45.2: Biologically Inspired Auditory Sensing System Interfaces on a Chip

Paul Hasler, Paul D. Smith, Rich Ellis, David Graham, and David Anderson *

Georgia Institute of Technology, Atlanta, GA 30332, phasler@ece.gatech.edu, (404) 894-2944

ABSTRACT

This paper describes our current effort creating cooperative analog/digital signal processing (CADSP) systems [1] towards auditory sensor and signal processing applications. We address resolution issues that affect the choice of signal processing algorithms arriving from an analog sensor. We discuss current analog circuit approaches towards the front-end signal processing. We discuss our current IC approaches using this technology for noise suppression, as well as our current analog signal processing front-end system for speech recognition. Experimental data is presented from circuits fabricated using a 0.5μm nwell CMOS process available through MOSIS.

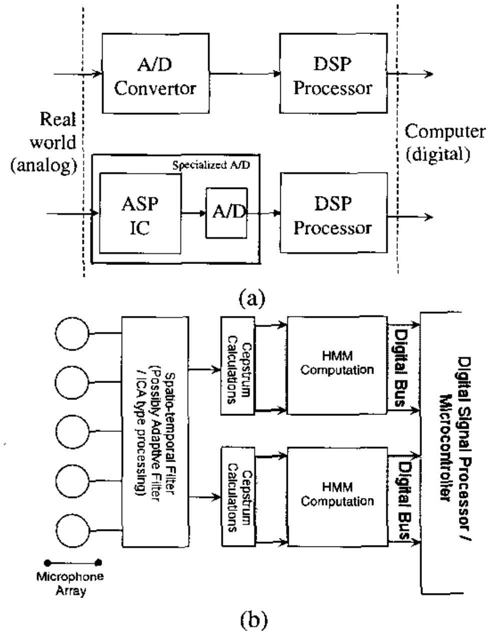

Fig. 1. Cooperative Analog–Digital Signal Processing (CADSP) applied towards Auditory Sensor processing. (a) We assume the typical model of signals coming from real-world sensors, which are analog in nature, that need to be utilized by digital computers. Our approach is to perform some of the computations using analog signal processing, requiring simplier A/D converters, and reducing the computational load of resulting digital processors. (b) Block diagram of a potential speech front-end system which takes the outputs of several microphones and could compute phonems for a higher level digital processing system.

This paper describes our current effort creating cooperative analog/digital signal processing (CADSP) systems [1] towards auditory sensor and signal processing applications. New advances in analog VLSI circuits have made it possible to perform operations that more closely reflect those done in DSP applications, or that are desired in future DSP applications. Further, analog circuits and systems can be *programmable*, reconfigurable, adaptive, and at a density comparable to digital memories (for example, 100,000+ multipliers on a single chip) Therefore, one might wonder if we have both digital and analog signal processing (DSP and ASP respectively) available, how does one choose a particular solution for a given auditory application. The question is where to partition the analog–digital boundry, as shown in Figure 1a, to enhance the overall functionality of a system by utilizing analog/digital computations in mutually beneficial way. By adding functionality to our analog systems, we enhance the capabilities of the controlling digital system, and therefore, the entire product under consideration. Further, this additional computational power allows for expansion of current DSP algorithms to incorporate more biologically inspired techniques in its algorithms.

We will discuss analog signal processing connected with incoming accoustical sensor inputs along several lines. First, we will address resolution issues that affect the choice of signal processing algorithms arriving from an analog sensor. Second, we will discuss current analog circuit approaches towards the front-end signal processing, and the relationship to modeling biological cochleas. Third, we will discuss our current IC approaches using this technology for noise suppression using gain-control algorithms expected in biological data [4]. Finally, we describe our current analog signal processing front-end system for speech recognition. Experimental data is presented from circuits fabricated using a 0.5μm nwell CMOS process available through MOSIS.

