AN FPGA-BASED ELECTRONIC COCHLEA WITH DUAL FIXED-POINT ARITHMETIC

C. K. Wong and Philip H. W. Leong

Department of Computer Science and Engineering,
The Chinese University of Hong Kong, Shatin, Hong Kong
kitwong@cse.cuhk.edu.hk, phwl@cse.cuhk.edu.hk

ABSTRACT

An improved FPGA implementation of an electronic cochlea filter is presented. We show that by using decimation, the computations of the electronic cochlea can be reduced. Furthermore, employing dual fixed-point arithmetic, gives a significant improvement in signal to noise ratio. A sequential architecture is described which employs pipelined infinite impulse response filter stages. The accuracy, performance and resource utilisation of a number of different implementations are compared.

1. INTRODUCTION

The human cochlea is a transducer which converts mechanical vibrations from the middle ear into neural electrical discharges, and additionally provides spatial separation of frequency information in a manner similar to that of a spectrum analyzer. It serves as the front-end signal processing for all functions of the auditory nervous system such as auditory localization, pitch detection, and speech recognition.

Although it is possible to simulate cochlea models in software, hardware implementations may have orders of magnitude of improvement in performance. Hardware implementations are also attractive when the target applications are on embedded devices in which power efficiency and small footprint are design considerations.

In human hearing, the eardrum vibrates in response to changes in sound pressure level. The vibrations are conducted to the oval window. The fluid-filled basilar membrane varies in stiffness along its length, the frequency response decreasing along its length from the base at the oval window to the apex. Hair cells along the basilar membrane are disturbed by the fluid motion, triggering neural responses which are sent to the higher levels of the auditory system [1], [2].

The electronic cochlea, first proposed by Lyon and Mead, is a cascaded series of biquadratic filter sections with exponentially decreasing cutoff frequencies (as shown in Fig. 1). Many audio signal processing systems such as pitch detection [3], spatial localization [4], a computer peripheral [5], amplitude modulation detection [6], correlation [7] and speech recognition [8] have successfully used the electronic cochlea model.

To the best of our knowledge, the only reported FPGA-based electronic cochlea implementation was a module generator that used distributed arithmetic to implement the biquadratic filters [9]. Using this module generator, designs with different numbers of inputs, filter coefficients and precision could be generated.

In this paper, we present an improved design for an electronic cochlea filter which employs decimation to reduce computation, and dual fixed-point arithmetic to improve dynamic range. The contributions of this work are:

- We propose using decimation to avoid redundant filter computations in the low frequency sections of the Lyon and Mead model.
- We compare a standard fixed-point implementation with one using dual fixed-point (DFX) arithmetic [10] which employs a single bit exponent to select between two different fixed-point representations.
- We present novel architectures for both pipelined and sequential implementations which use the new ideas introduced.

The resulting electronic cochlea implementations have significantly improved performance and accuracy compared with previous work.

The paper is organized as follows. In section 2, background describing the application of decimation to the electronic cochlea is presented. In section 3, architectures for the implementation of the electronic cochlea are introduced. A brief introduction to dual fixed-point arithmetic and the architecture of a dual fixed-point second-order IIR filter is given in section 4. Results are presented in section 5 and conclusions are drawn in section 6.

2. DECIMATION

In order to avoid aliasing, the sampling frequency of the electronic cochlea must be 2 times larger than the highest frequency content of the input signal.

$$B < fs / 2 \tag{1}$$

where B is bandwidth and fs is sampling frequency.

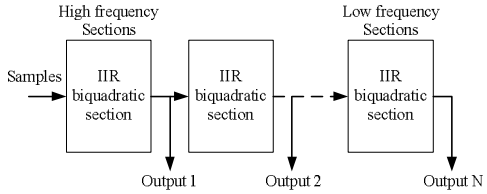

Fig. 1. Cascaded IIR biquadratic sections used in the Lyon and Mead cochlea model.

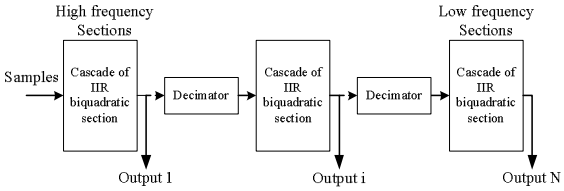

Fig. 2. Decimated electronic cochlea.

