# Analog MAP Decoder for (8, 4) Hamming Code in Subthreshold CMOS *

Chris Winstead, Jie Dai, Woo Jin Kim†, Scott Little,
Yong-Bin Kim†, Chris Myers, Christian Schlegel
*Department of Electrical Engineering*
*University of Utah*
*winstead@eng.utah.edu*

***Abstract***

*An all-MOS analog implementation of a MAP decoder is presented for the (8, 4) extended Hamming code. This paper describes the design and analysis of a tail-biting trellis decoder implementation using subthreshold CMOS devices. A VLSI test chip has recently returned from fabrication, and preliminary test results indicate accurate decoding up to 20 MBit/s.*

## 1: Introduction

A digital receiver must determine the intended content of an incoming message on the basis of some received analog signal which has been corrupted by noise. This inference may be made in a number of ways which vary in their complexity. The quality of the inference typically increases with the receiver's complexity. It is known that, in general, the receiver (decoder) makes the best possible inference if it uses knowledge of the transmitter (encoder) to determine, for each bit of information, its most probable value given an entire sequence of analog measurements. This is known as the *Maximum A-Posteriori* (MAP) approach. Digital implementation of MAP decoders is prohibitively complex for most practical systems, and other approaches which are slightly less than optimal have consequently been more popular.

More recent developments, including the advent of Turbo codes, have presented systems which require the use of MAP decoders. Interest has therefore begun to grow in the design of more efficient MAP decoder implementations. It has been suggested [1] that a MAP decoder may be built using analog computation. Some researchers [2, 3] have implemented fully analog BiCMOS MAP decoders.

* Supported by NSF grant CCR-9971168.
† Now with Northeastern University.

MOS transistors with gate voltages below their threshold voltage provide roughly the same functional behavior as bipolar transistors [6]. The types of analog computation performed in bipolar circuits can thus be performed using only MOS devices operating below threshold. Subthreshold MOS circuits may also provide some advantages: they consume very little power, they scale to smaller sizes than bipolar transistors, and they are easier to fabricate.

Analog implementations provide an elegance which their digital counterparts lack. A single analog signal represents a real number, taking the place of several bits. Only a few transistors are needed to perform multiplication or addition of signals. Analog design thus allows for much simpler circuits than would be possible for a digital system. Analog circuits also promise robustness under a wider range of operating conditions.

Our goal is to provide a "proof of concept" implementation of a subthreshold MOS analog decoder. To this end we have chosen a non-systematic (8, 4) Hamming code. The decoding algorithm is based on the trellis structure of the code, and the complexity of the corresponding circuits is directly related to the complexity of the trellis. We have implemented a test chip that has recently returned from fabrication. Our preliminary bench tests indicate that accurate decoding can be achieved at up to 20 MBit/s.

Section 2 gives a description of the minimal tail-biting trellis for the Hamming code and describes MAP decoding. Section 3 presents an overview of subthreshold MOS circuits. Section 4 presents the analog design of our MAP decoder. Section 5 gives our simulation and testing results. Our conclusions are given in Section 6.

## 2: The Hamming Code and MAP Decoding

Our designs derive from the minimal tail-biting trellis representation of the (8, 4) Extended Hamming code, as discussed in [4]. The Hamming encoder takes a block of four "information bits" $\underline{u}$ and produces a block of eight digital bits $\underline{x}$. Thus, while there are 256 possible 8-bit sequences, only 16 of them can be produced as the encoder's output. A valid output sequence is referred to as a codeword.

Any two codewords are different from each other in four bit positions, so that valid codewords may be easily distinguished from one another. The Hamming encoder may be represented by its generator matrix $\underline{G}$, such that $\underline{u} \times \underline{G} = \underline{x}$.

$$G_8 = \begin{bmatrix} 11 & 11 & 00 & 00 \\ 00 & 11 & 01 & 10 \\ 00 & 00 & 11 & 11 \\ 01 & 10 & 00 & 11 \end{bmatrix}$$

A *trellis diagram* for a code $\mathbf{C}$ provides a graphical description of the allowable codewords of $\mathbf{C}$. Trellises may also be thought of as state-diagrams, describing the distinct states which may be entered by an encoder as it produces a codeword. The encoder is thus regarded as a finite state machine whose input is a sequence of digital information

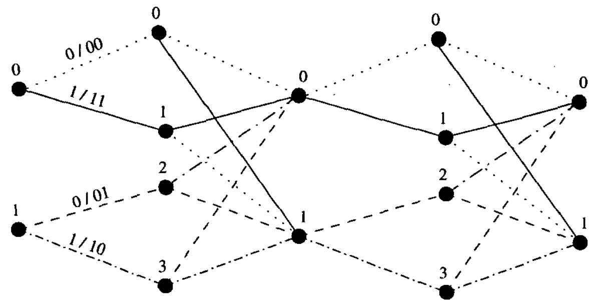


**Figure 1. Minimal Tailbiting Trellis for the (8, 4) Extended Hamming Code.**

bits. Its outputs, however, are not digital. The encoder instead selects its output from a set of physical signals. A transmittable signal is represented by a distinct symbol, or label. The set of these labels is referred to as the trellis alphabet.