1. SIGNAL-TO-NOISE VERSUS COST

Analog signal processing is capable of several linear and nonlinear operations [11, 9, 7]. Even if analog signal processing is capable of several important functions, and is programmable, the primary question is the effective resolution of these computing systems. The related question is identifying the *cost* of computation at a particular resolu-

*This work was partially supported by grants National Science Foundation (CISE-1068549, ECS (CAREER): 0093915, ECS-9988905) and by corportate donations to the Georgia Tech Analog Consortium by Texas Instruments and Motorola, Inc.

0-7803-7454-1/02/$17.00 ©2002 IEEE

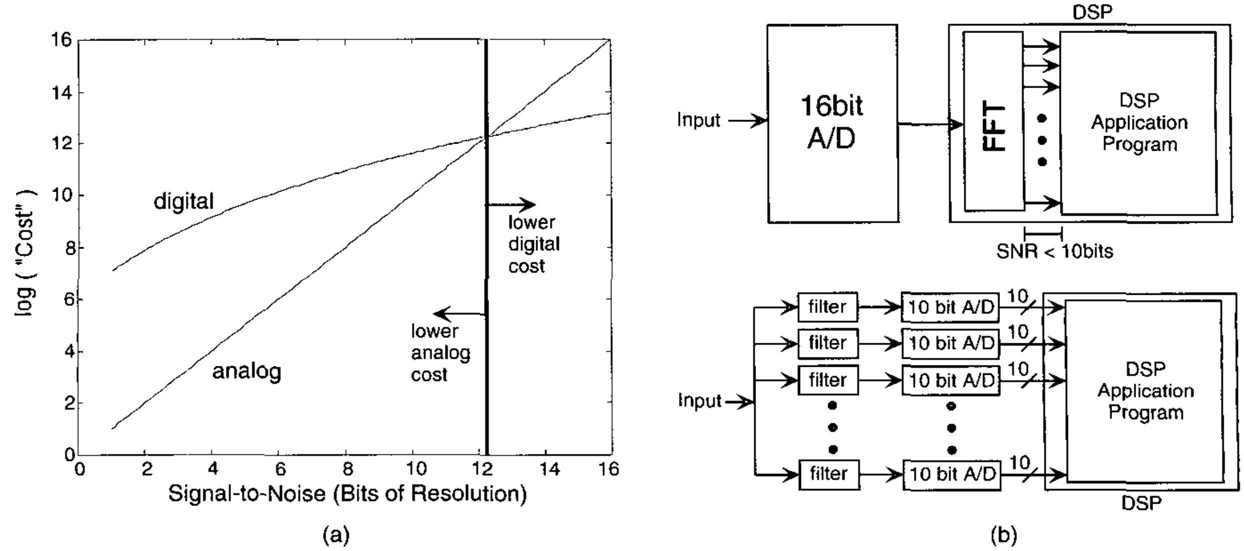

Fig. 2. Guidelines on using analog or digital signal processing depending upon required resolution (Signal–to–noise). (a) As discussed in several elsewhere [2], the computation cost of digital computation varies linearly with the required bits of resolution, while, the computation cost of digital computation varies exponentially with the required bits of resolution. This threshold is typically between 8bits to 14bits, depending upon the particular application. (b) An example comparison looking at the resulting SNR for two approaches for a particular applications: one case is a purely DSP solution, and the second case is a combined analog-digital solution. A practical example comparing using analog or digital signal processing for a particular output resolution (Signal–to–noise). One common signal processing step with incoming sensor data is taking an FFT, or equivalent Fourier based algorithm. For DSP computation, we would require a 16bit A/D converter to get some output channels at 10bit resolution. For ASP computation, we would require a bank of bandpass filters with 10bits of Signal-to-noise ratio coupled with a bank (or multiplexed) 10bit A/D converter to get the output channels at 10bit resolution. Both *analog* systems have similar design complexity. These computations are transparent (in resolution) to the engineers developing the remainder of the algorithm, and therefore tradeoffs could be made at these levels. In the end, either approach would give similar amount of information at each output channel

