A Compact 1.1-Gb/s Encoder and a Memory-Based 600-Mb/s Decoder for LDPC Convolutional Codes

Tyler L. Brandon, John C. Koob, *Member, IEEE*, Leendert van den Berg, *Member, IEEE*, Zhengang Chen, *Student Member, IEEE*, Amirhossein Alimohammad, *Member, IEEE*, Ramkrishna Swamy, Jason Klaus, *Member, IEEE*, Stephen Bates, *Senior Member, IEEE*, Vincent C. Gaudet, *Member, IEEE*, Bruce F. Cockburn, *Member, IEEE*, and Duncan G. Elliott, *Member, IEEE*

***Abstract*—We present a rate-1/2 (128,3,6) LDPC convolutional code encoder and decoder that we implemented in a 90-nm CMOS process. The 1.1-Gb/s encoder is a compact, low-power implementation that includes one-hot encoding for phase generation and built-in termination. The decoder design uses a memory-based interface with a minimum number of memory banks to deliver an information throughput of 1 b per clock cycle. The decoder shares one controller among a pipeline of decoder processors. The decoder dissipates 0.61 nJ of energy per decoded information bit at an SNR of 2 dB and a decoded throughput of 600 Mb/s. On-chip test circuitry permits accurate power measurements to be made at selectable SNR settings.**

***Index Terms*—CMOS integrated circuits, convolutional codes, forward error correction, iterative decoding, low-density parity-check (LDPC) codes.**

I. Introduction

It has been known for some time that low-density parity-check (LDPC) codes have excellent error-correcting properties [1], but it has only recently become practical for LDPC codes to reach their full potential due to their large computational requirements [2]. With a fixed-sized parity-check matrix, LDPC block codes (LDPC-BCs) have been the most widely studied LDPC variant to date. However, studies have shown that LDPC convolutional codes (LDPC-CCs), with a variable-length and potentially infinite parity-check matrix, also have capacity-approaching performance [3], [4]. LDPC-CCs may be more suitable than LDPC-BCs for applications where the data size is unbounded, such as streaming video, or variable-length, such as packet-switching networks [5]. As well, LDPC-CCs enable low-power encoder implementations with low hardware cost, which are suitable for distributed sensor networks. LDPC-CCs can achieve comparable performance to LDPC-BCs even though the convolutional constraint length may be much smaller than the block length [5]. Thus, an LDPC-CC decoder potentially uses less memory per iteration than a comparable LDPC-BC decoder.

Due to cost considerations and routing congestion, researchers have been investigating improved methods to schedule LDPC decoding. LDPC-BC decoders can be fully parallelized by physically implementing all variable and check-nodes to allow concurrent operations [6]–[8]. Other approaches include partially parallel decoders [9] and layered decoding [10]. Results have been reported for FPGA and ASIC LDPC-BC decoder implementations [11]–[13]. Recently, a highly parallel decoder architecture was proposed for time-invariant LDPC-CCs, achieving gigabit-per-second throughputs [14], [15].

In addition to LDPC decoders, researchers have recently published a number of implementations of LDPC block code encoders. For example, an encoder implemented in a 0.25-μm process achieved 860-Mb/s data throughput for a rate-7/8 code [16]. As well, a 78-Mb/s pipelined encoder was implemented on reconfigurable hardware [17].

We investigated the advantages of LDPC-CC encoders and decoders in previous implementations on FPGAs and ASICs. In [18] and [19], both memory-based and register-based architectures were proposed for LDPC-CC decoders. In [20], an LDPC-CC encoder and decoder chip was demonstrated for the first time, achieving a decoding throughput of 175 Mb/s. In [21], a serial LDPC-CC decoder architecture was proposed, which reduces the hardware cost for large codes but also lowers the throughput.

In this paper, we present a 600-Mb/s rate-1/2 (128,3,6) LDPC-CC implementation, which was summarized in [22], with a redesigned encoder and decoder that improve on our earlier work [20]. Our encoder features a compact built-in termination mechanism, reduced power consumption, and better performance compared with the previous encoder implementation. We will show how our new LDPC-CC decoder employs a pipelined architecture with identical processors concatenated together, which reduces routing congestion. The encoder and decoder architectures were implemented in a 90-nm CMOS test chip along with an integrated white Gaussian noise channel model and an architecture that permits power and performance measurements of individual system modules.

This paper is organized as follows. An overview of LDPC-CCs is provided in Section II. In Section III, the architectures of the encoder and decoder are discussed in

Manuscript received September 23, 2008; revised January 08, 2009. First published March 04, 2009; current version published May 20, 2009. This work was supported in part by the NSERC, the Alberta Ingenuity Fund, iCORE and CMC Microsystems. This paper was presented in part at the IEEE International Symposium on Circuits and Systems, Seattle, WA, May 2008. This paper was recommended by Associate Editor Y. Ismail.

T. L. Brandon, J. C. Koob, V. C. Gaudet, B. F. Cockburn, and D. G. Elliott are with the Department of Electrical and Computer Engineering, University of Alberta, Edmonton, AB, T6G 2V4 Canada (e-mail: jkoob@ece.ualberta.ca).

L. van den Berg and A. Alimohammad are with Ukalta Engineering Corporation, Edmonton, AB, T5J 4P6 Canada.

R. Swamy is with Aptina Imaging, Inc., San Jose, CA 95134 USA.

Z. Chen and S. Bates are with Raithlin Semiconductors, Inc., Canmore, AB, T1W 1A1 Canada.

J. Klaus is with Google, Inc., San Jose, CA 94043 USA.

Digital Object Identifier 10.1109/TCSI.2009.2016592

detail. Section IV summarizes the architecture of the test chip that enables on-chip at-speed 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 in 1999 [23]. LDPC-CCs and LDPC-BCs are both linear codes that generate code bits based on parity-check operations [24], and they both have capacity-approaching bit error rate (BER) performance [23], [25]. LDPC-CCs are codes with memory. Any given code bit $v(t)$ is generated using present and previous information bits $u(t)$ and previously generated code bits. An LDPC-CC can also encode frames of variable sizes [23], [25].

An LDPC-CC is defined as the null space of a parity-check matrix, which is the same way an LDPC-BC is defined. If we let the parity-check matrix be $\mathbf{H}$, then all valid codewords $\mathbf{v}$ satisfy $\mathbf{vH}^T = \mathbf{0}$. For an LDPC-CC of rate b/c ($b < c$), the parity-check matrix can be written as (1), shown at the bottom of the page, where $\mathbf{H}_i(t)$ ($i = 0, 1, \ldots, m_s$) is a submatrix of size $(c-b) \times c$. The parameter m_s in (1) will be called the *code memory* in this paper, in contrast to the memory definition used in circuits. The constraint length of the rate-b/c LDPC-CC is defined as $\nu = c \cdot (m_s + 1)$. For regular LDPC-CCs, both the column weight (K) and the row weight (J) of $\mathbf{H}^T$ are constant. We will call such a code an (m_s, J, K) LDPC-CC.

An LDPC-CC is time-invariant if the parity-check constraints defined by each row of $\mathbf{H}$ are fixed. Otherwise, the code is a time-varying LDPC-CC. The design in this paper targets periodically time-varying LDPC-CCs for stronger code performance. Let the period be T_s, then $\mathbf{H}_i(t) = \mathbf{H}_i(t + T_s)$ for all i and t. Due to the periodicity, a phase parameter ϕ ($\phi \in [0, T_s)$) is used to specify the parity-check constraint at each time instant, for both encoding and decoding. For any time t ($t \in [0, \infty)$), we have $\phi = t \bmod T_s$.

The parity-check matrix shown in (1) is infinite. An LDPC-CC, when applied to a finite-length frame, can be considered a special case of LDPC-BCs with a band-diagonal parity-check matrix. Note that the parity-check matrix is also lower triangular. Therefore, the LDPC-CC can be directly encoded from $\mathbf{H}$, given that the submatrix $\mathbf{H}_0(t)$ is always full rank [23]. On the other hand, the encoded frame needs to be properly terminated to ensure good error correction performance toward the end of the frame [26]. Termination is achieved by appending data-dependent tail bits. More information on LDPC-CC encoders and decoders appears in [20], [23].

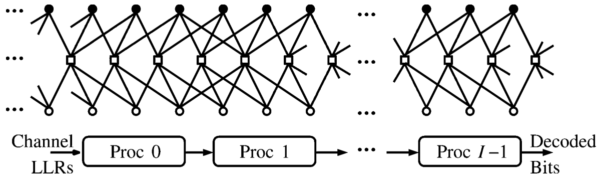


Fig. 1. Tanner graph of an LDPC-CC with sliding-window decoding.

For LDPC-CC decoding, the underlying Tanner graph can be infinite, as shown in Fig. 1. Because the code memory is finite, the neighboring variable nodes of each check node are located within a finite time range determined by m_s. Therefore, a sliding window can be applied upon the graph to perform appropriate node updates. The size of the sliding window is determined by ν. Furthermore, multiple sliding windows can be used to perform simultaneous decoding on different segments of the graph. In the implementation, these sliding windows are pipelined processors, as shown in Fig. 1. The processors are typically composed of variable/check nodes as well as memory for storing messages and channel log-likelihood ratios (LLRs).

