A Micropower Analog VLSI HMM State Decoder for Wordspotting

John Lazzaro and John Wawrzynek
CS Division, UC Berkeley
Berkeley, CA 94720-1776
lazzaro@cs.berkeley.edu, johnw@cs.berkeley.edu

Richard Lippmann
MIT Lincoln Laboratory
Room S4–121, 244 Wood Street
Lexington, MA 02173-0073
rpl@sst.ll.mit.edu

Abstract

We describe the implementation of a hidden Markov model state decoding system, a component for a wordspotting speech recognition system. The key specification for this state decoder design is microwatt power dissipation; this requirement led to a continuous-time, analog circuit implementation. We describe the tradeoffs inherent in the choice of an analog design and explain the mapping of the discrete-time state decoding algorithm into the continuous domain. We characterize the operation of a 10-word (81 state) state decoder test chip.

1. INTRODUCTION

Digital implementations of signal processing systems have numerous advantages over their analog counterparts, including accuracy, immunity from temperature- and supply-induced variations, and ease of design and test. These advantages make digital the default implementation choice in many application domains. In certain applications, however, a particular system requirement is much easier to achieve in an analog implementation.

Power consumption is a requirement that sometimes dictates an analog design. For example, implantable medical devices require signal processing systems that operate

continuously for many years from a small battery. A maximum power consumption specification of a few microwatts is not uncommon in these devices (Coggins *et al.,* 1995). For many of these micropower systems, an analog implementation may be the only realistic way to meet the power specification.

In this paper, we describe an analog implementation of a common signal processing block in pattern recognition systems: a hidden Markov model state decoder. The design is intended for applications such as voice interfaces for portable devices that require micropower operation. The paper focuses on the circuit and architecture choices that limit the impact of analog nonidealities on system performance.

The paper begins (Section 2) with an explanation of hidden Markov model state decoding, using the wordspotting speech recognition task as an application. In Section 3, the digital, discrete-time state decoding algorithm is recast in analog, continuous-time circuits and tradeoffs inherent in this mapping are detailed. Section 4 describes a test chip using this decoder. Section 5 shows experimental data from this chip.

The state-decoder design is a practical implementation of the Viterbi Net architecture originally described in (Lippmann and Gold, 1987). Analog circuit implementations of state decoding have previously been used in data storage applications (Matthews and Spencer, 1993).

2. HIDDEN MARKOV MODEL STATE DECODING

Hidden Markov models (HMMs) are used in the most successful modern speech recognition systems (Bourlard and Morgan, 1994). An HMM speech recognition system consists of a probabilistic state machine, and a method for tracing the state transitions of the machine for a given input speech waveform.

Figure 1 shows a state machine for a simple speech recognition problem: detecting the presence of keywords ("Yes," "No") in conversational speech. This type of recognition where a small set of words is detected in unconstrained speech is called wordspotting (Lippmann *et al.,*, 1994). Wordspotting can provide simple-to-use voice control of systems with little user instruction or training. As the speaker pronounces the word "Yes" the state machine sequences through states 1-10; while pronouncing the word "No" the machine sequences through 11-20. While speaking other words, and while only background environmental sounds are present, the machine stays in state 21 (the "Filler" state).

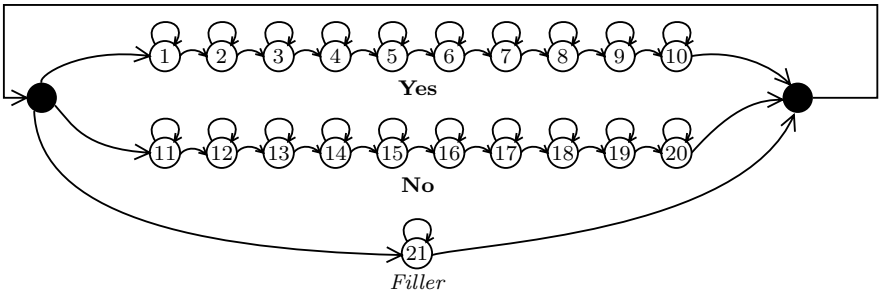

Figure 1. Grammar for a two-keyword ("Yes," states 1-10, "No," states 11-20) wordspotter.

Our goal during speech recognition is to trace out the most likely path through this state machine that could have produced the input speech waveform. This problem can be partially solved in a local fashion, by examining short (80 ms. window) overlapping (15 ms. frame spacing) segments of the speech waveform. We estimate the probability $b_i(n)$ that the signal in frame n was produced by state i, using static pattern recognition techniques. Static pattern recognition of short speech segments is a difficult task: state-of-the-art approaches provide about 60% accuracy (Bourlard and Morgan, 1994).

To improve the accuracy of these local estimates, we need to integrate information over the entire word. We do this by creating a set of state variables for the machine, called likelihoods, that are incrementally updated at every frame. Each state i has a real-valued likelihood $\phi_i(n)$ associated with it. Qualitatively, a likelihood value for a state indicates how likely it is that a path of states ending at that state best models the input speech signal. At each frame, the state with the largest likelihood is considered the current state of the machine.

Most states in Figure 1 have a stereotypical form: a state i that has a self-loop input, an input from state $i-1$, and an output to state $i+1$, with the self-loop and exit transitions being equally probable. For states in this topology, the update rule

$$\phi_i(n) = b_i(n)(\phi_i(n-1) + \phi_{i-1}(n-1)) \tag{1}$$

lets us estimate $\phi_i(n)$, given the prior state likelihoods $\phi_i(n-1)$ and $\phi_{i-1}(n-1)$ and the static probability estimate $b_i(n)$ as described above. In this equation, the transition probabilities have been left out for simplicity because they only introduce a constant scaling factor.

Figure 2 shows a complete system architecture which uses HMM state decoding to perform wordspotting. The "Feature Generation" and "Probability Generation" blocks comprise the static pattern recognition system, producing the probabilities $b_i(n)$ at each frame. Using these probabilities, the "State Decoding" block updates the likelihood state variables $\phi_i(n)$. The end-state likelihoods for each keyword and the filler state are monitored by the "Word Detection" block. The "Word Detection" block uses a simple, online algorithm to flag the occurrence of a word. Keyword end-state likelihoods are divided by the filler likelihood, and when this ratio exceeds a fixed threshold a keyword detection is signalled.

This paper primarily concerns implementations of the "State Decoding" block. In the remainder of this section, we focus on an issue central to the implementation of Equation 1: the dynamic range of $\phi_i(n)$. As the $b_i(n)$ are probabilities, $\sum_i b_i(n) = 1$. Therefore, all $b_i(n) \leq 1.0$, and with every iteration of Equation 1, most $\phi_i(n)$ values grow smaller.

