DIRECT PROCESSING OF MPEG AUDIO USING COMPANDING AND BFP TECHNIQUES

Christos Vezyrtzis, Aaron (Ari) Klein, Dan Ellis and Yannis Tsividis

Department of Electrical Engineering, Columbia University, New York

ABSTRACT

We present techniques for processing MPEG-audio encoded signals during the decoding process, using efficient fixed-point arithmetic operations. A large signal-to-quantization-noise-ratio is achieved over a large range of input levels. By taking advantage of MPEG-audio built-in properties, quantization distortion at the outputs of our systems is kept largely inaudible, even though only low-resolution fixed-point operations are used in the processing.

Index Terms— MPEG, DSP, companding

1. INTRODUCTION

The MPEG-1 audio coding standard is one of the most popular and widely used standards for efficient and perceptually lossless audio compression coding, since it achieves both high perceived audio fidelity and high compression rates. An excellent MPEG tutorial can be found in [1]. MPEG-audio uses a digital filterbank to create 32 narrowband filtered versions of a digital input signal, referred to as "subbands," each of which is downsampled by a factor of 32. The presence of a large signal in a particular subband makes noise in that subband perceptually inaudible; this phenomenon is known as "psychoacoustic masking". In MPEG-1 layers I and II, scale factors, coded as a 6-bit index corresponding to one of 64 high-precision predefined values, are used to compress the dynamic range of the subband samples (normalization). The actual value of each subband sample is given by the normalized subband sample multiplied with the corresponding scale factor; this multiplication is referred to as "denormalization".

Processing of MPEG-audio encoded signals is conventionally performed by first fully decoding the input stream and then performing the desired processing. This method, which we refer to as "classical DSP," completely ignores the features of MPEG-audio encoding. The processor must process a signal with high dynamic range, and with frequency content throughout the audio band. In [2, 3, 4, 5], among others, the problem of how to directly process the subband samples is addressed, but the processing proposed therein is done after denormalization, so the processed subband signals, though narrowband, have high dynamic range. As a result, to avoid introducing significant audible quantization distortion, these subband processors must be implemented in either very high resolution fixed-point or in floating point. For many applications, specifically those targeting low-power, battery-operated devices, it will often be very desirable to reduce the complexity of the computation required to implement the processing; using only low-resolution fixed-point operations for the processing would be very desirable. This is the case regardless of the implementation of the MPEG-1 decoder used, as this decoding is necessary, and essentially a fixed, "sunk" cost, independent of the implementation of the processing. Even if a floating-point decoder is used, great savings in dynamic power can be realized by improving the efficiency of the processing.

This work was supported in part by NSF Grant CCF 07-01766.

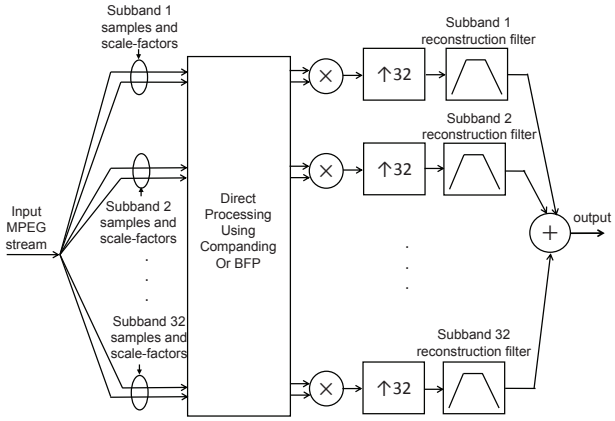

Fig. 1. Direct processing of MPEG-audio. Subband samples and corresponding scale factors are efficiently processed before denormalization by using syllabic companding or block-floating-point processors.

In this work, we process $prior$ to denormalization by using the syllabic companding (compressing-expanding) DSP technique, introduced in [6], or the block-floating-point (BFP) technique of [7]; these techniques process compressed input, along with corresponding input scale factors, and yield compressed output, along with corresponding output scale factors, provided that these scale factors correspond to the the time-domain envelope of the subband samples. The highest MPEG-audio layer for which this is the case is MPEG 1-Layer II (MP2), and this is the standard we used in this work. However, the techniques presented in this work can be similarly applied to any standard in which audio is encoded in the form of normalized time-domain subband samples and corresponding time-domain scale factors; we chose MP2 since it is used for many applications, including digital-video-broadcasting (DVB) and DVD players.