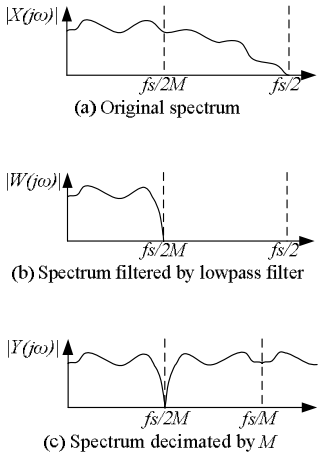

Fig. 3. Spectra for aliasing problem of sample rate reduction by a factor of *M*.

The signals in the low frequency sections of the cascaded series filters are of lower bandwidth than those in earlier sections since each IIR filter in the cascade has a low pass transfer function. Thus a lower sampling rate can be tolerated in later sections. Traditionally, implementations of the electronic cochlea have operated at a single sampling frequency, and data is processed at a higher sampling rate than necessary.

In order to reduce the computation rate of the low frequency sections, decimation can be used (as shown in Fig. 2). After decimation, the sampling frequency is reduced and the computation rate of subsequent filters is also reduced. For a parallel, pipelined implementation, a multi-rate system with high sampling frequencies for the sections near the based and lower operating frequencies at the apex can be employed. For a sequential implementation, decimation can be implemented by time-sharing IIR filters in such a way that computations for the high frequency contents are more frequent than for the low frequency sections.

2.1. Aliasing

To avoid aliasing, it is necessary to first filter earlier stages with a lowpass filter which approximates the ideal characteristic:

$$H(e^{j\omega}) = \begin{cases} 1, & |\omega| \le \pi / M \\ 0, & otherwise \end{cases} \qquad (2)$$

For the decimation of the signal *x(n)* by a factor of *M* (Fig. 3a), *x(n)* is first filtered by a lowpass filter with cutoff frequency of *fs/2M* (Fig. 3b). After filtering, the sample rate reduction is implemented by forming the new signal *y(m)* by extracting every M^{th} sample of the filtered output (Fig. 3c). If the frequency response of the lowpass filter is not sharp enough, aliasing will occur in which signals of frequency higher than *fs/2M* would corrupt the low frequency components [11].

As the Lyon and Mead cochlea model is composed of a series of lowpass filters with exponentially decreasing cutoff frequencies, they can also act as the anti-aliasing filter for the decimator. The remaining task for decimation is to determine suitable positions along the filter cascade to perform decimation.

In this work we simulate the cochlea filter with noise inputs to determine the decimation position. Each stage is examined in sequence and if its signal power at frequency $fs/2^n$ is smaller than the user-defined cutoff value in dB, it is decimated by a factor of 2^{n-1}. Fig. 4 shows the relationship between cutoff value and the normalized computation required.

2.2. Coefficient Modification after Decimation

After decimation, the filter coefficients need to be changed for the new operating frequency. This was done by regenerating the filter coefficients at the decimated sampling rate *fs/M*.

3. SYSTEM ARCHITECTURE

Fig. 5 shows the basic architecture of a second-order IIR filter which implements the filter with transfer function:

$$H(z) = \frac{b_0 + b_1 z^{-1} + b_2 z^{-2}}{1 - a_1 z^{-1} - a_2 z^{-2}} \qquad (3)$$

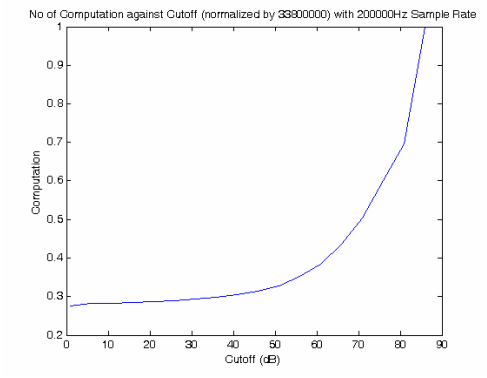

Fig. 4. Normalized computation rate against cutoff value of a system with sampling frequency of 200 kHz (169 stages).