tion. Figure 2a shows a typical plot of signal-to-noise as bits of resolution versus the net cost [2]. One gets similar results when computing cost using a wide range of metrics involving area, power dissipation, computational delay, required tools, expenses associated with the design and manufacture, and design time. The computation cost of digital computation varies linearly with the required bits of resolution, while, the computation cost of analog computation using a single wire varies exponentially with the required bits of resolution. As a result, computation requiring less resolution than a threshold is less expensive for analog computation, and computation requiring more resolution than a threshold is less expensive for digital computation. One careful study by Sarpeskar [2], showed that analog computation has significant advantages if the resolution of the incoming information is not sufficiently high, typically 10 bits or less. These concepts argue for analog implementations for many real-time sensor signal-processing/control problems.

The key in looking at the necessary resolution for either the analog or digital signal processing parts depends heavily on the amount of the incoming information and resolution needed to represent it. Figure 2b shows an example comparing how one might apply these results. One common signal processing step with incoming sensor data is taking an FFT, or equivalent Fourier based algorithm. For DSP computation, we would require a 16 bit A/D converter to get some output channels at 10bit resolution [12]. For ASP computation, we would require a bank of band-pass filters with 10 bits of Signal-to-noise ratio coupled with a bank (or multiplexed) 10 bit A/D converter to get the output channels at 10 bit resolution. Both *analog* systems have similar design complexity, because the design complexity of a 16 bit A/D converter is exponentially harder than the design complexity of a single or multiple 10 bit A/D converters. These computations are transparent (in resolution) to the engineers developing the remainder of the algorithm, and therefore trade-offs could be made at these levels. Modeling analog signal processing resolution, typically measured in signal-to-noise ratio (SNR) must consider the particular circuit effects and continuous-time signal processing to get an accurate estimate. Simply treating analog components as fixed-point arithmetic with finite register effects will always underestimate the SNR of actual computation.

2. SIGNAL PROCESSING CIRCUITS

We commonly use several basic circuit elements for our auditory signal processing structures, Figure 3 shows these circuits, we will look at these circuits, in turn, in the following sections. Floating-gate circuit techniques enable usin these