The minimal "tail-biting" trellis for the (8, 4) Hamming code was developed in [4] and is shown in Figure 1. It is called a "tail-biting" graph because the final state-space is connected to the initial state-space. This means that a path through the trellis is possible, or valid, only if it begins and ends *in the same state.* The branches in the diagram indicate allowable state-transitions, and are labeled with the outputs for those state-transitions (in the figure only the first is labeled; the rest follow from the type of line). If the encoder is given an input of '0', a state-transition occurs along the upper branch. If it is given an input of '1', a transition occurs along the lower branch. The sequence of labels corresponding to a valid path through the trellis is then a valid codeword.

The states are labeled from top to bottom beginning with '0'. Note that if a state is labeled with an even index, then it is fed only by branches which correspond to an input '0'. If a state has an odd labeling, then it is fed only by branches which correspond to an input '1'. Thus, from the receiver's perspective, if the probability of each state is known (given our received signal), then the probability of each input bit is also known.

The MAP decoder attempts to calculate the probability that a particular information bit is intended by the transmitter, given an entire sequence of samples $\underline{y}$ taken from the received signal. Figure 2 presents the relevant blocks of a typical communications system. The demodulator in this system does not output digital information. It instead provides "soft" information about the received signal (i.e., the probability that each bit received is a 0 or 1).

For binary signals under ideal circumstances, the demodulator might output +1 V when a '1' has been transmitted, and -1 V when a '0' has been transmitted. In reality, the received signal is corrupted by noise. The soft output of the demodulator reflects this fact. A particular sample $y_i$ is a Gaussian distributed channel measurement. Its mean is +1 V if a '1' is transmitted and -1 V if a '0' is transmitted.

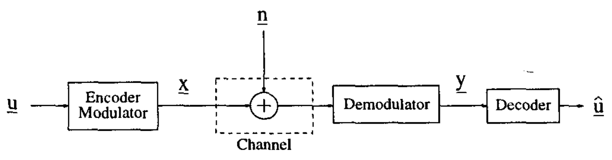


**Figure 2. A typical communications system.**

The decoder operates on these samples by propagating probability information through the trellis graph of the code. The MAP algorithm, also known as the BCJR algorithm, first appeared in [5]. Its approach is to separately propagate probability information forward and backward through the trellis. After iterating around the graph several times, the forward and backward estimates are combined to produce the final estimates of the information bits. In describing the algorithm, the following notation will be used:

- $\underline{u} = \{u_0, u_1, u_2, u_3\}$ is the encoder's input sequence.
- $\underline{x} = \{x_0, x_1, ..., x_7\}$ is the corresponding codeword.
- $\underline{y} = \{y_0, y_1, ..., y_7\}$ is the sequence of measurements provided by the demodulator.
- $\lambda_i(0)$ and $\lambda_i(1)$ are the probabilities $Pr(x_i = 0 \,|\, y_i)$ and $Pr(x_i = 1 \,|\, y_i)$, respectively, where $i = 0, 1, ..., 7$.
- $\gamma_j(p\,q)$ is the probability that the bit-pair $(x_i\, x_{i+1}) = (p\,q)$, given $y_i$ and $y_{i+1}$, where $p$ and $q$ are binary values, $j = 0, 1, ..., 3$ (the time-indices of the trellis) and $i = 2j$ denotes the codeword indices, as above. In other words, $\gamma_j(p\,q) = Pr(x_i = p \,\text{and}\, x_{i+1} = q \,|\, y_i, y_{i+1})$. There are four distinct $\gamma$ values for each trellis section (for '00', '01', '10' and '11').
- $\underline{y}_j^- = \{y_0, y_1, ..., y_{2j-2}, y_{2j-1}\}$ is the "backward-looking" sample sequence for trellis section $j$.
- $\underline{y}_j^+ = \{y_{2j}, y_{2j+1}, ..., y_7\}$ is the "forward-looking" sample sequence for trellis section $j$.
- $\alpha_j(s) = Pr(S_j = s \,|\, \underline{y}_j^-)$, where $S_j$ is the random variable representing the trellis state at time-index $j$, and $s$ is some particular state (0, 1, 2, or 3) which could occur in trellis section $j$. $\alpha$ is the forward-propagating probability. Every state in the trellis has its own $\alpha$.
- $\beta_j(s) = Pr(S_j = s \,|\, \underline{y}_j^+)$ is the backward-propagating probability.
- $\hat{\underline{u}} = \{\hat{u}_0, \hat{u}_1, \hat{u}_2, \hat{u}_3\}$ is the decoder's estimate of the intended input sequence. This is the decoder's output.

