An Analog LDPC Codec Core

David Haley[†*], Chris Winstead[‡], Alex Grant[†] and Christian Schlegel[‡]

[†]Institute for Telecommunications Research, University of South Australia, Mawson Lakes SA 5095, AUSTRALIA
[‡]Electrical Engineering Department, University of Alberta, Edmonton Alberta T6G 2V4, CANADA
E-mail: dhaley@spri.levels.unisa.edu.au

Abstract*: Motivated by the potential to reuse the architecture in place for an analog sum-product decoder, and thus reduce circuit space, we show how an encoder may be constructed for a class of reversible LDPC codes. We review the design of the analog sum-product decoder for subthreshold CMOS current mode operation and then show how the architecture may be reused for encoding. The encoder circuit operates by performing the Jacobi method for iterative matrix inversion of finite field matrices. We investigate both continuous and discrete time approaches. With the addition of only simple components into the sum-product decoder variable nodes, we provide a novel design for a time multiplexed analog codec.*

Keywords: Iterative encoding/decoding, low-density parity-check (LDPC) codes, analog VLSI, iterative matrix inversion, Jacobi method.

1 INTRODUCTION

Many high performance channel codes such as low-density parity-check (LDPC) codes [1] may be represented using factor graphs [2]. Iterative decoding algorithms for these codes, such as the sum-product algorithm, are then viewed as message passing on the graph. A standard approach to implementation has been to design digital circuit architectures which map the graph, with each node acting as a processor for message passing decoding. The highly parallel structure of the factor graph representation of LDPC codes therefore offers the potential for very high throughput decoder architectures to be built.

There has been a recent suggestion [3], [4] that analog circuits be used as decoders, in contrast to the standard digital implementation approach. An analog implementation offers the potential for high speed, low power decoding. The circuit is loaded with the received soft information and performs the calculation quickly in continuous time as it settles to the steady state decoder output.

So far the analog approach has been used to design trellis based BCJR-style decoders [5]–[7], sum-product decoders [8], and a decoder for block product codes [9]. Several of these designs have been successfully fabricated.

[*]This work was supported by Southern-Poro Communications and the Australian Government under ARC SPIRT C00002232.

In [10] we introduced a class of reversible LDPC codes that may be encoded using the Jacobi method for iterative matrix inversion over finite fields. We also outlined the potential that this approach provides for decoder architecture reuse. By reusing the decoder architecture for encoding, both operations can be performed by the same circuit on a time switched basis. Hence, by eliminating the need for a separate dedicated encoder circuit we aim to reduce the overall size of the communication system. The Jacobi algorithm operates using hard decisions in discrete time and hence applying this principle to the reuse of a digital decoder circuit should be straight forward. In this paper however, we target the reuse of an analog decoder architecture. We identify potential issues presented by the reuse of the analog components and incorporate solutions to these into the design.

2 THE ANALOG DECODER

It is well known that the sum-product decoder may be used to decode LDPC codes on a factor graph representation of the code [2]. The ability to implement this algorithm in analog VLSI has also been well documented [8].

The analog decoder circuit maps the network of the code factor graph. Variable and check nodes are implemented using analog equal and XOR gates respectively [8], [11]. These nodes operate in a similar manner to their digital equivalents, although instead of passing message probabilities in discrete time they receive and transmit probabilities in continuous time as currents. The probability of the variable x, is represented using the currents on two wires, as the vector (I_{p0},I_{p1}) corresponding to ($p(x=0)$,$p(x=1)$). We denote a probability of 1 by the unit current I_u and thus have $I_{p0}+I_{p1}=I_u$. Here we consider only current mode circuits which operate in subthreshold CMOS, although other approaches have also proven effective [6].

A two input check node performs a soft XOR operation on the input vectors $\mathbf{x}$ and $\mathbf{y}$ to form the output vector $\mathbf{z}$ according to the following expression:

$$\begin{bmatrix} p(z=0) \\ p(z=1) \end{bmatrix} = \begin{bmatrix} p(x=0)p(y=0)+p(x=1)p(y=1) \\ p(x=0)p(y=1)+p(x=1)p(y=0) \end{bmatrix}$$

A two input variable node performs a soft equality operation. The output vector $\mathbf{z}$ is formed according to the following expression, where the scale factor γ is chosen to ensure $p(z=0)+p(z=1)=1$:

$$\begin{bmatrix} p(z=0) \\ p(z=1) \end{bmatrix} = \gamma \begin{bmatrix} p(x=0)p(y=0) \\ p(x=1)p(y=1) \end{bmatrix}$$

The above equations show how the XOR and equal gates operate in one direction of message passing. Larger bi-directional nodes may be built by cascading the simple structures which implement these equations [8].