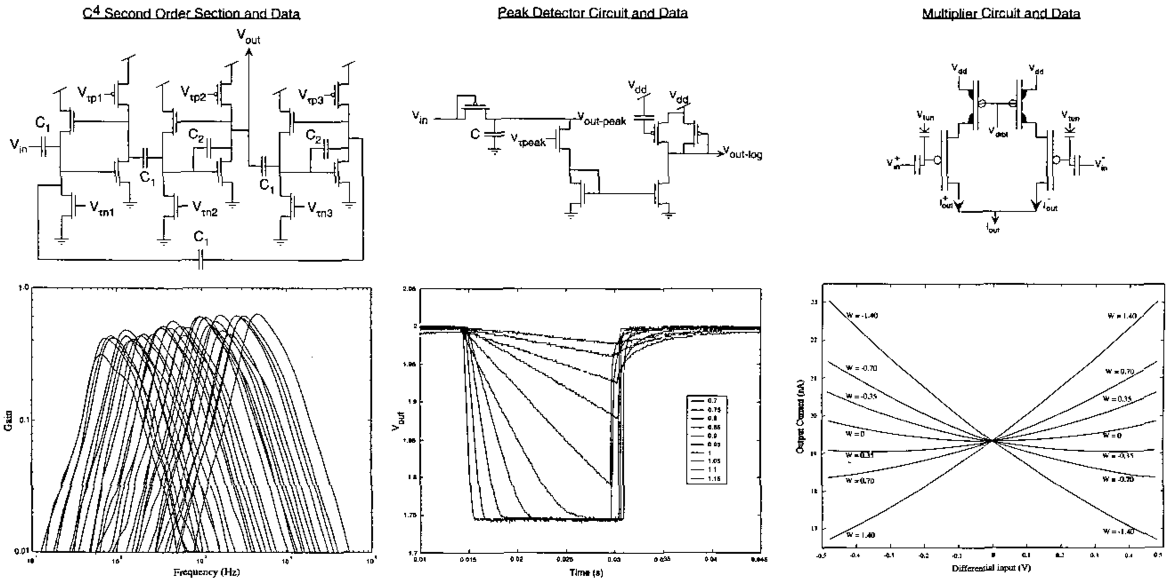

Fig. 3. Typical circuit elements used in auditory signal processing. **Second-Order section**: Floating-gate C^4 second-order-section and its corresponding frequency response. The high and low corner frequencies can be independently tuned for each filter bank. Arbitrarily programmable corner frequencies allow these filters to be spaced linearly, octave, logarithmically or any other values desired by the user. **Floating-Gate Multiplier**: Differential floating-gate multiplier structures multiply two differential signals by constant factors that are stored on the floating gate elements. **Floating-gate peak detectors**: The frequency response of the peak detector is controlled by a bias voltage which controls the gate of nFET M3. This element sets a constant resistance and the total R,C value shifts the high corner frequency. The frequency response is shown for different values of v_{tau}.

circuits for a wide range of signal processing functions [17].

2.1. Frequency Decomposition

We have been using coupled bandpass IC filter models for cochlear modeling, which are designed to be used for front-end signal processing [3]. The spectrum decomposition is done using differential C^4 second-order-section bandpass filters [3]. For simplicity only one half of the differential structure is shown in Fig. 3a. The spacing of the bandpass filters is arbitrary because each can be programmed to have a desired high-frequency corner and low-frequency corner [14]. Programming the C^4s is handled as if each filter were two floating-gate elements [13].

As a bandpass filter array, the C^4 SOS structure is not cascaded as in cochlea models [11], therefore eliminating the typical distortion or noise accumulation. In speech, particularly in noisy environments, the signal power is more evenly distributed across a broad frequency range than a simple tone, and therefore allowing for large input amplitudes with minimal output distortion (higher system signal-to-noise ratio). As a result, we typically have signal amplitudes through each filter that are 10mV to 30mV or less for input amplitudes between 0.25V and 1V, resulting in harmonic distortion through the system less than -30dB at each tap; differential circuits will further reduce these effects.

2.2. Amplitude Detection

The magnitude of each spectrum passes through a peak detector stage to produce a constant magnitude output. This magnitude is similar to taking the power spectrum density or real spectrum of an input signal. The circuit is shown in Fig. 3b. We program the peak detectors to the desired frequency response of each frequency band. The floating-gate transistor on the output provides an offset current to set the DC output voltage. Each peak detector has an individually programmable corner frequency. Because the output magnitude is continuous, this allows us to capture additional high frequency content within each band. The peak detector programming blocks are isolated similarly to the C^4s. The entire bank is treated as a single row and within that row the individual elements are accessed by column. Control circuitry on the rows and columns ensures isolation.

2.3. Weighted multiplication

Figure 3 shows our analog differential multiplier that multiplies the incoming differential voltage signal with a stored differential weight. We program the positive and negative weights by setting programmable floating-gate voltages. These values can be programmed to any arbitrary value, Their differential operation requires each pair to have a DC bias voltage.