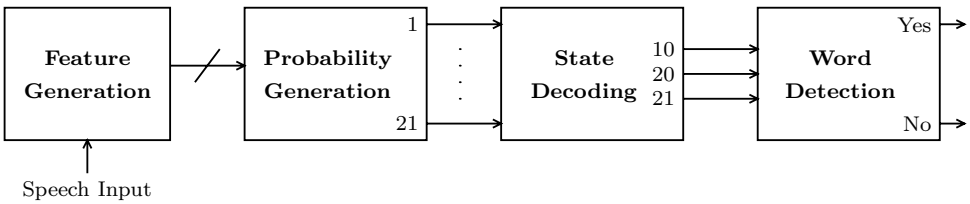

Figure 2. Block diagram for the two-keyword spotting system.

In our experience, a probability range of $0.0001 \leq b_i(n) \leq 1.0$ is necessary for best wordspotting accuracy. With this range of $b_i(n)$, any $\phi_i(n)$ may be reduced by a factor of 10,000 every frame. We use several techniques to handle this wide dynamic range. Equation 1 can be restated in the logarithmic domain, as:

$$\log(\phi_i(n)) = \log(b_i(n)) + \log(\phi_i(n-1) + \phi_{i-1}(n-1)), \tag{2}$$

and "log likelihood" values (i.e. $\log(\phi_i(n))$) become the stored state variables of the machine. Log likelihoods are negative numbers, whose magnitudes increase with each frame. Each unit of log likelihood represents an order of magnitude of likelihood. We limit the range of log likelihood values using a renormalization technique. Renormalization is implemented by maintaining a minimum log likelihood value for the overall state machine. If any log likelihood in the systems falls below this minimum value, a positive constant is added to all log likelihoods in the machine to prevent clipping.

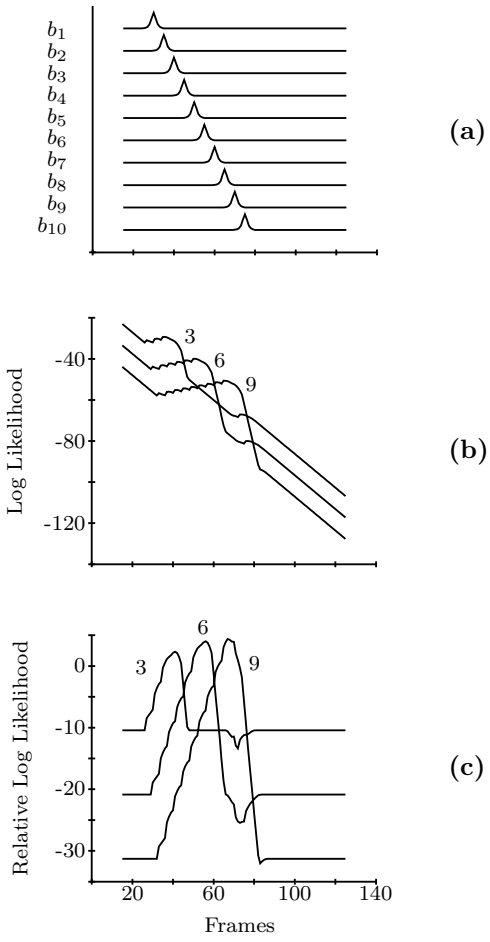

Figure 3. Simulation showing state decoder dynamic range requirements. **(a)** Word probability inputs $b_1 \ldots b_{10}$. **(b)** Log likelihood outputs from states 3, 6 and 9. **(c)** Data from (b), normalized by subtracting the log likelihood of the filler state.

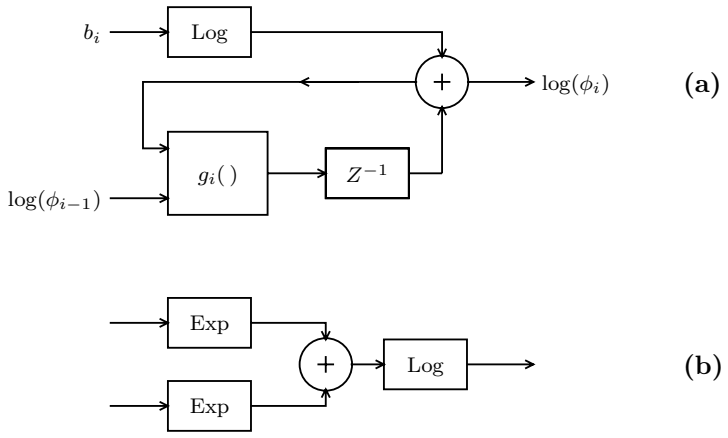

Figure 4. (a) Top-level block diagram implementation for Equation 2, the single-state decoding equation. **(b)** Implementation of the $g_i()$ function used in (a).

Figure 3 illustrates the range of log likelihoods needed for the wordspotting architecture. This figure shows data from a state decoder simulation for the state machine shown in Figure 1. The artificial input to this simulation (Figure 3a) is a simplified probability pattern that could be generated by speaking the word "Yes." This speech input produces sequential peaks in $b_1, b_2, \ldots b_{10}$.

Figure 3b shows the log likelihood values for states 3, 6, and 9 in the machine during this word, without renormalization. The log likelihood outputs trend downward throughout the simulation. Figure 3c removes this downward trend from the data, to expose the true dynamic range requirements of the system. We generated the data shown in Figure 3c by subtracting the log likelihood of the filler state from the data shown in Figure 3b, as is performed in the word detection stage of Figure 2.

The log likelihood signals shown in Figure 3c range over about 35 units during the word. In our experience with this wordspotting architecture, if the log likelihood values are constrained to operate over less than 35 units, a word that differs from "Yes" at its start (for example, "Less") may be incorrectly flagged as a "Yes."

Not all speech recognition systems require such a large log likelihood dynamic range. The wordspotting algorithm described in this section uses a simple word detection algorithm that requires a large dynamic range for accurate recognition. Other approaches that use more complex algorithms which function well with limited dynamic range would be infeasible for an analog implementation.

3. ANALOG CIRCUITS FOR STATE DECODING

The block diagram shown in Figure 4a is a digital discrete-time implementation of Equation 2. This section begins by introducing an analog discrete-time state decoder implementation, that shares the topology of Figure 4a. Then, we implement the renormalization procedure described in Section 2, and recast this design as a continuous-time circuit. Finally, the system architecture of the state decoder is described in detail.

