Digital Signal Processing 22 (2012) 376–390

Contents lists available at SciVerse ScienceDirect

Digital Signal Processing

www.elsevier.com/locate/dsp

Reconfigurable FPGA-based switching path frequency-domain echo canceller with applications to voice control device

Ka Fai Cedric Yiu [a,*], Yao Lu [a], Chun Hok Ho [b], Wayne Luk [b], Jiaquan Huo [c], Sven Nordholm [c]

[a] *Department of Applied Mathematics, The Hong Kong Polytechnic University, Kowloon, Hong Kong, PR China*
[b] *Department of Computing, Imperial College London, 180 Queen's Gate, London SW7 2BZ, United Kingdom*
[c] *Department of Electrical and Computer Engineering, Curtin University, Perth, Australia*

ARTICLE INFO

Article history:
Available online 4 November 2011

Keywords:
Echo cancellation
Double-talk
FPGA
Voice control
Speech recognition

ABSTRACT

Acoustic echo control is of vital interest for hands-free operation of telecommunications equipment. An important property of an acoustic echo canceller is its capability to handle double-talk and be able to operate in real time. When it is applied to intelligent voice control device, it is important to suppress the speech from the device and enhance the speech of the user for speech recognition, where double-talk situation is frequently occurred. In this paper, we propose a novel hardware architecture to support a robust adaptive algorithm in combination with a switching path model to tackle the double-talk situation. The proposed switching path model avoids adapting two filters at the same time during double-talk and prevents the disadvantage of the conventional two-path model. In order to achieve computational efficiency and to meet the rigorous timing requirements, the echo canceller is operated in the frequency domain and its computing power is raised by a hardware accelerator implemented in the FPGA fabric surrounding a PowerPC on a Xilinx XUP V2P platform. Results obtained show the echo canceller is successful in handling double-talk situation and the sub-band implementation has improved convergence significantly. An overall improvement by 82.5 times is achieved when a hardware accelerator is used to perform the critical part of the algorithm over a pure software implementation running on a 300 MHz embedded PowerPC processor.

1. Introduction

Echo arises at various points in a voice communication network, such as hands-free telephony, VoIP or intelligent voice control device. Without proper control, it can cause significant degradation in conversation quality. Adaptive filters are employed to identify the echo path and cancel the echo [5]. 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.

There are a lot of interests recently in intelligent voice control devices, which have many applications ranging from logistics warehouse control to intelligence home design [1,2]. In the electronics industry, it is also popular to add the speech control functionality to banking systems [3] and even interactive children books [4]. When conversation capability is included in the device, it resembles a hands-free communication system. When the device speaks, the voice will be fed back to the microphone creating an echo noise. If we try to issue commands to control the device at the same time, this will create the double-talk situation. Depending on the signal-to-echo ratio, the speech recognition performance of such device may deteriorate significantly.

When double-talk occurs, 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 [6] together with the robust normalised least-mean squares (NLMS) algorithm. Another approach is a two-path model that uses a foreground and a background filter [7,8]. 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

* Corresponding author. Fax: +852 23629045.
E-mail address: macyiu@polyu.edu.hk (K.F.C. Yiu).

1051-2004/$ – see front matter © 2011 Elsevier Inc. All rights reserved.
doi:10.1016/j.dsp.2011.10.008

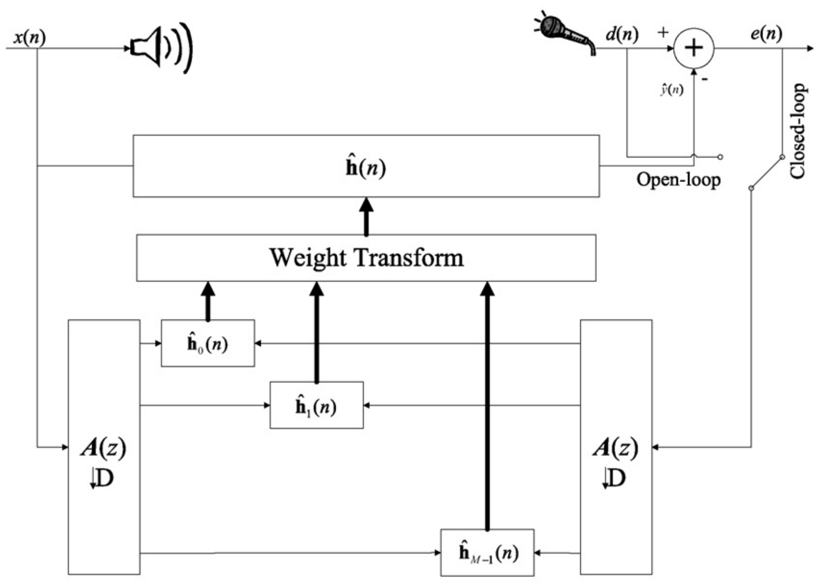


Fig. 1. Delay-less sub-band adaptive filter structure.

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.

In this paper, this problem is addressed. A robust adaptation technique is proposed to derive a switching scheme to transfer between two echo paths. This extends the two-path model and unlike [6], the proposed scheme has the advantage that only one adaptive filter is required at each time frame even during double-talk. 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 speedup the convergence of the echo canceller. In order to achieve real-time performance, the complete architecture is implemented on FPGA. Our implementation differs from the other approaches (such as [9,10]) where time domain is often used and double-talk is often ignored. In order to determine the filter length and the fixed point format, a commercial pre-trained speech recogniser together with a finite set of speech commands is used to assess the effectiveness in reducing echoes. The target is to achieved 100% accuracy for a given set of speech commands.

To summarise, the key contributions of this paper include the following. First of all, this is the first hardware architecture for a novel robust switching path sub-band echo canceller. The proposed design can handle double-talks effectively and efficiently, even when it occurs closely in time with echo path variations. The frequency domain implementation has improved the mis-alignment of the echo path which will give much better echo noise reduction. Second, suitable bitwidth of the system has been explored using optimisation based on bitwidth analysis. The optimised integer and fraction size using fixed-point format can reduce the overall circuit size by up to 80% when compared with a direct implementation of the software onto an FPGA platform. Third, hardware accelerator is equipped to perform the most time-consuming part of the algorithm. The acceleration is evaluated and compared with a pure software implementation running on a 300 MHz embedded PowerPC processor, showing that a speedup of 82.5 times is achieved. Fourth, the filter length and the fixed-point format are selected based on the performance on the enhancement of speech recognition using a given set of speech commands. Unlike other applications, it turns out that much shorter filter length is required even for very low signal-to-echo ratios.

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 delay-less sub-band echo canceller is illustrated in Fig. 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 [13]. 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.

The adaptive filter in the mth 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 delay-less 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 mth sub-band as

$$e_m(k) = d_m(k) - \hat{\mathbf{h}}_m^H(k)\mathbf{x}_m(k)$$
$$d_m(k) = d(n) \otimes f(n)|_{\downarrow D}$$
$$x_m(k) = x(n) \otimes f(n)|_{\downarrow D}$$
$$\mathbf{x}_m(k) = \begin{bmatrix} x_m(k) \\ x_m(k-1) \\ \vdots \\ x_m(k-N_s+1) \end{bmatrix} \tag{4}$$

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

$$e_m(k) = e(n) \otimes f(n)|_{\downarrow D} \tag{5}$$

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 subbands is necessary. The closed-loop configuration will be employed in this work.

3. Robust switching path adaptive filtering

Before describing the switching path model, we first discuss the background of the model. In an echo canceller employing a typical 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

$$e_f(n) = d(n) - \hat{\mathbf{h}}_f \otimes \mathbf{x}(n) \tag{6}$$

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

$$e_b(n) = d(n) - \hat{\mathbf{h}}_b \otimes \mathbf{x}(n) \tag{7}$$

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 order to control the copying of filter coefficients, the following hypotheses are to be tested

$$\begin{cases} H_1\colon & \|\Delta\mathbf{h}_b(n)\|_2 \ll \|\Delta\mathbf{h}_f(n)\|_2 \\ H_0\colon & \|\Delta\mathbf{h}_b(n)\|_2 \not\ll \|\Delta\mathbf{h}_f(n)\|_2 \end{cases} \tag{8}$$

where

$$\Delta\mathbf{h}_b(n) = \mathbf{h}_{opt} - \hat{\mathbf{h}}_b(n)$$
$$\Delta\mathbf{h}_f(n) = \mathbf{h}_{opt} - \hat{\mathbf{h}}_f(n)$$

are the background and foreground coefficient error vectors, and $\mathbf{h}_{opt}$ is the optimal filter coefficients.

