Reconfigurable acceleration of robust frequency-domain echo cancellation

Chun Hok Ho[1], Ka Fai Cedric Yiu[2], Jiaquan Huo[3], Sven Nordholm[3] and Wayne Luk[1]

[1]Department of Computing, Imperial College, The University of London, London SW7 2BZ, United Kingdom

[2] Department of Industrial and Manufacturing Systems Engineering, The University of Hong Kong, Pokfulam Road, Hong Kong, P.R. China

[3] Western Australian Telecommunications Research Institute (WATRI), The University of Western Australia, Perth, Australia

Abstract

Acoustic echo control is of vital interest for hands-free operation of telecommunications equipment. An important property of an acoustic echo controller is its capability to handle double-talk and be able to operate in real-time. In this paper, we propose a novel hardware architecture to support a robust adaptive algorithm in combination with a two-path model to tackle the double-talk situation. In order to achieve computational efficiency, the echo-canceller is implemented in the frequency domain and is improved by bitwidth optimisation. We show that the echo canceller is successful in handling double-talk and the sub-band implementation has improved convergence significantly. An implementation with 6 instances on a Xilinx XC4VSX55 FPGA at 165MHz can run 72 times faster than software on a 3.2GHz Pentium-4 PC.

1. Introduction

Echo arises at various points in a voice communication network, such as hands-free telephony or VoIP. Without proper control, it can cause significant degradation in conversation quality. Adaptive filters are employed to identify the echo path and cancel the echo [1]. In a normal office room environment the reverberation time will be several 100 ms, which corresponds to several hundred samples with discrete-time impulse response at a 8 kHz sampling rate. The large number of samples contribute to the complexity of an effective acoustic echo canceller.

Another problem is the presence of a strong near-end signal, which is referred to the case of double-talk. Under this situation, the adaptation of the filter coefficients becomes questionable. The adaptive algorithm mistakes the near-end signal as an echo and adjusts the filter coefficients in an inappropriate manner. This will cause the algorithm to diverge and make the echo cancellation fails.

A general way to handle double-talk is to stop the adaptation whenever a strong near-end signal is detected. One approach is to use a double-talk detection scheme [2] together with the robust NLMS algorithm. Another approach is a two-path model that uses a foreground and a background filter [3]. This model employs a continuously adaptive background filter to identify the echo path while the foreground filter is a fixed filter copied from the background filter constantly. There are certain disadvantages of using these two approaches. For the first approach, it requires to adapt two filters at the same time during double-talk, which increases the complexity significantly. For the second approach, the continuous adaptation of the background filter allows it to diverge from the true echo path during double-talk. As a result, the fixed foreground filter may not reflect the correct echo path for a certain duration of time. This is particularly serious when double-talk is followed immediately by echo path variations, where the two-path model fails to track any variation until the background filter is re-converged.

When designing an FPGA structure for an echo canceller [4, 5], the problems with double-talk are often ignored. Also, time domain is often used for simplicity of the design. This restricts the performance of the echo canceller significantly, both functionally and computationally.

In this paper, a robust adaptation technique is proposed to derive a switching scheme to transfer between two echo paths. This extends the two-path model and has the advantage that only one adaptive filter is required at each time frame. The problem of double-talk is tackled by switching between paths instead of using two adaptive filters or one fixed filter. In achieving computational efficiency, first of all, a frequency domain implementation (or sub-band processing) using a delay-less structure is employed to speed up the convergence of the echo canceller. Then, the complete architecture is realised in hardware to aim for real-time operation of the final robust echo canceller. To summarise, the key contributions of this paper include:

1. The first hardware architecture for a novel robust two-path sub-band echo canceller. The proposed design can handle double-talk which results in much higher sound quality, and the frequency domain implementation has improved the mis-alignment of the echo path which will give much better echo noise reduction.

2. Optimisation based on bitwidth analysis to explore suitable bitwidth of the system. The optimised integer and fraction size using fixed point arithmetic can reduce the overall circuit size by up to 80% when compared with a direct realisation of the software onto an FPGA platform.