3.1 ANALOG DISCRETE-TIME STATE DECODING

Figure 5a shows an analog discrete-time implementation of Equation 2. The delay element (labeled Z^{-1}) acts as a edge-triggered sampled analog delay, with full-scale voltage input and output. The delay element is clocked at the frame rate of the state decoder (15 ms. clock period). The "combinatorial" analog circuits must settle within the clock period.

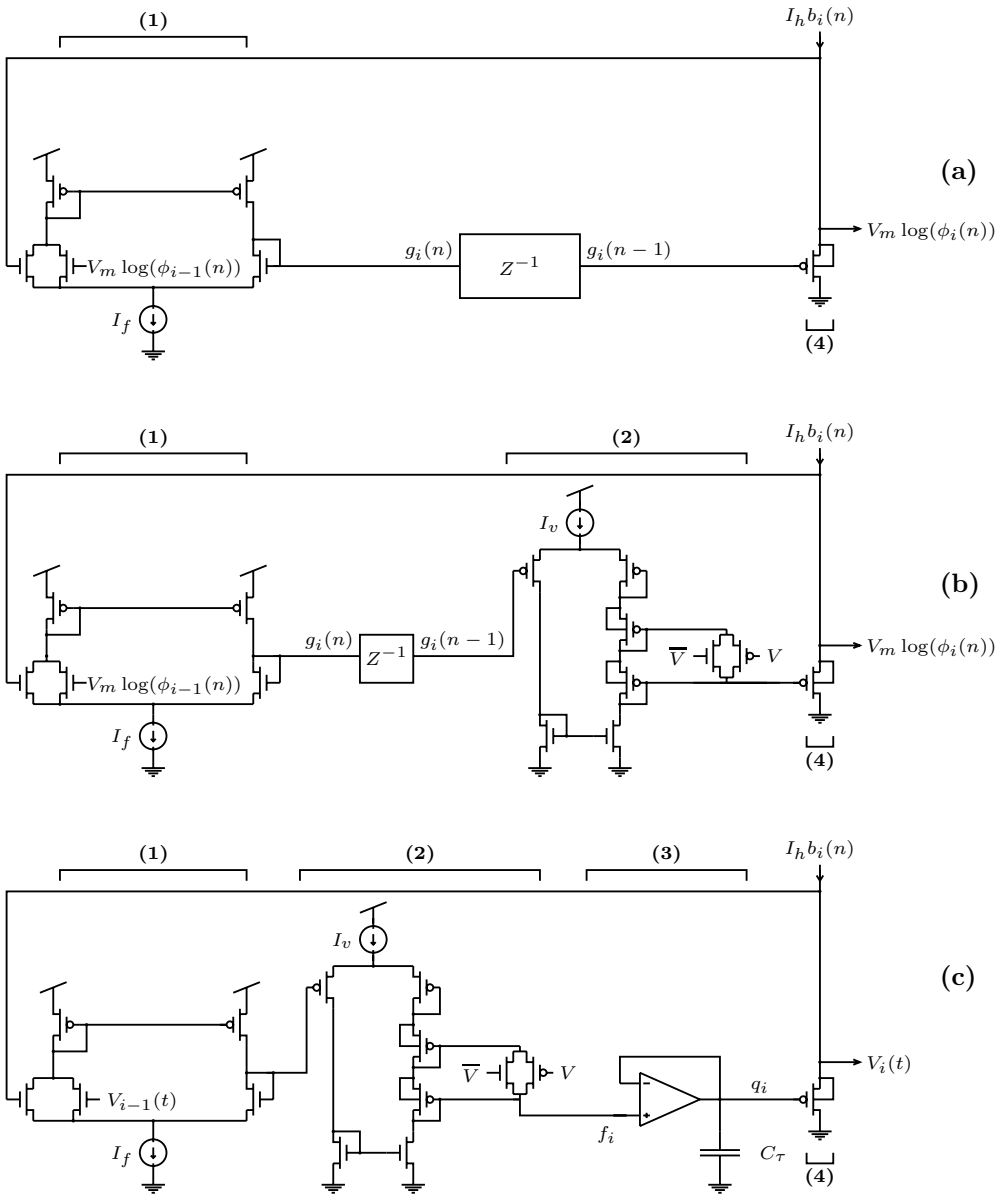

Figure 5. **(a)** Analog discrete-time circuit implementation of a single-state decoder. **(b)** Enhanced version of (a), includes the renormalization system. **(c)** Continuous-time extension of (b).

A clock period of 15 ms. allows a relatively long settling time, which enables us to make extensive use of submicroampere currents in the circuit design. The microwatt power consumption design specification drives us to use such small currents. As a result of submicroampere circuit operation, the MOS transistors in Figure 5a are operating in the weak-inversion regime. In this paper we use this model of weak-inversion MOS transistor operation (n-channel version):

$$I_{ds} = I_o \exp(\frac{\kappa V_g - V_s}{V_o}), \tag{3}$$

where $V_o = kT/q$, and parameters I_o and κ are functions of V_{sb}. We add the subscripts n or p to these parameters to denote n-channel or p-channel when needed.

Equation 2 uses two types of variables: probabilities and log likelihoods. As noted in Section 2, probabilities range over a factor of 10,000, and log likelihoods range over a factor of 35. In the implementation shown in Figure 5, we choose unidirectional current as the signal type for probability, and large-signal voltage as the signal type for log likelihood.

We can understand the dimensional scaling of these signal types by analyzing the floating-well transistor labeled **(4)** in Figure 5a. This transistor implements the blocks labeled "Log" and "+" in the diagram shown in Figure 4a. The equation

$$V_m \log(\phi_i(n)) = V_m \log(b_i(n)) + g_i(n-1) + V_m \log(\frac{I_h}{I_o}) \tag{4}$$

describes the behavior of this transistor, where $V_m = (V_o/\kappa_p)\ln(10)$, and $g_i(n-1)$ is the output of the delay element. This equation brings several design issues into clear focus:

• **Analog Nonidealities.** Both I_o and κ in Equation 3 are functions of V_{sb}. However, the floating-well topology of the transistor labeled **(4)** in Figure 5a ensures $V_{sb} = 0$ for this device, and thus V_m and I_o in Equation 4 are independent of changes in V_s. However, V_m and I_o are functions of ambient temperature; in addition, multiple copies of the circuit on the same die have different values of V_m and I_o, due to parameter variations.

