Decoding in Analog VLSI

Hans-Andrea Loeliger and Felix Tarköy, Endora Tech AG
Felix Lustenberger and Markus Helfenstein, ETH Zurich

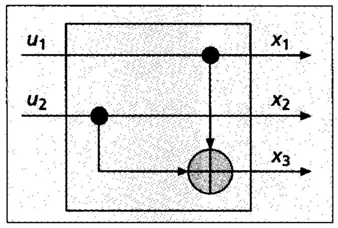
ABSTRACT The iterative decoding of state-of-the-art error correcting codes such as turbo codes is computationally demanding. It is argued that analog implementations of such decoders can be much more efficient than digital implementations. This article gives a tutorial introduction to recent research on this topic. It is estimated that analog decoders can outperform digital decoders by two orders of magnitude in speed and/or power consumption.

Advanced error correcting codes and/or coded modulation schemes are an essential part of most modern data transmission systems for both wireless and copper wire applications. As shown in Fig. 1, the coding system consists of two parts: an *encoder* in the transmitter and a *decoder* in the receiver. The *channel* in Fig. 1 comprises everything between the encoder output and decoder input, including modulation, demodulation, and equalization. The purpose of the coding system is to transform a noisy channel into a reliable bit pipe.

In contrast to many other signal processing tasks in communications, the encoding and decoding of error-correcting codes has always been implemented digitally. However, some researchers have recently become interested in analog decoders. For example, analog or hybrid implementations of the so-called Viterbi decoding algorithm for trellis codes have been proposed [1, 2]. The present article focuses on recent work on analog decoding by Hagenauer [3, 4] and by ourselves [5, 6] in the context of turbo coding and iterative decoding (see next section). This work suggests that analog decoders for such codes can be constructed that outperform digital implementations by a wide margin.

By its very nature decoding is nonlinear. In fact, it is almost the opposite of a linear operation: small disturbances of the decoder input signal should not affect the decoder output signal at all. Analog decoding has thus little in common with traditional analog signal processing such as linear filtering. It will be argued later in this article that decoding is fundamentally *better* suited for analog implementation than such linear operations.

Decoders of high-performance codes such as turbo and related codes may be viewed as being composed of building blocks that we will alternately refer to as *trellis modules* or *probability gates*. The former term refers to a type of computation that arises in turbo coding; the latter indicates that these building blocks may be viewed as a generalization of logic gates to probabilistic reasoning. We will give examples of such building blocks and indicate how they can be realized in analog very large-scale integration (VLSI).

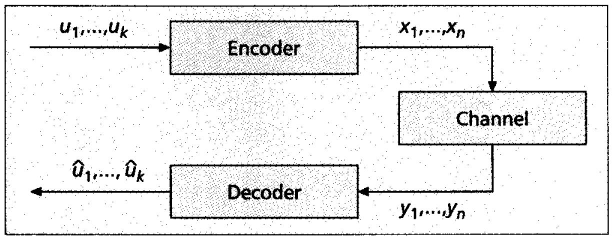


■ **Figure 2.** *Simple encoder. "Addition" is modulo 2 (exclusive OR).*

ALGEBRAIC AND PROBABILISTIC CODING

The field of error control coding is divided into the subfields of *algebraic coding* and *probabilistic coding*, with complementary strengths and weaknesses. Algebraic coding relies on the techniques of advanced abstract algebra and is most useful for providing very strong protection against low noise levels. Typical examples of codes are BCH codes and Reed-Solomon codes [7].

Probabilistic coding relies more on statistical decision techniques and is better suited to providing moderate protection against high noise levels. Typical examples of such codes are trellis codes, turbo codes, and low-density parity check codes. Turbo codes [8] represent a recent breakthrough in coding that moved the practically achievable data rates much closer to the theoretical limit (the channel capacity) than was earlier believed possible; low-density parity check codes were invented in 1963, but were only recently discovered to be almost as good as turbo codes [9].



