SPARSITY-BASED REAL-TIME AUDIO RESTORATION

P. Maechler[1], *D. Bellasi*[1], *A. Burg*[2], *N. Felber*[1], *H. Kaeslin*[1], *and C. Studer*[3]

[1]ETH Zürich, 8092 Zürich, Switzerland; e-mail: {maechler, felber, kaeslin}@iis.ee.ethz.ch
[2]EPFL, 1015 Lausanne, Switzerland; e-mail: andreas.burg@epfl.ch
[3]Rice University, Houston, TX 77005, USA; e-mail: studer@rice.edu

ABSTRACT

We demonstrate real-time audio signal restoration using techniques from compressive sensing (CS) and sparse signal recovery. In particular, we show an FPGA prototype implementation of the approximate message passing (AMP) algorithm, which removes clicks and pops from corrupted audio signals.

1. INTRODUCTION

Compressive sensing (CS) postulates to sample and reconstruct sparse signals using fewer measurements than the Nyquist rate suggests [1]. Since many natural or man-made signals exhibit sparse representations in certain bases, CS has the potential to lower the costs of sampling in a variety of applications. Moreover, the theory of CS and sparse signal recovery can be extended to the restoration of corrupted signals, images, and videos [2]. Unfortunately, the sparse signal recovery algorithms required to reconstruct signals from compressive measurements typically exhibit very high computational complexity, which renders the development of corresponding high-throughput hardware implementations a challenging task. Therefore, applications requiring the real-time recovery of sparse signals necessitate the design of dedicated and high-performance hardware implementations.

Contributions: We demonstrate the capabilities of approximate message passing (AMP) [3], an efficient and high-performance first-order sparse signal recovery algorithm, for the real-time restoration of audio signals corrupted by clicks/pops (as it is typical for, e.g, old phonograph recordings) on an off-the-shelf FPGA prototyping board.

2. SIGNAL RESTORATION

As shown in [2], the principles of CS and sparse signal recovery can be used to recover signals that are corrupted by, e.g., impulse noise, clicks/pops, or saturation artifacts. Consider a corrupted signal $\mathbf{z} \in \mathbb{R}^M$, which can be modelled as [2]

$$\mathbf{z} = \mathbf{A}\mathbf{a} + \mathbf{B}\mathbf{b} + \mathbf{n}. \tag{1}$$

The authors would like to thank R. G. Baraniuk, H. Bölcskei, and A. Maleki for inspiring discussions. The work of C. Studer was supported by the Swiss National Science Foundation (SNSF) under Grant PA00P2-134155.

Here, the matrix $\mathbf{A} \in \mathbb{R}^{M\times N_a}$ is used to sparsify the signal $\mathbf{s} = \mathbf{A}\mathbf{a}$ to be restored, the matrix $\mathbf{B} \in \mathbb{R}^{M\times N_b}$ is used to sparsify the (unwanted) corruptions (e.g., clicks/pops), and $\mathbf{n} \in \mathbb{R}^M$ is additive noise, which models small errors that cannot be sparsified in either basis.

Sparsity-based signal restoration: Sparsity-based signal restoration amounts to recovering the sparse vector $\mathbf{x} = [\hat{\mathbf{a}}\,; \hat{\mathbf{b}}]$ from $\mathbf{z}$ with the aid of a sparse signal recovery algorithm, followed by computing the (uncorrupted) signal estimate $\hat{\mathbf{s}} = \mathbf{A}\hat{\mathbf{a}}$. For the restoration to succeed, the matrices $\mathbf{A}$ and $\mathbf{B}$ must i) sparsify the signal $\mathbf{s}$ and corruptions $\mathbf{e}$, and ii) be incoherent. For example, a pair of matrices for audio-signal restoration satisfying these properties corresponds to $\mathbf{A}$ being the $M \times M$ discrete cosine transform (DCT) matrix to sparsify audio signals, and $\mathbf{B}$ being the identity basis to sparsify clicks/pops and saturation artifacts (see [2] for the details). An example of an old phonograph recording and the corresponding restored signal is shown in Fig. 1.