This contribution describes high-throughput encoder and decoder architectures that have a number of improvements on our previous work [20]. The new encoder uses techniques, such as one-hot encoding logic, to reduce power consumption and increase encoding performance. Although both implementations use registers for storage, the redesigned decoder accesses banks of registers with a memory-based interface for improved performance. Another departure from the previous architecture is the phase alignment of the decoder processors.

III. Encoder and Decoder Architecture

A. Encoder

Our discussion of LDPC-CC design first examines a compact encoder with built-in all-phase termination. The implementation of an LDPC-CC encoder is typically much simpler than an LDPC-BC encoder with similar capabilities, even though both types of encoders add redundancy to a stream of information bits. In the latest implementation, the encoder design has been improved compared to the previous implementation presented in [20]. This section will explain how the encoder was redesigned to consume less energy per bit while operating at higher frequencies of up to 1.1 GHz.

The structure of an LDPC-CC encoder consists of phase control, registers for information bits and code bits, and logic for code bit generation (see Fig. 2). The phase signals are created by

$$\mathbf{H} = \begin{bmatrix} \ddots & & \ddots & & & & \\ & \mathbf{H}_{m_s}(t) & \ldots & \mathbf{H}_0(t) & & & \\ & & \ddots & \vdots & \ddots & & \\ & & & \mathbf{H}_{m_s}(t+m_s) & \ldots & \mathbf{H}_0(t+m_s) & \\ & & & & \ddots & \vdots & \ddots \end{bmatrix} \tag{1}$$

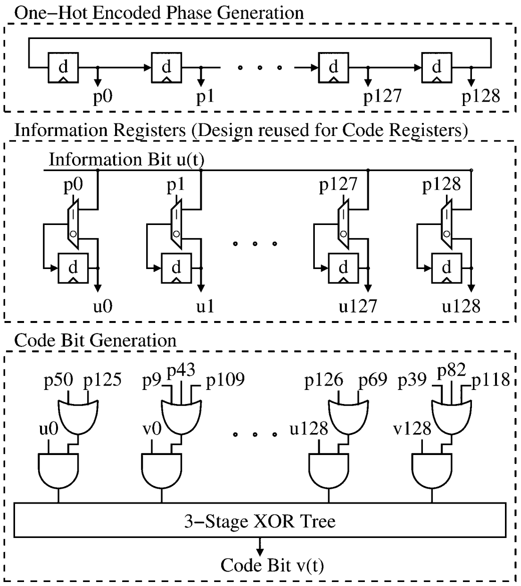


Fig. 2. Architecture of the 1.1-GHz encoder [22].

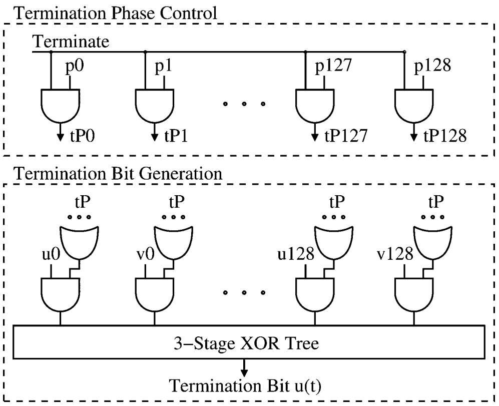


Fig. 3. Termination mechanism for the encoder [22].

one-hot encoding logic. This approach minimizes the fraction of encoder logic that is active in any given cycle, which reduces the power consumption and permits a higher frequency clock. As shown in Fig. 2, the one-hot encoded phase coordinates the time when 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 time-dependent set of previous code and information bits which are selected by the phase $p(t)$. The three-stage XOR tree in Fig. 2 is pipelined to increase the throughput.

Unlike block codes, convolutional codes require a termination scheme to ensure that the trailing information bits at the end of a transmission are fully protected. Truncating a data stream without termination results in loss of BER performance for those bits [26]. Termination is accomplished by returning the encoder 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 readily generate zeros at its input to complete the decoding process. Unfortunately, built-in termination introduces encoder area overhead, which is quantified in Section V-A.

The latest LDPC-CC encoder implementation utilizes the all-phase termination scheme described in [26]. Our design of the termination circuity is shown in Fig. 3. The value of the termination bit depends on the current state of the encoder. The termination bit is generated using the phase bits in a manner inspired by the code bit generation technique described above. 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. The encoder will reach the all-zero state after approximately M cycles, where M is the encoder memory size. At this point, the termination bit will remain at ’0’. The termination circuitry is inactive for most of the time, so its contribution to the overall encoder power consumption is primarily leakage power.

B. Decoder

An LDPC-CC decoder is significantly more complex than its corresponding encoder, but several design techniques can be used to simplify the decoder’s larger implementation. For instance, the decoder can be divided into a number of identical processors that are concatenated together. While this helps reduce routing congestion, opportunities still exist for eliminating replicated circuitry. For example, each processor had its own controller in a previous LDPC-CC implementation [19], [20]. Instead, one controller is shared among an arbitrary number $I \geq 1$ of processors in the redesigned decoder. Since each processor performs the same set of operations as defined by the code, it is possible to share one set of control circuitry. Data and control signals are registered between decoder processors to create pipelined decoding logic.

1) Decoder Data Path: The pipelined decoder processor data path manipulates LLRs, which are represented as 8-b words in Fig. 4. For the (128,3,6) code, a v-node consists of two variable-nodes with a degree of three, while a c-node is a single check-node with a degree of six. A given LLR can follow different paths through a processor. For example, if we assume that an LLR is read from a memory bank, it is registered at the output of the memory as well as at the input and the output of the switch matrix. If this LLR subsequently proceeds through the check-node, it is registered at the output. The LLR is registered two more times as it enters and leaves the switch matrix. The memory write itself counts as the seventh register in the pipeline. The critical path of the decoder starts at the registered output of the c-node, passes through the switch-matrix and ends at the registered output of the switch-matrix.

Our LDPC-CC implementation used 8-b LLRs, but equivalent BER performance would be achievable with 6-b LLRs. The LDPC-CC design was to support adjustable LLR widths that range from 4 to 8 b. However, this feature did not get implemented due to time constraints, so the current implementation only supports 8-b LLRs. A slightly shorter LLR width (such as 6 b) would be preferable because it would significantly reduce decoder area.

2) Decoder Control Path: As shown in Fig. 4, the decoder controller consists of memory controllers and a switch matrix

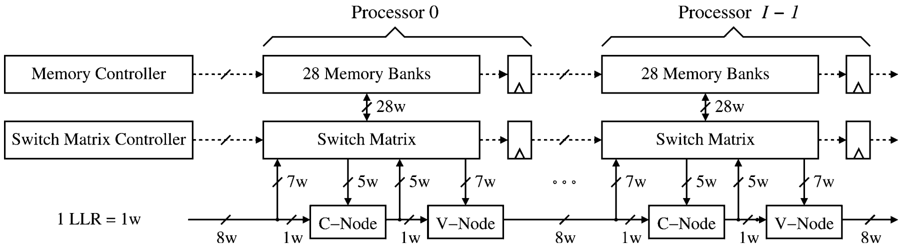


Fig. 4. Decoder controller and the decoder processor data path.

controller. In a given processor, the set of control signals is registered for one cycle before being passed to the adjacent processor. In 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 LLRs between the memory banks, one c-node, one v-node and the processor's inputs and outputs.

3) Decoder-Memory Interface: One goal of the new LDPC-CC design was to implement the convolutional code with a minimum number of memory banks while preserving an information throughput of 1 b per cycle. Only single-port memories were used since dual-port memories have approximately 1.5 times greater area as well as higher power consumption. 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 accesses 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 and check-node operations. In our case, the v-node and c-nodes receive two of the LLR inputs from other nodes and twelve other LLR inputs from the memory banks.

It was important to consider the constraints that are required to sort the LLRs during the design of the memory-decoder interface. The presence of seven internal pipeline stages in the data path increases the complexity of this problem. A graph-coloring algorithm was developed to find the minimum number of memory banks. As shown in Fig. 4, the algorithm indicated that 28 memory banks are required even though only 24 reads and writes occur in each cycle. The algorithm was also able to sort the LLRs so as to balance the number of LLRs in each memory bank.

4) Power Consumption Issues: A memory-based decoder offers a number of opportunities for reducing power consumption. The decoder was originally designed to function with custom SRAMs. However, since the SRAM blocks were unproven in silicon, registers were instantiated instead to ensure reliability. Ultimately, SRAM is preferable over registers because SRAM can reduce power consumption and 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. In the LDPC-CC decoder, the control of a memory bank is independent of the other memory banks and the rest of the system because LLRs are allocated to specific memory banks. Since the individual memory controller subcircuits are synchronized, each subcircuit can operate without communicating with any other subcircuit. 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.