In this paper we design a proof-of-concept codec for a small (3,6)-regular reversible LDPC code with the following parity check matrix.

$$\mathbf{H} = \begin{bmatrix} 1&1&0&1&0&0&0&0&1&0&0&1&1&0&0&0 \\ 0&1&1&0&1&0&0&0&1&1&0&0&0&0&1&0 \\ 0&0&1&1&0&1&0&0&0&1&0&0&0&1&0&1 \\ 0&0&0&1&1&0&1&0&0&1&1&0&1&0&0&0 \\ 0&0&0&0&1&1&0&1&0&0&0&0&1&1&1&0 \\ 1&0&0&0&0&1&1&0&1&0&1&0&0&0&0&1 \\ 0&1&0&0&0&0&1&1&0&0&0&1&0&0&1&1 \\ 1&0&1&0&0&0&0&1&0&0&1&1&0&1&0&0 \end{bmatrix} \quad (1)$$

The analog sum-product circuit which maps the factor graph of $\mathbf{H}$ is shown in Fig. 1. Each 6-edge check node is implemented using a 6-port bi-directional soft XOR gate and each variable node using a 3.5-port equal gate with an output bit slicer, as described by Lustenberger [8]. Channel observations are passed into the variable nodes and the decoded soft decisions are presented at the variable node outputs. Message probability vectors are passed as currents in a single direction along 2 wires and bidirectionally between nodes along 4 wires.

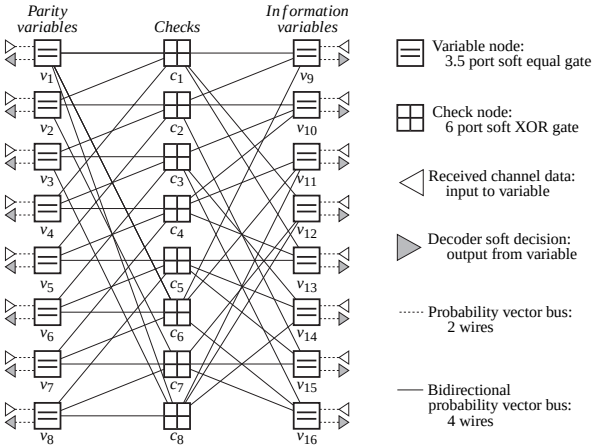

Figure 1: Analog decoder circuit for $\mathbf{H}$

3 REVERSIBLE LDPC CODES

Consider a binary systematic (n,k) code with codewords arranged as row vectors $\mathbf{x} = [\mathbf{x}_p \mid \mathbf{x}_u]$, where $\mathbf{x}_u$ are the information bits and $\mathbf{x}_p$ are the parity bits. Likewise partition the parity check matrix, $\mathbf{H} = [\mathbf{H}_p \mid \mathbf{H}_u]$. Defining $\mathbf{b} = \mathbf{H}_u \mathbf{x}_u^t$, encoding becomes equivalent to solving $\mathbf{H}_p \mathbf{x}_p^t = \mathbf{b}$. For $m \times m$ non-singular $\mathbf{H}_p$, we have $\mathbf{x}_p^t = \mathbf{H}_p^{-1}\mathbf{b}$. The Jacobi method for iterative matrix inversion is applied over $\mathbb{F}_2$ by performing the iteration

$$\mathbf{x}_{p:k+1} = (\mathbf{H}_p \oplus \mathbf{I})\mathbf{x}_{p:k}^t + \mathbf{b} \quad (2)$$

In previous work [10] we introduced a class of reversible LDPC codes which allow decoder architecture reuse to perform encoding using (2). The parity check matrix shown in (1) represents such a code, encodable in 4 iterations.

In this work we explore the reuse of an analog sum-product decoder architecture to build an encoder. To this end, we now consider two key differences between the continuous time, probabilistic analog current mode approach and the discrete time hard decision operation of (2). Firstly, the analog check node performs a soft XOR operation. The magnitude of certainty provided at the soft XOR output will be less than, or at most equal to, the certainty of its weakest input. Given that we are dealing with a feedback network of probabilities, if the magnitudes of the current pairs input to a check node stray below the extreme values, i.e. $(I_{p0}, I_{p1}) = (I_u, 0)$ or $(I_{p0}, I_{p1}) = (0, I_u)$, then this effect will cascade around the circuit and the output currents will settle at $I_{p0} = I_{p1} = I_u/2$. This effect is explained in more detail in [12]. The second point to consider involves setting the initial value. The algorithm will converge for any initial hard decision guess of $\mathbf{x}_p$ so we must initialize each element of the first guess $\mathbf{x}_{p:0}$ to something other than $p(x_{pi} = 0) = p(x_{pi} = 1) = 0.5$.