In a typical two-path model, such as the original two-path model (OTPAF) [7], it is suggested that the hypothesis H_1 to be selected when

$$\xi^{(e)}(n_r) = \frac{\sigma_{e_b}(n_r)}{\sigma_{e_f}(n_r)} \leqslant T_e \quad \forall r = 0, 1, \ldots(\Upsilon_{hold} - 1) \tag{9}$$

$$\xi^{(d)}(n_r) = \frac{\sigma_{e_b}(n_r)}{\sigma_d(n_r)} \leqslant T_d \quad \forall r = 0, 1, \ldots(\Upsilon_{hold} - 1) \tag{10}$$

$$\xi^{(x)}(n_r) = \frac{\sigma_d(n_r)}{\sigma_x(n_r)} \leqslant T_x \quad \forall r = 0, 1, \ldots(\Upsilon_{hold} - 1), \tag{11}$$

and H_0 be selected otherwise. In the conditions, $\sigma_\chi(n_r) = \sqrt{E(|\chi(n_r)|^2)}$ denotes the standard deviation of the corresponding signal $\chi(n)$, T_e, T_d and T_x are preset detection thresholds, Υ_{hold} is the hangover time, n_r's are the consecutive detection time instances. In practical implementation, $\sigma_\chi(n_r)$ is often replaced by the mean absolute deviation of the corresponding signal over an L point window as

$$\hat{\sigma}_\chi(n_r) = \frac{1}{L}\sum_{l=0}^{L-1}|\chi(n_r - l)|$$

in order to lower the computational complexity. From the above conditions (9)–(11), filter coefficient copying only occurs when [7]

- the background residual echo power is at least $-10\log_{10} T_e$ decibels lower than the foreground one (from (9));
- the background filter delivers at least $-10\log_{10} T_d$ decibels echo suppression (from (10));
- the echo signal is at least $20\log_{10}(1/T_d^2 - 1)$ decibels above the near-end signal (from (10));
- the echo signal is at least $20\log_{10}(T_x^2(1 - T_d^2))$ decibels below the microphone signal (from (10) and (11)).

The above information can be used in the choice of detection thresholds. Note that the original two-path model (OTPAF) was developed for line echo cancellation where a minimum echo path attenuation is guaranteed by industry standard. The condition (11) makes explicit use of this a priori information. However, in acoustics, echo paths are of very diverse characteristics and information about the echo path attenuation is not known a priori. Hence, in our situation, only conditions (9) and (10) are imposed.

It is well known that the OTPAF achieves fast converging and tracking with a background filter that is continuously updated and masks the performance degradation due to double-talk (DT) with a foreground filter that is kept fixed when the background filter adaptation is disrupted by the near-end speech. Nevertheless the background filter is expected to diverge during double-talk period due to its continuous adaptation. Also, it suffers from the problem of slow tracking after double-talk. This is due to the continuous adaptation during DT so that the background filter diverges during DT. After the near-end talker ceases talking, the background filter has to converge toward the new room impulse response from a considerably poorer initial guess. This give rise to a slow tracking speed after DT.

Another major problem of the OTPAF is the possibility for the background filter coefficients to be copied to the foreground when the foreground echo path model is considerably more accurate than its background counterpart. We call this false filter coefficient copying. The OTPAF relies on the comparison of the residual echo powers to determine whether the background echo path model is more reliable than the foreground one. Nevertheless, due to the non-uniform distribution of far-end speech signal energy over different frequency bands, large differences between the echo path and the background filter may not show up in the background error signal power. This allows the background filter to be regarded as a better model of the echo path than the foreground one when its overall modelling accuracy is actually considerably poorer in the sense that

$$\int_{-\pi}^{\pi} |\Delta H_b(\omega)|^2 \, d\omega \gg \int_{-\pi}^{\pi} |\Delta H_f(\omega)|^2 \, d\omega, \tag{12}$$

where $\Delta H_b(\omega)$ and $\Delta H_f(\omega)$ are the Fourier transform of the background and foreground coefficient error vectors.

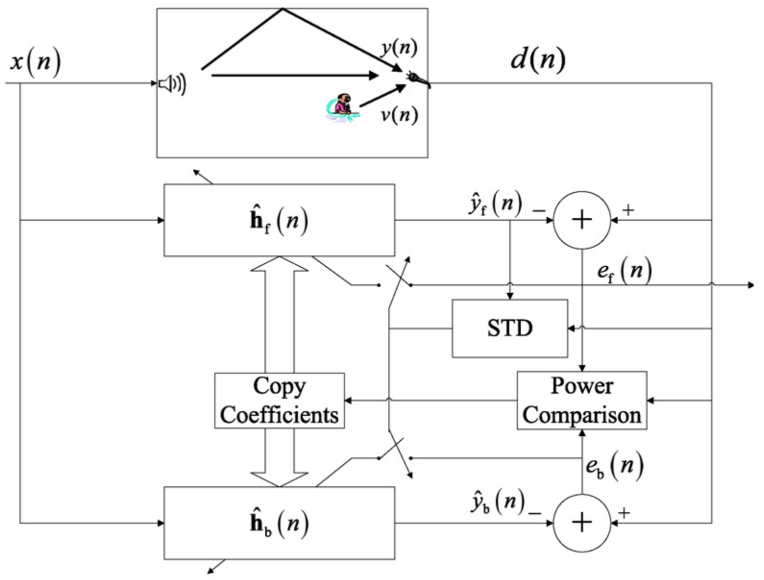


Fig. 2. Robust switching path adaptive filter block diagram.

Assume that in a given time period, the dominant part of the far-end signal energy falls in the frequency region Ω_0 and condition (9) is satisfied. The fact that (9) is satisfied over such a time period ensures that

$$\left|\Delta H_f(\omega)\right| \gg \left|\Delta H_b(\omega)\right| \quad \forall\omega \in \Omega_0 \tag{13}$$

but leaves the possibility of

$$\left|\Delta H_b(\omega)\right| \gg \left|\Delta H_f(\omega)\right| \quad \forall\omega \notin \Omega_0 \tag{14}$$

open. When this happens, the background filter coefficients will be copied to the foreground despite that the foreground filter may be a more accurate model of the echo path than the background one in the sense that the foreground misalignment is considerably smaller than the background misalignment

$$\varrho_f(n) \ll \varrho_b(n), \tag{15}$$

where the misalignments of a filter $\hat{\mathbf{h}}(n)$ is defined as

$$\varrho(n) = 20\log_{10}\frac{\|\mathbf{h}_{\text{opt}} - \hat{\mathbf{h}}\|_2}{\|\mathbf{h}_{\text{opt}}\|_2}. \tag{16}$$

Note that speech signals are non-stationary. When the far-end signal spectrum changes after a false filter coefficient copying and significant portion of the far-end signal energy falls in Ω_1, a residual echo of increased power will be sent to the far-end user.

The above discussion reveals that the OTPAF suffers from a limited tracking capability after DT and a possible false filter coefficient copying. These problems are caused by the divergence of the background filter during DT. In view of this, a novel robust switching path adaptive filtering algorithm (RSPAF) is proposed. The proposed RSPAF has the following properties:

- 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 Fig. 2 illustrates the proposed robust switching path adaptive filter, in which STD stands for single talk detector.

From the above analysis, it is realised that the OTPAF is allowed to stay in the state in which the accuracy of the background echo path model is substantially lower than that of the foreground one. This gives rise to the problems of slow tracking and false filter coefficient copying after DT. In order to alleviate these problems, we test the following three hypotheses:

$$\begin{cases} H_0: & \left\|\Delta\mathbf{h}_b(n)\right\|_2 \approx \left\|\Delta\mathbf{h}_f(n)\right\|_2 \\ H_1: & \left\|\Delta\mathbf{h}_b(n)\right\|_2 \ll \left\|\Delta\mathbf{h}_f(n)\right\|_2 \\ H_2: & \left\|\Delta\mathbf{h}_b(n)\right\|_2 \gg \left\|\Delta\mathbf{h}_f(n)\right\|_2. \end{cases} \tag{17}$$

We copy the filter coefficients from background to foreground (forward filter coefficient copying) when H_1 is selected and copy the filter coefficients from foreground to background (backward filter coefficient copying) when H_2 is selected. The copying of the filter coefficients in both forward and backward directions is referred to as two-way filter coefficient copying. In the case of H_0, the foreground filter is kept fixed and the background filter is updated as in the OTPAF.

With the two-way filter coefficient copying, the statistic $\xi^{(e)}(n)$ becomes a more reliable indicator of the relative quality of the echo path models. Consider that the background filter is excited in the frequency region Ω_1 during DT. This results in

$$\left|\Delta H_b(\omega)\right| \gg \left|\Delta H_f(\omega)\right| \quad \forall\omega \in \Omega_1$$

and

$$\sigma^2_{\epsilon_b} \gg \sigma^2_{\epsilon_f}. \tag{18}$$

This, in turn, triggers a backward filter coefficient copying. The probability of

$$\left|\Delta H_b(\omega)\right| \gg \left|\Delta H_f(\omega)\right| \quad \forall\omega \in \Omega_1$$

occurring when $\xi^{(e)} \leqslant T_e$ is observed is substantially diminished. Therefore, one could be much more confident that the background echo path model is more accurate and forward filter coefficient copying is much less likely to be false. The same argument also applies for backward filter coefficient copying. The hypotheses in (17) can be tested as follows:

- When (9) and (10) are satisfied, select H_1;
- When
$$\xi^{(e)}(n_r) \geqslant T_{e,\mathrm{b}} \quad \forall r = 0, 1, \ldots (\Upsilon_{\mathrm{hold,b}} - 1), \tag{19}$$
select H_2;
- Select H_0 otherwise.