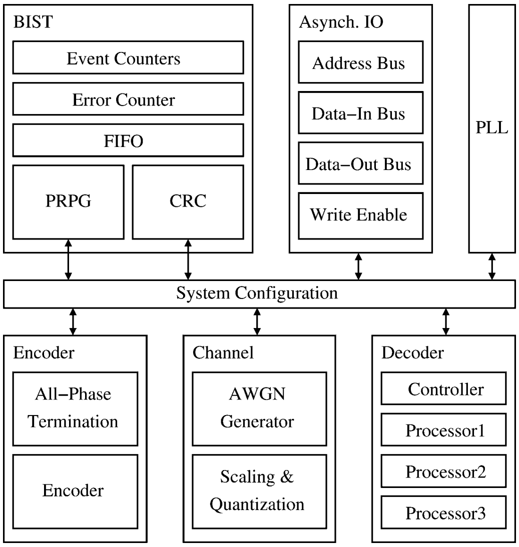


Fig. 5. LDPC-CC test chip block diagram [22].

IV. Test Chip Architecture

The architecture of the 90-nm LDPC-CC test chip is shown in Fig. 5. In addition to the encoder and decoder, the system modules include a built-in self-test (BIST) block, a phase-locked loop (PLL), a noise generator for the additive white Gaussian noise (AWGN) channel model, an asynchronous input/output (IO) interface, and a system configuration module. Here, we describe each module and explain how they help to facilitate testing and characterization at high operating frequencies.

A. BIST Module

The BIST module comprises several components that permit functional testing of the LDPC-CC test chip at operational speeds. The BIST module includes an unconventional pseudo-random pattern generator (PRPG), a cyclic-redundancy-checker (CRC), a variable-length first-in first-out (FIFO) buffer, and a counter-based controller.

1) PRPG for High-Speed Testing: The PRPG and CRC modules enable data to be generated and compacted on-chip, thus permitting high-speed on-chip operation without the need to interface to costly high-speed off-chip test equipment. Instead of a conventional linear feedback shift register (LFSR), a custom combined Tausworthe PRPG was used to generate billions of

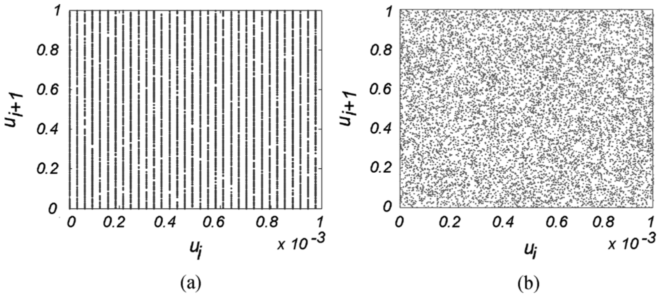


Fig. 6. Randomness of (a) an LFSR and (b) a Tausworthe generator [27].

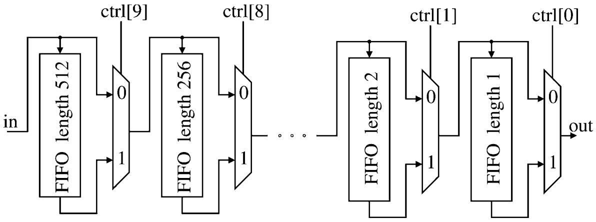


Fig. 7. Variable-length FIFO.

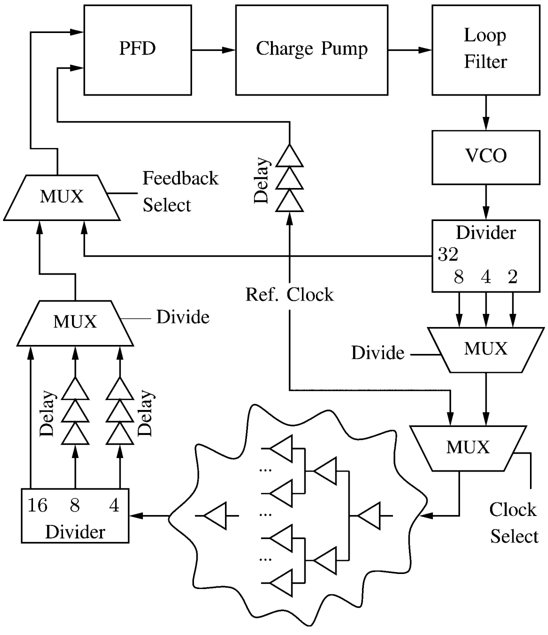


Fig. 8. PLL block diagram.

pseudo-random vectors for characterizing the various on-chip modules. Fig. 6(a) plots the two-dimensional (2-D) distribution of 10^7 pseudorandom number (PN) pairs (u_i, u_{i+1}) generated using a conventional 52-b LFSR PN generator with the primitive characteristic polynomial $p(x) = x^{52} + x^3 + 1$ and a period of 2^{52}. An undesirable lattice structure is clearly visible in Fig. 6(a). Without good randomness properties, a set of generated vectors might never excite certain hard-to-detect faults [28]. To reduce the inherent regular structure of linear PN generators without sacrificing the pseudorandom generation rate, one effective solution is to use combined linear PN generators [29]. A properly designed combined PN generator has better statistical properties than any of its component generators [30]. Thus, we implemented a PRPG with three component Tausworthe generators [31]. Our combined Tausworthe generator improves the randomness properties of generated PN pairs compared to that of a single LFSR, as shown visually by the 2-D distribution in Fig. 6(b) [27], [32].

2) CRC for Response Compaction: To verify the operation of individual modules at high speed, the CRC can generate a 32-b signature (hash) of the consecutive outputs of any module within the chip. The CRC uses the same polynomial as the IEEE 802.3 standard. The CRC can be programmed, via the control registers, to stop after a fixed number of cycles. Once stopped, the CRC will hold its value until reset. We can compare the signature values for a given module from simulation with those from the actual chip to verify that the module is behaving as expected.

3) Variable-Length FIFO: The BIST module includes a programmable variable-length FIFO capable of buffering 1-b inputs for 0 to 1023 cycles. The FIFO stores values from the BIST's PRPG so that they can be compared with the output of the decoder to calculate the BER at a later point in time. The variable-length FIFO is created from 10 FIFOs of length 1, 2, 4, 8, 16, 32, 64, 128, 256 and 512. A 10-bit control register acts as a bit-mask for the FIFOs. This bit-mask determines which FIFOs in the FIFO chain are enabled or bypassed. Fig. 7 illustrates this configuration.

4) Counter-Based Controller: The high degree of programmability in our BIST design is achieved by using counters rather than state machines. In the LDPC-CC chip, there are a number of modules that require different initialization sequences. For example, the encoder might need to be started after ten cycles, the noise generator after one cycle and the decoder after 22 cycles. All of these events are programmable via control registers in the system configuration module. Module reset counters determine when LDPC-CC module reset signals are de-asserted. Counters determine when control signals are asserted for each of the modules. Stop events send control signals to individual components after a specified number of cycles. The ability for tests to run for long periods of time is important for power measurements because our ammeter records measurements at 150-ms intervals. That is, the counters support up to $2^{64} - 1$ cycles, which is more than sufficient for any characterization measurement.

B. Integrated PLL

Our 90-nm LDPC-CC system was designed to operate at 600 MHz, but separate performance tests were needed to confirm that certain modules could function at clock speeds exceeding 1 GHz. The maximum external clock frequency is only 100 MHz due to the frequency limitations of the IO pad circuitry and the available digital test equipment. For economical high-speed testing, high-frequency clocks should be generated on chip. Thus, a custom clock-synthesis PLL was designed and integrated as a system module. The PLL was designed for testability by providing the ability to clock the LDPC-CC core at a wide range of frequencies.

The custom PLL design is shown in Fig. 8. The voltage-controlled oscillator (VCO) uses a five-inverter current-starved ring oscillator with a frequency range from 800 MHz to 5 GHz.

Using a divider, the VCO frequency can be reduced by factors of 2, 4, or 8, producing a clock frequency range from 100 MHz to 2.5 GHz. In order for the chip to operate at frequencies below 100 MHz, the user can select between the PLL-produced clock or the external reference clock signal and use either of these signals to feed the clock tree network. The output of the clock tree network is then divided down further to 1/32nd of the VCO frequency and phase-aligned with the incoming reference clock. Buffer elements are inserted to equalize the delay through the divide paths and to match the latency between the reference clock and feedback paths. The delay matching ensures that the clock edges at the leaf nodes of the clock network are aligned with the reference clock. A local feedback path can be used to avoid jitter in the feedback path from the clock tree network.

The phase-frequency detector (PFD) is a latch-based design from [33]. The charge pump is of a differential design with dummy branches to reduce ripple in its output current. The loop filter is a passive second-order Butterworth filter. Sensitive analog circuits, such as the VCO and charge pump, are powered by a separate supply to reduce jitter. These analog circuits and the passive loop filter components are placed in deep N-wells to isolate them from the substrate noise produced by the digital circuits on the chip. The PLL occupies a total area of 26 500 μm^2, with most of this area dedicated to the loop filter components. The PLL has a bandwidth of approximately 3 MHz and consumes less than 1 mW when programmed to produce a 2-GHz output clock.