The techniques presented in this work could also be used with standards utilizing frequency-domain subband samples, such as the popular MPEG 1- Layer III (MP3) standard, but with such standards, it would be necessary to first partially decode the input stream to obtain time-domain normalized subband samples and scale factors; for the most part, such partial decoding is part of the standard decoding process, so it is not an overhead.

The system-level block diagram for the technique we present in this paper is shown in Fig. 1, where the MPEG-audio encoded signal is processed during decoding, before denormalization, taking advantage of the compressed input and scale factors provided to us by the MPEG-audio standard. As in [3, 4], the processor shown in the figure is composed of 32 "subband processors," where each subband processor is a multiple-input, single-output system fed with subband samples from all 32 subbands; the i^{th} subband processor outputs the

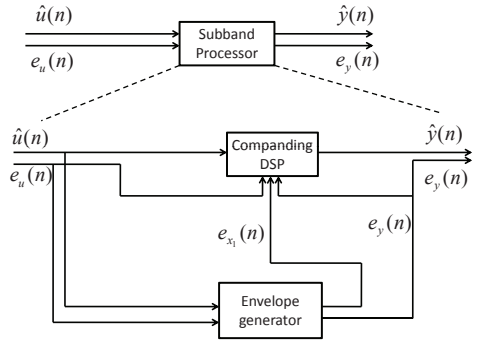

Fig. 2. A companding subband processor. For this case study, the processor block of Fig. 1 is composed of 32 identical copies of this subband processor, with the i^{th} processor taking as input only the i^{th} stream of subband samples and corresponding scale factors.

subband samples corresponding to the i^{th} subband. In contrast to [2, 3, 4, 5], the processor shown in Fig. 1 can use few bits and simple low-bit fixed-point operations. Due to the compressed dynamic range of the input, state, and output signals, the resulting output signal to quantization noise ratio (SNR) is always sufficiently high that the output quantization noise is inaudible due to the psychoacoustic masking properties exploited by the MPEG-audio reconstruction filter bank.

2. DIRECT PROCESSING OF MPEG-ENCODED SIGNALS

Due to lack of space, we present our techniques using a specific example: an all-pass digital reverberator. A non-allpass filter could theoretically filter out some artifacts caused by our technique; by using an allpass filter, we ensure that the absence of artifacts is a property of our technique, rather than an "accident" related to the type of filtering used.

2.1. Syllabic Companding Approach

We used the syllabic companding DSP technique, introduced in [6] and refined in [8], to implement a reverberator prototype, described in state-space by the following equations:

$$\begin{aligned} x_1(n+1) &= -0.8 \cdot x_L(n) + 0.2 \cdot u(n) \\ x_i(n+1) &= x_{i-1}(n), \quad 2 \le i \le L \\ y(n) &= 1.8 \cdot x_L(n) + 0.8 \cdot u(n) \end{aligned} \tag{1}$$

where $L = 2048$ and the sampling rate for the input $u(n)$, output $y(n)$, and states $x_i(n)$ of the prototype is $f_S = 44.1$ kHz. For this case, we could use the technique of [2], so that the processing is implemented by 32 identical "subband processors," with the i^{th} processor taking as input only the i^{th} stream of (denormalized) subband samples. Each of the 32 subband processors is described by (1), but with L replaced by $K = \frac{L}{32} = 64$, and with $u(n)$ and $y(n)$ now being the stream of samples at the input and output, respectively, of a particular subband processor; we will refer to such a subband processor as the "subband-prototype." Here, we would like to process samples before denormalization, so we will apply the companding DSP technique of [6] to the subband-prototype. Thus, as in [6], we introduce externally applied control signals $e_u(n)$, $e_y(n)$, and $e_{x_i}(n)$ signals, referred to as "e-controls", and normalized input, output and states $\hat{u}(n)$, $\hat{y}(n)$, and $\hat{x}_i(n)$, such that:

$$\begin{aligned} \hat{u}(n) &= \tfrac{u(n)}{e_u(n)} \\ \hat{y}(n) &= \tfrac{y(n)}{e_y(n)} \\ \hat{x}_i(n) &= \tfrac{x_i(n)}{e_{x_i}(n)}, \quad 1 \le i \le K \end{aligned} \tag{2a}$$

