LDPC-CC implementation in a 90-nm CMOS process that permits power and performance measurements of individual system modules. This implementation also features an improved termination mechanism in the encoder and an efficient decoder design.

A brief introduction to LDPC-CCs is provided in Section II. In Section III, the architecture of the encoder and decoder is discussed in detail. Section IV summarizes the main features that facilitate chip testing. Power consumption and performance measurements of the fabricated test chip are presented in Section V. We conclude the paper in Section VI.

II. OVERVIEW OF LDPC CONVOLUTIONAL CODES

The first LDPC convolutional codes were proposed by Jimenez-Felström and Zigangarov in 1999 [7]. LDPC-CCs and LDPC-BCs share some similarities: they both are linear codes that generate code bits based on parity-check operations [8] and they both have capacity-approaching bit error rate performance [7], [9]. However, for LDPC-CCs, any given code bit, \( v(t) \), is generated using present and previous information bits, \( u(t) \), and previously generated code bits. As well, an LDPC-CC can encode frames of variable size [7], [9].

While the performance of an LDPC-CC is directly related to the size \( M \) of its memory, the complexity of the encoder and decoder is also affected. For example, the code bit generation equation of the regular (128,3,6) rate 1/2 code we implemented can be written as

\[
v(t) = V(t) + U(t).
\]

If the parity-check equations are time-varying with period \( T \), at time \( t = nT + \phi \) we would have

\[
V(nT + \phi) = v(nT - M + \phi) + v(nT - \delta(v)(\phi))
\]

\[
U(nT + \phi) = \sum_{i=1}^{3} u(nT - \delta_i(u)(\phi))
\]

where \( n \geq 0, \phi \in \{0, 1, 2, \ldots, T-1\} \) and \( + \) implies the XOR operation. As well, \( \delta(v)(\cdot) \) and \( \delta_i(u)(\cdot) \in \{1, 2, \ldots, M-1\} \) are a function of the code and are related to the associated parity check matrix. The code bit generated at time \( t = nT + \phi \) forms a part of the parity check with previous information and code bits up to \( M \) clock cycles in the past. More information on LDPC-CC encoders and decoders appears in [4], [7].

III. ENCODER AND DECODER ARCHITECTURE

A. Encoder

The LDPC-CC design includes an encoder with built-in all-phase termination. For convolutional LDPC codes, the encoder adds redundancy to a stream of information bits and is typically far simpler than its block-based counterpart. In our 90-nm implementation described in this paper, the encoder design was optimized compared to the previous implementation in [6]. That is, the encoder was redesigned to consume as little power as possible while operating at 1.1 GHz.

As shown in Fig. 1, the structure of encoder an LDPC-CC encoder consists of phase control, information and code registers and code bit generation logic. The phase signal uses one-hot encoding, which reduces the power consumption and allows a faster clock. Using the one-hot encoded phase, the previous information and code bits are loaded into the appropriate registers. The current code bit is generated based on the XOR of a specific set of previous code and information.
Convolutional codes require a termination scheme to ensure that the trailing information bits at the end of a transmission are fully protected. However, built-in termination introduces encoder area overhead (see Section V). Termination is the method by which the encoder is returned to the all-zero state, where information bit zeros sent to the encoder result in zeroed code bits. This state is important because the decoder can then generate zeros at its input to complete the decoding process.

Our 90-nm LDPC-CC implementation uses the all-phase termination scheme described in [10]. Fig. 2 shows our architecture for the termination circuitry. The termination bit is generated from the current state of the encoder. Based on the phase, approximately half of the information and code bit values are XORed to form the termination bit, which is then fed back into the encoder as the current information bit. Within approximately $M$ cycles the encoder will reach the all-zero state, at which point the termination bit will remain at ‘0’. As the termination circuitry is inactive for most of the time, its effect on power consumption during normal operation consists only of leakage power.