4 THE ANALOG ENCODER

We now extend the analog decoder circuit described in Sec. 2 to allow encoding, while addressing the issues outlined in the previous section.

The information variable nodes $(x_9 \ldots x_{16})$ are extended to allow encoding as shown in Fig. 2 for the case of x_9. The transmission gates and multiplexer (also built from transmission gates) are used to switch the node between encode (enc) and decode ($\overline{enc}$) modes. In this figure 2-wire probability vector buses (solid lines) carry the currents (I_{p0}, I_{p1}) and where necessary these have been expanded into single wires (dotted lines). In decode mode ($enc = Gnd, \overline{enc} = V_{dd}$) the node is connected through the equal gate, as described in Sec. 2. Gate edge connections E1Out, E2Out and E3Out are routed to adjacent check nodes. Edge connections E1In, E2In and E3In receive messages routed from adjacent check nodes. The received channel observation and decoder decision for the bit are routed into and out of the GateIn and GateOut ports respectively. The first step toward encoding is to generate the vector $\mathbf{b} = \mathbf{H}_u \mathbf{x}_u^t$. In encode mode ($enc = V_{dd}, \overline{enc} = Gnd$) the equal gate that is used in decode mode is disconnected. Instead, the information bit value is presented at the node input and a current mirror block is used to replicate and pass this current along each outgoing edge of the variable node. Each check node then includes the three adjacent incoming information bits in the XOR operation, implicitly producing $\mathbf{b}$, without the need for modification. For example, the XOR operation at check c_1 includes information bits x_9, x_{12} and x_{13} which produce the equivalent of b_1.