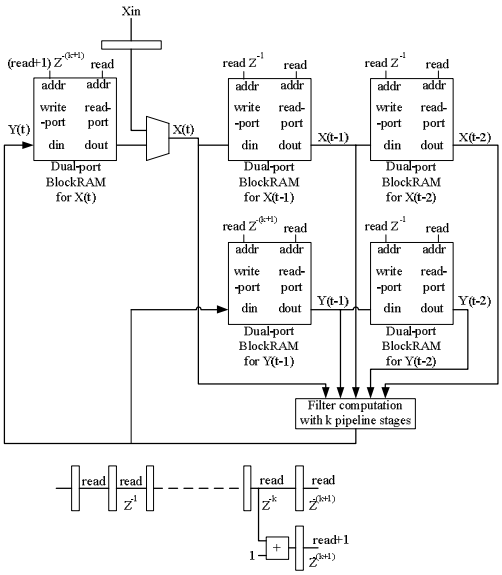

Fig. 6. Block diagram of the fully pipelined sequential processing electronic cochlea

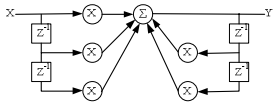

Fig. 5. The architecture of a second-order Infinite Impulse Response Filter.

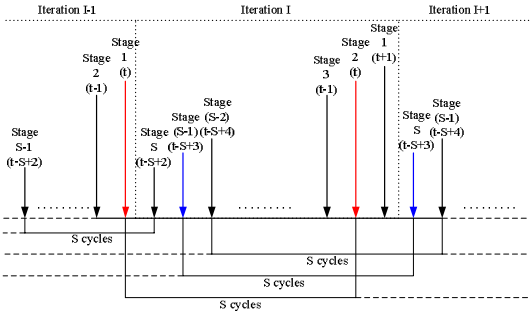

Fig. 7. The interleaved schedule for the pipelined electronic cochlea

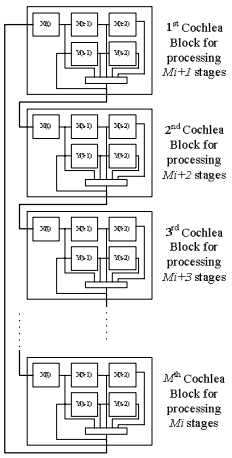

Fig. 8. Multiple Sequential Cores

The parallel architecture of Fig. 1 can be used only if the FPGA device has sufficient resources (multipliers in particular) to perform all the operations in parallel. For a cochlea system with S stages, $5 \bullet S$ multipliers are required for a parallel implementation.

If insufficient resources are available, or if maximum throughput is not the priority, a sequential approach can be used. For such a design, only 5 multipliers are required. Fig. 6 shows the architecture of a pipelined sequential processing electronic cochlea which uses 5 dual-port block RAMs for storing intermediate signals. As there are data dependencies between successive stages (the input X of time t of stage n is the output Y of time t of stage $n-1$), there would be $k-1$ idle cycles between the signal computations of time sample t in stage $n-1$ and stage n if the IIR filter is fully pipelined and has a latency of k.

Fig. 7 shows an interleaving scheme in the sequential processing electronic cochlea. It expands the time between computations of stage $n-1$ and stage n of the same time sample t, to S cycles, where S is the total number of stages. Since $S \geq k-1$, the dependencies are met, and the previous samples of other stages are interleaved. This scheme avoids idle cycles and since the IIR filter is fully pipelined, a new computation can be started every cycle. A total of S cycles are required to process the entire cochlea filter.

When decimation is applied, the schedule is further modified as computation is not required for all of the stages in each iteration.