**B. Decoder**

An LDPC-CC decoder can be divided into several processors. In a previous LDPC-CC implementation [4], each processor had its own controller. In the new design, one controller can be shared among an arbitrary number of processors. Each processor performs the same set of operations as defined by the code. By registering data and control signals between processors, it is possible to share one set of control circuitry.

1) **Decoder Data Path:** The decoder processor data path, as shown in Fig. 3, has seven pipeline stages and an 8-bit data path. Starting with a memory read, a log-likelihood ratio (LLR) is registered at the output of the memory, as well as at the input and the output of the switch matrix. The LLR then proceeds through the check-node or variable-node and is registered at the output. The LLR is registered as it passes back through the switch matrix and as it reaches the memory input. The memory write itself counts as the seventh register.

2) **Decoder Control Path:** The decoder controller consists of memory controllers and a switch matrix controller. Control signals are registered for one cycle in the current processor before being passed to the next processor. This way, two cycles are available to evaluate parity-check and variable node operations on incoming data before their effect on the output is passed out of the processor. The switch matrix routes signals to or from the memories, the check-node, the variable-node and the processor’s inputs and outputs.

3) **Decoder-Memory Interface:** One design goal was to implement the convolutional code using the smallest possible number of memory banks while preserving an information throughput of 1 bit per cycle. In addition, since dual-port memories have approximately 1.5-times greater area-usage and higher power consumption, only single-port memories were used. It is possible to more effectively utilize the memory banks by selecting larger codes, which results in more entries per memory bank. With larger codes, there are fewer conflicts in the memory operations because the LLRs reside in the memory for longer periods of time between memory accesses. A lower bound on the number of memory banks depends on
the degree of the variable-node and check-node operations. In our case, the variable and check node circuitry receive two of the LLR inputs from other nodes and twelve other LLR inputs from the memory banks.

During the design of the memory-decoder interface, it was important to consider the constraints that are required to sort the LLRs. The presence of seven internal pipeline stages in the data path increases the complexity of this problem. A graph-coloring algorithm was developed to not only find the minimum number of memory banks, but to sort the LLRs so as to balance the number of LLRs in each memory bank.

4) Power Consumption Issues: The decoder was originally designed to function with custom SRAMs, but registers were instantiated instead due to time constraints. Ultimately, SRAM is preferable over registers because SRAM can reduce power consumption, conserve area while fulfilling access time and bandwidth requirements.

In hardware design, it is advantageous when the control path is completely independent of the data path. With LLRs allocated to specific memory banks, the control of a memory bank is independent of the other memory banks and the rest of the system. Since the individual controller modules are synchronized, each controller can operate without communicating with any other module. This reduces the amount of long-distance routing and the associated power consumption. The same principle applies to the design of the switch matrix controller.

IV. TEST CHIP ARCHITECTURE

As shown in Figs. 4 and 5, the LDPC-CC is implemented as a complete system-on-chip. Built-in self-test (BIST) features, an additive white Gaussian noise (AWGN) channel, and a phase-locked loop (PLL) are critical for high-speed testing.

A. BIST Features for DFT

The BIST module includes a FIFO, an LFSR, event counters, a cyclic-redundancy-checker (CRC) and an error counter.

To verify the operation of individual modules at high speed, the CRC can generate a 32-bit signature. A flexible routing network between all modules permits various configurations of the flow of data through the modules. We clock-gated all modules to allow us to accurately measure module power.

B. Integrated AWGN Generator

To eliminate the need for an external noise generator, we implemented the compact and accurate Gaussian noise generator (GNG) described in [11]. The GNG can be readily configured to achieve arbitrary tail accuracy with only small variations in hardware utilization, without degrading the output sample rate. The accuracy of the GNG is important when evaluating the performance of an LDPC system at low BER regions. The adverse impact of an inaccurate GNG on performance evaluation is shown in [11].