The decoding algorithm is depicted graphically in Figure 3 for forward-propagating probabilities in the second trellis section. All $\alpha$ values are uniformly initialized (i.e., their values for one time-index are all equal and sum to one). In the forward direction, each branch takes the product of the $\alpha$ value which feeds it and the $\gamma$ value associated with its

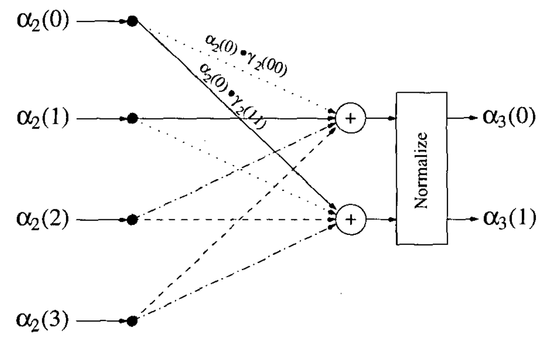


**Figure 3. Forward propagation of probabilities for the second trellis section.**

label. These products are shown for the top two branches in Figure 3. Each state node on the right sums over these products for the branches which feed it:

$$\alpha_{j+1}(s') = \kappa \sum_{s \in F_{s'}} \alpha_j(s) \cdot \gamma_j(l(s,\, s'))$$

where $F_{s'}$ denotes the set of states at time-index $j$ which are connected to state $s'$ at time-index $j+1$ (i.e., the set of branches which feed $s'$), and $l(s,\, s')$ denotes the bit-pair which labels the branch between states $s$ and $s'$. $\kappa$ is some constant chosen such that $\sum_{s'} \alpha_{j+1}(s') = 1$. The $\gamma$ values are given simply by $\gamma_j(p\,q) = \lambda_{2j}(p) \cdot \lambda_{2j+1}(q)$. The $\lambda$ values depend on the channel and on the design of the demodulator.

Propagation of $\beta$ values in the backward direction is simply the mirror image of Figure 3. The $\beta$ value for a state on the right is multiplied with the $\gamma$ values along the branches which are connected on its left. The states on the left sum over these products for the branches which leave them:

$$\beta_j(s) = \kappa \sum_{s' \in B_s} \beta_{j+1}(s') \cdot \gamma_j[l(s,\, s')]$$

where $B_s$ is the set of states at time-index $j+1$ which are connected by branches to state $s$ at time-index $j$, and $l(s,\, s')$ is defined as before. Again, $\kappa$ is chosen so that the sum over $\beta_j(\cdot)$ is equal to one. These forward and backward propagations continue independently around the trellis. The boundaries of the trellis are connected in a circle, so that the propagation continues in a feedback loop. After enough time[1] has elapsed, a decision can be made. The final MAP information bit estimates are given by:

[1]Convergence time for the algorithm is not known. The algorithm is not necessarily stable for all cases under this feedback arrangement. Simulations indicate that any instabilities in the algorithm are rare enough that the statistical performance does not suffer.

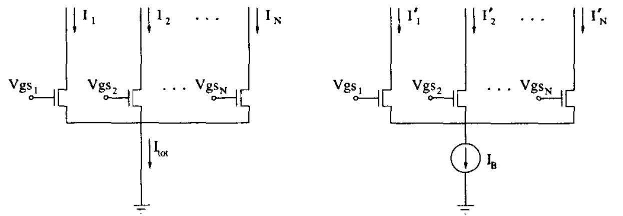


**Figure 4. Subthreshold multiplication.**

$$Pr(u_j = 0 \,|\, \underline{y}) = \sum_{s \in \{0,2\}} \alpha_j(s) \cdot \beta_j(s)$$

$$Pr(u_j = 1 \,|\, \underline{y}) = \sum_{s \in \{1,3\}} \alpha_j(s) \cdot \beta_j(s)$$

The output bit-decision $\widehat{u}_j$ is selected according to the greater of these probabilities.

## 3: Subthreshold MOS Circuits

If the gate-to-source voltage $v_{gs}$ of a MOS transistor is below the transistor's threshold voltage $V_{th}$ (slightly less than one volt for most processes), the transistor is said to be operating in the subthreshold region and obeys the following approximate model [6]:

$$I_D = I_0 \cdot \frac{W}{L} \cdot e^{v_{gs}/V_T} \cdot (1 - e^{v_{ds}/V_T})$$

where $V_T = \frac{k_B T}{q}$ and $I_0$ is a process dependent constant. If $v_{ds}$ is large enough that the $(1 - e^{v_{ds}/V_T})$ term may be neglected, then the transistor is said to be saturated. A saturated MOS device in subthreshold is then modeled by a single exponential term.

This behavior can be exploited to build simple current-mode analog multiplication and addition circuits, as illustrated in Figure 4, which presents the same circuit with and without a bias current. A bit of algebra reveals [7] that the drain currents of the biased transistors are linearly scaled so that their sum is equal to $I_B$. Thus:

$$I'_j = I_j \cdot \frac{I_B}{I_{tot}}$$

This property is the basis of the common Gilbert analog multiplier cell. Multiplication of analog currents in subthreshold MOS can thus be accomplished through biasing of current mirrors, and addition can be performed simply by shorting wires.