■ **Figure 1.** *Communication system.*

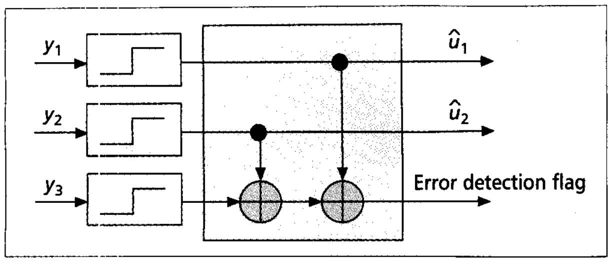
We will illustrate the nature of the computations in algebraic and probabilistic coding by the following simple example. Consider the encoder of Fig. 2. It takes as input two information bits, u_1 and u_2, and transforms them into three coded bits, x_1, x_2, and x_3. The first two coded bits, x_1 and x_2, are simply copies of the information bits u_1 and u_2, respectively; the third coded bit, x_3, results from exclusive-ORing (i.e., adding modulo 2) u_1 and u_2.

An algebraic decoder for this code is shown in Fig. 3. Because the code is too weak to allow for any error correction, this decoder performs only error detection. The computation in the decoder consists merely of two exclusive-OR operations.

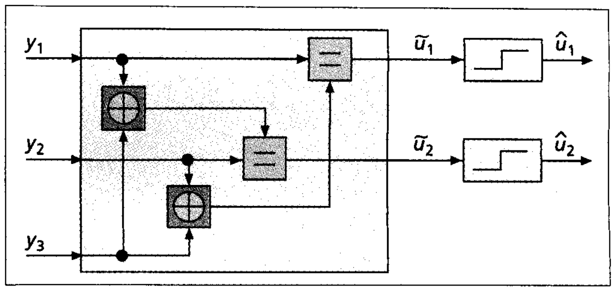
The decoder input signal $y_1, \dots, y_n$ in Fig. 1 is a sequence of real numbers (e.g., the matched filter output) that represent noisy bits. These numbers generally give an indication of the reliability of the received bit. In algebraic decoding, these "soft" bits are first converted into ordinary "hard" bits, as indicated in Fig. 3; the reliability information is discarded.

Probabilistic decoding, in contrast, operates directly with such soft bits; the conversion to hard bit estimates occurs after the decoding proper. For the example of Fig. 2, such a decoder is shown in Fig. 4. The gates in Fig. 4, which will be discussed later, involve real number operations.

In general, both encoding and algebraic decoding involve



■ **Figure 3.** *Error detection for the code of Fig. 2.*



■ **Figure 4.** *Decoding with "probability gates."*

mostly exclusive-OR operations, which are perfectly suited for digital implementation. Probabilistic decoding, on the other hand, uses computations with real numbers, which makes digital implementations of such decoders much more complex than those of algebraic decoders.

Moreover, the computation networks of decoders for high-performance codes such as turbo codes and low-density parity check codes contain loops. In digital implementations, these loops lead to iterative computations where the whole network is recomputed several times until a steady state is reached (or until the available time is over). Such iterations are also required if the soft output of the decoder is fed back to other parts of the receiver, as indicated in Fig. 5. (The use of such feedback loops is the subject of much current research.) The combined needs of real-number arithmetic and iterative decoding require substantial computational resources.

Graphical representations of codes and decoding networks as in Fig. 4 have recently become a subject of significant research interest [10–13].

Digital vs. Analog: Fundamentals

The very idea of changing from digital to analog will probably appear odd to most readers. The problems commonly associated with analog signal processing include sensitivity to component variations, susceptibility to noise and power supply disturbances, and temperature dependency, among others. However, the work by Mead [14] and others on *neuromorphic* analog VLSI has demonstrated that analog signal processing systems can be built that share the robustness of digital systems but outperform digital systems by several orders of magnitude in terms of speed and/or power consumption. This approach is characterized by exploiting, rather than fighting, the fundamental nonlinearities of transistor physics and by system designs that achieve global accuracy despite imprecise local subsystems and components.