C. Integrated Phase-Locked Loop

The on-chip PLL makes it possible to run high-speed tests on economical lower-speed digital test equipment. We designed an integer-N charge-pump PLL to synthesize the core clock frequency for our LDPC-CC test chip. A programmable divider allows the reference clock to be multiplied by a factor of 4, 8 or 16 and produces a lock range from 100 MHz up to 2 GHz. The PLL consumes less than 1 mW at 2 GHz.

V. TEST RESULTS

We tested the LDPC-CC chip shown in Fig. 5. While the encoder in our 90-nm implementation can operate independently at 1.1 GHz, the maximum core frequency of the entire LDPC-CC system is limited to 600 MHz. The silicon area consumed by key system modules is shown in Table I. The termination circuitry almost quadruples the area of the encoder. The three-processor decoder occupies ten times the area of the encoder.

For power measurements, the core power supply was 1.0 V, the SNR was set to 2.0 and the core clock frequency was 600 MHz. As discussed in Section IV, the power of individual modules can be measured independently. At first, only the power of the BIST, which includes leakage power, was measured. Then, the power of the encoder can be derived by enabling the encoder module, measuring the power consumption and then subtracting the power allocated to the BIST. Table II
lists each measurement and calculation step for obtaining the power consumed by the encoder, the AWGN generator and the decoder. With the simplifying assumption that the associated phase control generator draws negligible power, each of the three decoder processors consumes at most 123 mW. When operating at 1.1 GHz, the encoder module consumes 22 mW.

Fig. 6 (a) shows the energy per decoded information bit versus frequency for various SNR settings. SNR affects the power consumption by skewing the distribution of probabilities of a ‘1’ or a ‘0’. The decoder has less work to do when SNR is high because fewer channel samples are in error. Conversely, the decoder works harder when the SNR is low because the distribution of channel samples widens and more of the channel sample magnitudes deviate further from the nominal values. In Fig. 6 (a), the lowest energy per decoded bit occurs for infinite SNR, when the channel values are saturated and the decoder does not alter the original channel samples in any way. The remaining power is consumed by the clock-tree network, the control signals and the sign-bits flowing through the decoder data path.

For our implementation, the energy associated with correcting errors within the channel samples can be roughly approximated as the difference in energy between an SNR of infinity and an SNR 2.0. At an SNR of 2.0, roughly 20% of the decoder energy is due to error correction activity, while the other 80% is associated with the implementation. Power is linearly related to frequency and the energy per bit is a direct measure of the amount of work done to decode an information bit. However, in Fig. 6 (a), it is interesting to observe that the energy per bit actually decreases with frequency. This attributed to leakage currents. With shorter clock periods at higher frequencies, a constant leakage current occurs for infinite SNR, when the channel values are saturated and the decoder does not alter the original channel samples in any way. The remaining power is consumed by the clock-tree network, the control signals and the sign-bits flowing through the decoder data path.

Note that Fig. 6 (a) shows a decrease in energy with an increase in SNR. The energy for 3 processors at 600 MHz at an SNR of 2.0 is 0.61 nJ per decoded information bit. The energy per bit for 30 processors would actually be less than ten times this value (6.1 nJ) due to an effective decrease of the average SNR as LLRs pass through the 30-processor decoder.

Fig. 6 (b) shows the BER versus SNR for our LDPC-CC design with 10 and 30 processors using 8-bit LLRs. Two extra curves, for a 10-iteration LDPC-BC and a 10-processor LDPC-CC [6], are included in this figure for comparison.

VI. CONCLUSION

In this paper we presented an encoder and decoder architecture for LDPC convolutional codes that targets high-throughput operation. The encoder can function at 1.1 GHz while drawing 22 mW of power. When our three-processor decoder operates at 600 MHz with an SNR set at 2.0, it consumes 369 mW of power. A 30-processor decoder would dissipate less than 6.1 nJ per decoded information bit.

ACKNOWLEDGEMENTS

The authors would like to thank the Alberta Ingenuity Fund, NSERC and iCORE for their support. We also thank the Canadian Microsystems Corporation for their VLSI design infrastructure and chip fabrication grants.

REFERENCES