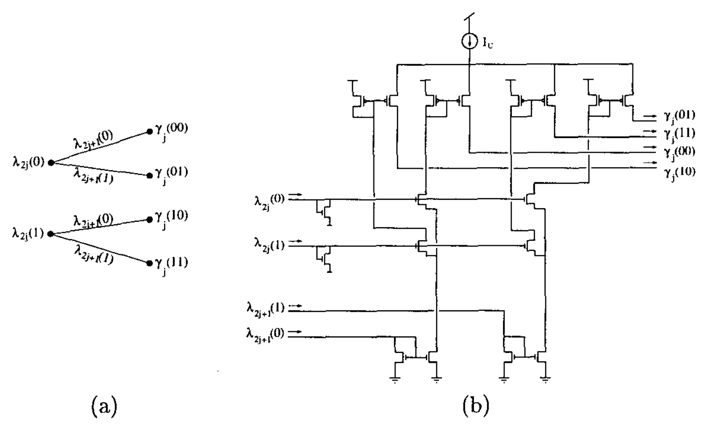


**Figure 5. Trellis-style bit-pair combination (a) and corresponding circuit (b).**

As an example, the relationship between $\gamma$ values and $\lambda$ values is depicted graphically in Figure 5(a). The graph has been drawn like a trellis in order to emphasize the simple correspondence between the trellis diagram and the probability propagation circuits. A subthreshold CMOS circuit implementing this computation is shown in Figure 5(b). The current mirrors at the bottom of the circuit bias the four interior transistors, resulting in a linear scaling. The diode-connected transistors on the left cause the interior transistors to have drain currents equal to their respective current inputs. The circuit's output for $\gamma_j(00)$, for example, is given by:

$$\gamma_j(00) = \frac{\lambda_{2j}(0) \cdot \lambda_{2j+1}(0)}{\lambda_{2j}(0) + \lambda_{2j}(1)}$$

Recalling the definition of $\lambda$ values as bit-probabilities, it follows that $\lambda_{2j}(0)+\lambda_{2j}(1) = 1$, and the circuit gives the desired function. The bias current $I_U$ provides an additional normalization on the outputs of the circuit. $I_U$ corrects for small non-ideal effects in the circuit. It also provides a reference current which corresponds to a probability of one. In other words, if $x$ is some event, and $I_x$ is the current which corresponds to the probability of that event, then $Pr(x) = 1$ means that $I_x = I_U$. The subscript 'U' therefore stands for "unit."

As a more substantive example, the backward-propagating equivalent of the second trellis section is shown in Figure 6(a), and its corresponding circuit in Figure 6(b). This circuit uses the same grid-style arrangement as in Figure 5(b). All combinations of *row* · *column* products are made available by the grid. Any unused outputs would be connected to $V_{DD}$ (in this example all combinations are used).

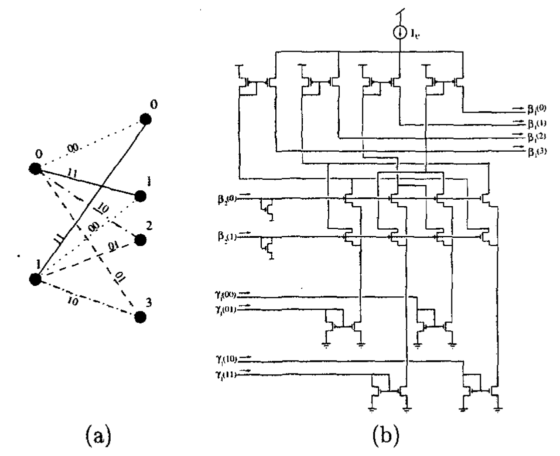


**Figure 6. Backward-propagating trellis (a) and corresponding circuit (b) for the second trellis section.**

## 4: Analog Design of an (8,4) Hamming Code MAP Decoder

The structure of the Hamming decoder chip is shown in Figure 7. The chip takes sixteen voltage inputs which correspond to the $\lambda_i$ values defined above. Each of the inputs is converted to a subthreshold-level current by a transconductor block. These currents feed into the subthreshold architecture. The subthreshold blocks are identified as follows:

- The bit-pair block produces $\gamma_j$ values from $\lambda_i$ values, as in Figure 5.
- F1 and F2 are the forward-propagating cells associated with the first and second trellis sections (which repeat, see Figure 1).
- B1 and B2 are the backward-propagating cells associated with the first and second trellis sections (an example circuit is shown in Figure 6).
- The 2-State and 4-State blocks perform the final sum over state-probabilities to yield the information bit probabilities.