C. Integrated AWGN Generator

To verify the behavior of very low BER systems, such as LDPC implementations, a Gaussian noise generator (GNG) must produce samples with accurate statistics. These Gaussian noise samples lie at multiple standard deviations away from the mean (i.e., at the tails of the distribution). The first challenge is that the distribution of generated samples must be as close as possible to the theoretical Gaussian probability density function (PDF) as any deviation from the ideal PDF can degrade test results. In addition, these noise samples should be uncorrelated. The adverse impact of an inaccurate GNG on performance evaluation is presented in [27]. The second challenge is to generate Gaussian samples as fast as possible to reduce the time required for the desired BER calculations and to keep pace with the rest of the system. Thus, to eliminate the need for an external and expensive noise generator, we implemented the compact and accurate white Gaussian noise generator described in [27]. Our on-chip GNG design is parameterizable at the register transfer level (RTL) and it is possible to select a tail accuracy for the AWGN without degrading the output sample rate.

D. Asynchronous IO Interface

The external IO interface of the LDPC-CC chip is accessed in the manner of an asynchronous memory interface. This approach simplifies testing because the external and internal clock domains do not need to be synchronized. As shown in Fig. 5, this interface consists of a 14-b address bus, a 16-b data input bus and a 16-b data output bus. The *write-enable* signal controls the passage of data into the chip between the internal and external clock domains. Invalid data on the input databus and address bus has no effect when the *write-enable* signal is not asserted. To write data to the chip, this signal is asserted after the address bus and input databus have stabilized. To acquire data from the chip, the output databus values are selected by the address.

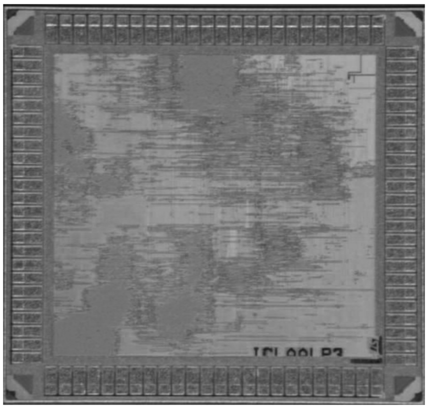


Fig. 9. LDPC-CC packaged die photograph with a 2-mm^2 core [22].

E. System Configuration Module

The system configuration module shown in Fig. 5 contains control registers that are used to configure all aspects of the operation of the LDPC-CC design. This includes the flow of data between modules, module initialization, module configuration and timing events. All modules are programmable via the IO interface.

The system configuration module improves system testability because it provides a configurable data path. It is customary to design communications systems with a data path that connects the modules together in a fixed sequence. That is, data would ordinarily progress from one end of a chain of modules to the other. However, for maximum flexibility during testing and characterization, the system configuration module has direct control over the data path so that it can be configured to interconnect any of the modules. Therefore, it can be used as a chip-level test bus. The data path passes from each module into and out of the system configuration module like spokes on a wheel. With the aid of the control registers, data can be directed from a given module to any other module in the system, including the input and output data buses in the asynchronous IO module. With this approach, a given module can be tested independently of the other modules. For example, the encoder can be tested if the output of the PRPG is attached to the input of the encoder and the output of the encoder is connected to the output databus. Alternatively, the input databus can be connected to the encoder, and the encoder's output would be directed to the CRC unit in the BIST module.

V. Test Results

A. Test Chip Organization

We tested the 90-nm LDPC-CC test chip shown in Fig. 9. The silicon areas of our implemented system modules are shown in Table I. The area of the first three testing modules and the AWGN generator are 6.9% and 14.6% of the total core silicon area, respectively. The encoder and decoder occupy the rest

TABLE I
SYSTEM MODULE SILICON AREA

System Module	Area (μm^2)	Percentage
System Configuration	59,185	3.0
BIST	52,233	2.6
PLL	26,500	1.3
AWGN Generator	292,362	14.6
Encoder	62,304	3.1
Decoder	1,507,416	75.4

TABLE II
POWER MEASUREMENT RESULTS

Chip Module	Measured[1] Power (mW)	Allocated[2] Power (mW)	Net[3] Power (mW)
BIST and Leakage	33.2	–	33.2
Encoder	45.3	33.2	12.1
Noise Generator	163.1	33.2	129.9
Decoder	543.9	175.2	368.7

[1] The measured power consumption of one or more active modules
[2] Power already allocated for one or more other active modules
[3] The calculated power consumption of a system module

of the 2-mm^2 core chip area. The percentage overhead of the testing modules would decrease if additional decoder processors were used in the implementation of the decoder. Although it is not listed separately in Table I, the termination circuitry almost quadruples the area of the encoder. Note that the three-processor decoder occupies three-quarters of the core area, making it a candidate for further optimizations.

B. Test Setup

Our test system consists of an HP81200 digital integrated circuit tester and a programmable Agilent E3647A power supply. While the tester has limited IO performance, signal bandwidth is further limited by the design-under-test (DUT) board design and the 100-MHz IO drivers in the test chip. However, our LDPC-CC modules were designed to run at significantly higher speeds. For example, the encoder module can operate independently at 1.1 GHz and the maximum core frequency of the entire system is 600 MHz. As discussed in Section IV-B, the on-chip PLL is available to produce clock frequencies that exceed the capabilities of the IO drivers.

C. General Test Chip Measurements

The power consumption of test chip modules can be determined from the current drawn by a module or a set of modules. These current measurements must be made by exercising the encoder, channel and decoder with over 3.2 billion vectors at various frequencies of operation. Billions of vectors are necessary to provide sufficient time to make current measurements at 150-ms intervals. Unless otherwise indicated, the power measurements were made for a signal-to-noise ratio (SNR or E_b/N_0) of 2 dB and a core clock frequency of 600 MHz.

As discussed in Section IV, the power of individual modules can be measured independently. Initially, the power of the BIST, including 9.3 mW due to chip leakage, was measured. The power of the encoder can be derived by enabling the encoder module as well as the BIST, 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 BIST, the encoder, the AWGN generator and the decoder for 600-MHz operation. 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.

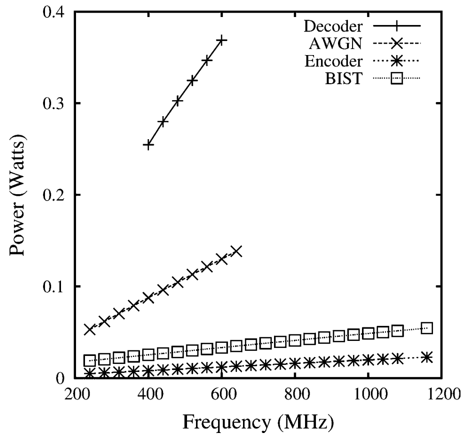


Fig. 10. Module power consumption versus frequency for a 1-V supply.

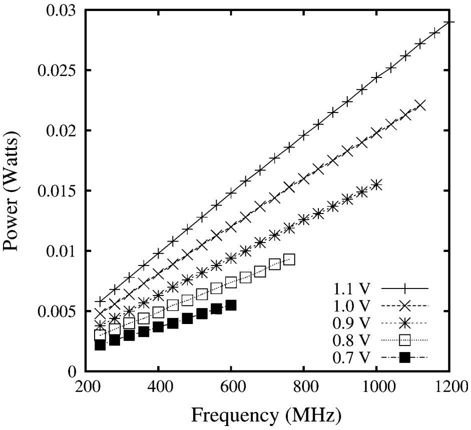


Fig. 11. Encoder module power consumption versus frequency.

The architecture of the LDPC-CC test chip offers the additional advantage that both the power and the performance of individual modules can be measured independently, as shown in Fig. 10. For these results, the power consumption of different chip modules was calculated from current measurements at different frequencies. Some chip modules, such as the AWGN generator and the decoder, have a narrow frequency of operation. Other modules, such as the encoder and BIST, can function at frequencies as high as 1.1 GHz.

D. Encoder Measurements

The power consumption of the encoder for various core voltages and operating frequencies is shown in Fig. 11. Each data point on this graph is calculated from power measurements in the manner described in Section V-C. In Fig. 11, the maximum operating frequency of the encoder is 1.1 GHz for a core supply voltage of 1.0 V. When the encoder is operated at 600 MHz, approximately 12 mW of power is consumed, which agrees with

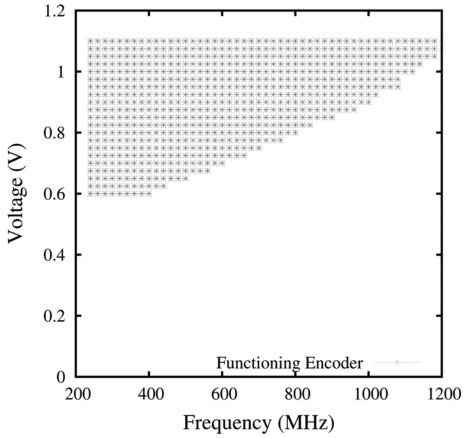


Fig. 12. Encoder module Shmoo plot.