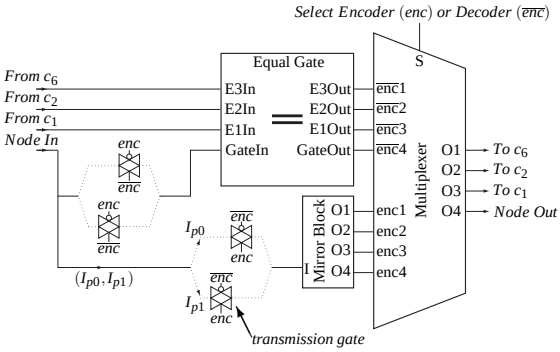

Figure 2: Information variable node structure for x_9

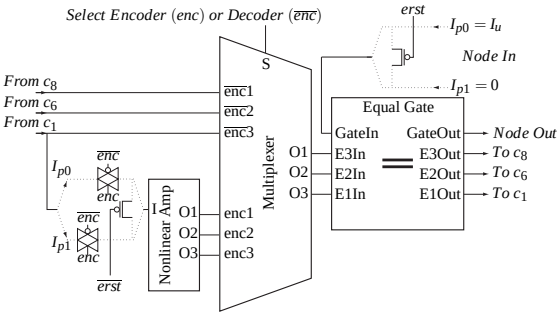

Figure 3: Parity variable node structure for x_1

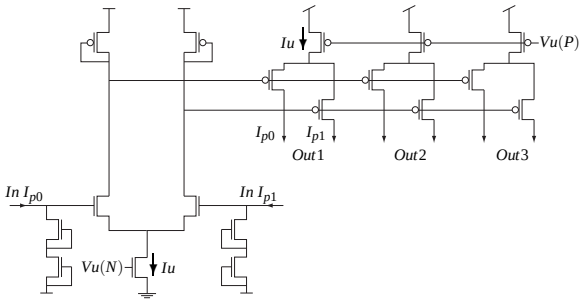

Figure 4: Nonlinear amplifier

We now extend the parity variable node design to allow encoding, as shown in Fig. 3 for the case of x_1. In decode mode the node is multiplexed through the equal gate to form the connections described in Sec. 2. In encode mode, to implement the Jacobi iteration (2) we require only the check node output for the probability current pair passed along the path $\mu_{c_i \to v_i}$ for each v_i representing the parity bit x_{pi}. We must then feed the current pair for v_i back into the checks $c \in \mathcal{A}(v_i) \setminus c_i$. It is also this current pair that will form the final decision for x_{pi}. Here we use a nonlinear amplifier, shown in Fig. 4, to boost the soft decision and prevent the issue of current degradation outlined in the previous section. The amplifier employs a nonlinear differential pair [13] which has a stack of diode connected input transistors. It also has wide transistors in the differential pair and long load transistors which drive the p-type output differential pairs. The amplifier outputs are presented as input to the equal gate which boosts the certainty of the decision further and then routes it to the necessary checks. To set the initial value $\mathbf{x}_{p:0} = \mathbf{0}$ we reuse the decoder reset transistors. These set a uniform distribution, $p(x_{pi} = 0) = p(x_{pi} = 1) = 0.5$, by connecting individual gate input wires together. When held in reset ($erst = V_{dd}, \overline{erst} = Gnd$) the feedback reset transistor is closed and the node input reset transistor open. Hence, by applying a zero external parity input vector, the only non-uniform decision presented at the equal gate is the zero decision coming from the external input. This value is therefore passed as the variable output. We reverse the polarity of $erst/\overline{erst}$ to release the encoder from reset and start the encoding process.

5 CODEC CIRCUIT SIMULATIONS

We have built a T-SPICE model of the complete analog codec core for the **H** structure of (1). A unit current of $I_u = 100nA$ was used for the experiments in this section.

The circuit output current I_{p1} for each $p(x_i = 1)$, for the simulation of a single block being decoded is shown in Fig. 5. In this example the decoder is released from reset at $5\mu s$ and the channel has flipped bits x_7 and x_{11}, which the decoder successfully corrects to arrive at the codeword $x = [0011001000101010]$.

To demonstrate the operation of the encoder we apply the information vector $x_{9\ldots16} = [00110111]$ and expect the parity vector $x_{1\ldots8} = [11111011]$. For this example, the current I_{p1} representing $p(x_i = 1)$ for each parity output bit is shown in Fig. 6. After applying the information vector at time zero and releasing the reset latch $5\mu s$ later, the circuit settles to the correct steady state solution, i.e. x_6 is the only output bit to settle at $p(x_i = 1) = 0$.

After testing all possible codewords for this code we have observed that it is possible, although not common, for the iterative processing to extend beyond that shown in Fig. 6. As the arrival of messages is not guaranteed to be synchronized for continuous time processing, it is possible for the circuit to stray temporarily from the iterative Jacobi path before settling to the correct steady state solution. This is shown in Fig. 7 for $x_{9\ldots16} = [01001000]$.

We expect that for the case of irregular code structures the above effect may become more of an issue due to un-

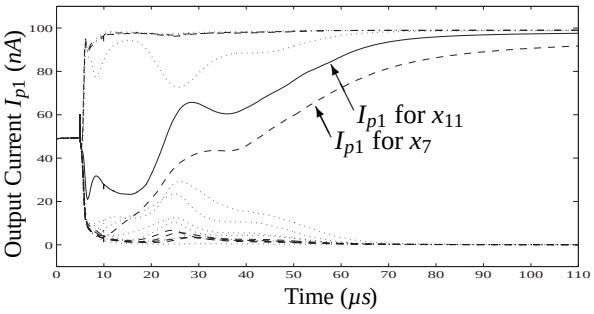

Figure 5: Decoder output with bits x_7 and x_{11} corrected

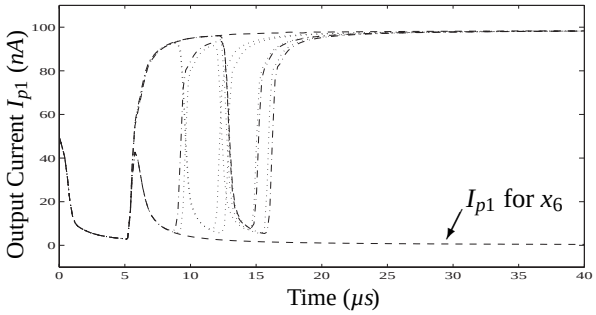

Figure 6: Encoder parity bit output decisions

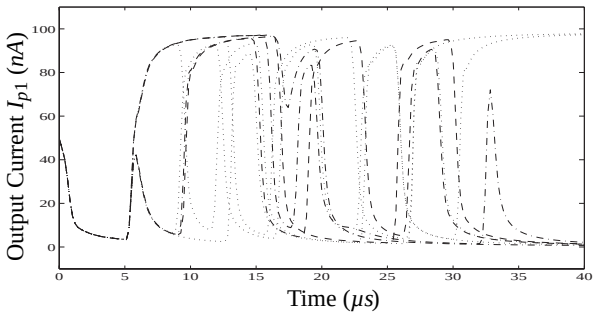

Figure 7: Encoder output decisions stray then return

even node computation times. An obvious solution is to latch the feedback and force the circuit to update each iteration in discrete time. This may be done by incorporating two basic sample and hold cells [9] into the nonlinear amplifier immediately after the diode connected input transistors. These cells represent a small addition as they each consist of only a single transmission gate and capacitor. For this case the above experiment is repeated in Fig. 8. After reset, the sample and hold cells are used to latch through the next feedback decision value. The check node calculations are then allowed to settle before the process is repeated.