By substituting (2a) in (1), we find that all subband processors are identical and described by the state equations:

$$\begin{aligned} \hat{x}_1(n+1) &= \frac{-0.8 e_{x_K}(n)}{e_{x_1}(n+1)} \cdot \hat{x}_K(n) + \frac{0.2 e_u(n)}{e_{x_1}(n+1)} \cdot \hat{u}(n) \\ \hat{x}_i(n+1) &= \hat{x}_{i-1}(n), \quad 2 \le i \le K \\ \hat{y}(n) &= \frac{1.8 e_{x_K}(n)}{e_y(n)} \cdot \hat{x}_K(n) + \frac{0.8 e_u(n)}{e_y(n)} \cdot \hat{u}(n) \end{aligned} \tag{2b}$$

with $K = 64$ and $e_{x_K}(n) = e_{x_1}(n - K + 1)$. The e-controls are constrained to be integer powers of 2, so that the ratios in (2) are efficiently implemented as subtractions of (integer) base-2 logarithms, and multiplying by the ratios is efficiently implemented with arithmetic bit-shift. Information about the input envelope for each subband is provided in MPEG-audio in the form of a signal scale-factor. From this, the $e_u(n)$ control signal is generated via a lookup table (LUT). The LUT has a 14-bit input: the 8-bit normalized input sample, concatenated with its corresponding 6-bit scale-factor index. The LUT outputs a 4-bit integer corresponding to the base-2 logarithm of the lowest integer power of 2 greater than the scale-factor, and a new 8-bit compressed subband sample corresponding to this power-of-2 scale factor. The new 8-bit sample is used as $\hat{u}(n)$ in (2), while the power-of-2 scale factor is used as $e_u(n)$ in (2). As shown in [6], the remaining e-controls should be chosen to correspond, at least roughly, to the envelopes of the corresponding signals in the prototype, in order to maximize the dynamic range of the subband processor, and minimize the quantization distortion. Thus, as seen in Fig. 2, we use an "Envelope generator" block to obtain the remaining e-controls required by the companding DSP. In [6], a replica DSP was used to calculate the remaining required e-controls. We could do this here as well, using 32 low-resolution fixed-point implementations of the subband-prototype. However, implementing the replica DSPs adds significant overhead, so we have devised a more efficient technique for estimating the remaining e-controls. Our algorithm, shown in block diagram format in Fig. 3, takes advantage of the narrowband nature of the subbands, and is described in detail in the following.

When a signal $u(n)$, narrowband around a frequency ω_1, is processed with an LTI filter, one can approximate $u(n)$ with a single tone at frequency ω_1, so that the output is roughly $\tilde{y}(n) = A_1 \cdot u(n - n_1)$, where A_1 is the magnitude of the filter's transfer function at frequency ω_1, and n_1 is the group delay of the filter, rounded to the nearest integer, at frequency ω_1. Thus, the envelope of $y(n)$, $e_y(n)$, can be approximated with $A_1 \cdot e_u(n - n_1)$. Similar results hold for the filter states.

The above discussion only applies when there is no sudden change in the input, $u(n)$, since until the system resettles after the sudden change, it cannot be viewed as above. We have determined empirically that abrupt changes in $u(n)$ are indicated by changes of more than a factor of 8 between consecutive values of $e_u(n)$ in (2a). When no such change is detected, we can consider the subband signal to be narrowband. For our subband-prototypes, all input-state and input-output transfer functions are normalized such that their maxima are at 0 dB, so $A_1 = 1$. Thus, in (2), we can approximate

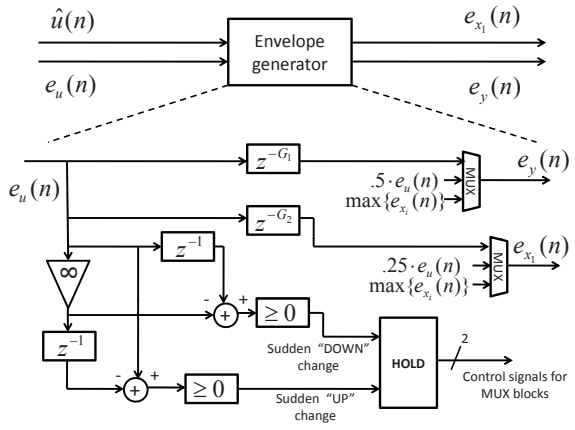

Fig. 3. Subband processor without a replica DSP.

the output envelope of the companding DSP's output, $e_y(n)$, by $e_u(n-G_1)$ and the first state's envelope, $e_{x_1}(n)$, by $e_u(n-G_2)$, where G_1 and G_2 are the corresponding group delays, rounded to the nearest integer.