The outputs of the 2-State and 4-State blocks are converted back to voltage by an output buffer. The chip has two outputs for each bit of decoder output. One is the probability that the decoded bit is a 1, and the other is the probability that it is a 0. The reference current $I_U$ was selected to be $10\mu A$. All subthreshold transistors are sized $2\mu m \times 2\mu m$. The circuit was fabricated through MOSIS using an AMI $0.5\mu m$ process. The total chip size is $1500\mu m \times 1500\mu m$. A photo of the chip is presented in Figure 8.

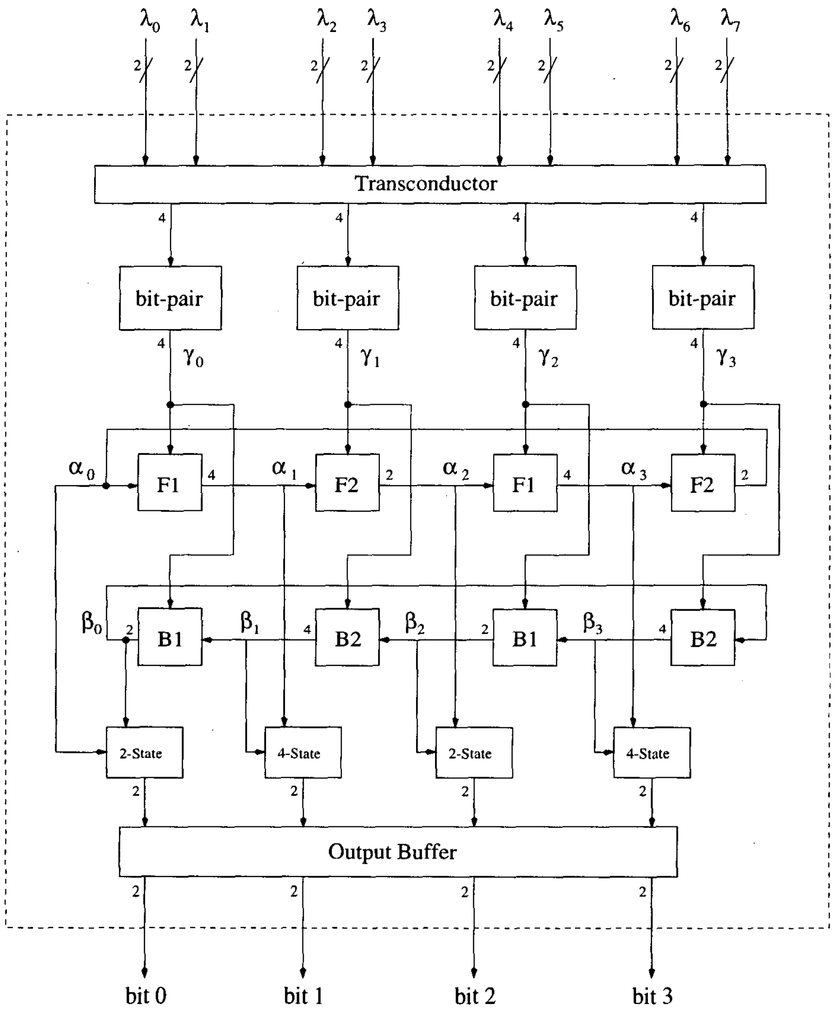


**Figure 7. Block Diagram of the Decoder.**

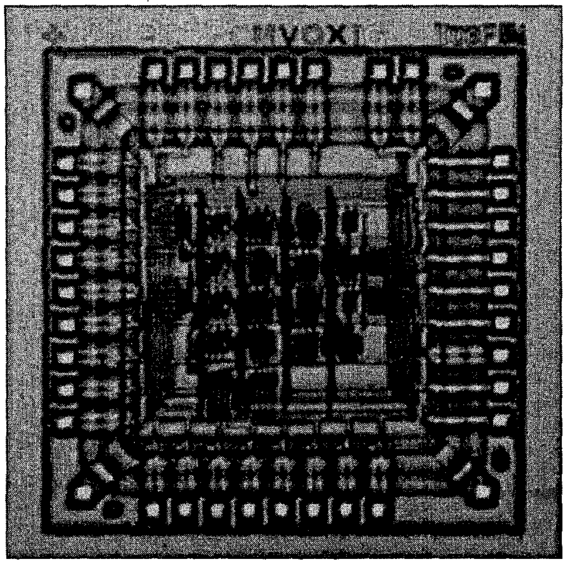


Figure 8. Die photo of the Hamming decoder.

## 5: Simulation Results

To validate the circuit design, VHDL simulations are used to estimate the bit-error rate (BER) of the decoder at different signal-to-noise ratios (SNR's). It is not possible to determine BER using SPICE-level simulation as millions of vectors must be applied at high SNR's. For comparison, a theoretical upper bound for the Hamming code's performance is estimated using an exhaustive codeword search algorithm.

For the VHDL simulation, the circuit blocks of Figure 7 are modeled by their ideal functional behavior. Each block is also assumed to introduce a unit of delay between its inputs and outputs. The signals passed between blocks are modeled by real values. The result of the block-level VHDL simulation, shown in Figure 9, is very close to the theoretical curve. This result confirms that the architecture of Figure 7 represents an optimal MAP decoder.