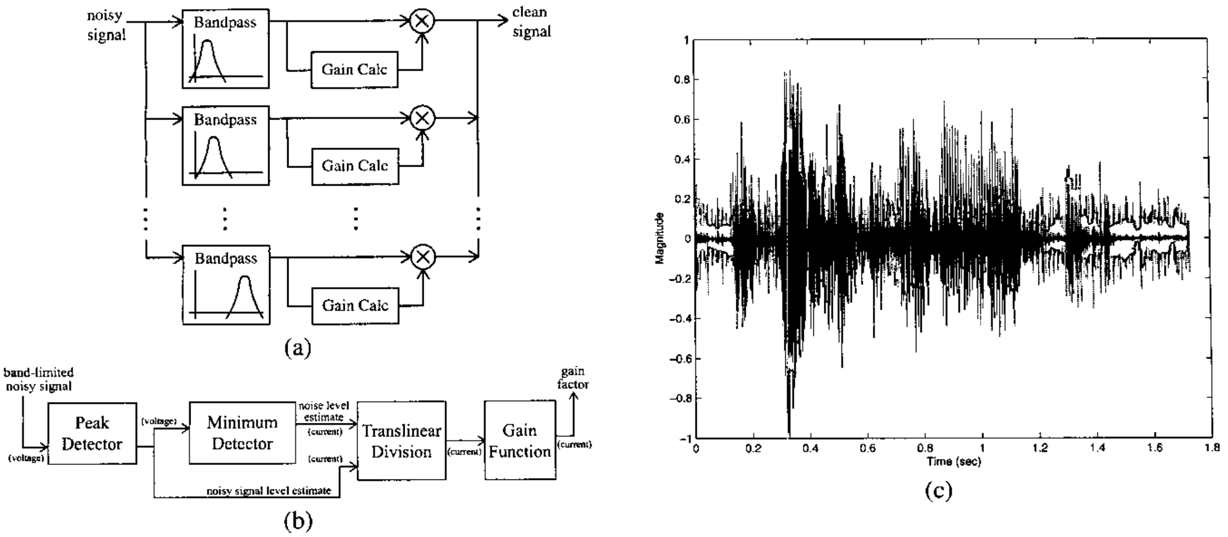

Fig. 4. Our continuous-time noise supression system. (a) The overall structure of the system. The incoming noisy signal is divided into exponentially-spaced frequency bands using C^4 second-order sections. Next, the optimal gain (gain calculation block) for each band is computed. If the band has sufficient estimated SNR, then the signal passes through with maximal gain, otherwise the gain is reduced dependent upon the the estimated SNR in that particular band. The resulting gain factor is multiplied with the band-limited noisy signal to produce a band-limited "clean" signal. Finally, the output of all of the bands are summed to reconstruct the signal with the noise components significantly reduced. (b) The details of the gain calculation block. Within each frequency band, the noisy signal envelope is estimated using a peak detector. Based on the voltage output of the peak detector, the noise level is estimated using a minimum detector operating at a slower rate than the peak detector. The currents representing the noisy signal and noise levels are input to a translinear division circuit, which outputs a current representing the estimated signal-to-noise ratio. A nonlinear function is applied to the SNR current. (c) Experimental measurements of noise suppression in one frequency band. The light gray data is the subband noisy speech input signal; the black waveform is the corresponding subband output, after the gain function has been applied.

3. ANALOG CEPSTRUM PROCESSOR FOR AS PART OF A SPEECH RECOGNITION FRONT-END

Audio signal enhancement by removing additive background noise from a corrupted noisy signal is not a new concept. However, with the prosperity of portable communication devices, it has recently received increased attention. While most noise suppression methods are focused on the processing of discrete-time sampled audio signals, we use a technique for noise suppression in the continuous-time domain. We are building a system that operates in real time and uses extremely low amounts of power. The result is a system that performs a function normally reserved for digital computation, freeing those resources for other operations in the digital domain. We present detailed motivation for these concepts elsewhere [8, 4] We present in detail the algorithm for gain calculation and the elements that perform this functionality elsewhere [4]. We present the details of the signal processing theory behind it elsewhere [8].

3.1. Structure of Suppression System

Figure 4a shows the structure of a continuous-time noise suppression system for real-time analog implementation. The goal is to design a real-time system that generates some optimal estimate of the actual signal from additive mixture of signal and noise. We assume that the additive noise is stationary over a long time period relative to the short term non-stationary patterns of normal speech. A filter bank separates the noisy signal into 32 bands that are exponentially spaced in frequency, similar to the human auditory system for frequency domain processing.