The inherent nature of decoding makes it attractive for such an approach. Decoding operates on noisy signals that are gradually transformed into "clean" bits; since the input signals are noisy anyway, they need not be represented and processed with high precision. On the other hand, a large dynamic range is required, which can be provided by a suitable analog design. Also, analog processing with continuous-time networks eliminates the iterations mentioned in the previous section, which are replaced by the settling behavior of the network.

Decoding networks (as in Fig. 4) typically contain only a small number of different types of trellis modules (probability gates). This means that the design of such decoding networks in analog VLSI is similar in spirit to the design of digital networks with logic gates. Moreover, in the approach of [6], all module types can be realized as variations of the same simple transistor circuit.

The Implementation of Probability Gates

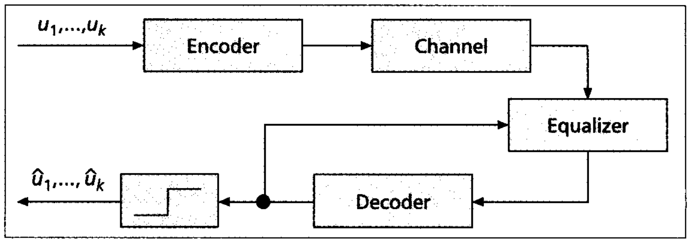
We now describe the operation and implementation of the two types of probability gates in Fig. 4. As we will see, the description of these operations depends strongly on the representation used for the soft bits. We begin with the following representation. Each soft bit is represented by *two* nonnegative real numbers, $p(0)$ and $p(1)$, that add up to one. The value $p(0)$ is (or is interpreted as) the probability that the bit is zero; $p(1)$ is (or is interpreted as) the probability that the bit is one. For this representation, the two gate types of Fig. 4 are defined as in Fig. 6. The scale factor γ in Fig. 6 is chosen such that $p_3(0) + p_3(1) = 1$.

The two gate types of Fig. 6 suffice for low-density parity check codes; they do not suffice for trellis codes and turbo codes. The additional trellis modules that are needed for the latter codes also involve nonbinary states.

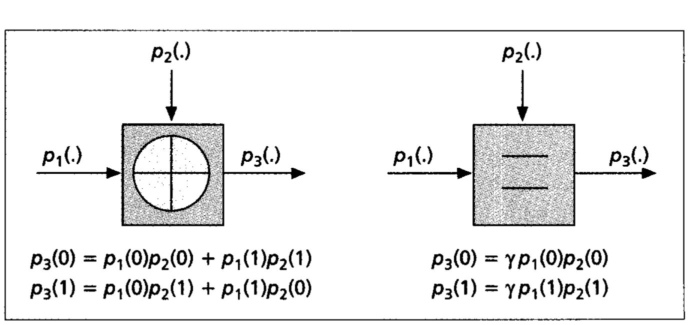
Log-Likelihood-Ratio Representation

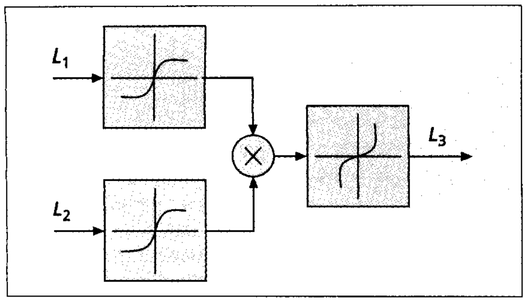
An alternative representation of a soft bit is the single number $L = \log(p(0)/p(1))$, which is commonly referred to as the *log-likelihood ratio*. In this representation, the operation of the "equals-sign" gate (right in Fig. 6) becomes simple addition: $L_3 = L_1 + L_2$. The "soft-EXOR" gate (left in Fig. 6) is expressed by the formula $L_3 = 2\tanh^{-1}(\tanh(L_1/2)\tanh(L_2/2))$, which is illustrated in Fig. 7.