Approximate message passing (AMP): AMP [3] is an iterative sparse-signal recovery algorithm that exhibits comparatively low computational complexity and requires low arithmetic precision. Moreover, AMP was shown to provide excellent recovery performance for a large number of applications (see, e.g., [4]). All these properties render AMP suitable for the efficient implementation in hardware. The AMP algorithm can be summarized as follows. After initializing the algorithm with $\mathbf{r}^0 = \mathbf{z}$, $\mathbf{x}^0 = \mathbf{0}$, the residual error vector $\mathbf{r}^i$ is update iteratively (for $i = 1, \ldots, I_{\text{max}}$) according to

$$\begin{aligned}\mathbf{x}^i &= \eta_\theta(\mathbf{x}^{i-1} + \mathbf{D}^T\mathbf{r}^{i-1})\\ \mathbf{r}^i &= \mathbf{z} - \mathbf{D}\mathbf{x}^i + b\mathbf{r}^{i-1}.\end{aligned}$$

Here, $\eta_\theta(\mathbf{x}) = \operatorname{sign}(\mathbf{x})\max\{|\mathbf{x}| - \theta, 0\}$ is an element-wise soft thresholding operator, with a threshold $\theta > 0$ chosen according to the root mean squared error of the residual [4], and b is given by $b = 1/M\,\|\mathbf{x}^i\|_0$, where $\|\mathbf{x}^i\|_0$ represents the number of non-zero coefficients in the vector $\mathbf{x}^i$.

3. FPGA PROTOTYPE IMPLEMENTATION

In order to restore a corrupted 16 bit stereo audio signal at 44.1 ksample/s in real time, we developed an efficient FPGA

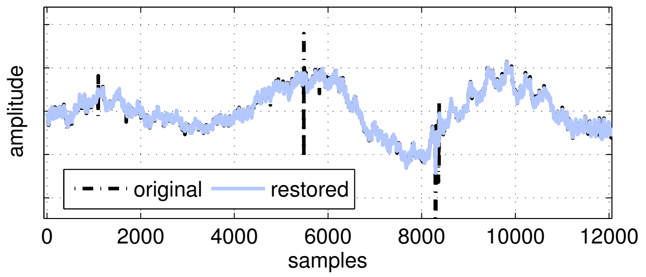

Fig. 1. Example of an old phonograph recording before and after restoration on the FPGA prototyping board.

implementation of the AMP algorithm. Signal restoration is performed using the DCT–identity pair on (independent but overlapping) blocks of length $M = 512$, which was found to deliver good restoration quality at reasonable hardware costs. The implementation performs a maximum of $I_{\text{max}} = 20$ AMP iterations, to ensure good convergence properties.

DCT-based AMP architecture: The main computational complexity of AMP lies in the matrix-vector multiplications with $\mathbf{D}$ and $\mathbf{D}^T$ required in each iteration. For the targeted audio restoration application, however, the matrix $\mathbf{D}$ has a fast transform, i.e., multiplying a vector with $\mathbf{D}$ and $\mathbf{D}^T$ can be carried out very efficiently using a fast DCT (FCT) and its inverse (IFCT) compared to a straightforward implementation using explicit matrix-vector multiplications.

The main unit of the developed AMP architecture (shown in Fig. 2) performs the FCT/IFCT. In order to arrive at an efficient (in terms of hardware resources and power consumption) FCT/IFCT architecture, we decided to use the algorithm proposed in [5], which builds upon an $M/2$-point fast (inverse) Fourier transform (FFT). This algorithm first transforms the input data into a real-valued M-point (inverse) FFT problem, which can then be computed efficiently using a complex-valued $M/2$-point FFT. This FFT-based approach enables us to take advantage of the regular structure and low complexity of state-of-the-art FFT architectures.