In other words,

$$H_2\colon \quad \|\Delta\mathbf{h}_{\mathrm{b}}(n)\|_2 \gg \|\Delta\mathbf{h}_{\mathrm{f}}(n)\|_2\colon \quad \frac{\sigma_{e_{\mathrm{b}}}(n_r)}{\sigma_{e_{\mathrm{f}}}(n_r)} \geqslant T_{e,\mathrm{b}} \tag{20}$$

$$H_1\colon \quad \|\Delta\mathbf{h}_{\mathrm{b}}(n)\|_2 \ll \|\Delta\mathbf{h}_{\mathrm{f}}(n)\|_2\colon \quad \begin{cases} \frac{\sigma_{e_{\mathrm{b}}}(n_r)}{\sigma_{e_{\mathrm{f}}}(n_r)} \leqslant T_e \\ \frac{\sigma_{e_{\mathrm{b}}}(n_r)}{\sigma_d(n_r)} \leqslant T_d \end{cases} \tag{21}$$

$$H_0\colon \quad \|\Delta\mathbf{h}_{\mathrm{b}}(n)\|_2 \approx \|\Delta\mathbf{h}_{\mathrm{f}}(n)\|_2\colon \quad \text{otherwise.} \tag{22}$$

In addition to the three hypotheses in (17), the algorithm has to test one more hypothesis

$$H_3\colon \quad \|\Delta\mathbf{h}_{\mathrm{f}}(n)\|_2 \approx 0, \quad \sigma_v^2 \approx 0 \tag{23}$$

for the detection of steady state. The above hypothesis can be easily tested with simple correlation analysis [11]. The hypothesis H_3 being true is equivalent to

$$\hat{y}_{\mathrm{f}}(n) \approx y(n) \tag{24}$$

$$d(n) \approx y(n) \tag{25}$$

and thus equivalent to

$$\hat{y}_{\mathrm{f}}(n) \approx d(n). \tag{26}$$

Therefore,

$$\rho(n) = \frac{|r_{\hat{y}d}(n)|^2}{r_{\hat{y}\hat{y}}(n) r_{dd}(n)} \approx \frac{|r_{yy}(n)|^2}{r_{yy}(n) r_{yy}(n)} = 1, \tag{27}$$

where

$$r_{ab}(n) = E\big(a(n)b(n)\big). \tag{28}$$

In practise, $r_{ab}(n)$ are estimated as

$$\hat{r}_{ab}(n) = \frac{1}{L_\rho} \sum_{l=0}^{L_\rho - 1} a(n-l) b(n-l) \tag{29}$$

where L_ρ is the window length for estimating the correlation. A steady state will be declared when a certain threshold is met as follows:

$$\hat{\rho}(n_r) = \frac{|\hat{r}_{\hat{y}d}(n)|^2}{\hat{r}_{\hat{y}\hat{y}}(n)\hat{r}_{dd}(n)} \geqslant T_\rho \quad \forall r = 0, 1, \ldots (\Upsilon_{\mathrm{hold},\rho} - 1). \tag{30}$$

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, similar to [6], robust statistics [12] based adaptive filtering algorithms are used to adapt the foreground filter as well. These algorithms update the adaptive filter coefficients as

$$\hat{\mathbf{h}}(n+1) = \hat{\mathbf{h}}(n) + \mu\psi\big(e(n), \hat{\sigma}(n)\big)\mathbf{x}(n) \tag{31}$$

where μ is the stepsize and $\psi(\nu, \iota)$ is a nonlinear function of ν with a scaling parameter ι. Different algorithms differ from each other in the nonlinear function $\psi(\cdot)$ employed and the way the scaling parameter $\hat{\sigma}(n)$ is computed. Here, we employ Huber's nonlinear function and a scaling parameter computed with Huber's method in the proposed AEC together with a closed-loop delay-less sub-band adaptive filter [13] for the adaptation of the foreground filter.

The proposed robust two-path adaptive filter in its complete form is summarised in the flowchart displayed in Fig. 3. In the flowchart,

$$\hat{\mathbf{h}}_{\mathrm{f}} = \hat{\mathbf{h}}_{\mathrm{f}} + \mu f(\nabla\hat{\mathbf{h}}_{\mathrm{f}})$$

denotes adapting the foreground filter with a robust algorithm, and

$$\hat{\mathbf{h}}_{\mathrm{b}} = \hat{\mathbf{h}}_{\mathrm{b}} + \mu\nabla\hat{\mathbf{h}}_{\mathrm{b}}$$

denotes adapting the background filter with a fast converging algorithm. Same as the OTPAF, the RSPAF also operates in a block by block fashion for each block of D signal samples.

4. Performance on voice control device

In order to assess the performance, a pre-trained speech recogniser based on the principle of hidden Markov model is employed. A fixed set of n voice commands, denoted by $\{s^1, s^2, \ldots, s^n\}$, is built into the dialogue between the system and users. A dialogue is defined as a finite state machine, which consists of states and transitions. A dialogue state represents one conversational interchange between the system and user, typically consisting of a prompt and then the user's response. The system constantly listens to the trigger phrase in the system standby phase. As soon as the user say the general-purpose trigger phrase, the system will respond with an acknowledge tone. The caller is response to specify the desired transaction. The caller responds in variety of ways but must include one of several keywords that define a supported transaction. In the case of a user profile transaction, the application will retrieve the pre-programmed setting of the specified user, and prompt the user with confirmation before going back to the system standby state.

Due to the presence of echo noise, the input commands are usually distorted noise, given by

$$x^i = s^i + n^i \quad i = 1, \ldots, n. \tag{32}$$

With echo cancellation, the estimated command signals are given by $\hat{s}^i$. For the received ith command, a vector of scores is calculated, denoted by

$$\big\{L_1(\hat{s}^i), \ldots, L_n(\hat{s}^i)\big\} \tag{33}$$

where $L_j(\hat{y}^i)$ stands for the likelihood that the received command is the jth command. The estimated command is taken to be

$$\hat{i} = \arg\max_j \big\{L_j(\hat{s}^i)\big\}. \tag{34}$$

We can find the percentage of correct recognition by counting the number of correct estimates.

5. Design and implementation

5.1. Overview

In the time domain, the main operations of the robust two-path echo canceller are the error calculations given by Eqs. (6) and (7). The only difference between the two equations is the filter coefficients. For a filter length of 1024 taps and a sample rate of 8 kHz,