After the incoming noisy signal has been band-limited by the filter bank, a gain factor is calculated based on the the envelopes of each observed subband signals and subband noise signals. The first step in the gain calculation algorithm (shown in Fig. 4) is to estimate both the levels of the noisy signal and the noise. Because one can not accurately estimate the actual signal component of the incoming signal, so the noisy signal is accepted as a reasonable estimate. The circuit outputs both a voltage and current that are representative of the noisy signal level. The output current, I_{SNR}, can be represented by $I_{SNR} = I_{scale}(\frac{I_{signal}}{I_{noise}} - 1)$ and represents the estimated SNR. Multiplication and division operations can be performed, which we present elsewhere [4].

This gain is applied to the subband signals and the signals are combined to form the optimal estimate of the actual signal. The resulting gain factor is then multiplied with the original band-limited signal. Finally, the band-limited signals are summed to reconstruct the full-band signal estimate, without the additive noise components.

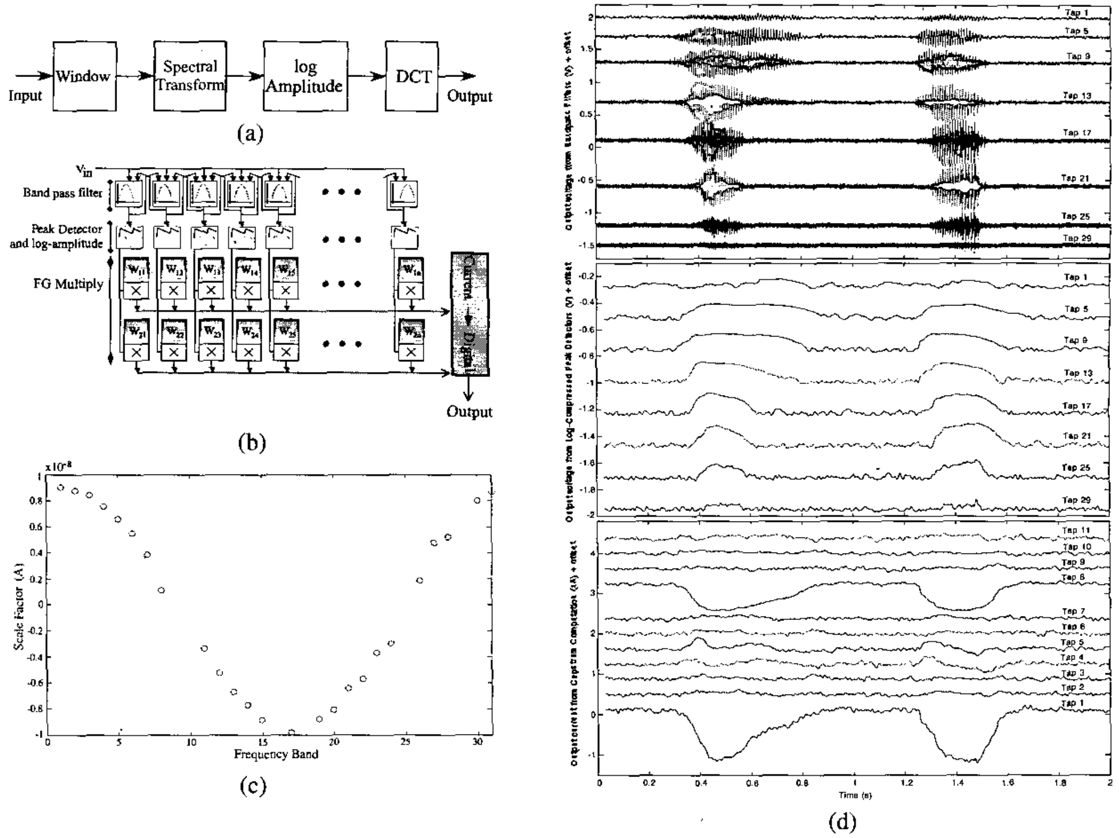

Fig. 5. A continuous-time cepstrum computation. (a) The traditional cepstrum computation as performed in digital circuitry. (b) Block diagram of a floating-gate system to perform cepstrum front-end computation for speech processing systems. The system contains 32 frequency taps that can be spaced arbitrarily by programming the corner frequencies for the bandpass filter banks. The peakdetectors provide a power spectrum of the input signal for any given time slice. (c) Programmed differential weights to the floating-gate multiplier circuits for the second row (a single cosine period) (d) Cepstrum system output. The system input is a sequence of speech using a standard speech database; each letter or phrase is separated by a short period of silence. There are 12 continuous cepstrum coefficients calculated for this section of speech and more coefficients is only a matter of chip area since the calculation is performed in parallel analog circuits. From the graph one can see the two distinct periods of speech.

3.2. System Results

The experimental results presented in this paper are from tests on individual components that have not yet been integrated into a larger system. Figure 4c shows a noisy speech signal that has been processed by the components in our system. The system is effective at adaptively reducing the amplitude of noise-only portions of the signal while leaving the desired portions relatively intact. Any noise or distortion created by the gain calculation circuits minimally affects the output signal because these circuits are not directly in the signal path. While the bandpass filters and the multipliers will inject a certain amount of noise into each frequency band, this noise will be averaged out by the summation of the signals at the output of the system. Distortion in the signal path will arise from the bandpass filters and the multiplier.

4. ANALOG SIGNAL PROCESSING FRONT END FOR SPEECH RECOGNITION

This section discusses our current work on a continuous-time mel-frequency cepstrum encoding IC using analog circuits and floating-gate computational arrays (more detail given in [5]. This approach is based upon our previous research in programmable analog filters [13, 14, 15]. Experimental data is presented from circuits fabricated on a 0.5μm nwell CMOS process available through MOSIS.