Other simulation results reveal a relationship between error rate and speed. Let $t_0$ be the instant when all inputs have been provided to the decoder. If more delay is given between $t_0$ and the final bit decision, then the decoder's performance improves, particularly at high SNR's. The results show that ten unit delays are required before the decoder's performance converges to the theoretical performance. Ten unit delays are allowed for the simulations depicted in Figure 9.

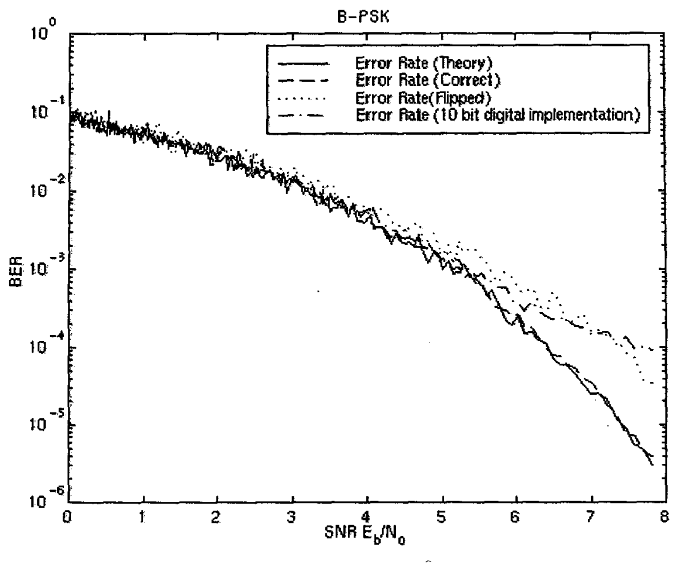


**Figure 9. Simulation Results with and without the Error.**

After the chip design was finalized and sent for fabrication, an error was discovered in the layout of the B1 block. The effect of this error is a swapping of $\gamma(01)$ and $\gamma(10)$ in the first and third trellis sections in the backward propagation only. The remainder of the circuit still weighs against this error, but a degradation of performance does result. The VHDL simulation was revised to reflect this error. The result is presented in Figure 9 as the curve labeled "flipped." This result may be interpreted as a loss in performance, but it may be equally regarded as a testament to the robustness of the analog approach. Devices built on analog principles are expected to degrade gracefully under imperfect conditions. In this case we see graceful degradation even in response to a design error.

As an additional comparison, Figure 9 presents simulation results for a 10-bit digital decoder implementation. In the digital model, all values passed between blocks are linearly quantized to ten bits. This result reveals that quantization errors can result in significant degradation in bit-error rate for high SNR's. This indicates that a large number of bits may be needed for a digital implementation of this design.

In the future, the functional models of the circuit blocks will be replaced by piecewise linear models which more closely represent their analog behavior. The simulations will also be expanded to include the effect of analog noise within the circuit. These additions will make the simulations more realistic so that they may provide greater benefit to the circuit design.

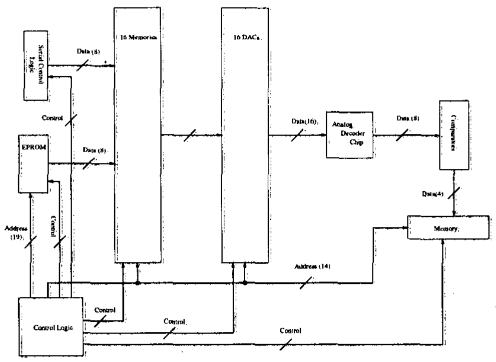


**Figure 10. Block diagram of the statistical test structure.**

## 6: Measured Results

The chip recently returned from fabrication, and its static decoding behavior has been successfully verified for a number of cases. To test the circuit's dynamical behavior, the chip's inputs are provided with a "worst case" pattern: the transition from the all-one codeword to the all-zero codeword. This is the most dramatic input transition the chip may face. It is therefore an appropriate test to yield a conservative measure of the chip's performance.

Bench tests cannot truly determine the decoder's performance. That must be assessed using a statistical measure. The circuit's true dynamic error performance for different noise environments will be evaluated using the test structure shown in Figure 10, which should be completed soon. Tests performed with oscilloscope and function generator thus give an initial indication of the chip's behavior and limitations.

Figure 11 shows the chip's response to a sudden change in input probabilities from a definite all-one codeword to a definite all-zero codeword. This shift is achieved by a complementary change from $1V$ to $0V$ for the input bit-probabilities. An input of $P(1) = 1V$ means that the channel has delivered a coded '1' with absolute certainty. The corresponding input $P(0)$ is always equal to $1 - P(1)$. The input transition is produced by a 100kHz square wave with a DC offset of $.5V$ and a peak-to-peak level of $1V$. The input transition causes a change in the decoder's codeword decision. The all-one codeword corresponds to the information sequence 1010. The all-zero codeword corresponds to the information sequence 0000. The output for bits 0 and 2 should therefore switch with the input while bits 1 and 3 should remain constant. Because of the symmetry, only bits 0 and 1 are shown in Figure 11. The chip's output probabilities behave as expected. As the output for bit 0 changes, a correct decision can be made once the two curves cross. This transition is quite rapid.