3. Several FPGA implementations to support the most time-consuming calculations in hardware. The acceleration is evaluated and compared with a software version running on a 3.2GHz Pentium-4 machine, showing that the FPGA-based implementation at 165MHz with 6 instances can achieve speedup of 72 times.

2. Background

Consider transmitting signals over hands-free telephony systems. Let $x(n)$ be the input calibration signal to the system and $y(n)$ be the return signal. Without echo cancellation, the return signal can be written as

$$y(n) = \sum_{k=0}^{\infty} h(k)x(n-k) + v(n), \tag{1}$$

where $v(n)$ is the background noise plus the possible speech of the near-end speaker. The echo cancellation is achieved by finding an estimate of the echo and subtracting it from the return signal. Let $\hat{y}(n)$ be the estimate of the echo, it can be written as

$$\hat{y}(n) = \sum_{k=0}^{N-1} \hat{h}(k)x(n-k), \tag{2}$$

where $\hat{h}(k)$ is the estimate of the impulse response of the echo path with a filter length N. The error of the signal is therefore

$$e(n) = y(n) - \hat{y}(n). \tag{3}$$

The least-squares method is frequently used to measure the error and a fast convergent NLMS algorithm can be derived. However, it is well-known that large errors in even one data point will seriously degrade the least-squares estimation. This is the case for double-talk where the error function is perturbed badly. In view of this, more resistant methods are often necessary, and for example the use of the least absolute distance criterion or l_1 norm can be appropriate in this situation as it tends to give less weights to the effect of gross errors or wild points. There are several ways to combine these properties to give hybrid cost functions. Applying within a stochastic gradient framework using NLMS, the most popular class of functions, known as the Huber function [6] which consists of a parabola prolonged by two tangents, can be written as

$$J(e) = \begin{cases} E[e^2], & if \ |e| \le ks \\ E[ks(2|e| - ks)], & if \ |e| > ks. \end{cases} \tag{4}$$

The first order derivative neglecting the expectation is given by

$$\nabla_h J(e) = \begin{cases} 2ex & if \ |e| \le ks \\ 2sign(e)x & if \ |e| > ks. \end{cases} \tag{5}$$

The final NLMS algorithm is given by $\hat{h}(n) = \hat{h}(n-1) +$

$$\begin{cases} (\mu e(n)/x(n)^T x(n))x(n) & if \ |e(n)| \le ks \\ sign(e(n))(\mu s/x(n)^T x(n))x(n) & if \ |e(n)| > ks. \end{cases} \tag{6}$$

The delayless sub-band echo canceller is illustrated in Figure 1. The echo path is modelled in sub-bands with a set of parallel adaptive filters. The sub-band filters are then collectively transformed to a single full-band filter via a weight transform. In this paper the DFT-FIR weight transform method is used [7]. This full-band filter models the acoustic channel. By separating the paths for adaptation and echo cancellation, the analysis/synthesis system in the signal path, and thus the signal path delay, is avoided whilst the desired features of sub-band processing such as signal de-correlation and computational efficiency are retained.

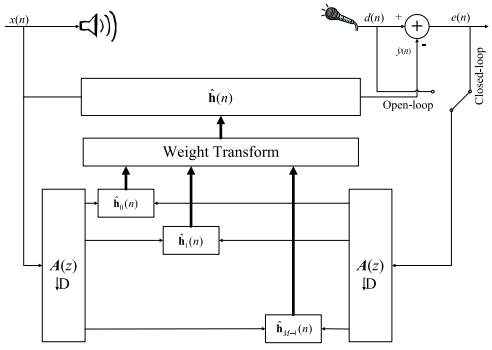


Figure 1: Delay-less sub-band adaptive filter structure.