This cepstrum processor can act as the front-end for larger digital or analog speech processing systems. This cepstrum processor is one part of our current analog signal processing front-end system for speech recognition. comprised of an analog Cesptrum-like processor [5], a Vector Quantization stage [6], and a continuous-time HMM block built from programmable analog waveguide stages [7]. Early data from a related project gives confidence that this approach will improve the state of the art at a given power dissipation level [10]

4.1. Analog and Digital Mel-Cepstral Analysis of Speech Signals

The Mel-cepstrum is often computed as the first stage of a speech recognition system [16]. Implemented in the discrete domain, the Mel-cepstrum may be calculated by combining the output of the $\log|S(\omega)|$ into critical band energies and then performing the discrete cosine transform (DCT) on the sequence of critical band energies [16] (See Fig. 5a). The mel-cepstrum, as used in digital signal processing (DSP) is based on a signal sampled in time and in frequency. Figure 5b shows the block diagram for the analog cepstrum which is an approximation to either the mel-cepstrum or cepstrum (depending on the filter corner frequencies) in which frequency is sampled but time is not. The output of each filter contains information similar to the short-time Fourier transform and can likewise be assumed to represent the product of the excitation and vocal-tract within that filter band. The primary difference here is that the DSP mel-cepstrum approximates the critical band log frequency analysis of the human ear by combining DFT bands while the analog system actually performs a critical band-like analysis on the input signal. Thus higher frequency critical band energies are effectively computed using shorter basis functions than the lower frequency bands. This is more in agreement with analysis in the human auditory system and is better suited to identifying transients. We present a detailed discussion on the signal processing foundation of analog and digital Mel-Cepstrum computations elsewhere [5].

4.2. Implementation and Experimental Results for an Analog Cepstrum

The basic building block of the cepstrum begins with a continuous spectrum decomposition and amplitude detection, similar to a Discrete-Fourier Transform (DFT). The spectrum decomposition is done using differential C^4 second-order-section bandpass filters. The magnitude function (inside the log) is estimated using a peak detector rather than using the true magnitude of the complex spectrum. Finally, we compute a DCT on these results using a matrix multiply using arrays of floating-gate circuits where each row of the matrix is another DCT basis vector. Figure 5c shows the 32 programmed weight values (difference between a positive and negative weight) for a single row of multipliers programmed to a cosine function (row 2 of a DCT). Figure 5d shows experimental results from different stages of our Cepstrum computation. The system output from the analog peak detector was computed using MATLAB multiplier models that agree closely with experimental data on multipier arrays. The 14 output tap of our analog cepstrum computation closely agrees to the DSP equivalent algorithm, when starting from a set of bandpass filter elements.