• **Input Probability Encoding.** Note that $b_i(n)$ is a unitless quantity, identical to the probability $b_i(n)$ in Equation 2. This unitless quantity scales the fixed unidirectional current I_h, defining the current flowing through the transistor. The current I_h is chosen to be the largest current that keeps the transistor in the weak-inversion regime. We define I_l to be the smallest value for $I_h b_i(n)$ that allows the circuit to settle within the clock period. The ratio I_h/I_l sets the supported range of $b_i(n)$. For typical layouts and process parameters, $I_h/I_l \approx 10,000$ are possible, meeting the design specification for this parameter.

• **Log Likelihood Encoding.** Note that $\log(\phi_i(n))$ is a unitless quantity, identical to the log likelihood $\log(\phi_i(n))$ in Equation 2. This unitless quantity scales the voltage V_m to form a large-signal voltage encoding of log likelihood. For typical values of κ_p at $V_{sb} = 0$ in a 2μm n-well process, a nominal value for $V_m = (V_o/\kappa_p)\ln(10)$ is 85mV at 300K. To support a log likelihood range of 35 (the design specification for this parameter in Section 2) requires a large-signal voltage range of about 3 volts (i.e. $35V_m$).

The term $g_i(n-1)$ in Equation 4 is shown as the output of the circuit labeled **(1)** in Figure 5a. Comparing Equations 2 and 4, we notice that ideally, the term $g_i(n-1)$ in Equation 4 should implement the expression $V_m \log(\phi_i(n-1) + \phi_{i-1}(n-1))$. In practice, the circuit implements the function

$$V_m \log((e^{r \ln \phi_i(n-1)} + e^{r \ln \phi_{i-1}(n-1)})^{1/r}) \tag{5}$$

where $r = \kappa_n / \kappa_p$, with κ_p constant at its $V_{sb} = 0$ value, and κ_n weakly dependent on the input voltages of the subcircuit. Equation 5 behaves identical to $V_m \log(\phi_i(n-1) + \phi_{i-1}(n-1))$ if the two input likelihoods are sufficiently different in value. For most applications, this asymptotic property is sufficient for correct system behavior.

Note that if $r = 1$ Equation 5 reduces to $V_m \log(\phi_i(n-1) + \phi_{i-1}(n-1))$. A variant of this circuit that exchanges the role of p-channel and n-channel transistors, and uses floating-well p-channel transistors for the differential pair, has $r = 1$ for all input voltage values. This design could be used if an application requires an exact implementation of $V_m \log(\phi_i(n-1) + \phi_{i-1}(n-1))$.

3.2 RENORMALIZATION

As noted in Section 2, the computed log likelihood $\log(\phi_i(n))$ in Equation 2 decreases every frame. The circuit shown in Figure 5a does not behave in this way because the voltage $V_m \log(\phi_i(n))$ *increases* every frame. This difference in behavior is attributable to the constant term $V_m \log(I_h / I_o)$ in Equation 4, which is not present in Equation 2, and is always larger than the negative contribution from $V_m \log(b_i(n))$.

Figure 5b adds a new circuit (labeled **(2)**) to Figure 5a, that allows the constant term in Equation 4 to be altered under control of the binary input V. If V is V_{dd}, the circuit in Figure 5b is described by

$$V_m \log(\phi_i(n)) = V_m \log(b_i(n)) + g_i(n-1) + V_m \log(\frac{I_h I_o}{I_v^2}), \tag{6a}$$

where the term $V_m \log((I_h I_o)/I_v^2)$ should be less than or equal to zero. If V is grounded, the circuit is described by

$$V_m \log(\phi_i(n)) = V_m \log(b_i(n)) + g_i(n-1) + V_m \log(\frac{I_h}{I_v}). \tag{6b}$$

where the term $V_m \log(I_h / I_v)$ should have a positive value of at least several hundred millivolts.

The goal of this design is to create two different operational modes for the system. One mode, described by Equation 6a, corresponds to the normal state decoder operation described in Equation 2. The other mode, described by Equation 6b, corresponds to the renormalization procedure described in Section 2, where a positive constant is added to all likelihoods in the system. During operation, a control system alternates between these two modes, to manage the dynamic range of the system.

Since system operation is under closed-loop control, the exact values of the constant terms in Equations 6a and 6b are not critical. For the fabrication process we used for the test chip presented in this paper (2μ n-well), we calculate nominal values for these constants. In this process, an I_h of 660 nA and an I_o of 1e-16 are typical, and an I_l of 66 pA and an I_v of 20 pA easily meet settling time requirements. These bias currents set the term $V_m \log(I_h I_o / I_v^2)$ in Equation 6a to -0.07V, and the term $V_m \log(I_h / I_v)$ in Equation 6b to 0.38V. These values are in the correct range to implement the operational modes of the system. For fine-line processes, however, a modified implementation of the circuit labeled **(2)** in Figure 5b may be needed for proper operation.

Analog nonidealities relating to the V_{sb} dependence of the parameters I_o and κ are avoided in the circuit labeled **(2)** in Figure 5b, by using a floating-well configuration for the key p-channel level-shifting transistors in the circuit. Mismatch in the terms $V_m \log(I_h I_o / I_v^2)$ (Equation 6a) and $V_m \log(I_h / I_v)$ (Equation 6b) across different circuits on the same die is the main inaccuracy in the renormalization circuits. Current-source mismatch, and I_o and κ mismatch, are the contributing causes of this nonideality.

3.3 ANALOG CONTINUOUS-TIME STATE DECODING

Section 2 formulated HMMs as discrete-time systems. However, there are significant advantages in replacing the Z^{-1} element in Figure 5b with a continuous-time delay circuit. The switching noise of a sampled delay is eliminated. The power consumption and cell area specifications may benefit from continuous-time implementation as well.

Fundamentally, a change from discrete-time to continuous-time is not only an implementation change, but also an algorithmic change. In this section, we present a continuous-time state decoder whose observed behavior is qualitatively similar to a discrete-time decoder. We present a simple analysis of continuous-time state decoder operation to explain this observed similarity. However, a more complete analysis would be required to quantify the limitations and appropriate application domains of continuous-time decoders.

Figure 5c shows the continuous-time state decoder circuit. The delay circuit, labeled **(3),** uses a linear transconductance amplifier in a follower-integrator configuration. The capacitance C_τ and the tunable transconductance G set the time constant for the delay. The output of the state decoder is the continuous-time voltage $V_i(t)$. In the following analysis, we link the analog discrete-time behavior of the circuit shown in Figure 5b, whose behavior is described in Equations 6a and 6b, with the continuous-time circuit shown in Figure 5c.