Hagenauer's approach to analog decoding is based on this log-likelihood representation and on realizing these two for-



■ **Figure 5.** *Soft feedback from decoder to equalizer.*


$p_2(.)$
$p_1(.)$
$p_3(.)$
$p_3(0) = p_1(0)p_2(0) + p_1(1)p_2(1)$
$p_3(1) = p_1(0)p_2(1) + p_1(1)p_2(0)$
$p_2(.)$
$p_1(.)$
$p_3(.)$
$p_3(0) = \gamma p_1(0)p_2(0)$
$p_3(1) = \gamma p_1(1)p_2(1)$
■ **Figure 6.** *The "probability gates" from Fig. 4.*



■ **Figure 7.** *Soft-exclusive-OR gate for log-likelihood-ratio representation.*

mulas as analog circuits. A number of simulations with ideal analog cells of this type have been reported. These simulations show that continuous-time decoding with networks as in Fig. 4 can work well even if the network has loops. Transistor-level circuits have not yet been reported.

CURRENT-VECTOR REPRESENTATION

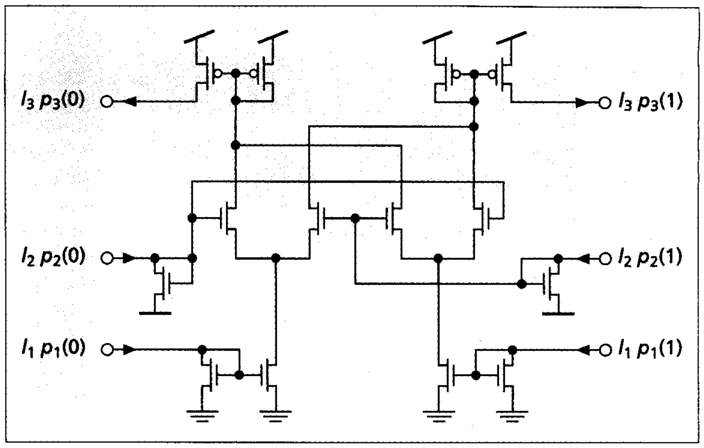
Our own approach, in contrast, is based on representing each soft bit by *two* currents, $I{\cdot}p(0)$ and $I{\cdot}p(1)$, where the sum current I can be chosen freely. This representation has the advantage that all signals/currents are unipolar and cannot overflow. Moreover, it has led to a large family of simple transistor circuits for essentially all trellis modules (even with nonbinary states) that are of interest in coding [6].

The corresponding circuit for the soft-EXOR gate (left in Fig. 6) is shown in Fig. 8. The transistors are assumed to operate in so-called subthreshold mode; alternatively, bipolar transistors can be used. The particular circuit of Fig. 8 happens to be (a version of) the so-called Gilbert multiplier, which is a standard circuit *for real number multiplication*; that it is, in fact, an *exact* soft-EXOR seems not to have been noticed before. The corresponding circuit for the equals-sign gate differs from Fig. 8 only in a slight change in the wiring.

We have made extensive transistor-level simulations with decoding networks of such modules, and a working proof-of-concept chip has been manufactured. We found that such decoding networks work very well and are robust against the nonidealities of real transistors. From this experience, we expect that such decoders will outperform digital implementations by two orders of magnitude in terms of speed (when designed for the same power level) or power consumption (when designed for the same speed).

CONCLUSIONS

We have described recent research results on analog VLSI decoding networks for error correcting codes. Such decoders appear attractive for applications where speed and/or power consumption are critical; the expected gains amount to two orders of magnitude. The versatile new "probability gates" are likely to be useful beyond decoding.