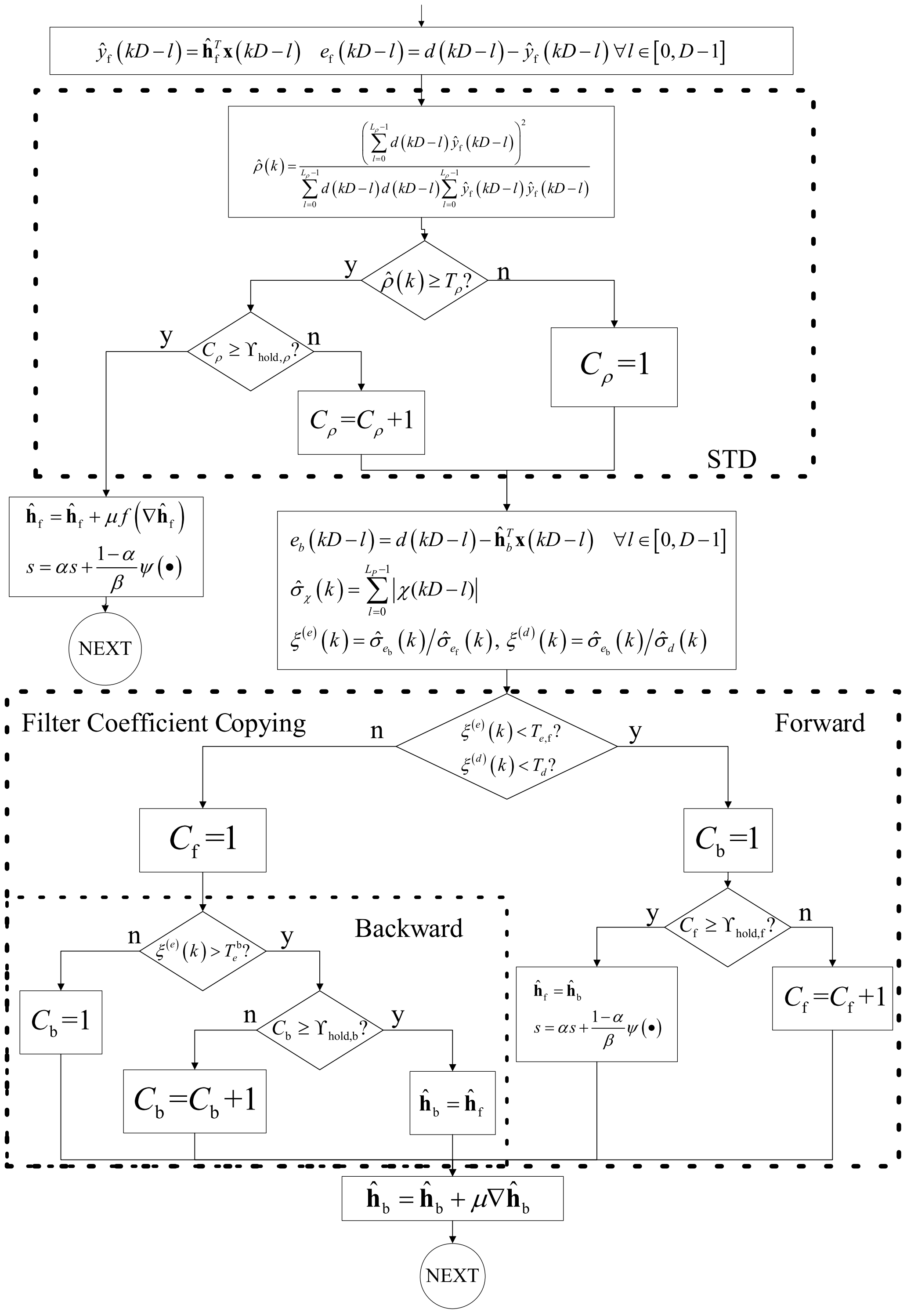

$\hat{y}_f(kD-l)=\hat{\mathbf{h}}_f^T\mathbf{x}(kD-l)\quad e_f(kD-l)=d(kD-l)-\hat{y}_f(kD-l)\ \forall l\in[0,D-1]$
$\hat{\rho}(k)=\frac{\left(\sum_{l=0}^{L_\rho-1}d(kD-l)\hat{y}_f(kD-l)\right)^2}{\sum_{l=0}^{L_\rho-1}d(kD-l)d(kD-l)\sum_{l=0}^{L_\rho-1}\hat{y}_f(kD-l)\hat{y}_f(kD-l)}$
$\hat{\rho}(k)\ge T_\rho?$
y
n
$C_\rho \ge \Upsilon_{\text{hold},\rho}?$
$C_\rho=1$
$C_\rho=C_\rho+1$
STD
$\hat{\mathbf{h}}_f=\hat{\mathbf{h}}_f+\mu f(\nabla\hat{\mathbf{h}}_f)$
$s=\alpha s+\frac{1-\alpha}{\beta}\psi(\bullet)$
NEXT
$e_b(kD-l)=d(kD-l)-\hat{\mathbf{h}}_b^T\mathbf{x}(kD-l)\quad \forall l\in[0,D-1]$
$\hat{\sigma}_\chi(k)=\sum_{l=0}^{L_P-1}|\chi(kD-l)|$
$\xi^{(e)}(k)=\hat{\sigma}_{e_b}(k)/\hat{\sigma}_{e_f}(k),\ \xi^{(d)}(k)=\hat{\sigma}_{e_b}(k)/\hat{\sigma}_d(k)$
Filter Coefficient Copying
Forward
$\xi^{(e)}(k)<T_{e,f}?$
$\xi^{(d)}(k)<T_d?$
$C_f=1$
$C_b=1$
Backward
$\xi^{(e)}(k)>T_e^b?$
$C_b \ge \Upsilon_{\text{hold,b}}?$
$C_b=C_b+1$
$\hat{\mathbf{h}}_b=\hat{\mathbf{h}}_f$
$C_f \ge \Upsilon_{\text{hold,f}}?$
$\hat{\mathbf{h}}_f=\hat{\mathbf{h}}_b$
$C_f=C_f+1$
$\hat{\mathbf{h}}_b=\hat{\mathbf{h}}_b+\mu\nabla\hat{\mathbf{h}}_b$

Fig. 3. Control of adaptation and coefficient copying in RSPAF.

the processor has to perform at least 48 million arithmetic operations per second because of the computational-intensive time-domain convolution operation. This is greatly reduced by carrying out the actual filtering in the frequency domain and transforming the results back to the time domain described using the dataflow shown in Fig. 4, the main operations of which involves:

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

The Xilinx XUP V2P board is used as a hardware platform to implement the echo canceller. This board contains a Virtex II Pro FPGA with 256 MB external DDR memory. Despite the

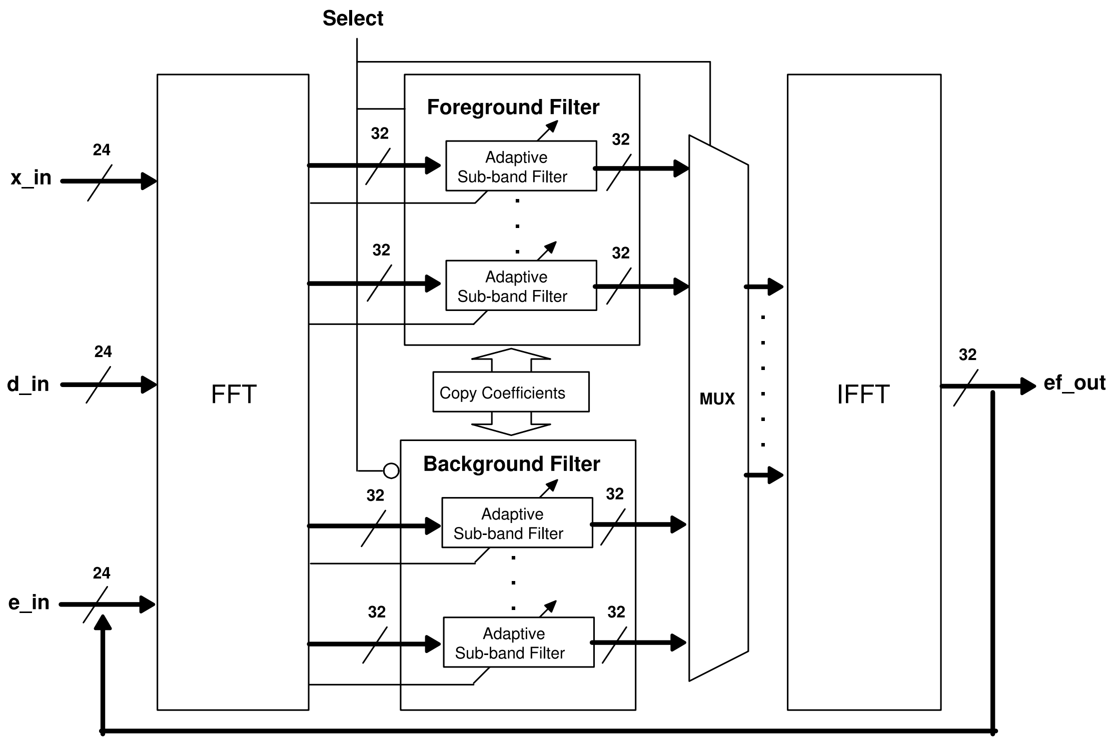


Fig. 4. Dataflow of the main operations.

reconfigurable logic fabric, there are two PowerPC 405 processors in the FPGA which can operate at 300 MHz.

The echo canceller is implemented using a hardware software co-design flow. The implementation begins with a pure software system implemented on a processor in an FPGA. It acts as a reference implementation and can be used for profiling to determine the critical computation in the system. Once the time consuming operations are identified. Those operations can then be implemented on the hardware using FPGA fabric. FPGA device embedded with processor such as Virtex II Pro is a good candidate for this implementation [14]. A dedicated processor allows generic computation on the software part while computationally intensive part can be implemented using reconfigurable fabric.

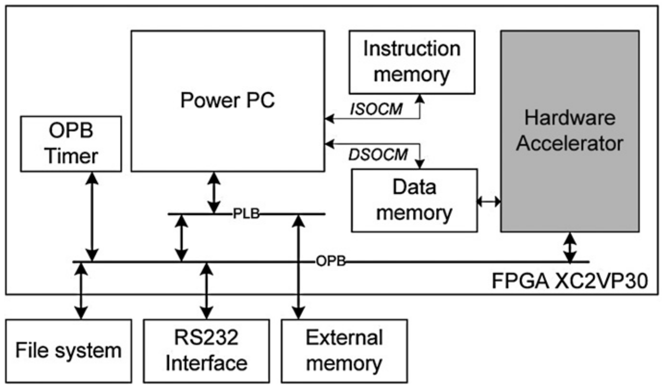


Fig. 5. Hardware/software co-design system.

5.2. Pure software implementation

A block diagram of the echo canceller architecture is shown in Fig. 5. As it is based on hardware-software co-design approach, the architecture involves general processor component, FPGA fabric and the connection interface between them. While a pure software implementation does not require FPGA fabric for implementing computation part, the FPGA fabric is still used in implementing the interface to connect different peripherals with corresponding buses.

Instructions are stored in the on-chip memory and can be accessed by the processor using instruction-side on-chip memory bus (ISOCM). User inputs are stored in external file system initially and are transferred to external memory during the initialisation stage. External memory is attached to the processor using processor local bus (PLB). Data are then fetched to the on-chip memory for processing via data-side on-chip memory bus (DSOCM). In addition, the processor is connected to a RS232 interface and a timer for user communication and profiling the results respectively. Low speed peripherals such as the RS232 interface, the timer and the file system device are connected to the PowerPC using on-chip peripherals bus (OPB).

The algorithm is first described in Matlab and is translated to C program. It is described using relatively straightforward, hardware independent C code, with some minor optimisations to increase the performance. It is then profiled using a timer attached to the processor. The timer is a hardware counter which runs at system bus clock (100 MHz). Because the counter is a hardware timer, it guarantees the accuracy of our measurements. Reading the value from the timer usually takes 10–20 clock cycles which does not significantly affect our accuracy since the computation time is ranged from several thousands to millions of clock cycles.