In this analysis, we assume that the input probability $b_i(n)$ remains a discrete-time variable. We begin our analysis at the end of frame $n-1$, and end it at the end of frame n. During this time interval of length τ the value of $b_i(n)$ is constant. We also assume that $V_i(t) \gg V_{i-1}(t)$ throughout the interval; this assumption lets us treat the circuit labeled **(1)** in Figure 5c as a simple voltage follower. Given these assumptions, we can write differential equations for the decoder circuit, for normal decoder operation (Equation 6a)

$$\frac{V_m}{C_\tau/G}(\log(\frac{I_o I_h}{I_v^2}) + \log(b_i(n))) = \frac{dV_i(t)}{dt} \tag{7a}$$

and renormalization operation (Equation 6b)

$$\frac{V_m}{C_\tau/G}(\log(\frac{I_h}{I_v}) + \log(b_i(n))) = \frac{dV_i(t)}{dt}. \tag{7b}$$

We solve these equations via direct integration, using the boundary conditions for the beginning and end of a frame:

$$\frac{V_m}{C_\tau/G}(\log(\frac{I_o I_h}{I_v^2}) + \log(b_i(n))) \int_0^\tau dt = \int_{V_m \log(\phi_i(n-1))}^{V_m \log(\phi_i(n))} dVi(t) \tag{8a}$$

$$\frac{V_m}{C_\tau/G}(\log(\frac{I_h}{I_v}) + \log(b_i(n))) \int_0^\tau dt = \int_{V_m \log(\phi_i(n-1))}^{V_m \log(\phi_i(n))} dVi(t). \tag{8b}$$

Evaluating these integrals, and rearranging terms, results in the final expressions:

$$V_m \log(\phi_i(n)) = \alpha V_m \log(b_i(n)) + V_m \log(\phi_i(n-1)) + \alpha V_m \log(\frac{I_h I_o}{I_v^2}) \tag{9a}$$

$$V_m \log(\phi_i(n)) = \alpha V_m \log(b_i(n)) + V_m \log(\phi_i(n-1)) + \alpha V_m \log(\frac{I_h}{I_v}), \tag{9b}$$

where $\alpha = \tau/(C_\tau/G)$. If G is tuned so that $\alpha = 1$, Equation 9a is identical to the discrete-time formulation of Equation 6a, and Equation 9b is identical to Equation 6b. Note the that $g(n-1)$ term in Equations 6a and 6b can be simplified to $V_m \log(\phi_i(n-1))$, if $V_m \log(\phi_i(n-1)) \gg V_m \log(\phi_{i-1}(n-1))$.

Unfortunately, this analysis technique does not easily extend to conditions other than $V_i(t) \gg V_{i-1}(t)$. For these other conditions, multiple states must be analyzed simultaneously, and a set of coupled nonlinear differential equations must be solved. However, simulations of multi-state systems indicate a qualitative similarity between continuous-time and discrete-time decoder architectures.

Several analog nonidealities affect the continuous-time state decoding architecture. One issue surrounds the tuning of the conductance parameter G to meet the condition $\tau/(C_\tau/G) = 1$. This condition requires G to be compensated for ambient temperature and power-supply variations. Recent advances in micropower compensation circuits (Oguey and Aebischer, 1996) could be leveraged to solve this problem within the power consumption specifications of the system.

A second issue involves the voltage range of linearity of the transconductance amplifier in the delay circuit. To bring this issue into focus, consider a simple amplifier implementation, that has the weak-inversion transfer function $I_{\text{out}} = I_\tau \tanh((\kappa(V_{\text{plus}} - V_{\text{minus}})/(2V_o)))$ where the bias current I_τ sets the amplifier transconductance. The equations

$$V_m \log(\phi_i(n)) = V_m \log(\phi_i(n-1)) + \frac{\tau I_\tau}{C_\tau} \tanh(0.5 \ln(b_i(n)) + 0.5 \ln(\frac{I_h I_o}{I_v^2})) \quad (10a)$$

$$V_m \log(\phi_i(n)) = V_m \log(\phi_i(n-1)) + \frac{\tau I_\tau}{C_\tau} \tanh(0.5 \ln(b_i(n)) + 0.5 \ln(\frac{I_h}{I_v})) \quad (10b)$$

describe the operation of the continuous-time state decoder, if this simple transconductance amplifier is used in the delay circuit. Using the bias current values for a 2μ n-well process detailed earlier in this section, we can rewrite these equations as:

$$V_m \log(\phi_i(n)) = V_m \log(\phi_i(n-1)) + \frac{\tau I_\tau}{C_\tau} \tanh(1.15 \log(b_i(n)) - 0.9) \quad (11a)$$

$$V_m \log(\phi_i(n)) = V_m \log(\phi_i(n-1)) + \frac{\tau I_\tau}{C_\tau} \tanh(1.15 \log(b_i(n)) + 5.2) \quad (11b)$$

For all but the largest $b_i(n)$ values, the tanh function of Equation 11a will be saturated to an output of -1; for all but the smallest $b_i(n)$ values, the tanh function of Equation 11b will be saturated to an output of +1. To restore the desired system behavior shown in Equations 9a and 9b, the simple transconductance amplifier circuit could be replaced by an amplifier design with an extended linear range. Alternatively, a first-order log domain filter could be substituted for the follower-integrator delay circuit.

In the test chip presented in this paper, we used a follower-integrator delay implementation with a simple transconductance amplifier. As explained in Section 4, we were able to demonstrate the qualitative performance of the state decoding system, despite the reduced $b_i(n)$ range depicted in Equation 11a and 11b.

3.4 STATE DECODING SYSTEMS

The single-state decoder circuit shown in Figure 5c satisfies our primary design specifications. It supports an input probability range of 10,000, and for the typical bias current values detailed in Section 3.2, it supports a voltage range in excess of 3V, providing a factor of 35 in log likelihood range. In this section, we describe a state decoding system architecture that uses this circuit as a core building block.

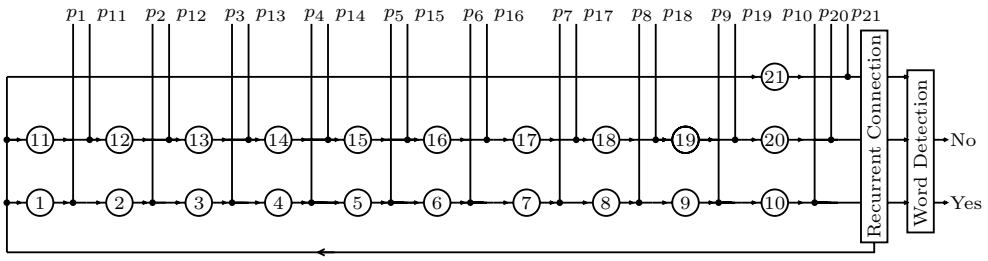

Figure 6. State decoder system for grammar shown in Figure 1.