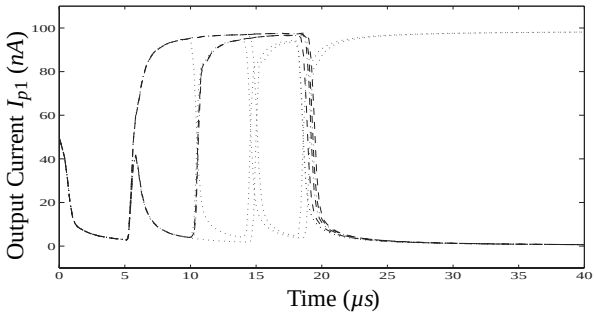

Figure 8: Latched encoder output decisions

6 CONCLUSIONS

In this paper we have reviewed the architecture of the analog sum-product decoder. By extending this circuit we have presented a novel design, and simulation results, for the core of a reversible LDPC code encoder. The small proof-of-concept design explored in this paper may be scaled up for use with larger reversible LDPC code structures such as those investigated in [10]. The encoder reuses the decoder architecture with only a small additional device overhead and represents, to our knowledge, the first analog codec design.

REFERENCES

[1] R. G. Gallager, *Low-density parity check codes*, MIT Press, Cambridge, MA, 1963.

[2] F. R. Kschischang, B. J. Frey, and H.-A. Loeliger, "Factor graphs and the sum-product algorithm," *IEEE Trans. Inform. Theory*, vol. 47, no. 2, pp. 498–519, Feb. 2001.

[3] J. Hagenauer and M. Winklhofer, "The analog decoder," in *Proc. IEEE Int. Symp. on Inform. Theory*, Cambridge, MA, 1998, p. 145.

[4] H.-A. Loeliger, F. Tarköy, F. Lustenberger, and M. Helfenstein, "Decoding in analog VLSI," *IEEE Commun. Magazine*, vol. 37, no. 4, pp. 99–101, Apr. 1999.

[5] F. Lustenberger, M. Helfenstein, H.-A. Loeliger, F. Tarköy, and G. S. Moschytz, "All-Analog decoder for a binary (18,9,5) tail-biting trellis code," in *Proc. European Solid-State Circuits Conference*, Duisburg, 1999, pp. 362–365.

[6] M. Mörz, T. Gabara, R. Yan, and J. Hagenauer, "An analog 0.25μm BiCMOS tailbiting MAP decoder," in *Proc. International Solid-State Circuits Conference*, San Francisco, 2000, pp. 356–357.

[7] C. Winstead, J. Dai, W. J. Kim, S. Little, Y. B. Kim, C. Myers, and C. Schlegel, "Analog MAP decoder for (8,4) Hamming code in subthreshold CMOS," in *Proc. ARVLSI (Advanced Research in VLSI)*, Salt Lake City, Utah, 2001, pp. 132–147.

[8] F. Lustenberger, *On the design of analog iterative VLSI decoders*, Ph.D. thesis, ETH, Zürich, Switzerland, 2000.

[9] C. Winstead, C. Myers, C. Schlegel, and R. Harrison, "Analog decoding of product codes," in *Proc. Information Theory Workshop*, Cairns, Australia, 2001, pp. 131–133.

[10] D. Haley, A. Grant, and J. Buetefuer, "Iterative encoding of low-density parity-check codes," in *Proc. GLOBECOM 2002*, Taipei, Taiwan, 2002, vol. 2, pp. 1289–1293.

[11] J. Hagenauer, M. Mörz, and E. Offer, "Analog turbo-networks in VLSI: The next step in turbo decoding and equalization," in *Proc. Int. Symp. on Turbo Codes and Related Topics*, Brest, France, 2000, pp. 209–218.

[12] D. Haley, C. Winstead, A. Grant, and C. Schlegel, "Architectures for error control in analog subthreshold CMOS," in *Proc. 4th Australian Communications Theory Workshop*, Melbourne, Australia, 2003, pp. 75–80.

[13] R. G. Benson, *Analog VLSI supervised learning system*, Ph.D. thesis, California Inst. of Technology, Pasadena, 1994.