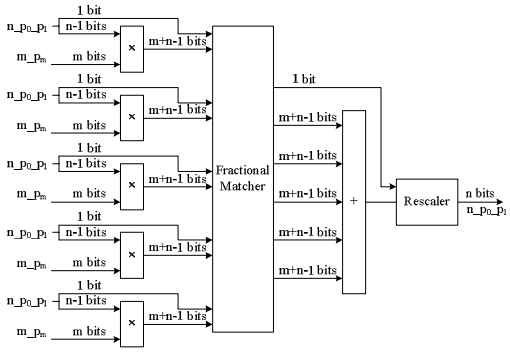

Fig. 9. Filter computation module for dual fixed-point $n\_p_0\_p_1$

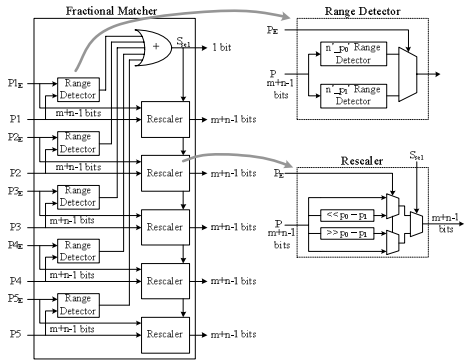

Fig. 10. Dual fixed-point fractional matcher

1.1. Multiple cores

By using multiple numbers of sequential cores described in Fig. 6, a parallel electronic cochlea can be implemented (Fig. 8). If P cochlea blocks are used to implement an electronic cochlea system, the schedules of each cochlea block can be modified so that the n^{th} cochlea block, only processes stages i, where $i \bmod P = n$. The signal $X(t)$ is forwarded from the $(n-1)^{th}$ cochlea block, which calculates the $Y(t)$ signal of the previous stages. Hence, using multiple cores, the maximum sample rate of the system can be increased by P.

2. DUAL FIXED-POINT (DFX) IIR FILTER

2.1. Introduction of Dual Fixed-point (DFX)

Dual fixed-point (DFX) is a data representation that employs a single bit exponent E to select between two different fixed-point scalings, *Num0* and *Num1*. To describe a DFX number system, we use the notation $n\_p_0\_p_1$ where n is the word length including the exponent bit E and $p_0 > p_1$.

$$value = \begin{cases} X \cdot 2^{-p0} & if \quad E = 0 \\ X \cdot 2^{-p1} & if \quad E = 1 \end{cases} \tag{4}$$

A boundary value B, equal to the range limit of the fixed-point number $(n-1)\_p_0$, ($n-1$ is the word length and p_0 is the number of bits used in the fractional part in fixed-point.) is used to decide the best scaling to use and the value of E is calculated by:

$$E = \begin{cases} 0 \, if & -2^{(n-2-p0)} \le D < 2^{(n-2-p0)} - 2^{-p0} \\ 1 \, if \; D < -2^{(n-2-p0)} & or \quad D \ge 2^{(n-2-p0)} - 2^{-p0} \end{cases} \tag{5}$$

If the input value is in the *Num0* range, all the bits above the boundary, $2^{(n-2-p0)}$, would be *0*'s or *1*'s. Therefore, the Boolean expression to generate E is:

$$E = \overline{(d_{n_{in}-1} \cdot d_{n_{in}-2} \cdot \ldots \cdot d_{p_{in}+(n-2-p_0)}} + \overline{d_{n_{in}-1}} \cdot \overline{d_{n_{in}-2}} \cdot \ldots \cdot \overline{d_{p_{in}+(n-2-p_0)}})} \tag{6}$$

2.2. Architecture of DFX second order IIR filter

Fig. 9 and Fig. 10 show how the DFX filter computation module is organised. In the DFX filter, *(n-1)*-bit-×-*m*-bit fixed-point multipliers and *(m+n-1)*-bit fixed-point adders are used. In implementing DFX, additional blocks are range detectors, a fractional matcher and a rescaler as illustrated in Fig. 9 and Fig. 10. Range detectors are used to determine the exponent bit of the input and the output expression is similar to the exponent E expression in Eq. 6. The fractional matcher is used to normalise the products to the same exponent.

3. RESULTS

Using Xilinx ISE 8.1i, the sequential processing electronic cochlea presented in section 3 and 4 was described in VHDL, synthesized and tested on a Xilinx Virtex–II Pro board XC2VP100-6-ff1704. The coefficients for the Lyon and Mead cochlea model [1] were generated using Slaney's Auditory Toolbox [12] in Matlab 7.2. A random signal was used as the input test case since this has energy at all frequencies. Sythesis results are given in Table 1 and Table 2. The resulting signal to noise ratio (SNR) of different implementations are shown in Fig. 11 and Fig. 12. The SNR is with reference to double precision floating-point results. The relative SNR is calculated as the difference in SNR over the floating-point computation electronic cochlea using fixed-point coefficients.