The magnitude of the transfer function from the subband prototype's input, $u(n)$, to its K^{th} state, $x_K(n)$, was simulated to range from -15 dB to 0 dB. Thus, when there have been no recent abrupt input envelope changes, $e_u(n)$ and $e_{x_K}(n)$ differ by at most one order of magnitude. When there are abrupt input envelope changes, $e_u(n)$ will temporarily be either much larger or much smaller than $e_{x_K}(n)$. In the subband prototypes, given by (1), but with L replaced by $K = \frac{L}{32}$, we see that $x_1(n+1)$ and $y(n)$ are both composed of two components: one depending on the input, $u(n)$, and the other on the K^{th} state, $x_K(n)$. When there is an abrupt input envelope change, one or the other component will dominate in determing the envelopes of $x_1(n+1)$ and $y(n)$, allowing us to use simple approximations for these envelopes. Specifically, for sudden increases in $e_u(n)$, $e_u(n)$ temporarily becomes significantly larger than $e_{x_K}(n)$, so in (2), we can approximate $e_y(n)$ as $.8 \cdot e_u(n)$, and $e_{x_1}(n)$ as $.2 \cdot e_u(n)$. Since we always use exact integer powers of 2 for $e_u(n)$, and we also want $e_y(n)$ and $e_{x_1}(n)$ to be exact integer powers of 2, we further approximate $e_y(n)$ as $.5 \cdot e_u(n)$ and $e_{x_1}(n)$ as $.25 \cdot e_u(n)$. This also results in a far simpler implementation, as $e_y(n)$ and $e_{x_1}(n)$ can be easily computed from $e_u(n)$ by simply subtracting 1 or 2, respectively, from the integer power of 2 stored for $e_u(n)$. These assignments must be carried for at least G_1 samples, after which the envelopes can again be estimated via the group delays, until a new abrupt input jump is detected. Similarly, for sudden decreases in $e_u(n)$, both $e_y(n)$ and $e_{x_1}(n)$ can be approximated as $\max\{e_{x_i}(n)\}$ until a new abrupt input jump is detected.

The above described functionality is shown in Fig. 3. Even though only low-resolution fixed-point operations, along with a minimal amount of extra hardware, are used in the implementation described above, its performance will be seen to yield high output SNR over a large input dynamic range, and excellent perceived audio quality.

2.2. Block Floating Point Approach

Another way to process samples before denormalization is to apply a block floating point (BFP) technique, based on [7], but modified as in [8] to allow for input and output compression in addition to state-variable compression. We introduce scaling signals $g_u(n)$, $g_y(n)$, and $g_i(n)$, referred to as "g-controls", and normalized input, output and states $\hat{u}(n)$, $\hat{y}(n)$, and $\hat{x}_i(n)$, such that:

$$\begin{array}{rcll} \hat{u}(n) & = & g_u(n) \cdot u(n) & \\ \hat{y}(n) & = & g_y(n) \cdot y(n) & \\ \hat{x}_i(n) & = & g_i(n) \cdot x_i(n), & 1 \le i \le K \end{array} \tag{3}$$

The derivation of the BFP technique is not included here due to space constraints, but it is in [7, 8]; here, we apply this technique to the subband prototypes of the previous subsection. In general, the BFP technique of [7] obtains an intermediate "partially compressed" state vector, $\tilde{x}(n)$, and output, $\tilde{y}(n)$, from the compressed input, $\hat{u}(n)$, the compressed state vector, $\hat{x}(n)$, and the g-controls. For the subband prototypes, this is accomplished as follows:

$$\begin{aligned} \tilde{x}_1(n+1) &= -0.8\,\tfrac{g_1(n)}{g_K(n)}\,\hat{x}_K(n) + 0.2\,\tfrac{g_1(n)}{g_u(n)}\,\hat{u}(n) \\ \tilde{y}(n) &= 1.8\,\tfrac{g_y(n-1)}{g_K(n)}\,\hat{x}_K(n) + 0.8\,\tfrac{g_y(n-1)}{g_u(n)}\,\hat{u}(n) \end{aligned} \tag{4a}$$