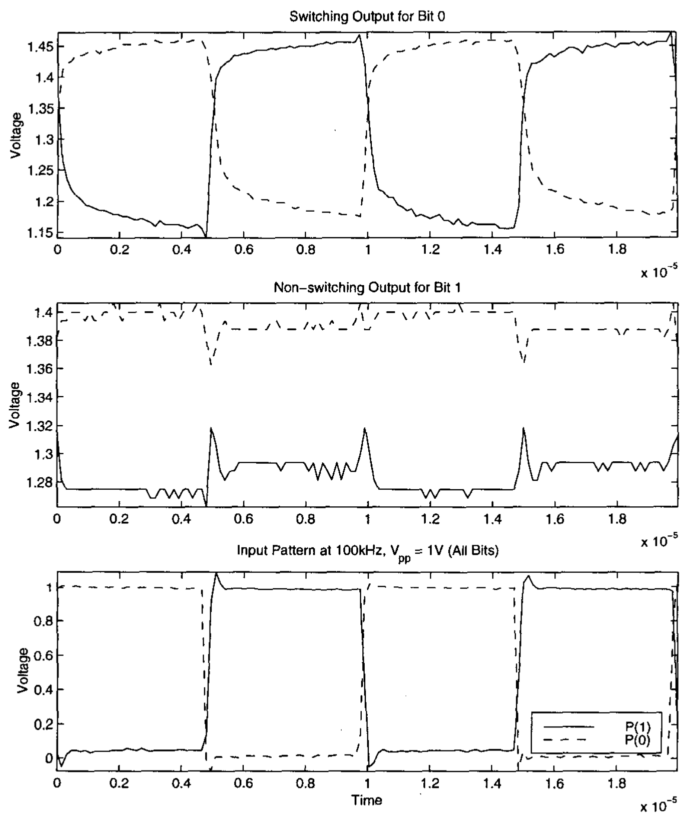


**Figure 11. The circuit's response to input transitions at 100kHz.**

As an additional test, the input probabilities may be reduced by lowering the peak-to-peak voltage of the input signal. Such a signal bears a very rough analogy to values delivered by a channel with higher noise. The circuit's response to a peak-to-peak transition of $250mV$ is shown in Figure 12. The decoder still resolves a decision, and the response time does not change.

At higher frequencies all differential outputs begin to attenuate. Clear decisions are still made at 2MHz, as shown in Figure 13(a). As the peak-to-peak input level is reduced, the decoder's decision begins to be threatened by noise. This is evident in Figure 13(b). In spite of the prevalence of noise at frequencies above 3MHz, correct decisions may still be discerned on an oscilloscope at 5MHz. An AC analysis performed using SPICE reveals that 5 MHz is the AC cutoff frequency of the decoder circuit. A blurred but correct output is still visible at 5MHz when the peak-to-peak level has been reduced to $250mV$.

Much of the chip's noise is expected to originate at the pin. A fast on-chip comparator might thus be able to resolve correct decoder decisions at or above 5MHz. Our experimental setup also suffers from high distortion in the input pattern at frequencies above 2MHz. This makes it difficult to accurately evaluate the circuit's behavior at those frequencies. The bench performance of the chip is in a sense better than expected, in that accurate decoding can conceivably take place as high as the chip's AC cutoff frequency. If these results hold for statistical tests, we can expect the chip to operate at nearly 20 MBit/s.

## 7: Conclusions

MAP decoding can be performed using all-analog computation with only MOS devices. Subthreshold MOS circuits provide the same computations as their BiCMOS cousins. They also scale to smaller transistor sizes and consume very little power. The average power consumed by our chip is 3.3mW and ranged between 2.8mW and 3.7mW. Subthreshold circuits do not yet approach the speeds of similar BiCMOS circuits. The BiCMOS decoder of [3] is reported to operate at 400 Mbit/s, which vastly outpaces the expected maximum 20 Mbit/s speed of the Hamming decoder. More study is needed to determine how subthreshold MOS devices may be optimized for speed.

The robustness and elegance of analog circuits is well established. Judging only from their parallel design, they are expected to degrade gracefully under suboptimal conditions. This is observed in the physical implementation of the Hamming decoder. The analog technique thus shows great promise for the implementation of more complex decoding systems, and may provide a practical alternative to digital probability propagation algorithms.