5. REFERENCES

[1] P. Hasler and D. Anderson, "Cooperative Analog-Digital Signal Processing," *International Conference on Accoustics, Speech, and Signal Processing*, Orlando, May 2002.

[2] Rahul Sarpeshkar, *Efficient precise computation with noisy components: extrapolating from an electronic cochlea to the brain*, PhD thesis, California Institute of Technology, Pasadena, CA, 1997.

[3] David Graham and Paul Hasler, "C^4 SOS sections for cochlea modeling," *International Symposium on Circuits and Systems*, Phoenix, May 2002.

[4] Rich Ellis, Heejong Yoo, David Graham, Paul Hasler, and David Anderson, "A Continuous-time Speech Enhancement Front-End for Microphone Inputs," *International Symposium on Circuits and Systems*, Phoenix, May 2002.

[5] Paul Smith, Matt Kucic, Rich Ellis, and Paul Hasler, "A Floating-Gate Cepstrum IC," *International Symposium on Circuits and Systems*, Phoenix, May 2002.

[6] Paul Hasler, "Continuous-Time Feedback in Floating-Gate MOS Circuits," *IEEE Transactions on Circuits and Systems*, vol. 48, no. 1, January 2001, pp. 56 - 64.

[7] Paul Smith and Paul Hasler, "Low-power speech recognition using analog signal processing techniques," *IEEE International Conference on Accoustics, Speech, and Signal Processing*, Orlando, May 2002.

[8] H. Yoo, D.V. Anderson, and P. Hasler, "Continuous-time audio noise suppression and real-time implementation," in Proceedings of the IEEE International Confreence on Acoustics, Speech, and Signal Processing, Orlando, 2002.

[9] David Anderson and Paul Hasler, "Cooperative Analog-Digital Signal Processing," *World Multiconference on Systems, Cybernetics, and Informatics*, Orlando, 2001, pp. 496-501.

[10] Todd M. Massengill, Denise M. Wilson, Paul Hasler, and David Graham, "Emperical Comparison of Analog and Digital Auditory Perprocessing for Automatic Speech Recognition," *International Symposium on Circuits and Systems*, Phoenix, May 2002.

[11] C.A. Mead, *Analog VLSI and neural systems*, Addison-Wesley, Reading, Massachusetts, 1989.

[12] P.E. Allen and D.R. Holberg, *CMOS Analog Circuit Design*, Prentice-Hall, 2002.

[13] Matt Kucic, Paul Hasler, Jeff Dugger, and David Anderson, "Programmable and Adaptive Analog Filters using Arrays of Floating-Gate Circuits," *IEEE Advanced Research in VLSI*, Salt Lake City, UT, March 2001.

[14] Matt Kucic, AiChen Low, Paul Hasler, and Joe Neff, "A Programmable Continuous-time Floating-Gate Fourier Processor," *IEEE Transactions on Circuits and Systems II*, vol. 48, no. 1, January 2001, pp. 90-99.

[15] Matt Kucic, AiChen Low, and Paul Hasler, "A Programmable Continuous-time Analog Fourier Processor based on Floating-Gate Devices," *IEEE International Symposium on Circuits and Systems*, Geneva, Switzerland, May 2000, vol.3, pp. 351-354.

[16] John R. Deller, John G. Proakis, and john H.L. Hansen, *Discrete-time Processing of Speech Signals*, IEEE Press, New York, 2000.

[17] Paul Hasler and Bradley A. Minch, **Floating-Gate Devices, Circuits, and Systems**, in Press.