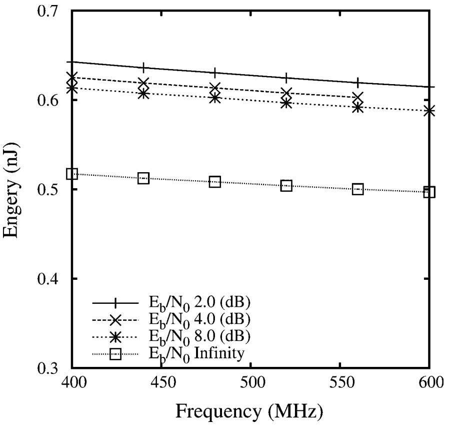


Fig. 14. Measured energy per decoded bit using three processors [22].

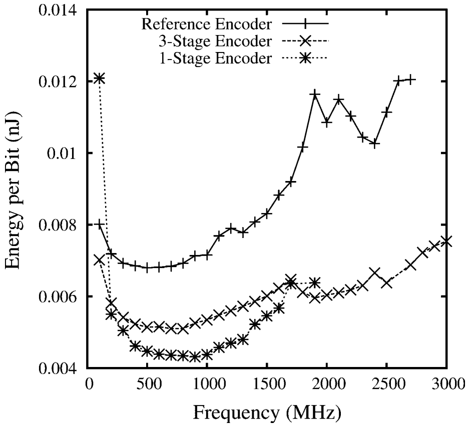


Fig. 13. Encoder energy per bit from synthesis data compared with [20].

the result reported in Table II. The encoder can operate at voltages lower than 1.0 V for additional power savings, but this reduces the range of operating frequencies where the encoder produces correct output.

The Shmoo plot in Fig. 12 identifies the ranges of voltages and frequencies for which the encoder correctly encodes the pseudorandom samples. These results were obtained from the encoder operating without the termination mechanism.

It is useful to investigate the power consumption of the encoder in terms of energy per bit and compare this with our previous LDPC-CC encoder design [20]. Fig. 13 shows the energy per bit using data obtained from prelayout synthesis reports. Fig. 13 also shows the maximum predicted operating frequency of each encoder architecture. The encoders were synthesized without termination, and the redesigned encoder design was synthesized with both one and three stages of pipelining. Using techniques discussed in Section III-A, the redesigned encoder in Fig. 13 consumes 40% less energy per bit at 1.1 GHz than the reference encoder in [20].

E. Decoder Measurements

Fig. 14 shows the measured energy per decoded information bit versus frequency for various values of E_b/N_0. Changing E_b/N_0 affects the power consumption by skewing the distribution of probabilities of a "1" or a "0." The decoder has a lower activity factor when E_b/N_0 is high because fewer channel samples are in error. Conversely, the decoder works harder when E_b/N_0 is low because the distribution of channel samples widens and more of the channel sample magnitudes deviate further from the nominal values. In Fig. 14, the lowest energy per decoded bit occurs for infinite E_b/N_0, 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 is apparently due to a 25% increase in decoder activity versus the activity associated with an E_b/N_0 of infinity.

Dynamic power is linearly related to frequency and the energy per bit is a direct measure of the amount of computation done to decode an information bit. However, in Fig. 14, it is interesting to observe that the energy per bit decreases slightly with frequency. As the operating frequency is increased, we suspect that the voltage across some standard cell supply rails droops slightly due to IR drops. This effect in turn reduces the current, which also reduces the energy per bit. This observation suggests that there are further opportunities for power reduction by reducing the supply voltage.

Note that Fig. 14 shows a decrease in energy with an increase in E_b/N_0. The energy for three processors operating at 600 MHz with an E_b/N_0 of 2 dB 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 in the average E_b/N_0 as LLRs pass through the 30-processor decoder.

Fig. 15 shows the decoder power versus frequency for various core voltages. The top curves in Figs. 10 and 15 represent the same power measurements of the decoder for a 1-V supply. The minimum operating frequency of the PLL at a frequency multiple of 8 is 240 MHz, which is why no data is shown for the decoder lower than 240 MHz for the other three supply voltages. At a supply voltage of 1 V, it is interesting to note that the decoder has a minimum operating frequency of 400 MHz rather than 240 MHz. We hypothesize that this is due to hold time violations in the decoder at frequencies below 400 MHz, during which the power rail is close to the 1-V supply voltage. For frequencies of 400 MHz and higher, we believe that the internal

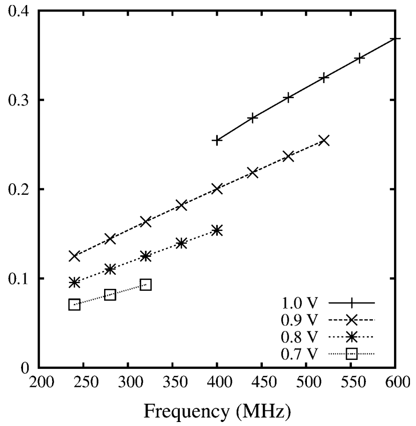


Fig. 15. Decoder module power consumption versus frequency.

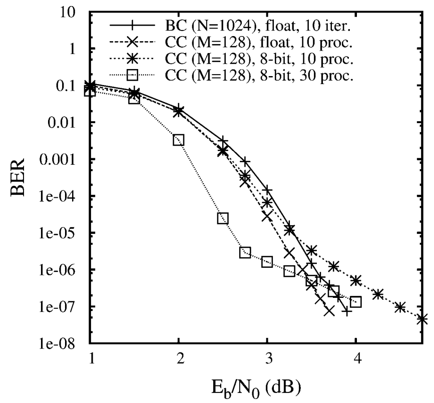


Fig. 16. Decoder BER versus E_b/N_0[22].

core voltage drops slightly below 1 V due to IR drops, which causes the hold time violations to disappear. This hypothesis is supported by observed correct operation at lower frequencies at a core voltage of 0.9 V.

Although our LDPC-CC implementation was allotted limited silicon area, the LDPC-CC decoder design can be scaled up for further performance measurements by simulation. If approximately 50 mm^2 is available, a competitive implementation of a 30-processor LDPC-CC decoder is possible. Fig. 16 shows the BER versus E_b/N_0 for 10 and 30 processors using 8-b LLRs. Two extra curves, for a conventional 10-iteration LDPC-BC decoder [23] and a 10-processor LDPC-CC decoder [20], are included in this figure for comparison. The (128,3,6) LDPC-CC starts to show an error floor at the BER of approximately 1e-06. An error floor also appears in the LDPC-BC begins at a BER of 1e-06, but it is not as noticeable as the error floor for the LDPC-CC decoder.

F. Decoder Throughput and Power Comparison

While different LDPC-CC decoders might have similar architectures, their implementations can differ dramatically. To create normalized comparisons, various design parameters will need to be scaled in an attempt to eliminate any dependency on the implementation as much as possible. For example, parameters that might require scaling include process technology, message width and operating frequency. Note that the scaling factors are first-order estimates based on device scaling rather than actual measured characterizations.

TABLE III
LDPC-CC VS LDPC-BC THROUGHPUT COMPARISON

Category	LDPC-CC (This Work)	LDPC-BC from [13]
Code Size	constraint length = 258	block length = 1296
Code Rate	1/2	1/2
Parallelism	1	27
Max. Iterations	25 serial processors	25 iterations
CN Algorithm	min-sum (*simple*)	3-min (*complex*)
Message Width	8 bits (*scaled to 6*)	6 bits
Technology	90 nm	65 nm (*scaled to 90*)
Clock Frequency	600 MHz	289 MHz (*scaled*)
Silicon Area	9.375 mm^2	1.04 mm^2 (*est.*)
Throughput	**600 Mbps**	**39 Mbps** (*scaled*)
Latency	5.75 µs	8.03 µs (*scaled*)
Throughput/Area	**64 Mbps/mm^2**	**38 Mbps/mm^2**

1) Throughput Comparison With an LDPC-BC Decoder: We now compare the throughput of our LDPC-CC decoder with the IEEE 802.11n LDPC-BC decoder implementation in [13]. To form a valid comparison, we need to make some assumptions and scale some parameters, which are summarized in Table III. Each row of this table will be explained in sequence. The implementation in [13] is a multicode decoder, with a code block length of 648/1296/1944 and code rate ranging from 1/2 to 5/6. To compare with our rate-1/2 single-code LDPC-CC decoder with $\nu = 258$, we choose the rate-1/2 N $= 1296$ LDPC-BC decoder because it has the same code rate and the most comparable BER performance among the three block lengths. The LDPC-BC decoder in [13] can support 27 to 81 parallel operations for throughputs ranging from 54 to 281 Mb/s, with 20–25 iterations. In this multi-code decoder, the decoding throughput varies due to factors including code rate, parallelism and number of iterations. Rate-1/2 LDPC-BCs certainly account for the low end of the throughput range. Obviously, the low-end throughput (54 Mb/s) is also achieved with the lowest parallelism (27) and the maximum number of iterations (25). Our LDPC-CC decoder can be considered to function serially (parallelism of 1) if the multiple decoder processors are equated to iterations. Thus, our LDPC-CC decoder can be scaled to use 25 decoding processors to match the 25 iterations required by the decoder in [13]. The LDPC-CC decoder utilized a simple min-sum algorithm, while the LDPC-BC decoder features a more complex 3-min algorithm. To normalize the message width in this throughput comparison, we modified the LDPC-CC decoder simulator by scaling the 8-b LLRs down to 6 b. Thus, the message width scaling factor is $s_w = 6/8$. Similarly, we scaled the LDPC-BC decoder as if it were implemented in a 90-nm process technology rather than the original 65-nm process. Thus, using a generalized scaling factor of $s = 90/65$ [34], we define the scaling factors for clock frequency, $s_f = 1/s$, and area, $s_a = s^2$. The measured operating frequency of the LDPC-CC decoder is 600 MHz, but the measured operating frequency of the LDPC-BC decoder is 400 MHz [13]. In a 90-nm process, the scaled clock frequency of the LDPC-BC decoder is 400 MHz $\times s_f$ = 289 MHz (see Table III).