The total execution time required to process a 21 s wave file sampled at 8 kHz in pure software implementation is about 661 s. In other words, the pure software implementation can be able to process about 254 samples per second. The profiling results of the main operations are shown in Table 1(a), indicating that the FFT/IFFT are the most time consuming operations. Both transformations consume 97% of the processor time. In addition, the complex convolution are the third most computationally intensive operations and consume 2% of processor time. Other operations, such as file I/O, initialisation, voice activity detection and coefficients copy, consume approximately around 1% of processor time. A more detail analysis is performed and the number of clock cycle required for individual operation is obtained. The results is shown in Table 1(b).

5.3. Hardware/software implementation

To achieve higher performance, we introduce dedicated hardware for computationally intensive operations using reconfigurable resources on the FPGA. This approach guarantees computational efficiency by taking advantage of the parallelism property of the algorithm running in the frequency domain, which can be exploited at several levels:

- Loop level parallelism, consecutive loop iterations can be executed in parallel;
- Task level parallelism, that entire procedures inside the program can be executed in parallel;
- Data parallelism.

Table 1
Profiling pure software implementation.

Function	# Execution	% Overall time
24-bit FFT (128pt)	3948	12%
Complex Conv (16 taps)	256 620	2%
24-bit IFFT (128pt)	25 052	85%
File I/O, initialisation, misc.	n/A	1%

(a) Main operations

Function	Cycle count
24-bit FFT (128pt)	2 106 334
Complex Conv (16 taps)	5868
24-bit IFFT (128pt)	2 390 317

(b) Number of clock cycle for individual operation

The software implementation is analysed to determine an optimised mapping to the available hardware. Since the software implementation consists of control part and a computation part, the first step is to identify computational kernels of the algorithms. As shown in Section 5.2, profiling results suggest that the FFT/IFFT and complex convolution operations are the best candidates to be implemented on as hardware accelerator on the reconfigurable fabric. The remaining parts in the code, such as, initialisation, control logic, voice activity detection, coefficients copying and file I/O are implemented by software running on the processor.

The architecture of hardware implemented is illustrated in Fig. 5, where the shaded box indicates the newly introduced reconfigurable hardware. The design focuses on the flexibility and portability in which a single description can derive different implementations based on the quality of the filter and the targeting platform. Therefore, HDL descriptions with parametrised constructs are employed to specify the architectural parameters such as the bus width, the polarity of control signals and the number of functional units.

Since functional units can operate independently in the subband frequency domain, different functional units can execute the band in parallel without affecting each other. Depending on the reconfigurable resources of the targeting platform, it is possible to instantiate more than one functional unit to speedup the computation. In this case, a multi-port register file can be employed to store the data which allows the concurrent write back of corresponding results.

The internal architecture of the hardware accelerator is described in Fig. 6, where it is depicted at logic block level. The core contains an operation unit, direct memory access (DMA) controllers and control registers. The DMA controllers provide a communication channel for the accelerator to access the shared memory between processor and accelerator. In order to maximise the system performance, the FFT, the IFFT and the complex multiplier are implemented using core generators provided by the vendor

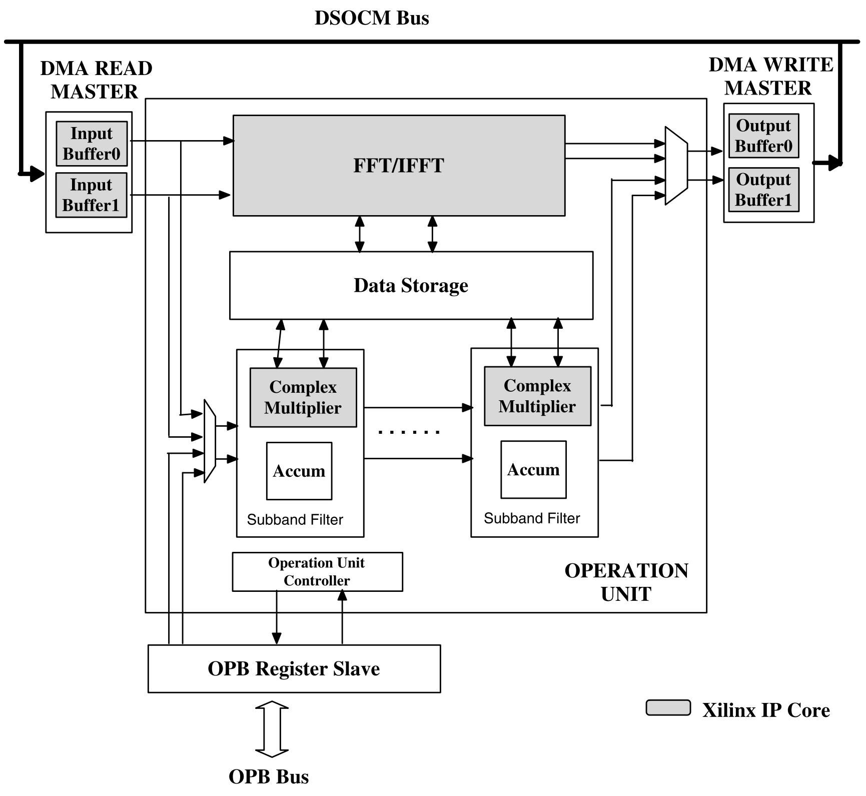


Fig. 6. Block diagram of the hardware accelerator.

tools [17]. The FFT and IFFT components are based on a radix-2 implementation and require 128-point, 24-bit data. The cores allow continuous data processing and achieve fast transformation time. The complex multiplier has been configured to support 32-bit input and 64-bit output. The final results are truncated to 32-bit so that they can be fitted into the data bus. Additional saturation arithmetic logic is implemented to handle the overflow condition. The saturation arithmetic logic detects if there is any overflow at the output of the complex multiplier and assigns the output to maximum value instead of the overflow one [15]. Despite the FFT, IFFT and complex multiplier cores, several different configurations of FIFOs are used as I/O buffers in the DMA controllers.

One of the key challenges in hardware-software co-design system is the interface between them. It is because the communication overhead is usually the bottleneck in such systems. A protocol is established such that the hardware knows when the data transfer is completed and start the processing. When the processing is completed, the hardware has to notify the software and transfer the data back to software for further processing.

To reduce the overhead in transferring the data between hardware and software, shared memory architecture is employed. The shared memory architecture can reduce the transfer time significantly since both software and hardware can access the same piece of memory. Therefore, data written by the software can be seen from the hardware immediately and vice versa. A dual-port block memory in the reconfigurable device is a suitable candidate to implement the shared memory system. One side of the port is connected to the processor and the other side of the port is connected to the reconfigurable hardware directly as shown in Fig. 5.

Data coherency is one of the major concern in the shared memory system. In the proposed system, only one port can access the memory at a time to resolve the data coherency problem. In particular, hardware is not allowed to access the memory before receiving a start signal from the software and after sending a finish signal to the software. Software is not allowed to access the memory after sending the start signal and before receiving the finish signal. In our implementation, we used OPB registers to transfer the start and finish signals. In addition, each sub-band filter has independent memory addressing range. It ensures that the each sub-band filter does not have race condition even if they operate in parallel.

The hardware has dedicated DMA controllers to access the data in the shared memory. The DMA controllers are written in VHDL using behavioural description and used for control of data transfer from/to the shared memory, DSOCM. The DMA read master is a read-only master that reads data from the DSOCM memory to the input buffers of the hardware accelerator. Real number input data will be stored into the buffer0, while imaginary number will be stored into the buffer1. Once both real and imaginary data are ready to be used in the input Buffers, Operation Unit will then be enabled to do the computation. The DMA read master can be configured to perform bursts read of different word length at the beginning of each read transaction. The assumes the use of a linear frame buffer in which all data are contiguous. The read start address and burst length of the DMA from the DSOCM is programmed via the OPB register slave interface. The DMA master may be configured to generate an interrupt request at the end of each frame read from memory. Similarly, DMA write master is a write-only master that writes to the DSOCM in system memory. The data of the output buffer0 are used to store the real product and are first written to the DSOCM. While the output buffer1 used to store the imaginary product are written next.

The operation unit is written in VHDL, and can perform two major operations, namely time-frequency domain transformation and sub-band filtering. The transformation is implemented by using FFT IP core offered by the vendor. The sub-band filtering can be decomposed into a series complex multiplications and accumulations. So the sub-band filter is implemented by complex multipliers and accumulators.

The operation mode is selected at the beginning of each operation via the OPB register slave. Once the processor has one complete block of data in the DSOCM to be processed, the processor asserts a signal to operation unit to trigger the computation. When the calculation is completed and the results are all written to the DSOCM, the operation unit asserts another signal to indicate the computation is completed. All the assertion signals are transferred using OPB interface. As the processor requires to wait for the results from the operation unit for further processing, there is no penalty to use polling scheme to detect the completion of operational unit.

Even though we have used DMA to handle the data delivery between DSOCM to the computation core, transferring large chunk of data between the accelerator and DSOCM imposes a penalty on system performance due to the overhead associated with the DMA control. In order to reduce the use of DMA, a data storage is implemented in the operation unit and is located between FFT/IFFT core and complex multiplication cores. The moral of the scheme is to cache the intermediate results from FFT/IFFT so that the results can be fed to the next stage immediately without writing back to the DSOCM. Therefore we can reduce the number of access of the DMA. As shown in Fig. 7, results from the FFT/IFFT is written to the data storage on a “column by column” basis. Once the FFT/IFFT stage is completed, the complex multiplier can fetch data from the data storage on a “row by row” basis for processing. Using this scheme, we can reduce the number of DMA access by half.