The adaptive filter in the m-th sub-band, $\hat{\mathbf{h}}_m(k)$, is adapted by the signals in that sub-band, $x_m(k)$ and $e_m(k)$. Depending on how $e_m(k)$ is constructed, the delayless sub-band adaptive filter can be configured in either a open-loop and closed-loop way. In the open-loop configuration, the error signal $e_m(k)$ is generated locally in the m-th sub-band

as

$$\begin{aligned} e_m(k) &= d_m(k) - \hat{\mathbf{h}}_m^H(k)\mathbf{x}_m(k) \\ d_m(k) &= d(n) \otimes f(n)|_{\downarrow D} \\ x_m(k) &= x(n) \otimes f(n)|_{\downarrow D} \\ \mathbf{x}_m(k) &= \begin{bmatrix} x_m(k) \\ x_m(k-1) \\ \vdots \\ x_m(k-N_s+1) \end{bmatrix} \end{aligned} \tag{7}$$

where $\otimes$ denotes a convolution operation, $\cdot|_{\downarrow D}$ denotes D fold downsampling, and $f(n)$ is the analysis filter. In the closed-loop configuration, $e_m(k)$ is obtained from the full-band error signal $e(n)$ as

$$e_m(k) = e(n) \otimes f(n)|_{\downarrow D} \tag{8}$$

presenting an implementation of the 'synthesis dependent' solution. By utilising the full-band error signal, it is possible for a closed-loop sub-band adaptive filter to converge to the optimal Wiener solution. Moreover, the closed-loop configuration yields better computational efficiency because no convolution in the sub-bands is necessary. The closed-loop configuration will be employed in this work.

3. Robust Two-Path Adaptive Filtering

In an echo canceller employing the original two-path adaptive filter structure, the echo is cancelled using the non-adaptive foreground filter $\hat{\mathbf{h}}_f(n)$. The resulting foreground error signal

$$e_{\mathrm{f}}(n) = d(n) - \hat{\mathbf{h}}_{\mathrm{f}} \otimes \mathbf{x}(n) \tag{9}$$

is transmitted to the far-end. The echo path is identified in the background using an adaptive background filter $\hat{\mathbf{h}}_{\mathrm{b}}(n)$. The background error signal

$$e_{\mathrm{b}}(n) = d(n) - \hat{\mathbf{h}}_{\mathrm{b}} \otimes \mathbf{x}(n) \tag{10}$$

is fed back for the update of the background filter coefficients. The signal powers are compared. When the background filter provides a more reliable estimate of the echo path than the foreground filter, its coefficients are copied to the foreground.

In this section, a novel robust two-path adaptive filtering algorithm (RTPAF) is proposed. The proposed RTPAF is different from the original two-path model in the following ways:

- Instead of a binary logic, we employ a three-value logic for filter coefficient copying (referred to two-way filter coefficient copying in this paper). The two-way filter coefficient copying mitigates the aforementioned problems of slow tracking and false coefficient copying after double-talk.
- It performs adaptation directly on the foreground filter in steady state in which the foreground filter provides a good estimate of the echo path and the near-end speaker is silent. This helps to eliminate an extra convolution needed for running the background filter when possible.

The block diagram in figure 2 illustrates the proposed robust two-path adaptive filter.

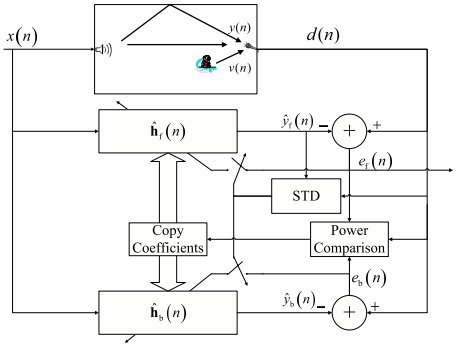


Figure 2: Robust two-path adaptive filter block diagram.

It should be noted that due to statistical fluctuation, steady state detection errors are inevitable. In order to prevent these detection errors causing the foreground filter to diverge, robust statistics based adaptive filtering algorithms are used to adapt the foreground filter as well.