Other specialized hardware units are used to compute square roots (RMSE), to update the residual $\mathbf{r}$ (R-CALC), to perform the thresholding operation (TRSH), and to calculate the parameter b (L0). Please refer to [4] for more details.

Stereo audio interface: The FPGA prototype implementation includes an AC'97 audio interface to acquire stereo audio signals at 16 bit 44.1 ksample/s from an analog line-in port and to feed the signals to the AMP unit, where the main restoration task is performed. Subsequently, the restored audio signals are streamed back to the AC'97 codec for digital-to-analog conversion.

As the developed AMP-architecture performs signal restoration in a block-wise manner, input and output buffers for both channels are required. To avoid unwanted boundary artifacts, these blocks are windowed (using a triangular window) and overlapped by 8 samples.

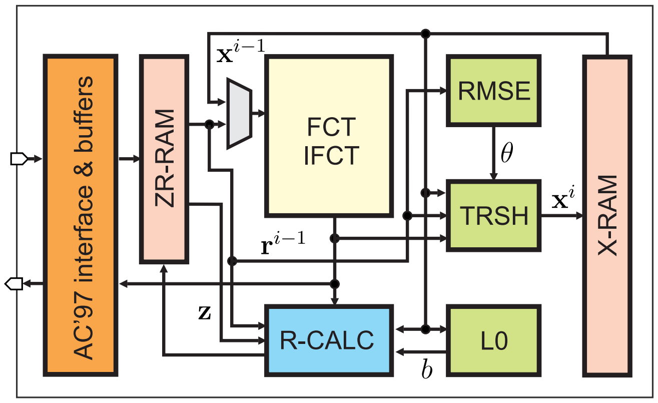

Fig. 2. High-level block diagram of the AC'97 audio interface and the AMP architecture for audio signal restoration.

Implementation results and demonstration: The developed AMP signal restoration architecture and the audio interface were mapped to a low-cost Xilinx Spartan-6 XC6SLX45-3 FPGA on a Digilent Atlys prototyping board. The final prototype runs at 35.4 MHz and occupies only 1200 logic slices, 15 DSP slices, and 11 block RAMs. Note that the economic size of the AMP implementation would allow for real-time processing on an even smaller XC6SLX9 FPGA.

4. CONCLUSIONS

We have developed the first real-time implementation of a sparse signal recovery algorithm for the restoration of audio signals corrupted by, e.g., clicks/pops or saturation artifacts. The implemented approximate message passing (AMP) algorithm enables the fast restoration of audio signals at low hardware requirements, thanks to using a low-area fast cosine transform (FCT)-based VLSI architecture. The proposed FPGA demonstrator shows that clicks/pops and saturation artifacts from, e.g., old phonograph recordings, can be mitigated in real-time at very good perceived audio quality.

5. REFERENCES

[1] E. Candès and M. Wakin, "An introduction to compressive sampling," *IEEE Signal Process. Mag.*, vol. 25, no. 2, pp. 21–30, March 2008.

[2] C. Studer and R. G. Baraniuk, "Stable restoration and separation of approximately sparse signals," *submitted to Appl. Comput. Harmon. Anal., ArXiv preprint: 1107.0420*, 2011.

[3] D. Donoho, A. Maleki, and A. Montanari, "Message-passing algorithms for compressed sensing," *Proc. of the National Academy of Sciences*, vol. 106, no. 45, pp. 18 914–18 919, Sept. 2009.

[4] P. Maechler *et al.*, "VLSI implementation of approximate message passing for signal restoration and compressive sensing," *IEEE J. on Emerging and Sel. Topics in Circuits and Systems*, vol. 2, no. 3, 2012.

[5] J. Makhoul, "A fast cosine transform in one and two dimensions," *IEEE Trans. Acoust., Speech, Signal Process.*, vol. 28, no. 1, pp. 27–34, Feb. 1980.