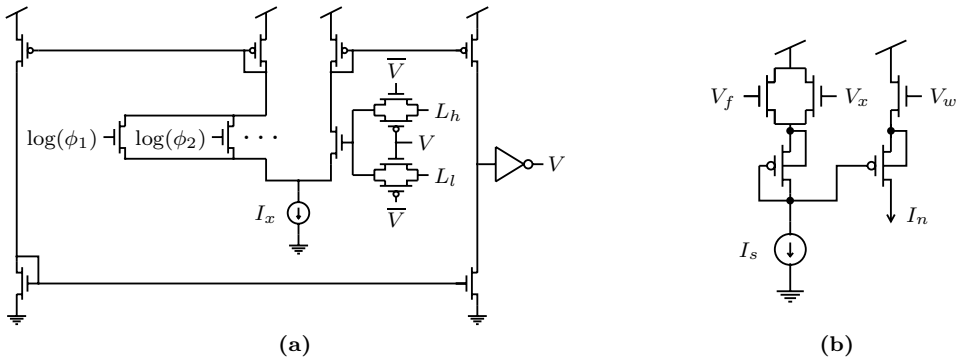

Figure 7. (a) Circuit for renormalization control. **(b)** Circuit for word detection.

Figure 6 shows a state decoding system that corresponds to the grammar shown in Figure 1. Each numbered circle corresponds to the circuit shown in Figure 5c. The signal flows of this architecture support a dense layout: a rectangular array of single-state decoding circuits, with input current signal entering from the top edge of the array, and end-state log likelihood outputs exiting from the right edge of the array. States connect to their neighbors via the $V_{i-1}(t)$ and $V_i(t)$ signals shown in Figure 5c. For notational convenience, in this figure we define the unidirectional current $p_i(t)$ to be $I_h b_i(t)$.

In addition to the single-state decoder circuit, several other circuits are needed to complete the system. The "Recurrent Connection" block in Figure 6 implements the loopback connecting the filled circles in Figure 1. We implement this block using a 3-input version of the voltage follower circuit labeled **(1)** in Figure 5c. This implementation is subject to the same nonidealities described in Section 3.1.

The "Word Detection" block in Figure 6 performs a simple function. Each keyword end-state log likelihood is subtracted from the filler state log likelihood. The circuit shown in Figure 7b performs this function, subtracting the end-state word log likelihood V_w from the maximum of the filler state log-likelihood V_f and the limiting voltage V_x, producing the output current

$$I_n = I_s \exp\left(\frac{V_w - \max(V_f, V_x)}{V_o/\kappa} \right). \tag{12}$$

Mismatches in the κ values for n-channel and p-channel transistors, and the dependence of κ on V_{sb}, are the sources of inaccuracy in Equation 12.

To complete the system, we implement the renormalization algorithm, using the circuit shown in Figure 7a. The circuit takes as input the log likelihood signals from all states ($\log(\phi_1), \log(\phi_2), \ldots$ in Figure 7a) and generates the binary control signal V, that is distributed to all states in this system. This control signal determines whether the single-state decoding circuits exhibit normal behavior (Equation 6a) or renormalization behavior (Equation 6b).

The renormalization control circuit works as follows. Initially, the behavior of Equation 6b is selected (renormalization mode). In this mode, log likelihood voltages

tend to increase over time, with the the most likely states leading the ascent. This mode stays in effect until one of the log likelihood voltages reaches the voltage L_h (shown in Figure 7a). At this point, the control circuit toggles the value of V, so that the behavior of Equation 6a is selected. In this mode, log likelihood voltages tend to decrease over time. Once the largest state log likelihood voltage falls to the voltage L_l (shown in Figure 7a, set to be several hundred millivolts below L_h), renormalization behavior is again selected, starting another cycle. Note that unlike the classical renormalization algorithm described in Section 2, the circuit shown in Figure 7a operates by detecting overflow, not underflow.

There are several deficiencies in this design. The switching of the control signal V (and its complement $\overline{V}$) produces glitches at the node labeled f_i in Figure 5c; these glitches are integrated by the continuous-time delay circuit, introducing error into the likelihood values. In addition, unless low-power design techniques are used to implement the inverter in the circuit shown in Figure 7a, a sizable current glitch occurs during each change of V, impacting system power dissipation. Finally, as indicated by Equations 11a and 11b, the switching approach to renormalization is the underlying reason that a transconductance amplifier with a large linear input range is needed in the continuous-time delay circuit.

Circuit-level improvements can lessen these deficiencies. An alternative approach is to abandon the discrete-time renormalization algorithm, in favor of a continuous-time algorithm. Continuous-time modulation of the current I_v in the circuit labeled **(2)** to Figure 5b could be the basis for such a continuous-time renormalization system.

4. STATE DECODER TEST CHIP

We fabricated a state decoder test chip in the 2μm, n-well process of Orbit Semiconductor, via MOSIS. The chip has been fully tested and is functional. The chip decodes a grammar consisting of eight ten-state word models and a filler state. The state decoding and word detection sections of the chip contain 2000 transistors, and measure $586\,\mathrm{x}\,2807\mu$m ($586\,\mathrm{x}\,2807\lambda$, $\lambda = 1.0\mu$m).

In this section, we show test results from the chip, to illustrate the qualitative behavior of the system. In these tests, we apply a temporal pattern of probability currents to the ten states of one word in the model (numbered 1 through 10), and observe the log likelihood voltage of the final state of the word (state 10). The filler state input probability current is constant. The probability currents of all states of the other word models are set to the minimum value, essentially removing these word models from the system.

The chip includes circuits for generating test patterns of input probability currents. A radial-basis function circuit (Delbruck, 1991), shown in Figure 8a, is the core element in the test generation system; the circuit shown generates the output current

$$\frac{I_u}{4}\mathrm{sech}^2\left(\frac{\kappa(V_p - V_n)}{2V_o}\right) + I_l \tag{13}$$

as a function of its differential voltage input. An array of these circuits, shown in Figure 8b, is used to generate the probability currents for the word model.