6. Results

6.1. Individual operation

The performance of individual operation of the hardware accelerator is shown in Table 2. We measure the clock cycle count using the same OPB timer as the one discussed in Section 5.2. The results are then compared with the pure software one as shown in Table 1(b). The clock cycle count is significantly reduced by using hardware accelerator. For FFT/IFFT operations, the speedup is approximately 1000 times and the hardware convolution can achieve 30 times speedup. As a results, the corresponding execution time of the operations accelerated by hardware can be reduced dramatically as illustrated in Table 3.

6.2. Echo canceller

We can access the performance of the echo canceller using simulation based approach. By using simulation, we can adjust parameters of the echo canceller such as the filter length, number of sub-band relatively easy to tune the optimal performance of the accelerator.

The performance of the proposed sub-band robust echo canceller is first assessed. Fig. 8 illustrates the inputs of the echo canceller. The data are collected in an office of $456 \times 324 \times 270$ cm in dimensions at 8 kHz sampling rate. The full band filter length is set to $N = 1024$, the number of sub-bands is $M = 128$ with a decimation factor of $D = 64$. For a typical application without double-talk, the performance of the sub-band algorithm relative to the time-domain implementation is illustrated in Fig. 9. Clearly, the sub-band algorithm outperforms the full-band algorithm in terms of the tracking efficiency and mis-alignment accuracy.

In the second experiment, we compare the performance of the proposed RSPAF against those of the OTPAF and of the DTD based on normalised cross-correlation [18] (referred to as NCC in the following text). The parameter settings for the three algorithms are as follows.

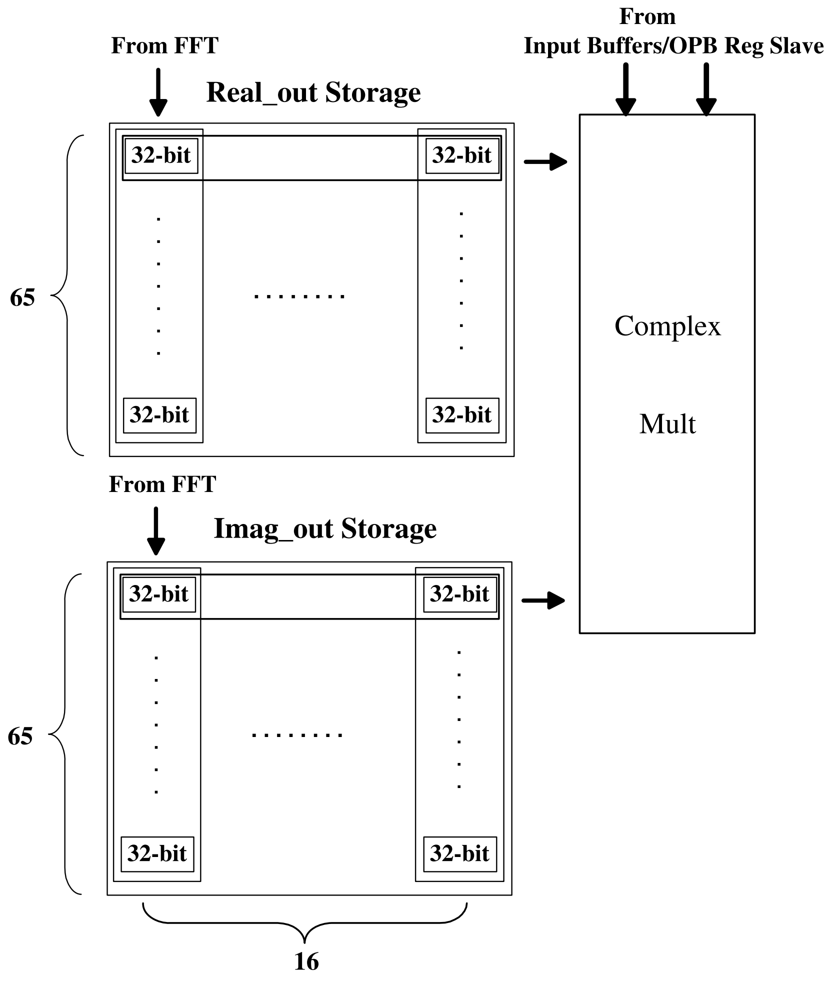


Fig. 7. Dataflow of data storage module.

Table 2
Performance of individual operation in the RSPAF algorithm after using hardware accelerator.

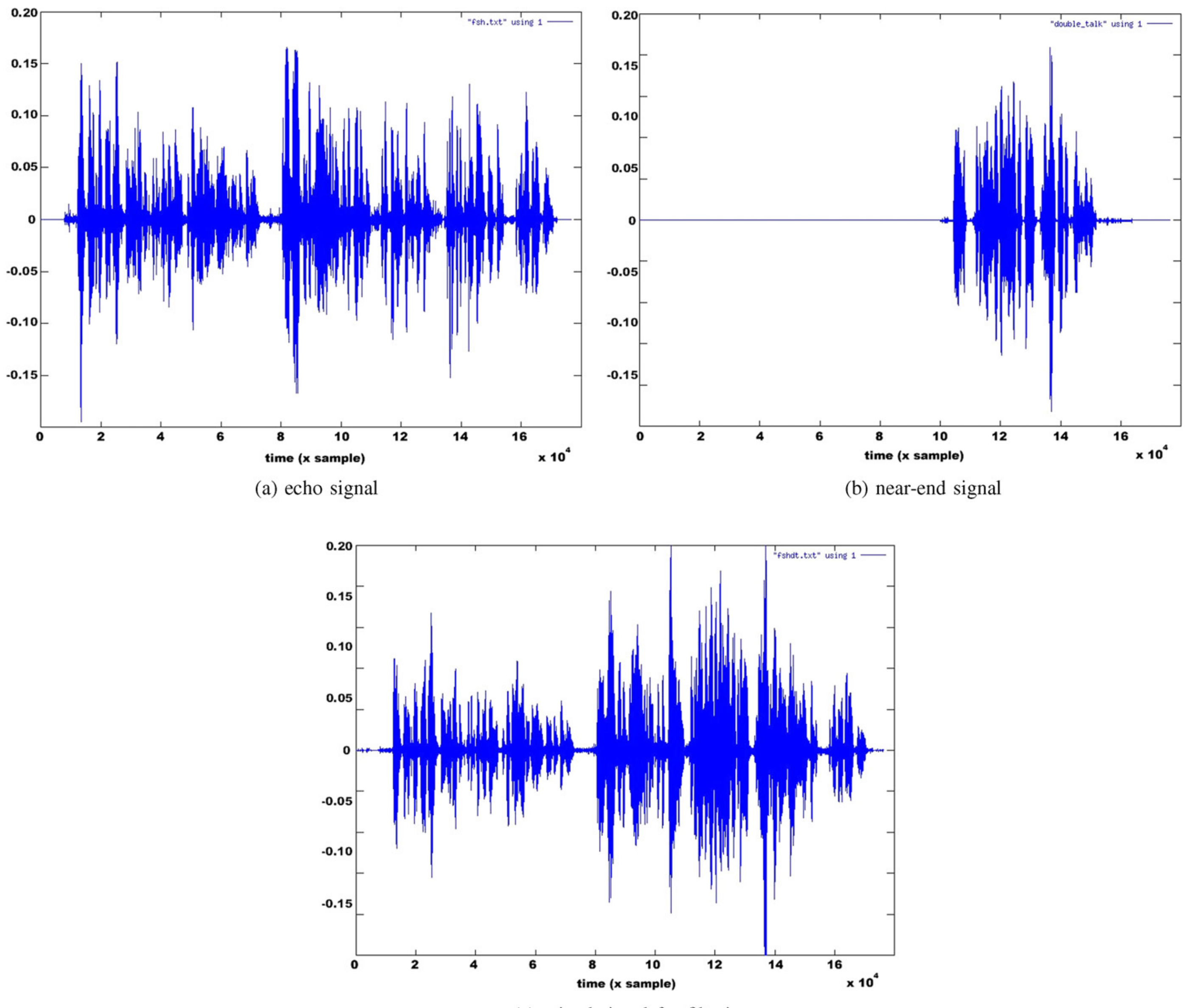
Operation	Cycle count
24-bit FFT (128 pt)	2317
Complex Conv (16 taps)	201
24-bit IFFT (128 pt)	2019

Table 3
Execution time of the RSPAF algorithm after using hardware accelerator.