Uniform DFT-modulated FIR filter banks are used to analyse the signals and to transform the sub-band filters to the full-band one [7]. A uniform DFT-modulated filter bank is a set of filters $F_m(z)$ within which each of the filters $F_m(z)$ is related to a single L-tap prototype filter $F_0(z)$ by a complex modulation

$$F_m(z) = F_0(zW_M^m) \tag{11}$$

where $W_M^m = e^{-j2m\pi/M}$. By making use of the modulation structure, a uniform DFT-modulated FIR filter bank can be implemented efficiently.

The DFT-FIR weight transform, implemented with this polyphase-FFT network, requires LM point IFFT for computing the vector multiplication, which is $LM/2\log_2 M$ complex multiplications, and LQ real multiplications for convolving the result from FFT with z^{-nM}. The total operation is about $2LM\log_2 M + LQ$ real multiplications. The overall process can be illustrated by figure 3.

4. Hardware Architecture and Design

In the time domain, the main operation of the robust two-path echo canceller is the error calculations given by equations 9 and 10. The only difference between the two

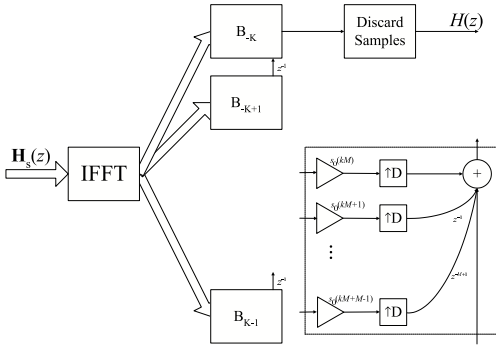


Figure 3: DFT-FIR weight transform.

equations is the filter coefficients. For a filter length of 512 taps and a sample rate of 8 kHz, the processor has to perform at least 24 million arithmetic operations per second because of the convolution operator. These operations are greatly reduced by carrying out the actual filtering in the frequency domain and transforming the results back to the time domain. All the filter coefficients are therefore represented in frequency domain. The proposed hardware design will perform the following calculations:

1. Analyse the input and error signal to their frequency domain representations via FFT;

2. Filter the sub-band signals by the sub-band impulse response estimates. The multiplication itself is a complex dot-vector product circuit;

3. Synthesise the impulse response estimates back to the time domain via IFFT (inverse FFT).

Once the full-band impulse response estimate is reconstructed, the error signal can easily be calculated by software implementation.

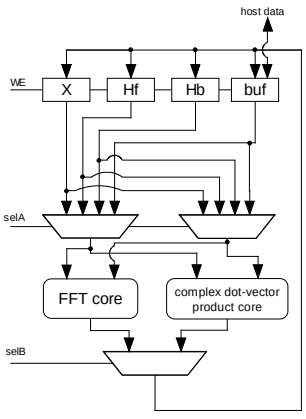


Figure 4: Hardware design for two-path adaptive filter.

A hardware design has been developed to support the calculations. The design have two main processing cores: the complex dot-vector product core and FFT core as shown in figure 4. The FFT core transforms the input and error signal from the time domain to the frequency domain and similarly for IFFT, while the multiplication core performs the filtering in the frequency domain.

To support the robust two-path adaptive filtering algorithm, there are three memory elements for storing filter coefficients ($\hat{\mathbf{h}}_b^T(n)$, $\hat{\mathbf{h}}_f^T(n)$) and input signal $x(n)$. A state machine is developed which generates all the control signal in the design, including the address counter, write-enable and finish signal. When the full band filter length N is equal to 1024, the number of sub-bands M is 128 and the decimation factor D is 64, the depth of the memory requirement for filter coefficients is $(D+1) \times \frac{N}{D} = 1040$. In addition, there is a dual-port block memory *buf* for storing temporal data and interfacing with the host processor, while the depth of the memory requirement for the input and the output signal is 128. It means the hardware will manipulate 128 samples of time domain signal for each iteration.