## 8: Acknowledgments

We would like to thank Professor Reid Harrison of the University of Utah for his comments and advice on this work especially in regards to the testing of the chip. We express

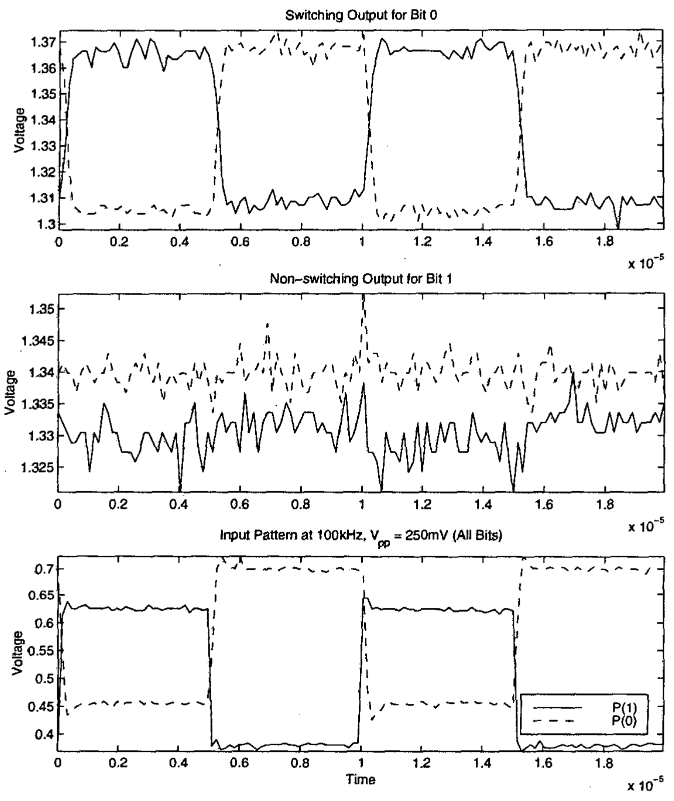
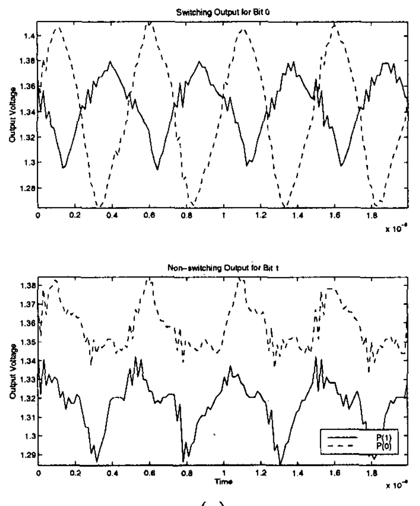
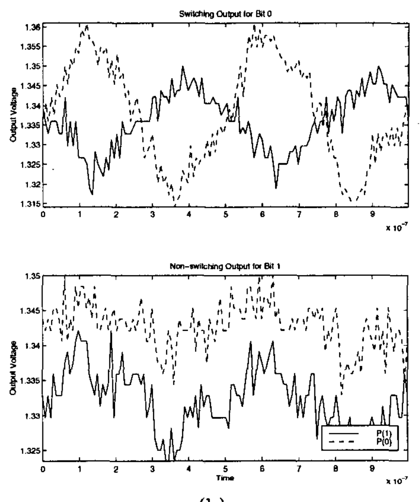


**Figure 12. Lower peak-to-peak input transitions at 100kHz.**



(a)



(b)

**Figure 13. (a) The circuit's response to 1V peak-to-peak transitions at 2MHz. (b) The circuit's response to 250mV peak-to-peak transitions at 2MHz.**

our appreciation to Hans-Andrea Loeliger of the Swiss Federal Institute of Technology for introducing us to the idea of using analog circuits to perform digital decoding. Finally, we thank Matthias Moerz and Thad Gabara for several stimulating discussions on their experiences in the design of analog decoders.

## References

[1] H.-A. Loeliger and F. Tarkoy, "Decoding in Analog VLSI," *IEEE Comm. Mag.*, vol. 37, April 1999, pp. 99-101.

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

[3] M. Moerz, T. Gabara, R. Yan, and J. Hagenauer, "An analog .25$\mu$m BiCMOS tailbiting MAP decoder," *IEEE Proc. International Solid-State Circuits Conference*, pp. 356-357, San Francisco, February 2000.

[4] A. R. Calderbank, G. D. Forney, Jr., and A. Vardy, "Minimal Tail-Biting Trellises: The Golay Code and More," *IEEE Trans. Inform. Theory*, vol. 45, pp. 1435-1455, July 1999.

[5] L. R. Bahl, J. Cocke, F. Jelinek, and J. Raviv, "Optimal decoding of linear codes for minimizing symbol error rate," *IEEE Trans. Inform. Theory*, vol. IT-20, pp. 284-287, March 1974.

[6] C. A. Mead, *Analog VLSI and Neural Systems*, Reading, MA, Addison-Wesley, 1989.

[7] Andreas Andreou, "Neural Information Processing II," *Analog VLSI: Signal and Information Processing*, ed. Mohammed Ismail and Terri Fiez, New York, McGraw-Hill.