As the wordlength is increased, the SNR of the fixed-point designs (FIX) increase as expected. For the

Table 1. Resource usage and timing comparsion. FIX x_y refers to a fixed point system with total wordlength x bits and fractional wordlength y bits. DFX x_y_z is explained in Section 2.1

Arithmetic	Design	Length of Cascaded Series of Filters	Block RAM (18 kb)	Block Multipliers (18-bit × 18-bit)	Slices	Number of Pipeline Stages	Minimum Period (ns)
Fixed-point	FIX 26_24	88	22	20	1034	8	4.526
	FIX 28_26	88	22	20	1067	8	4.352
Dual Fixed-point	DFX 27_28_24	88	22	20	1366	10	6.214
	DFX 27_30_24	88	22	20	1370	10	6.279
Fixed-point	FIX 38_36	169	28	30	1826	9	5.293
	FIX 40_38	169	28	30	1862	9	5.442
	FIX 42_40	169	28	30	1900	9	5.482
Dual Fixed-point	DFX 39_40_36	169	28	30	2276	11	6.342
	DFX 39_42_36	169	28	30	2282	11	7.027
	DFX 39_46_36	169	28	30	2281	11	6.385

Table 2. Maximum processing rates for different designs.

Design	Non-decimated processing rate (kHz)	-70 dB cutoff decimated		-50 dB cutoff decimated	
		Processing rate (kHz)	Decimation level	Processing rate (kHz)	Decimation level
FIX 26_24	2510	3167	2	4657	5
FIX 28_26	2611	3294	2	4843	5
DFX 27_28_24	1828	2307	2	3392	5
DFX 27_30_24	1809	2283	2	3357	5
FIX 38_36	1117	2348	4	3559	8
FIX 40_38	1087	2284	4	3461	8
FIX 42_40	1079	2267	4	3436	8
DFX 39_40_36	933	1960	4	2970	8
DFX 39_42_36	842	1769	4	2680	8
DFX 39_46_36	926	1947	4	2950	8

implementations with 88 cascaded filter stages, DFX 27_28_24 has an SNR which is 20 dB better than the FIX 26_24 implementation at the same wordlength (Fig. 11). For 169 filter stages, the SNR of the DFX 39_42_36 is 30 dB higher than the FIX 38_36 which uses the same numbers of multipliers and adders (Fig. 12). DFX 39_42_36 also has a better SNR than FIX 40_38.

The resource overhead for the DFX implementation is approximately 350 slices, this being mainly for registers in the additional 2 pipeline stages in the filter computation module.

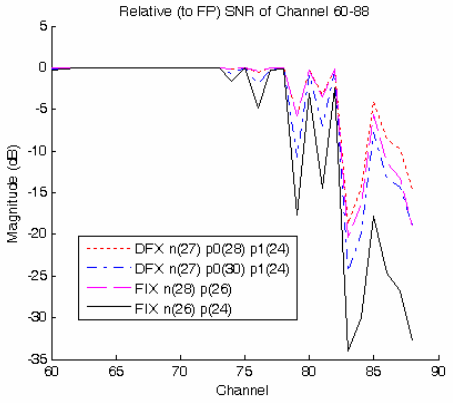

Fig. 11. Relative SNR of 88 cascaded filters systems of different arithmetic and wordlength

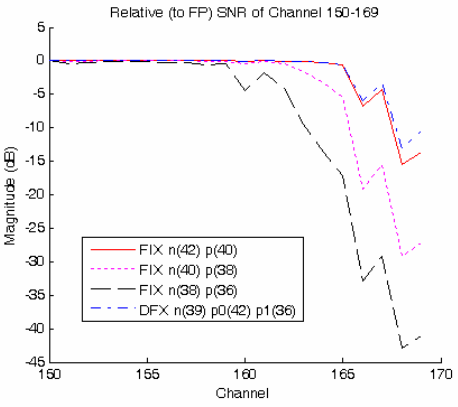

Fig. 12. Relative SNR of 169 cascaded filters systems of different arithmetic and wordlength

The FIX 42_40 and DFX 39_42_36 designs have similar SNR. The area-speed product of the DFX 39_42_36 implementation is 16035 and FIX 42_40 is 10415, so if the design is constrained by logic resources, the fixed-point implementation may be advantageous. Both designs used the same number of 18-bit-×-18-bit multipliers and so if that is the main resource constraint, DFX offers better SNR.