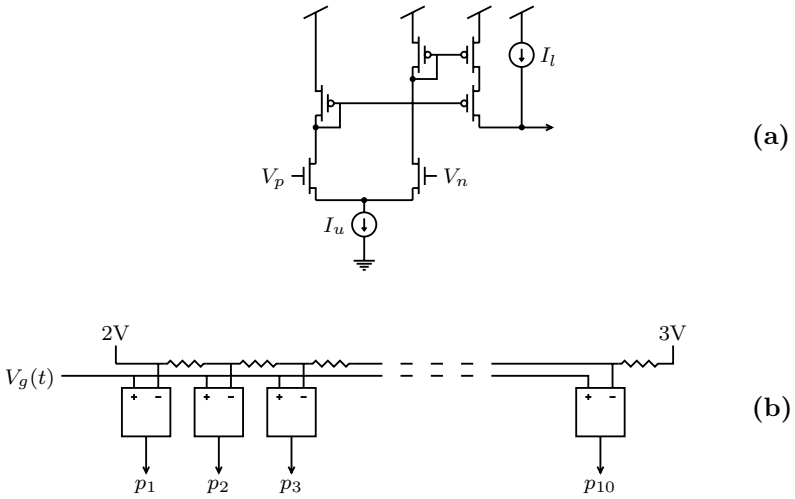

Figure 8. (a) Radial basis function circuit. **(b)** Pattern generation system for the test chip.

A low-frequency ramp waveform is applied to the input $V_g(t)$ to produce a temporal pattern of current outputs. By adjusting the amplitude and frequency of this ramp, a variety of probability patterns can be produced. In an electrical sense, this pattern generation system provides a realistic proxy for the "Probability Generation" system (Figure 2) of a speech recognition system, since radial basis functions are often used in the final processing stage of speech classifiers (e.g. (Lippmann *et al.*, 1994)).

As detailed in Section 3.3, the test chip uses a simple circuit implementation for the continuous-time delay, that has a limited range of linearity as quantified in Equations 11a and 11b. To work around the problem, we restrict our maximum input probability current to a small multiple of I_l. This restriction forces the tanh function to be saturated to an output of -1 during renormalization (Equation 11a), and operates the tanh function over its linear regime during normal operation (Equation 11b).

Figure 9 contains simulated results, allowing us to show internal signals in the system that were not brought off-chip. Figure 9a shows the temporal pattern of input probability currents $p_1 \ldots p_{10}$, that correspond to a simple simulated input word.

Figure 9b shows the log likelihood voltage waveform for the end-state of the word (state 10). The waveform plateaus at L_h, the limit of the operating range of the state decoder system. During this plateau this state has the largest log likelihood in the system. Figure 9c is an expanded version of Figure 9b, showing in detail the renormalization cycles described in Section 3.4.

Figure 9d shows the output computed by the "Word Detection" block in Figure 7. Note that during the peak plateau in Figure 9b, the Word Detection output is not constant, reflecting changes in the filler state log likelihood. Also note the smoothness of the waveform, unlike Figure 9c. By subtracting the filler-state log likelihood from the end-state log likelihood, the Word Detection block cancels the common-mode renormalization waveform.

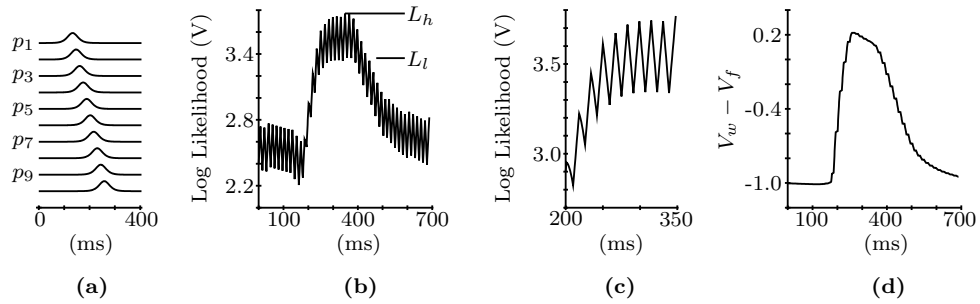

Figure 9. Simulation of state decoder: **(a)** Inputs patterns, **(b), (c)** End-state response, **(d)** Word-detection response.

Figure 10 shows a series of four experiments that confirm the qualitative behavior of the state decoder system. This figure shows experimental data recorded from the fabricated test chip. Each experiment consists of playing a particular pattern of input probability currents $p_1 \ldots p_{10}$ to the state decoder many times; for each repetition, a certain aspect of the playback is systematically varied. We measure the peak value of the end state log likelihood during each repetition, and plot this value as a function of the varied input parameter. For each experiment shown in Figure 10, the left plot describes the input pattern, while the right plot is the measured end-state log likelihood data.

The experiment shown in Figure 10a involves presenting complete word patterns of varying durations to the decoder. As expected, words with unrealistically short durations have end-state responses below L_h, and would not produce successful word detection. The experiment shown in Figure 10b also involves presenting patterns of varying durations to the decoder, but the word patterns are presented "backwards," with input current p_{10} peaking first, and input current p_1 peaking last. The end-state response never reaches L_h, even at long word durations, and (correctly) would not trigger a word detection.

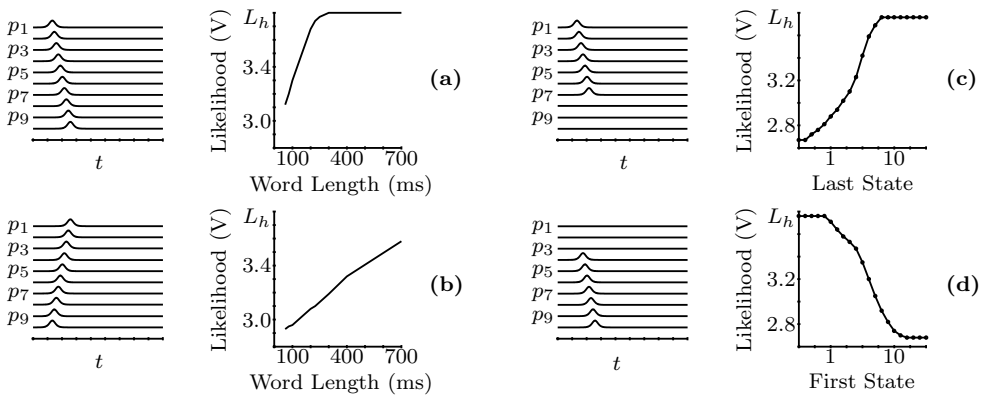

Figure 10. Measured chip data for end-state likelihoods for long, short, and incomplete pattern sequences.