In contrast to traditional software development, designing a system on an FPGA platform always involves an estimation of the length of bitwidth which affects the circuit size, the system performance and the quality of the calculation. As the design employs a fixed point representation and the saturation arithmetic [8] to avoid the overflow case, a set of fixed point library is developed which allows the exploration of how the bitwidth affects the quality of the signal during echo cancellation. Bitwidth analysis can identify a near-optimal bitwidth for the hardware which can ensure the quality of the signal with less area consumption.

In the analysis, the input signal is purely an echo signal which does not mix with any near-end signal. The expected output is some residual white noise with most of the echo noise suppressed. As a result, a smaller amplitude in the output means a better quality of the system.

The architecture has been implemented on a FPGA platform using VHDL. It is synthesised using Synplify Pro 8.1, placed and routed on the Xilinx XC4VSX55-12-FF1148 [9] and XC3S5000-5-FG1156 [10] FPGA chips using Xilinx ISE 8.1 FPGA design package. In order to maximise the system performance, all the processing cores are implemented using a core generation engine provided by the vendor tools. The core generation engine can generate high speed processing cores by describing the interconnection of the components and the placement of circuits using netlist files. In addition, this approach reduces the development time significantly and the quality of the design can be ensured. The processing cores generated include a 24-bit 128-point FFT core, a 28-bit complex number multiplication core and different configurations of block RAM memory.

Multiple instances of our robust two-way adaptive filter can be packed in a single FPGA to boost the performance, which would be useful especially when the design has multiple channels. This technique can fully utilise the resource on the FPGA and gain massive speedup.

The performance of the FPGA implementation is compared with a pure software implementation. The equivalent software implementation is developed based on MATLAB and compiled to native machine code.

5. Results

In the simulation, the full band filter length is set to $N = 1024$, the number of sub-bands is $M = 128$ with a decimation factor of $D = 64$. The performance of the sub-band algorithm relative to the time-domain implementation is illustrated in Figure 5a. Clearly, the sub-band algorithm outperforms the full-band algorithm in terms of the tracking efficiency and mis-alignment accuracy.

To simulate the double-talk situation, a near-end speech is introduced from sample 80,000 to 100,000. Echo path variation is simulated by switching the desired signal to that from the microphone 24 cm apart from the original one at sample 110,000. This simulates the difficult situation where double-talk is followed immediately by an echo path variation. Simulation results have confirmed that the algorithm performs well during double-talk, and be able to track echo path variation quickly. This is the case even when the near-end speech has a low energy level which resembles the scenario of whispering in front of the microphone.

Another setup has been introduced for bitwidth analysis. Figure 6 illustrates the inputs and outputs of the echo-canceller. Given an echo signal and a mixed signal as shown in figure 6a and 6c respectively, where the near-end speech is introduced from sample 100,000 to 160,000, the echo-canceller produces a filtered signal with most of the echo noise eliminated to recover the near-end speech as illustrated in figure 6b. Figure 6d shows the filtered signals given by the echo-canceller using a double precision floating point arithmetic. Note that the first 20,000 samples are the transient state where the echo-canceller starts to converge. After the filter coefficients have been trained, most of the echo noise can be filtered effectively.

In the present simulation, the amplitude of the echo noise and the near-end speech are designed to be close to each other. In reality, the echo noise should be much weaker than the near-end speech and therefore the echo-canceller can perform even better.

First we run the experiment using a fixed fraction size, say 32-bit while varying the integer size and disable the saturation arithmetic in order to determine the suitable integer size in this system. In case of any overflow, the coefficients will change dramatically and the results will be invalid. The integer size finally determined as 10. In practice it may possible to overflow occasionally even if the integer size is 10, so saturation arithmetic has been employed in the hardware design to minimise the impact.

Figure 7 shows the performance indicator, which is the output signal without introducing any near-end speech in the input. The echo-canceller was simulated using the fixed-point library with different fraction sizes and a fixed integer size equal to 10. It shows that the quality is rather poor when the fraction size is equal to 10. However, by increasing the fraction size, the error signal converges quickly and have no significant change once the fraction size exceeds 18. Based on this analysis, the integer size and the fraction size is chosen to be 10 and 18, respectively, for the FPGA implementation. The corresponding output signal after introducing near-end signal in the input is shown in figure 6e.