■ **Figure 8.** *Soft-exclusive-OR gate for current-vector representation.*

REFERENCES

[1] M. H. Shakiba, D. A. Johns, and K. W. Martin, "BiCMOS Circuits for Analog Viterbi Decoders," *IEEE Trans. Circuits and Sys. II*, vol. 45, Dec. 1998, pp. 1527–37.
[2] X. Wang and S. B. Wicker, "An artificial neural net Viterbi decoder," *IEEE Trans. Commun.*, vol. 44, Feb. 1996, pp. 165–71.
[3] J. Hagenauer, "Der Analoge Dekoder," German Pat. Appl. no. 197 25 275.3, filed June 14, 1997.
[4] J. Hagenauer, "Decoding of Binary Codes with Analog Networks," *Proc. 1998 Info. Theory Wksp.*, San Diego, CA, Feb. 8–11, 1998, pp. 13–14.
[5] M. Helfenstein *et al.*, "Verfahren und Schaltung zur Signalverarbeitung, insbesondere zur Berechnung einer Wahrscheinlichkeitsfunktion," Swiss Pat. Appl. no. 1998 0375/98, Feb. 17, 1998.
[6] H.-A. Loeliger *et al.*, "Probability Propagation and Decoding in Analog VLSI," *Proc. 1998 IEEE Int'l. Symp. Info. Theory*, Cambridge, MA, Aug. 16–21, 1998, p. 146.
[7] R. E. Blahut, *Theory and Practice of Error Control Codes*, Addison-Wesley, 1984.
[8] C. Berrou, A. Glavieux, and P. Thitimajshima, "Near Shannon-limit error-correcting coding and decoding: turbo codes," *Proc. ICC '93*, Geneva, Switzerland, May 1993, pp. 1064–70.
[9] D. J. C. MacKay and R. M. Neal, "Good codes based on very sparse matrices," *Cryptography and Coding. 5th IMA Conf.*,C. Boyd, Ed., no. 1025, *Lecture Notes in Computer Science*, Berlin: Springer, 1995, pp. 100–11.
[10] N. Wiberg, "Codes and Decoding on General Graphs," Ph.D. thesis, Univ. Linköping, Sweden, 1996.
[11] B. J. Frey, *Graphical Models for Machine Learning and Digital Communications*, Cambridge, MA: MIT Press, 1998.
[12] G. D. Forney, Jr., "The forward-backward algorithm," *Proc. 34th Allerton Conf. Commun., Control, and Comp.*, Monticello, IL: Allerton House, Oct. 2–4, 1996, pp. 432–46.
[13] B. J. Frey *et al.*, "Factor graphs and algorithms," *Proc. 35th Allerton Conf. Commun., Control, and Comp.*, Monticello, IL: Allerton House, Sept. 29–Oct. 1, 1997, pp. 666–80.
[14] C. Mead, *Analog VLSI and Neural Systems*, Addison-Wesley, 1989.

BIOGRAPHIES

HANS-ANDREA LOELIGER (haloeliger@endora.ch) is a consultant in signal processing, digital communications, and error-correcting codes. He received a diploma in electrical engineering from the Swiss Federal Institute of Technology (ETH), Zurich, Switzerland, in 1985 and a Ph.D. degree, also from ETH Zurich, in 1992. From 1992 to 1995 he was assistant professor at Linköping University, Sweden. He is now with Endora Tech AG, Basel, Switzerland, which he co-founded in 1995.

FELIX LUSTENBERGER (lustenbe@isi.ee.ethz.ch) received his diploma (M.Sc.) in micro engineering from the Swiss Federal Institute of Technology (EPFL), Lausanne, Switzerland, in 1995. Currently he is a research assistant at the Signal and Information Processing Laboratory of ETH Zurich, Switzerland, where he is working toward his Ph.D. degree in electrical engineering. His main research interests are in analog and neuromorphic VLSI design.

FELIX TARKÖY (ftarkoey@endora.ch) is a consultant in multi-user communications, signal processing, and error-control coding. He received a diploma (M.Sc.) and a Ph.D. degree from the Swiss Federal Institute of Technology (ETH), Zurich, Switzerland, in 1987 and 1994, respectively. From 1987–1989 he worked for Ascom Tech AG in Solothurn, Switzerland. He is now with Endora Tech AG, Basel, Switzerland, which he co-founded in 1995.

MARKUS HELFENSTEIN (helfenst@isi.ee.ethz.ch) received a diploma (M.Sc.) and a Ph.D. degree in electrical engineering from the Swiss Federal Institute of Technology (ETH), Zurich, Switzerland, in 1992 and 1997, respectively. From 1986 to 1988, he worked as an IC designer for Mettler Instrumente AG in Zurich, Switzerland. He is now a senior assistant at ETH Zurich. His scientific interest is in analog integrated circuit design, including sampled-data filters, high-speed op-amps, and analog decoding techniques for VLSI.