where $K = 64$. Eqn. (4a) is not a standard state space, as it relates $\tilde{x}(n+1)$ to $\hat{x}(n)$. As in the previous subsection, we use a LUT to convert from the MPEG normalized subband samples and scale factors to scale factors that are integer powers of 2, along with the corresponding normalized subband samples. These are used as $g_u(n)$ and $\hat{u}(n)$ in (4a). As in [7], the remaining g-controls are derived recursively by introducing "p-controls." Since for our case study, $g_K(n) = g_1(n-K+1)$, we only need to derive $g_1(n)$ and $g_y(n-1)$, so we only need $p_1(n)$ and $p_y(n)$. The former is obtained from $\tilde{x}_1(n)$:

$$p_1(n) = \begin{cases} \frac{1}{4} & \alpha 2^N < |\tilde{x}_1(n)| \\ \frac{1}{2} & \alpha 2^{N-1} < |\tilde{x}_1(n)| \le \alpha 2^N \\ 1 & \alpha 2^{N-2} < |\tilde{x}_1(n)| \le \alpha 2^{N-1} \\ 2 & |\tilde{x}_1(n)| \le \alpha 2^{N-2} \end{cases} \tag{4b}$$

where N is the number of bits used for compressed states, input, and output, and α is a constant "safety factor" set to be slightly less than unity. Similarly, $p_y(n)$ is obtained by an equation identical to (4b), but with $\tilde{y}(n)$ replacing $\tilde{x}_1(n)$. The p-controls are used to recursively obtain g-controls:

$$\begin{array}{rcl} g_1(n) & = & p_1(n) \cdot g_1(n-1) \\ g_y(n) & = & p_y(n) \cdot g_y(n-1) \end{array} \tag{4c}$$

The p-controls are also used to obtain the fully compressed $\hat{x}_1(n)$ and $\hat{y}(n)$ from the partially compressed $\tilde{x}_1(n)$ and $\tilde{y}(n)$:

$$\begin{array}{rcl} \hat{x}_1(n) & = & p_1(n) \cdot \tilde{x}_1(n) \\ \hat{y}(n) & = & p_y(n) \cdot \tilde{y}(n) \end{array} \tag{4d}$$

The K^{th} state is simply obtained as: $\hat{x}_K(n) = \hat{x}_1(n-K+1)$.

The $p(n)$ and $g(n)$ signals in (4) are all integer powers of 2, and they are stored as those powers. Thus, although (4) contains ratios and products, these are implemented as additions and subtractions of powers of 2, and bitshifts by these powers. No division is performed, and the only multipliers are the coefficient multipliers.

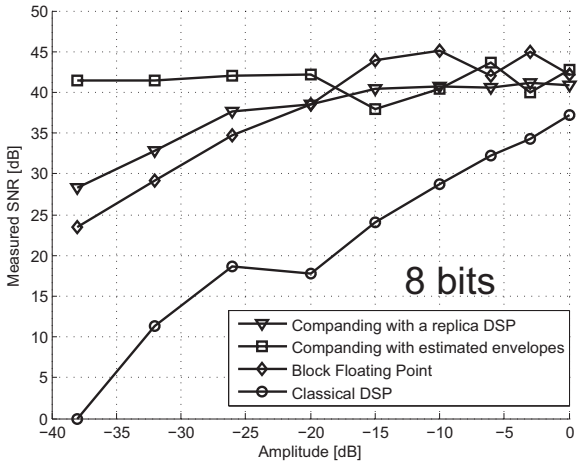

Fig. 4. SNR comparison for a 500 Hz input tone.

3. MEASUREMENTS

The systems discussed were implemented and simulated in Matlab/Simulink with both pure-tone and speech inputs. Fig. 4 shows the SNR for all systems when their inputs are a 500 Hz encoded tone. All systems operate in 8-bit, fixed-point arithmetic, meaning that they use 8-bit registers and multipliers, and 16-bit accumulators, adders, subtracters and shifters. As shown, the SNR at the output of the companding and BFP systems is very close to the full-scale SNR over a large input dynamic range (DR); such is not the case for the 8-bit classical system. Thus, for a fixed target SNR, the companding and BFP systems can provide a much larger DR than a classical system using the same number of bits.