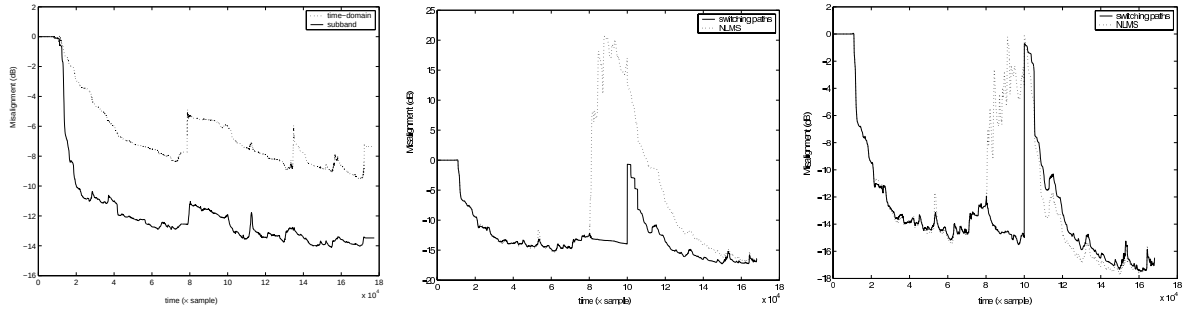
Table 1 represents the implementation results of the proposed hardware design on both high-end Xilinx XC4VSX55-12-FF1148 and low-end Xilinx XC3S5000-5-FG1156 FPGA chips.

FPGA chip	XC4VSX55	XC3S5000
Slices used	4372 (17%)	5255(15%)
DSP48/MULT used	52 (10%)	48 (46%)
Block RAM used	24 (7%)	24 (23%)
Maximum Frequency	180.0MHz	98.8MHz

Table 1: Implementation results of the robust two-path echo canceller, DSP48 is the coarse-grained DSP embedded block in Virtex-4 Series FPGA for multiplication while MULT is a block multiplier in Spartan-3 Series FPGA.

An estimation has been made to evaluate the performance of the FPGA-based echo-canceller. Assuming one block of data contains 128 samples under a 8kHz sampling rate, the number of clock cycle required for processing the block of data in the frequency domain is measured as 1841. Therefore, given that the period of one clock cycle is $1/(180MHz) = 5.55ns$, the FPGA-based echo-canceller can perform one step of echo cancelling in $10.23\mu s$, or equivalently $12.5M$ samples per second. For the low-end FPGA-based echo-canceller using SPARTAN-3 device, it can process $6.87M$ samples per second.

An equivalent software version is developed in MATLAB and compiled to native machine code using the MATLAB supplied compiler (mcc). It should be noted that the performance of the algorithm compiled using MATLAB library is not worse than an equivalent software implementation using native C compilation. It is because MATLAB library has made a lot of optimisation in both matrix and FFT computations, which is used intensively in the echo-