Operation	# Execution	% Overall time
24-bit FFT (128 pt)	3948	0.1%
Complex Conv (16 taps)	256 620	1%
24-bit IFFT (128 pt)	25 052	0.9%

- OTPAF:
  - The background filter is adapted in 128 frequency bins using the multi-delay frequency domain adaptive filtering algorithm [19] with the stepsize μ set to 1.
  - Thresholds and hangover time for filter coefficient copying are $T_e = 0.875$, $T_d = 0.125$ and $\Upsilon = 4$ (32 ms).
- RSPAF
  - The background filter is adapted exactly them same as it is in the OTPAF.
  - The foreground filter is adapted with a multi-delay version of the algorithm described in [20] with $\lambda = 0.95$, $\beta = 0.60665$ and $k_0 = 1.5$.
  - Thresholds and hangover for forward coefficient copying are $T_e = 0.875$, $T_d = 0.125$ and $\Upsilon_{\text{hold,f}} = 4$.
  - Thresholds and hangover for backward coefficient copying are $T_{e,\text{b}} = 1.125$ and $\Upsilon_{\text{hold,b}} = 6$.
  - The squared correlation coefficient ρ is estimated with $L_\rho = 64$ and the STD threshold and hangover time are $T_\rho = 0.9$ and $\Upsilon_{\text{hold}} = 5$.
- NCC
  - The normalised cross-correlation coefficient is computed with a 1024 tap auxiliary filter. The auxiliary filter is adapted toward the echo path in the same way the background filter of an OTPAF is adapted.
  - The echo cancellation filter is adapted in the same way as the foreground filter is adapted in an RTPAF.
  - Detection threshold and hangover are set as $T_{\text{ncc}} = 0.8$ and $\Upsilon_{\text{ncc}} = 4$.

For all three algorithms, a far-end speech energy detector is also included to stop the adaptation in the frequency bins where there is no sufficient excitation. This energy detection is done by continuously monitoring the far-end background noise power in each frequency bin with the fast-slow average method as presented in [21] and performing update only in those frequency bins where the instantaneous far-end signal power exceeds 2.5 times the estimated background noise power. The foreground misalignment and the residual echo power are taken as the indexes of performance.



(a) echo signal

(b) near-end signal

(c) mixed signal for filtering

Fig. 8. Input signals.

In the experiment, we compare the performance of the three algorithm in the presence of a near-end speech signal. A speech segment of 20 001 samples is added to the echo between sample 60 000 and 80 000. The near-end speech signal is scaled so that the echo to near-end speech energy ratio during DT is 64/49. The misalignment curves are plotted in Fig. 10. The OTPAF falsely copied the diverged background filter coefficients to the foreground at around sample 100 000 and struggled to converge back to the same level it had before DT occurred. On the other hand, the RSPAF and the NCC worked on refining the echo path estimate right after the near-end speech stopped. It is also noticed in the misalignment curves that the NCC made a misclassification during DT, witnessed as a jump in misalignment between sample 60 000 and 80 000, while the RSPAF was almost unaffected by the near-end speech.

The third example demonstrates the different capability of tracking abrupt echo path change during DT for the three algorithms. The abrupt echo path change is simulated by displacing the microphone by 4 cm at sample 90 000. Near-end speech is added to the echo between sample 80 000 and 100 000 with an echo to near-end speech energy ratio of 64/49. The misalignment curves in Fig. 11 indicate that the OTPAF completely failed to track the echo path change due to the divergence of the background filter during DT. Both the RSPAF and the NCC are capable of tracking the echo path change, while the tracking of the RSPAF is faster.

In summary, substantial performance improvement over the OTPAF is achieved with the proposed RSPAF. When compared with the recently proposed NCC, the RSPAF delivers comparable performance. Moreover, the RSPAF is advantageous in terms of computational efficiency in that it only runs one adaptive filter at any given time instance while the NCC needs to employ two filters.

6.3. Bitwidth analysis

To further exploit the reconfigurability of the FPGA, bitwidth analysis is introduced [16]. Bitwidth analysis can reduce the use of reconfigurable fabric and increase the circuit clock rate while maintaining the quality of the filters. Fig. 12 shows the performance which is the output signal without introducing any

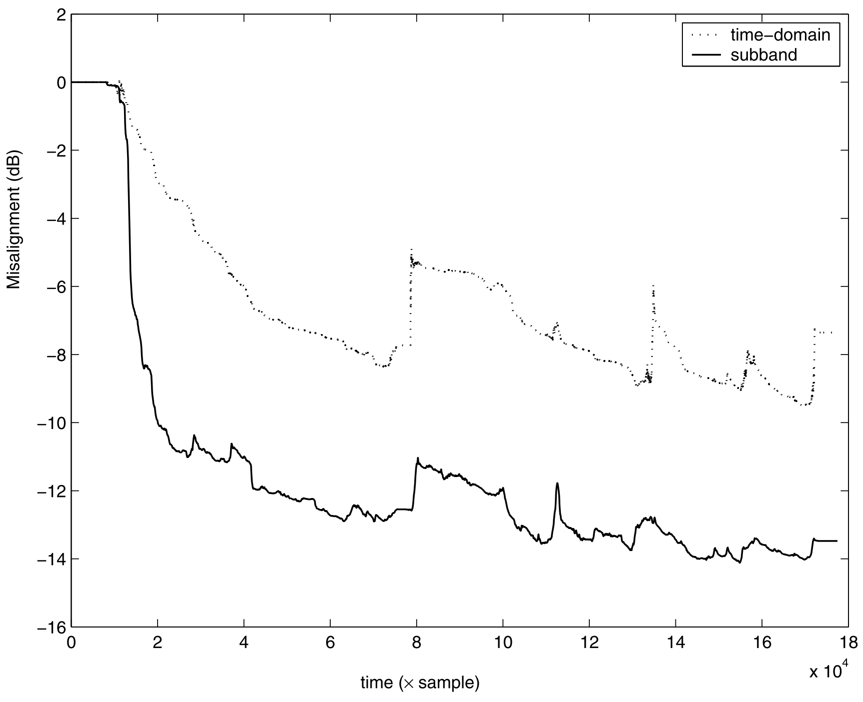


Fig. 9. The performance of the sub-band implementation.

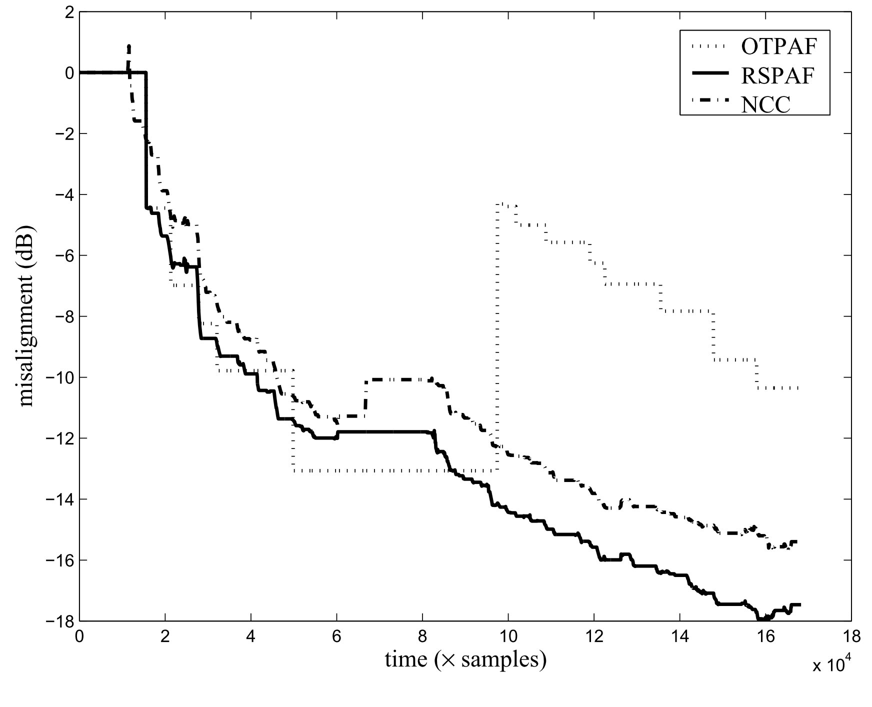


Fig. 10. Performance of the algorithms in the presence of double-talk.

near-end speech in the input. The echo canceller is 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. The different fixed point format will be further assessed below for voice control applications.

Given an echo signal and a mixed signal as shown in Figs. 8(a) and 8(c) respectively, when 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. Figs. 13(a) and 13(b) show the filtered signals produced by the echo canceller using a double precision floating-point arithmetic and fixed-point FPGA implementation. Note that the first 20 000 samples is the transient state where the echo canceller starts to converge. After the filter coefficients have been trained, most of the echo noise has been filtered effectively.