The experiments shown in Figure 10c and 10d involve presenting partially complete word patterns to the decoder. In both experiments, the duration of the complete word pattern is 250 ms, the minimum duration resulting in an end-state response of L_h. Figure 10c shows words with truncated endings, while Figure 10d shows words with truncated beginnings. In Figure 10c, end-state log likelihood is plotted as a function of the last excited state in the pattern; in Figure 10d, end-state log likelihood is plotted as a function of the first excited state in the pattern. In both plots, as expected, the end-state log likelihood falls below L_h as significant information is removed from the word pattern.

While performing the experiments shown in Figure 10, the state-decoder and word-detection sections of the chip had a measured average power consumption of 141 nW ($V_{dd} = 5V$). More generally, however, the power consumption, input probability range, and the number of states are related parameters in the state decoder system. The power dissipation of a state decoding system with N states is

$$NV_{dd}(I_f + I_v + I_\tau + I_l) \; + \; P_r(N, V_{dd}) \; + \; V_{dd} I_h \sum_{i=1}^{N} (b_i(t) - (I_l/I_h)). \tag{14}$$

The first term of this Equation is the static power consumption of the array of state decoder circuits; I_τ is the static current of the analog delay circuit in Figure 5c. For the experiments shown in Figure 10, the per-state power consumption due to this term is approximately 1.5 nW, resulting in 125 nW power dissipation for the 81-state decoder array.

The second term of Equation 14, $P_r(N, V_{dd})$, accounts for power dissipation in the renormalization circuit, and in the circuits that implement the "Word Detection" and "Recurrent Connection" functions. This term is a complex function of V_{dd}; some circuits in these blocks scale linearly with V_{dd} while other circuits scale quadratically with V_{dd}. This term also increases with N; the bias currents are determined by parasitic capacitances which scale linearly with the number of states in the system. For the experiments shown in Figure 10, this term is approximately 16 nW.

The final term of Equation 14 accounts for power consumption due to input probability currents above the minimum I_l current. Due to the small value of I_h used in the experiments shown in Figure 10, this term makes a negligible contribution to the total power consumption in this experiment. However, if I_h is set to the maximum acceptable value for the fabricated test chip (660nA, for an I_h/I_l of 10,000), the power consumption of this term ($3.3\mu W$) would dominate system power consumption.

5. DISCUSSION

This paper presents a preliminary design of an analog implementation of HMM state decoding. Intermixed with the design exposition in Section 3 is a critical assessment of the shortcomings of each part of the design, along with suggestions for improved implementations. These suggestions are a starting point for a second-generation state-decoder design, that would serve as a practical building block for pattern recognition systems. Two key design goals for a second generation design are improving the continuous-time delay implementation and replacing the switched renormalization system with a continuous-time approach.

In addition to these circuit improvements, future work involves the examination of the issues involved in integrating the state decoder into a complete system. To create a micropower implementation of the wordspotting architecture shown in Figure 2, each block must be implemented using micropower techniques, not just the state decoding subsystem.

Fortunately, many signal processing blocks needed for this application have already been the subject of micropower implementation research. Several generations of micropower audio pre-processing systems (Watts *et al.,* 1992; van Shaik *et al.,* 1996) have been implemented and these designs are good candidates for the "Feature Detection" block in Figure 2. Micropower implementations of multi-layer perceptrons (Coggins *et al.,* 1995), radial basis function networks (Delbruck, 1991), or Gaussian mixture models are suitable for the "Probability Generation" block.

The design of individual signal processing blocks is perhaps the easiest challenge of this implementation project. Harder tasks include the interfaces between these blocks, and the compensation of the entire system to temperature and power-supply variations. Perhaps the hardest task is the creation of an efficient, high-level simulation model for the complete design, that captures the analog nonidealities of each block. The ultimate design goal of a speech recognition system is low recognition error. Achieving this goal requires circuit-level and microarchitecture optimizations. A system simulation that takes into account the nonidealities of the underlying analog implementation is an essential tool for guiding these optimizations.

Acknowledgments

We thank Herve Bourlard, Dan Hammerstrom, Brian Kingsbury, Alan Kramer, Nelson Morgan, Stylianos Perissakis, and Su-lin Wu for comments on this work. We also thank the anonymous reviewers for many helpful suggestions. This work was sponsored by the Office of Naval Research (URI-N00014-92-J-1672) and by the Department of Defense Advanced Research Projects Agency. Opinions, interpretations, conclusions, and recommendations are those of the authors and are not necessarily endorsed by the United States Air Force.

References

Bourlard, H. and Morgan, N. (1994). *Connectionist speech recognition : a hybrid approach.* Boston: Kluwer Academic Publishers.

Coggins, R., Jabri, M., Flower, B., and Pickard, S. (1995). "A low-power network for on-line diagnosis of heart patients," *IEEE Micro,* vol 15, no. 3, pp. 8-25.

Delbruck, T. (1991). "Bump circuits for computing similarity and dissimilarity of analog voltages," *Proceedings of the International Joint Conference on Neural Networks (IJCNN-91-Seattle),* vol 1, pp. 475-479.

Lippmann, R. P., Chang, E. I., and Jankowski, C. R. (1994). "Wordspotter training using figure-of-merit back-propagation," *Proceedings International Conference on Acoustics, Speech, and Signal Processing,* Vol. 1, pp. 389-392.

Lippmann, R. P. and Gold, B. (1987). "Neural-net classifiers useful for speech recognition," *Proceedings of the International Conference on Neural Networks (ICNN),* vol. IV, pp 417-425.

Matthews, T. W. and Spencer, R. R. (1993). "An integrated analog CMOS Viterbi detector for digital magnetic recording," *IEEE Journal of Solid-State Circuits*, vol. 28, no. 12, pp. 1294-302.

Oguey, H. and Aebischer, D. (1996). "CMOS current reference without resistance" *Proceedings European Solid State Circuits Conference* pp. 104-107.

van Schaik, A., Fragniere, E., and Vittoz, E. (1996). "Improved silicon cochlea using compatible lateral bipolar transistors," in *Advances in Neural Information Processing Systems 8,* D. Tourestzky, M. C. Mozer, and M. E. Hasselmo, Eds, Cambridge, Mass: MIT Press.

Watts, L., Kerns, D. A., Lyon, R. F., and Mead, C. (1992). "Improved implementation of the silicon cochlea," *IEEE Journal Solid State Circuits,* vol 27, no. 3, pp. 692-700.