Fig. 4 alone does not fully determine the performance of the systems when subject to signals of varying envelopes; such performance will depend on both the SNRs in Fig. 4 and the accuracy of the envelope calculations. As such, we also fed the presented systems with audio signals, including speech signals. Sample audio clips have been posted on a web site [9]; the use of only 8 bits allows us to audibly demonstrate the noise reduction achievable using the techniques presented, which would not be the case if more bits had been used, as the noise in all cases would then be largely imperceptible, due to limitations of the medium. Listening tests confirmed that the quantization noise of the companding and BFP systems is significantly reduced relative to that of the classical DSP, due to the higher SNRs shown in Fig. 4 and the masking properties of the MPEG-audio reconstruction filter bank.

4. CONCLUSION

We have applied both syllabic companding and BFP to directly process MPEG-audio encoded signals before denormalization. Our proposed technique takes advantage of the compressed subband samples and scale factors already provided in the MPEG-audio encoded signal. The compressed input and scale factors are used as inputs to low-resolution syllabic companding or BFP processors, and processing is thus accomplished with low-resolution fixed point arithmetic. For the number of bits used, relatively large SNR is achieved over a large input dynamic range. The companding nature of the processing ensures that significant quantization distortion is only present in subbands that also simultaneously contain significant signal. This property, combined with the psychoacoustical masking properties of the MPEG-audio reconstruction filter bank, ensures that even though our processor uses low-resolution fixed-point arithmetic, the resulting quantization distortion at the processor output is significantly reduced relative to that of the classical DSP. Thus, this work leverages the well-known psychoacoustical masking properties of the MPEG-audio reconstruction filter bank, already present in any standard MPEG encoder/decoder, to acheive relatively large SNR with relatively low-resolution fixed-point direct processing of the subband samples.

We chose to use 8-bit systems to make the noise reduction apparent in both our measurements and our audio clips, so that there is an audible difference between the clips posted in [9]; more bits would be used in commercial applications to further reduce the resulting quantization noise and render it perceptually inaudible. Our results imply that by using companding or BFP in lieu of classical processing, fewer bits are required to achieve inaudible quantization noise.

Due to space constraints, the only system we discussed in this paper was a reverberator with a delay given by a multiple of 32. Deriving the corresponding set of 32 subband processors was straightforward, as in [2]. Our proposed technique, though, is far more general, and can be applied to any set of subband processor prototypes. For example, we have applied our proposed technique to a linear-phase, finite impulse response filter, using the method in [3] to derive the subband processor prototypes. For the reverberator case study presented in this work, the computational complexity of the reverberator is small relative to that of the MPEG decoder, so our presented technique only grants efficiency improvements for a small fraction of the total processing required. Our proposed technique would, in a relative sense, be more beneficial for cases where the digital filtering would require large computational complexity, such as for high-order FIR filtering. In this work, we chose to use an all-pass system for the reasons discussed at the beginning of Section 2.

5. REFERENCES

[1] D. Pan, "A tutorial on MPEG/Audio compression," in *IEEE Mult. Med.*, Summer 1995, pp. 60–74.

[2] S.N. Levine, "Effects processing on audio subband data," in *ICMC Proceeding*, 1996.

[3] C.A. Lanciani and R.W. Schafer, "Subband-domain filtering of MPEG audio signals," in *Proc. 1999 IEEE ICASSP*, 1999.

[4] Christopher A. Lanciani, *Compressed-Domain Processing of MPEG Audio Signals*, Ph.D. thesis, Georgia Institute of Technology, Atlanta, GA, 1999.

[5] A.B. Touimi, "A generic framework for filtering in subband-domain," in *Proc. of the Ninth DSP Workshop (DSP2000)*, 2000.

[6] A. Klein and Y. Tsividis, "Companding digital signal processors," in *Proc. 2006 IEEE ICASSP*, May 2006, vol. 3, pp. III–700 – III–703.

[7] S. Sridharan, "Implementation of state-space digital filters using block-floating point arithmetic," in *Proc. 1987 IEEE ICASSP*, 1987, pp. 908–911.

[8] A. Klein and Y. Tsividis, "Externally linear time invariant digital signal processors," *IEEE Trans. on Signal Processing*, vol. 58, pp. 4897–4909, Sept 2010.

[9] "http://www.ee.columbia.edu/companding/spconf2011.htm," .