The silicon area occupied by 25 LDPC-CC decoder processors is $0.5 \times 25 \times s_w = 9.375$ mm^2. A decoder for a

TABLE IV
LDPC-CC VS PARALLEL LDPC-BC COMPARISON

Category	**LDPC-CC (This Work)**	**LDPC-BC from [6]**
Code Size	constraint length = 258	block length = 1024
Code Rate	1/2	1/2
Parallelism	No	Yes
Max. Iterations	30 serial processors	64 iterations
Message Width	8 bits (*scaled to 4*)	4 bits
Technology	90 nm	160 nm (*scaled to 90*)
Clock Frequency	600 MHz	64 MHz (*scaled: 114*)
Throughput	600 Mbps	889 Mbps (*scaled*)
Silicon Area	7.5 mm^2	16.6 mm^2 (*est.*)
Throughput/Area	**80 Mbps/mm^2**	**54 Mbps/mm^2**
Power Dissip.	1845 mW (*scaled*)	276 mW (*scaled*)
Energy/bit	**3.1 nJ/decoded-bit**	**0.3 nJ/decoded-bit**

TABLE V
LDPC-CC ENERGY COMPARISON

Category	**LDPC-CC (This Work)**	**LDPC-CC from [20]**
Code Size	constraint length = 258	constraint length = 258
Code Rate	1/2	1/2
Processors	10	10
CN Algorithm	min-sum	min-sum
Message Width	8 bits (*scaled to 6*)	6 bits
Technology	90 nm	180 nm (*scaled to 90*)
Frequency	600 MHz	350 MHz (*scaled*)
Throughput	600 Mbps	350 Mbps (*scaled*)
Silicon Area	3.75 mm^2	2.40 mm^2 (*scaled*)
Throughput/Area	**160 Mbps/mm^2**	**146 Mbps/mm^2**
Power Dissip.	923 mW (*scaled*)	539 mW (*scaled*)
Energy/bit	**1.5 nJ/decoded-bit**	**1.5 nJ/decoded-bit**

single rate-1/2 N=1296 LDPC-BC with a parallelism of 27 is expected to be smaller than the multicode decoder in [13]. Due to memory/node reuse, the memory area of the multicode decoder is determined by the maximum block length for lowest code rate, and the node area is determined by the maximum parallelism. Thus, the area of the rate-1/2 N = 1296 LDPC-BC decoder is estimated to be $(0.096+0.395) \times (27/81)(\text{nodes}) + 0.467 \times (1296/1944)(\text{memory}) + 0.065(\text{network}) = 0.54$ mm^2. Note that this is a simplified estimation due to incomplete information about the multi-code implementation. Scaled to a 90-nm process, the area is $0.54 \times s_a = 1.04$ mm^2. The throughput of the LDPC-CC decoder is 600 Mb/s, while the LDPC-BC decoder's throughput is $54 \times s_f = 39$ Mb/s. The latency of the 25 LDPC-CC decoder processors is $0.23 \times 25 = 5.75$ μs, while the LDPC-BC decoder in [13] is $5.8/s_f = 8.03$ μs. Finally, the ratios of decoder throughput to silicon area are shown in Table III.

2) Comparison With a Parallel LDPC-BC Decoder: We compare the throughput and energy usage of our serial LDPC-CC with a parallel LDPC-BC from [6] in Table IV. The 8-b LLRs in the LDPC-CC decoder need to be scaled to 4 b to match the decoder in [6], so the message width scaling factor becomes $s_w = 4/8$. We scaled the parallel LDPC-BC decoder as if it were implemented in a 90-nm process rather than the original 160-nm process. Thus, using the factor $s = 160/90$ from generalized scaling theory [34], we redefine the scaling factors for frequency, $s_f = s$, and area, $s_a = 1/s^2$. The maximum operating frequency of the LDPC-CC and LDPC-BC decoders are 600 and 64 MHz, respectively [6]. If fabricated in a 90-nm process, an estimated clock frequency of the LDPC-BC is $64 \times s_f = 114$ MHz, which is summarized in Table IV. Scaling our LDPC-CC decoder to more than 30 decoder processors offers little improvement in BER performance. Thus, the silicon area occupied by 30 LDPC-CC decoder processors is $0.5 \times 30 \times s_w = 7.5$ mm^2. The 52.5-mm^2 LDPC-BC decoder in [6] would occupy $52.5 \times s_a = 16.6$ mm^2 in a 90-nm process. For the 90-nm LDPC-BC decoder, the decoded throughput would be about $500 \times s_f = 889$ Mb/s. Although the LDPC-CC decoder only achieves 600 Mb/s, Table IV shows that it has better throughput per area than the block-code decoder in [6]. The maximum power dissipation of a 30-processor LDPC-CC decoder is $30/3 \times 369 \times s_w = 1845$ mW. The LDPC-BC decoder in [6] consumed 690 mW at 1.5 V. According to [34], the LDPC-BC power consumption would become $690 \times 1.5^2/s^3 = 276$ mW. The scaled energy per decoded information bit in our decoder is higher than that for the decoder from [6] because the power scaling to 30 processors is linear (see Table IV). If an early stopping rule is used for our LDPC-CC decoder, the power consumption would be significantly reduced.

3) Power Comparison With an LDPC-CC Decoder: Finally, we compare our LDPC-CC decoder with the convolutional decoder from [20]. As evident in Table V, the features of the two decoders are very similar, but there still are several implementation differences. To scale the 180-nm decoder from [20] to a 90-nm process, we define the generalized scaling factor as $s = 180/90$. We estimate that the clock frequency would increase by $s_f = s$ to 350 MHz. Thus, the throughput would become 350 Mb/s, which is lower than our 600-Mb/s LDPC-CC. The silicon area occupied by 10 LDPC-CC decoder processors is $0.5 \times 10 \times s_w = 3.75$ mm^2, where s_w was defined in Section V-F1. From [20], the decoder occupies 9.6 mm^2 on a die manufactured using a 180-nm process. This can be normalized by scaling the 180-nm decoder by a factor of $s_a = 1/s^2 = 1/4$. With this scaling factor, Table V shows the area occupied by the decoder from [20] is similar to the area of our latest LDPC-CC decoder. The throughput per area of our LDPC-CC decoder is 160 Mbps/mm^2, which is 10% better than the scaled version of the decoder in [20].

We will now compare the power consumption of the two LDPC-CC decoders. Since a three-processor decoder consumed 369 mW, the maximum power dissipation of a 10-processor LDPC-CC decoder is $10/3 \times 369 \times s_w = 923$ mW. The decoder from [20] consumes 1330 mW in a 180-nm process at 1.8 V. In a 90-nm process, this would scale to $1330 \times 1.8^2/s^3 = 539$ mW (see Table V) [34]. The energy per decoded information bit calculated in [20] was 7.6 nJ/bit at the measured throughput of 175 Mb/s. Since the throughput of a 90-nm decoder could likely increase to approximately 350 Mb/s, the energy per decoded bit becomes 1.5 nJ/bit. A 10-processor LDPC-CC decoder would consume at most 1.5 nJ/bit. Although this upper bound equals the energy required by the decoder from [20], a simple linear scaling of the number of processors ignores possible reductions of switching activity in some processors.

VI. CONCLUSION

We have presented an encoder and decoder architecture for LDPC convolutional codes that targets high-throughput operation. Our test chip design features an architecture that enables power and performance measurements to be made at operating speeds. The low power encoder design consumes 40% less energy per bit than a reference convolutional encoder design. The implemented encoder can function at 1.1 GHz while drawing 22 mW of power. When our three-processor decoder operates at 600 MHz with E_b/N_0 set at 2 dB, it consumes 369 mW of power for a 1.0-V supply. A 30-processor decoder would dissipate less than 6.1 nJ per decoded information bit. We project that this architecture would make LDPC convolutional codes attractive for such applications as streaming video, packet-switching networks, and distributed sensor networks.