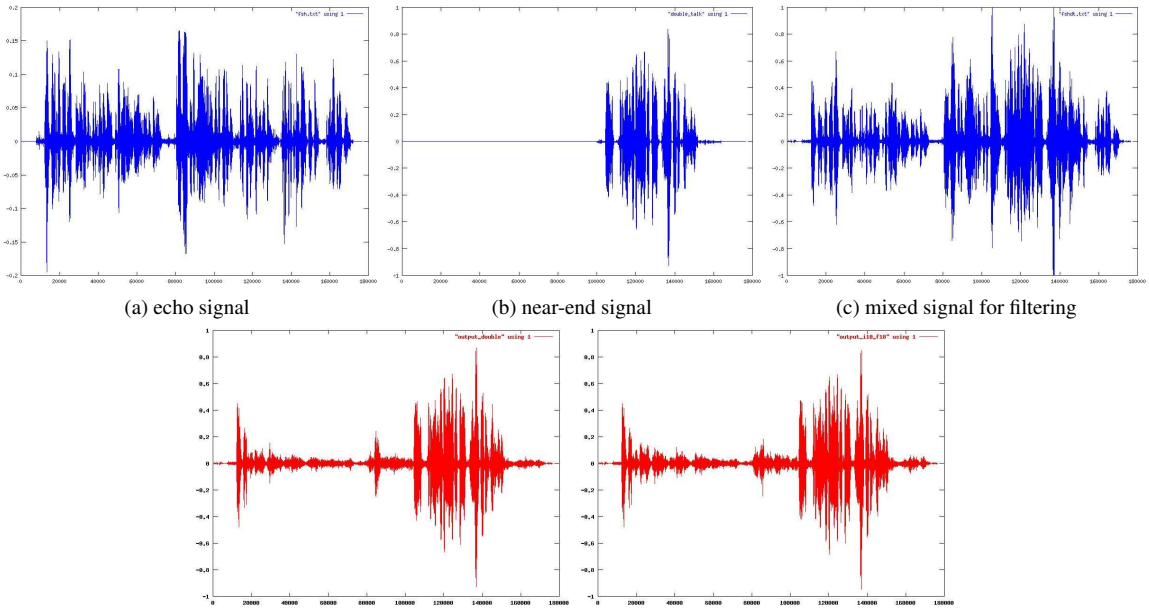


(a) The performance of the sub-band implementation.

(b) Performance of the proposed AEC with closely placed double-talk and echo path variation.

(c) Robustness against whispering near-end talker and echo path variation.

Figure 5: Simulation results for the two-way robust filter algorithm.



(a) echo signal

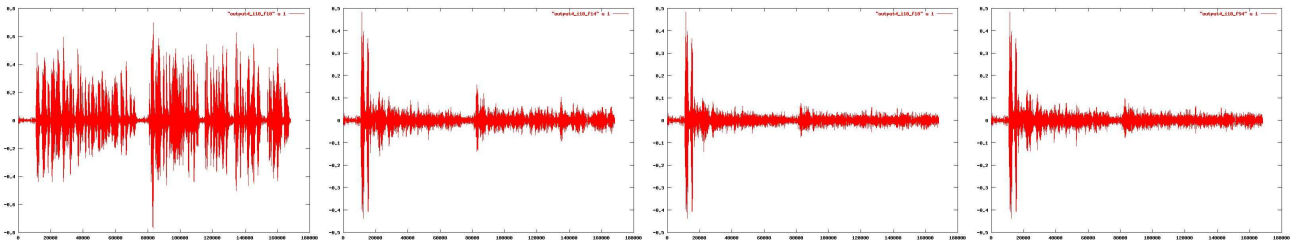
(b) near-end signal

(c) mixed signal for filtering

(d) filtered signal using double precision floating point number

(e) filtered signal using fixed point number, integer size = 10, fraction size = 18

Figure 6: Input signal and results of the robust two-way adaptive filter.



(a) fraction size = 10

(b) fraction size = 14

(c) fraction size = 18

(d) fraction size = 54

Figure 7: Performance indicator on the time domain with different fraction size, where integer size is 10.

canceller. A test is performed by providing 2^{24} samples to the program and measure the time required to finish all the calculations. The test is carried on a Pentium-4 3.2GHz machine with 1GB memory, and it takes an average of 17.5 seconds to finish the calculations. Therefore, the software performance is $2^{24}/17.5 = 0.95$ million samples per second. It shows that the FPGA-based echo-canceller can achieve 13 times speedup when compared with software running on a 3.2GHz PC.

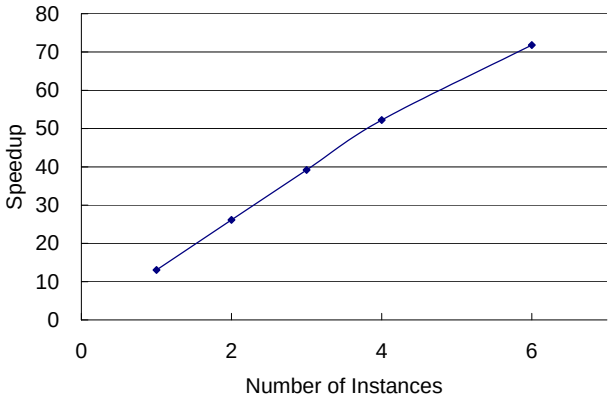


Figure 8: Variation of speedup against the number of robust two-path adaptive filter instances. The speedup does not scale linearly with the number of instances.