By using decimation, the maximum processing rate of each design can be increased. For decimation with a cutoff value of -70 dB, the maximum processing rate of the 88-stage systems and the 169-stage systems are increased by 26% and 110% respectively. If cutoff value is chosen to be -50 dB, higher levels of decimation can be used and the maximum processing rates are 86% and 218% higher than the non-decimated 88-stage and 169-stage systems respectively.

It is interesting to compare the FPGA-based DFX sequential processing system with our previous parallel distributed arithmetic (PDA) implementation [9]. The PDA implementation processed all the stages in parallel and used LUTs for implementing the distributed arithmetic. This requires more logic resources than the implementation in this paper. Although processing stages in parallel leads to higher speed, there is a limit on the size of the cochlea filter that can be implemented since the resource requirements grow linearly with the number of stages. In contrast, the sequential cochlea can implement filters with any number of stages, although the speed decreases linearly with the number of stages. This can be improved to some degree by employing multiple cores as described in Section 1.1. Of course, decimation and DFX arithmetic can be also applied to a PDA-based implementation.

4. CONCLUSION

In this work, an improved hardware implementation for an electronic cochlea employing decimation and DFX arithmetic was presented. The sequential architecture employed a pipelined IIR filter and has modest resource requirements. We showed that decimation can improve the processing rate by more than 200% and DFX arithmetic can improve the SNR, albeit at an extra cost in hardware. Thus this new design has significantly improved performance and accuracy compared with previous work.

5. REFERENCES

[1] R. F. Lyon and C. Mead, "An analog electronic cochlea," *IEEE Trans. Acoustics, Speech, and Signal Processing*, vol. 36, no. 7, pp. 1119–1134, 1988.

[2] J. O. Pickles, *An Introduction to the Physiology of Hearing*, Academic Press, London, UK, 1988.

[3] J. P. Lazzaro and C. Mead, "Silicon models of pitch perception," *Proc. National Academy of Sciences*, vol. 86, no. 23, pp. 9597-9601, 1989.

[4] J. P. Lazzaro and C. Mead, "Silicon models of auditory localization," *Neural Computation*, vol. 1, pp. 47-57, Spring 1989.

[5] J. P. Lazzaro, J. Wawrzynek, and A. Kramer, "Systems technologies for silicon auditory models," *IEEE Micro*, vol. 14, no. 3, pp. 7-15, 1994.

[6] A. van Schaik and R. Meddis, "Analog very large-scale integrated (VLSI) implementation of a model of amplitude modulation sensitivity in the auditory brainstem," *Journal of the Acoustical Society of America*, vol. 105, no. 2, pp. 811-821, 1999.

[7] C. A. Mead, X. Arreguit, and J. P. Lazzaro, "Analog VLSI model of binaural hearing," *IEEE Transactions on Neural Networks*, vol. 2, no. 2, pp. 230-236, 1991.

[8] J. P. Lazzaro, J. Wawrzynek, and R. P. Lippmann, "Micro power analog circuit implementation of hidden Markov model state decoding," *IEEE Journal Solid State Circuits*, vol. 32, no. 8, pp. 1200-1209, 1997.

[9] M. P. Leong, Craig T. Jin and Philip H. W. Leong, "An FPGA-based Electronic Cochlea," *EURASIP Journal on Applied Signal Processing*, no. 7, pp. 629-638, 2003.

[10] Chun Te Ewe, Peter Y. K. Cheung, and George A. Constantinides, "Dual Fixed-Point: An Efficient Alternative to Floating-Point Computation," *Field-Programmable Logic and Applications 2004*, pp. 200-208, 2004.

[11] R. E. Crochiere and L. R. Rabiner, "Interpolation and Decimation of Digital Signals – A Tutorial Review," *Proc. of IEEE*, vol. 69, no. 3, pp. 300-331, March 1981.

[12] Malcom Slaney, "Auditory Toolbox: A Matlab Toolbox for Auditory Modeling Work," Technical Report 1988-010, Interval Research Coporation, Palo Alto, Calif, USA, 1998, Version 2.

[13] Richard G. Lyons, *Understanding Digital Signal Processing*, Prentice Hall PTR, 2001.

[14] Alan V. Oppenheim, Ronald W. Schafer and John R. Buck, *Discrete-time Signal Processing*, Prentice Hall PTR, 1999.