References

[1] R. Gallager, "Low-density parity-check codes," *IRE Trans. Inform. Theory*, vol. IT-8, no. 1, pp. 21–28, Jan. 1962.

[2] D. MacKay and R. Neal, "Near Shannon limit performance of low density parity check codes," *Electron. Lett.*, vol. 32, no. 18, pp. 1645–1646, Aug. 1996.

[3] A. E. Pusane, K. S. Zigangirov, and D. J. Costello , Jr, "Construction of irregular LDPC convolutional codes with fast encoding," in *Proc. IEEE Int. Conf. Commun.*, Istanbul, Turkey, Jun. 2006, vol. 3, pp. 1160–1165.

[4] D. J. Costello , Jr, A. E. Pusane, C. R. Jones, and D. Divsalar, "A comparison of ARA- and protograph-based LDPC block and convolutional codes," in *Proc. IEEE Inf. Theory and Applications Workshop*, San Diego, CA, Jan. 2007, pp. 111–119.

[5] Z. Chen, S. Bates, and X. Dong, "Low-density parity-check convolutional codes applied to packet based communication systems," in *Proc. IEEE Global Telecommun. Conf.*, St. Louis, MO, 2005, vol. 3, pp. 1250–1254.

[6] A. Blanksby and C. Howland, "A 690-mW 1-Gb/s 1024-b, rate-1/2 low-density parity-check code decoder," *IEEE J. Solid-State Circuits*, vol. 37, no. 3, pp. 404–412, Mar. 2002.

[7] T. Brandon, R. Hang, G. Block, V. Gaudet, B. F. Cockburn, S. Howard, C. Giasson, K. Boyle, P. Goud, S. S. Zeinoddin, A. Rapley, S. Bates, D. G. Elliott, and C. Schlegel, "A scalable LDPC decoder architecture with bit-serial message exchange," *VLSI J.*, vol. 41, no. 3, pp. 385–398, May 2008.

[8] A. Darabiha, A. C. Carusone, and F. R. Kschischang, "Block-interlaced LDPC decoders with reduced interconnect complexity," *IEEE Trans. Circuits Syst. II, Exp. Briefs*, vol. 55, no. 1, pp. 74–78, Jan. 2008.

[9] H. Zhong and T. Zhang, "Block-LDPC: A practical LDPC coding system," *IEEE Trans. Circuits Syst. I, Reg. Papers*, vol. 52, no. 4, pp. 766–775, Apr. 2005.

[10] M. Mansour and N. Shanbhag, "High-throughput LDPC decoders," *IEEE Trans. Very Large Scale Integr. (VLSI) Syst.*, vol. 11, no. 6, pp. 976–996, Dec/ 2003.

[11] L. Yang, H. Liu, and C.-J. R. Shi, "Code construction and FPGA implementation of a low-error-floor multi-rate low-density parity-check code decoder," *IEEE Trans. Circuits Syst. I, Reg. Papers*, vol. 53, no. 4, pp. 892–904, Apr. 2006.

[12] M. Mansour and N. Shanbhag, "A 640-Mb/s 2048-bit programmable LDPC decoder chip," *IEEE J. Solid-State Circuits*, vol. 41, pp. 684–698, 2006.

[13] T. Brack, M. Alles, T. Lehnigk-Emden, F. Kienle, and N. Wehn, "Low complexity LDPC code decoders for next generation standards," in *Proc. Design, Autom. Test in Europe Conf. (DATE)*, Nice, France, Apr. 2007, pp. 331–336.

[14] E. Matus, M. B. Tavares, M. Bimberg, and G. Fettweis, "Towards a Gbit/s programmable decoder for LDPC convolutional codes," in *Proc. IEEE Int. Symp. Circuits Syst. (ISCAS)*, New Orleans, LA, May 2007, pp. 1657–1660.

[15] M. B. Tavares, E. Matus, S. Kunze, and G. Fettweis, "A dual-core programmable decoder for LDPC convolutional codes," in *Proc. IEEE Int. Symp. Circuits Syst. (ISCAS)*, Seattle, WA, May 2008, pp. 532–535.

[16] L. H. Miles, J. W. Gambles, G. K. Maki, W. E. Ryan, and S. R. Whitaker, "An 860-Mb/s (8158,7136) low-density parity-check encoder," *IEEE J. Solid-State Circuits*, vol. 41, no. 8, pp. 1686–1691, Aug. 2006.

[17] Z. Khan and T. Arsian, "Pipelined implementation of a real time programmable encoder for low density parity check code on a reconfigurable instruction cell architecture," in *Proc. Design, Autom. Test in Europe Conf. (DATE)*, Nice, France, Apr. 2007, pp. 349–354.

[18] S. Bates and G. Block, "A memory-based architecture for low-density parity-check convolutional decoders," in *Proc. IEEE Int. Symp. Circuits Syst. (ISCAS)*, Kobe, Japan, May 2005, vol. 1, pp. 336–339.

[19] R. Swamy, S. Bates, and T. Brandon, "Architectures for ASIC implementations of low-density parity-check convolutional encoders and decoders," in *Proc. IEEE Int. Symp. Circuits Syst. (ISCAS)*, 2005, vol. 5, pp. 4513–4516.

[20] R. Swamy, S. Bates, T. L. Brandon, B. F. Cockburn, D. G. Elliott, J. C. Koob, and Z. Chen, "Design and test of a 175-Mb/s, rate-1/2 (128,3,6) low-density parity-check convolutional code encoder and decoder," *IEEE J. Solid-State Circuits*, vol. 42, no. 10, pp. 2245–2256, Oct. 2007.

[21] S. Bates, Z. Chen, L. Gunthorpe, A. E. Pusane, K. S. Zigangirov, and D. J. Costello , Jr, "A low-cost serial decoder architecture for low-density parity-check convolutional codes," *IEEE Trans. Circuits Syst. I, Reg. Papers*, vol. 55, no. 8, pp. 1967–1976, Aug. 2008.

[22] T. Brandon, J. C. Koob, L. van den Berg, Z. Chen, A. Alimohammad, R. Swamy, J. Klaus, S. Bates, V. C. Gaudet, B. F. Cockburn, and D. G. Elliott, "A 600-Mb/s encoder and decoder for low-density parity-check convolutional codes," in *Proc. IEEE Int. Symp. Circuits Syst. (ISCAS)*, Seattle, WA, May 2008, pp. 3090–3093.

[23] A. Jiménez-Felström and K. S. Zigangirov, "Time-varying periodic convolutional codes with low-density parity-check matrix," *IEEE Trans. Inf. Theory*, vol. 45, no. 5, pp. 2181–2191, Sep. 1999.

[24] C. Schlegel and L. Perez*, Trellis and Turbo Coding*, 1st ed. New York: IEEE Press, 2004.

[25] D. J. Costello, A. E. Pusane, S. Bates, and K. S. Zigangirov, "A comparison between LDPC block and convolutional codes," in *Proc. IEEE Inf. Theory and Applications Workshop*, San Diego, CA, Feb. 2006.

[26] Z. Chen, S. Bates, D. Elliott, and T. Brandon, "Efficient encoding and termination of low-density parity-check convolutional codes," in *Proc. IEEE Global Telecommun. Conf.*, San Francisco, CA, 2006, pp. 1–5.

[27] A. Alimohammad, S. F. Fard, B. F. Cockburn, and C. Schlegel, "A compact and accurate Gaussian variate generator," *IEEE Trans. Very Large-Scale Integr. (VLSI) Syst.*, vol. 16, no. 5, pp. 517–527, May 2008.

[28] P. H. Bardell and W. H. McAnney, "Pseudorandom arrays for built-in tests," *IEEE Trans. Comput.*, vol. C-35, pp. 653–658, 1986.

[29] G. Marsaglia, A. Zaman, and W. W. Tsang, "Towards a universal random number generator," *Stat. Probabil. Lett.*, vol. 8, pp. 35–39, 1990.

[30] D. E. Knuth*, The Art of Computer Programming*. Reading, MA: Addison-Wesley, 1997.

[31] P. L'Ecuyer, "Maximally equidistributed combined Tausworthe generators," *Math. Computation*, vol. 65, no. 213, pp. 203–213, 1996.

[32] A. Alimohammad, S. F. Fard, B. F. Cockburn, and C. Schlegel, "On the efficiency and accuracy of hybrid pseudo-random number generators for FPGA-based simulations," in *Proc. IEEE Int. Symp. on Parallel and Distrib. Process.*, Apr. 2008, pp. 1–8.

[33] M. Mansuri, D. Liu, and C.-K. K. Yang, "Fast frequency acquisition phase-frequency detectors for Gsamples/s phase-locked loops," *IEEE J. Solid-State Circuits*, vol. 37, no. 10, pp. 1331–1334, Oct. 2002.

[34] G. Baccarani, M. R. Wordeman, and R. H. Dennard, "Generalized scaling theory and its application to a 1/4 micrometer MOSFET design," *IEEE Trans. Electron Devices*, vol. ED-31, no. 4, pp. 452–462, Apr. 1984.