Instances	Speed	Slices used	Speedup
1	180MHz	17%	13
2	180MHz	35%	26
3	180MHz	44%	39
4	180MHz	52%	52
6	165MHz	86%	72

Table 2: Slices used, maximum frequency and speedup when implementing multiple instance of the robust filter on an XC4VSX55-12-FF1148 FPGA chip.

Table 2 summarises the implementation results when adding more instances of the filter in an XC4VSX55-12-FF1148 FPGA chip. Figure 8 shows how the number of robust two-path adaptive filter instances affects the speedup. Ideally, the speedup would scale linearly with the number of two-path adaptive filter instances. In practice, the speedup grows slower than expected while the logic utilisation increases because the clock speed of the design deteriorates as the number of instances increases. This deterioration is probably due to the increased routing congestion and delay. A XC4VSX55-12-FF1148 can pack at most 6 instances of the two-path adaptive filter, so the speedup will be 72 times.

6. Conclusions

This paper proposes a novel hardware architecture for two-path frequency domain echo cancellation. The proposed echo-canceller is robust against double-talk and is sufficiently fast in tracking echo path variation for real-time applications. Also, bitwidth optimisation reduces the circuit size while maintaining the quality of the result. Multiple instances of echo-canceller have been packed into an FPGA to boost the speed of the filter. Current and future work includes optimisations for reducing power and energy consumption, and extensions to exploit the reconfigurability of FPGAs to support run-time customisation [11] for adaptive filtering.

References

[1] S.L. Gay. An introduction to acoustic echo and noise control. In S.L. Gay and J. Benesty, editors, *Acoustic signal processing for telecommunication*, chapter 1. Kluwer Academic Pubishers, 2000.

[2] T. Gänsler, S.L. Gay, M.M. Sondhi, and J. Benesty. Double-talk robust fast converging algorithms for network echo cancellation. *IEEE Trans. Speech and Audio Proc.*, (6):656–663, 2000.

[3] K. Ochiai, T. Araseki, and T. Ogihara. Echo canceler with two echo path models. *IEEE Transactions on Communications*, 25(6):589–595, 1977.

[4] W.C. Chew and B. Farhang-Boroujeny. FPGA implementation of acoustic echo cancelling. In *IEEE TENCON*, pp. 263–266, 1999.

[5] S.A. Jang, Y.J. Lee, and D.T. Moon. Design and implementation of an acoustic echo canceller. In *IEEE Asia-Pacific Conference on ASIC Proceedings*, pp. 299–302, 2002.

[6] P.J. Huber. Robust estimation of a location parameter. *Annals of Mathematical Statistics*, 35:73–101, 1964.

[7] J. Huo, S. Nordholm, and Z. Zang. New weight transform schemes for delayless subband adaptive filters. In *Globecom 2001*, 2001.

[8] G.A. Constantinides, P.Y.K. Cheung, and W. Luk. Synthesis of saturation arithmetic architectures. *ACM Transactions on Design Automation of Electronic Systems*, 8(3):334–354, 2003.

[9] Xilinx Inc. Virtex-4 Family Overview. *http://direct.xilinx.com/bvdocs/publications/ds112.pdf*, 2004.

[10] Xilinx Inc. Spartan-3 FPGA Family: Complete Data Sheet. *http://direct.xilinx.com/bvdocs/publications/ds099.pdf*, 2005.

[11] T.J. Todman et al. Reconfigurable computing: architectures and design methods. *IEE Proc.-Comput. Digit. Tech.*, 152(2):193–207, March 2005.