Tyler L. Brandon received the B.Sc. and M.Sc. degrees in electrical engineering from the University of Alberta, Edmonton, AB, Canada, in 1999 and 2003, respectively, where he is currently working toward the Ph.D. degree.

His research focuses on the areas of ASIC architectures for logarithmic-based processing, parallel computing architectures and LDPC codes. He holds five patents in the areas of homogeneous three-dimensional integrated circuit design, multilevel hybrid ROM and SRAM-ROM and IC circuit design.

Mr. Brandon was a corecipient of the Best Student Paper Award at the 2008 IEEE International Symposium on Circuits and Systems.

John C. Koob (S'94–M'96–S'01–M'09) received the B.Sc. degree in computer engineering (with distinction) and the M.Sc. degree in electrical and computer engineering from the University of Alberta, Edmonton, AB, Canada, in 1995 and 2004, respectively, where he is currently working toward the Ph.D. degree.

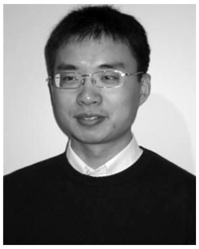
He was with Saville Systems, Edmonton, from 1995 to 1999. He cofounded Ukalta Engineering in 2008. His primary research involves the evaluation of economical file memory for multi-level buffer cache architectures. He holds three U.S. patents.

Mr. Koob is a member of the IEEE Computer Society and the IEEE Solid-State Circuits Society. He is a registered Professional Engineer in the Province of Alberta. He was a recipient of the NSERC PGS Award, the Alberta Ingenuity Studentship Award, and the iCORE Postgraduate Award. He was a corecipient of the Best Student Paper Award at the 2008 IEEE International Symposium on Circuits and Systems.

Leendert van den Berg (S'08–M'09) was born in The Netherlands in 1981. He received the B.Sc. degree in computer engineering and the M.Sc. degree in electrical and computer engineering from the University of Alberta, Edmonton, AB, Canada, in 2004 and 2008, respectively.

He is a cofounder of and currently employed by Ukalta Engineering, Edmonton, a start-up company commercializing wireless channel simulators. His research interests include frequency synthesis architectures, high-voltage circuits, and VLSI implementation aspects of communication systems.

Mr. Van den Berg is registered with the Association of Professional Engineers, Geologists, and Geophysicists of Alberta (APEGGA) as an Engineer-in-Training. He was a corecipient of the Best Student Paper Award at the 2008 IEEE International Symposium on Circuits and Systems.



Zhengang Chen (S'01) received the B.S. and M.S. degrees in electronic engineering from Tsinghua University, Beijing, China, in 1999 and 2002, respectively, and the Ph.D. degree in electrical and computer engineering at University of Alberta, Edmonton, AB, Canada, in 2009.

He is currently with Raithlin Semiconductors, Inc., Canmore, AB, Canada. His research interests include error control coding, communication systems, digital application-specific integrated circuit (ASIC) design, and pattern recognition.

Dr. Chen was a corecipient of the Best Student Paper Award at the 2008 IEEE International Symposium on Circuits and Systems.

Amirhossein Alimohammad (M'03) received the B.Sc. degree from the University of Isfahan, Isfahan, Iran, in 1995, the M.Sc. degree from the University of Tehran, Tehran, Iran, in 1998, and the Ph.D. degree from the University of Alberta, Edmonton, AB, Canada, in 2007.

From 1998 to 2000, he was with Informatics Services Corporation, Tehran, Iran, while he was a Ph.D. student at the Sharif University of Technology, Tehran. In 2001, he was with the Institute of Microelectronics, University of Ulm, Ulm, Germany, Atmel Germany GmbH, and Get2Chip GmbH, Munich, Germany. He was a Postdoctoral Fellow with the Department of Electrical and Computer Engineering, University of Alberta, and he is now with Ukalta Engineering, Edmonton. His research interests include reconfigurable computing, VLSI design and test, wireless communications, and signal processing.

Ramkrishna Swamy received the B.Sc. (Cooperative) degree in computer engineering and the M.Sc. degree in electrical and computer engineering from the University of Alberta, Edmonton, AB, Canada, in 2004 and 2007, respectively.

He was a Semiconductor Product Engineer with Micron Technology, Boise, ID. He is currently with Aptina Imaging, San Jose, CA. He was the primary VLSI designer of the first LDPC-CC encoder and decoder ASIC. His other research interests include circuit design and synthesis, digital ASIC design, and low-power system architectures for high-speed applications.

Mr. Swamy was the recipient of the Best Paper in Session Award at the Semiconductor Research Corporation's (SRC) TECHCON 2005 conference and was a coauthor of paper that received the Best Student Paper Award at the 2008 IEEE International Symposium on Circuits and Systems.

Jason Klaus (M'09) received the B.Sc. degree in computer engineering and the M.Sc. degree in electrical and computer engineering from the University of Alberta, Edmonton, AB, Canada, in 2004 and 2008, respectively.

He was a Hardware Engineer with YottaYotta and a Software Engineer with IBM. He is currently a Hardware Engineer with Google, Mountain View, CA. His current field of interest is Internet routing.

Mr. Klaus was a recipient of the NSERC CGS M Award, the Alberta Ingenuity Studentship Award, and the iCORE Postgraduate Award. He was a corecipient of the Best Student Paper Award at the 2008 IEEE International Symposium on Circuits and Systems.

Stephen Bates (S'94–M'03–***SM'XX***) received the Ph.D. degree from Edinburgh University, Edinburgh, U.K., in 1997.

He then joined Massana, an Irish fabless semiconductor company specializing in high-speed Ethernet transceiver design. In 2003, he joined the Department of Electrical and Computer Engineering, University of Alberta, Edmonton, AB, Canada, as an Assistant Professor. He is currently with Raithlin Semiconductors, Inc., Canmore, AB, Canada. His research interests include circuits and systems for high-speed wireline communication protocols, circuits for advanced forward error correction schemes, and the use of reconfigurable logic to solve problems related to communications and bio-informatics.

Vincent C. Gaudet (S'97–M'03) received the B.Sc. degree in computer engineering from the University of Manitoba, Winnipeg, MB, Canada, in 1995 and the M.A.Sc. and Ph.D. degrees from the University of Toronto, Toronto, ON, Canada, in 1997 and 2003, respectively.

In 2002 he joined the Department of Electrical and Computer Engineering University of Alberta, Edmonton, AB, Canada, where he is currently an Associate Professor. From February to July 2002, he was a Research Associate with the Département Electronique, École Nationale Supérieure des Télécommunications de Bretagne, Brest, France. During 2008–2009, he is on sabbatical leave at Northeastern University, Boston, MA, and at Tohoku University, Sendai, Japan. His research interests focus on information processing microsystems and more specifically on energy-efficient graph-based decoding of error control codes such as Turbo and LDPC codes.

Dr. Gaudet is licensed as a Professional Engineer in the Province of Ontario. He has previously served as the University of Alberta's CMC Faculty Representative and on the NSERC Scholarships and Fellowships Committee 176. He currently serves as Associate Editor for the IEEE Transactions on Circuits and Systems II: Express Briefs, and as Digest Editor for the 2009 IEEE International Solid-State Circuits Conference.

Bruce F. Cockburn (S'86–M'95) received the B.Sc. degree in engineering physics from Queen's University, Kingston, ON, Canada, in 1981 and the M.Math. and Ph.D. degrees in computer science from the University of Waterloo, Waterloo, ON, Canada, in 1985 and 1990, respectively. His doctoral thesis considered the analysis and design of provably optimal test algorithms for semiconductor memories.

He is currently an Associate Professor with the Department of Electrical and Computer Engineering, University of Alberta, Edmonton, AB, Canada. From 1981 to 1983, he was a Test Engineer and Software Designer with Mitel Corporation, Kanata, ON, Canada. His research interests include VLSI design and test, parallel signal processing algorithms and architectures, and high-performance application-specific architectures with application to broadband digital communications.

Dr. Cockburn is a member of the IEEE Computer Society, the IEEE Communications Society, the IEEE Solid-State Circuits Society, the IEEE Signal Processing Society, and the Association for Computing Machinery. He is a registered Professional Engineer in the Province of Alberta, Canada.

Duncan G. Elliott (M'97) received the B.A.Sc. degree in engineering science and the Ph.D. degree in electrical and computer engineering from the University of Toronto, Toronto, ON, Canada, in 1998.

He is an Associate Professor with the Department of Electrical and Computer Engineering, University of Alberta, Edmonton, AB, Canada. Previously, he was with Nortel in data communications, MOSAID Technologies as a DRAM designer, and IBM Microelectronics as a contractor in application-specific memory design. He spent his last sabbatical at the MIT Microsystems Technology Laboratories. His research interests include logic-enhanced memories, computer architecture, 3-D IC architecture, high-speed circuits, and merged microfluidic-microelectronic systems.

Prof. Elliott is a member of the IEEE Solid-State Circuits, Circuits and Systems, Computer, and Communications Societies and the ACM. He was the 2001 winner of the Colton Medal in microelectronics for his work on Computational RAM, which has been commercialized.