In applying the echo canceller for voice control device and to assess the performance of the different filter lengths and fixed point format, one set of voice commands are created to test the proposed method. The set consists of names of Christmas songs

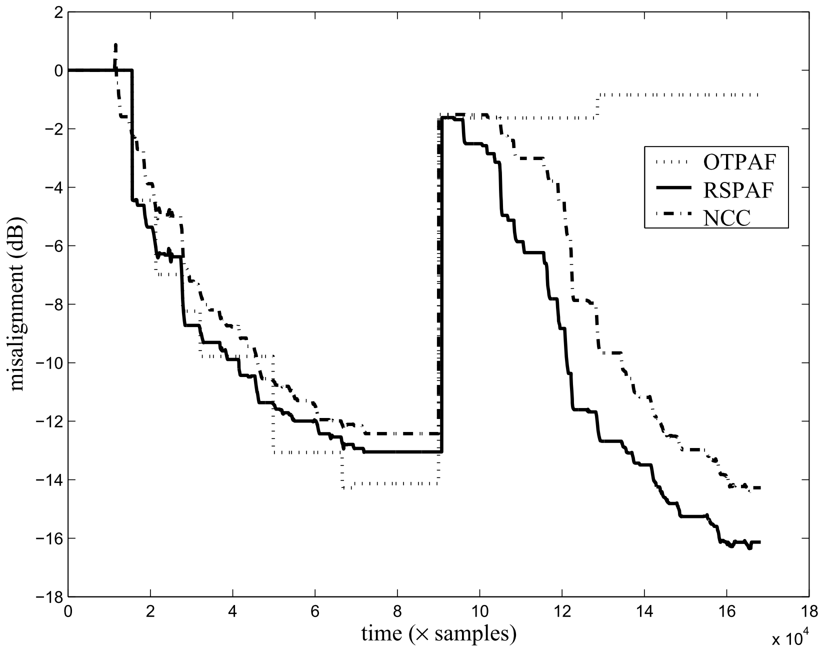
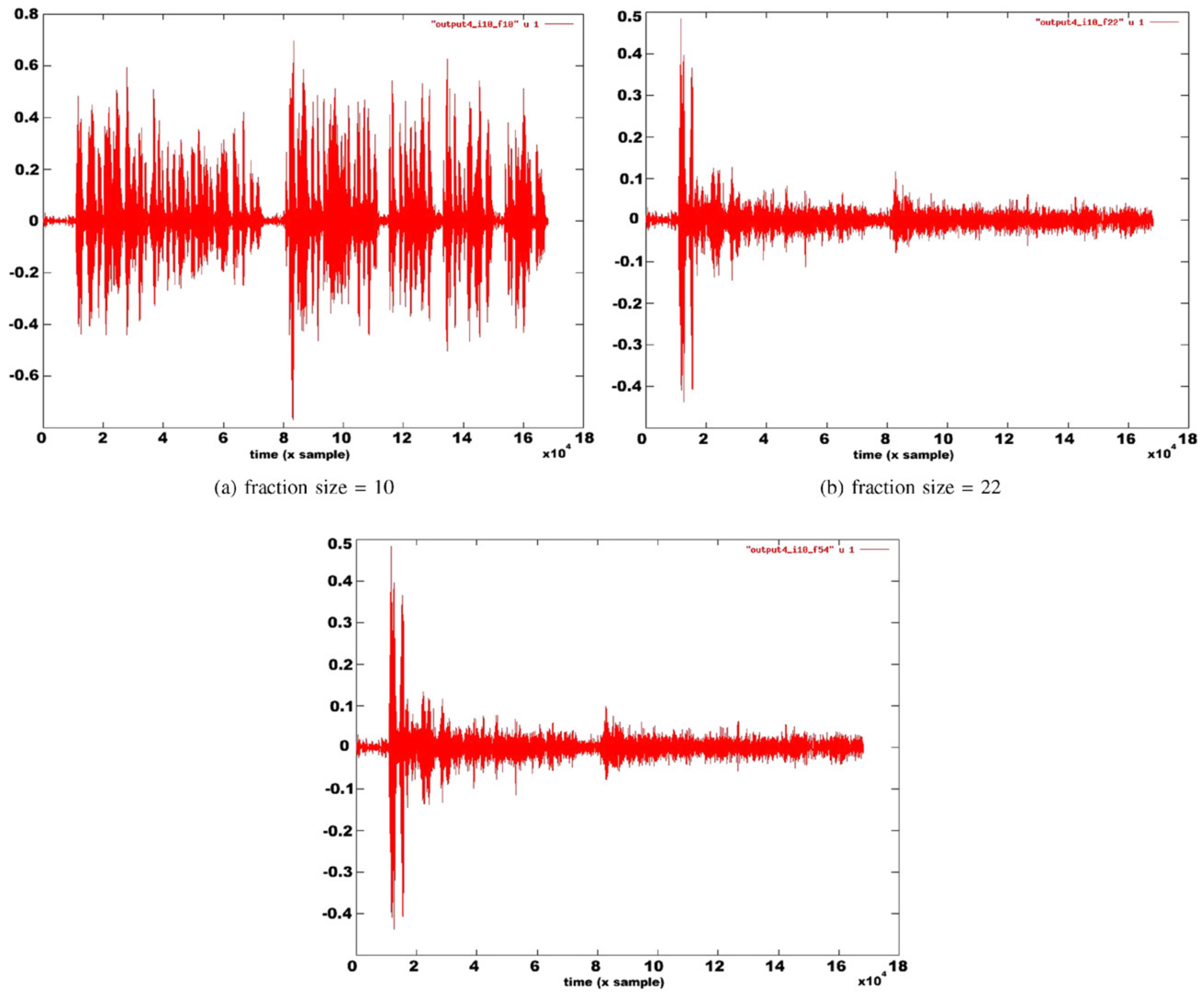


Fig. 11. Robustness against tracking of echo path variation occurring during double-talk.

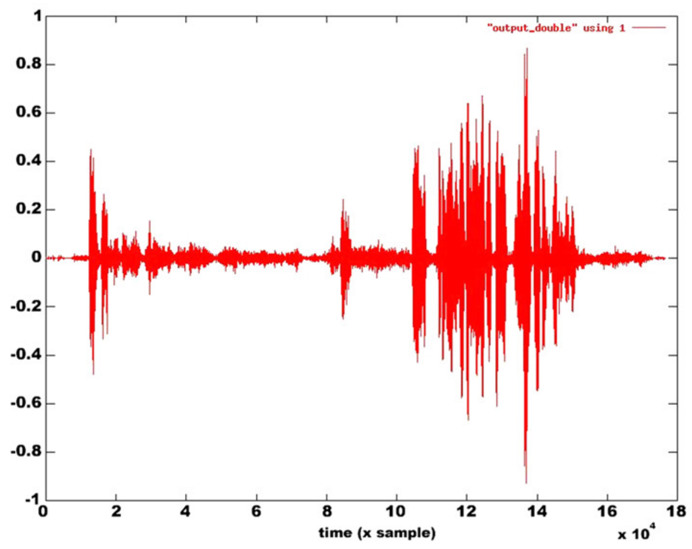


(a) fraction size = 10

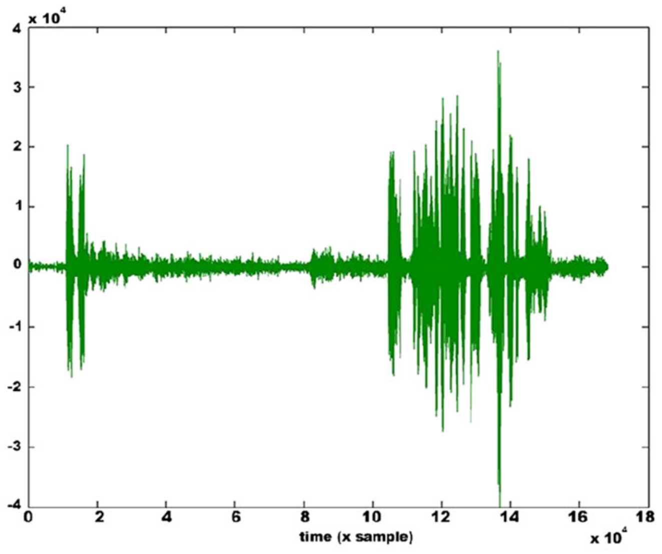
(b) fraction size = 22

(c) fraction size = 54

Fig. 12. Performance with different fraction size, where integer size is 10.



(a) Filtered signal using double precision floating-point representation.



(b) Filtered signal using fixed-point representation, integer size = 10, fraction size = 22.

Fig. 13. Results of the robust switching path adaptive filter.

Table 4
Implementation results of RSPAF algorithm.

Arch.	SW	SW + HW
Slices (27 392)	1779 (6%)	10 753 (39%)
LUTs (27 392)	1642 (5%)	8244 (30%)
Block RAMs (136)	32 (23%)	49 (36%)
MULT18X18s (136)	0	48 (35%)

Table 5
Speedup for different platform. Input is 21 s voice data sampling at 8 kHz.

Platform	Time (s)	Speedup (normalised)
XC2VP30 (PowerPC only)	660	1
TMS320C641	53.2	12.4
XC2VP30 (PowerPC + FPGA)	8	82.5

(*jingle bells*; *santa claus is coming to town*; *sleigh ride*; *let it snow*; *winter wonderland*) typically using in a musicbox. This is a typical command set with phrases. We denote this set of commands by Musicbox. The command set is encoded into a commercial speech recogniser "Sensory's FluentSoft" for experiments. The range of signal-to-echo ratio from 0 db to −10 db was tested. We started with a filter length $N = 1024$ to achieve 100% recognition rate for the chosen command set when signal-to-echo ratio was equal to −10 db. We then shortened the filter length in a binary fashion and found that the filter length could be shortened to $N = 256$ still maintaining 100% recognition rate.

6.4. Hardware accelerator

The FPGA-based echo canceller is implemented on the Xilinx XUP V2P board. The m:st major concern of the echo canceller is to confirm the real-time operation under severe conditions where double-talk is followed immediately by an echo path variation. The wave input samples are stored in the CompactFlash card and be read into the DDR SDRAM once the system is initialised. After the data processing is done, the result is written back to the CompactFlash card and be verified on a desktop PC. The bus clock frequency on the XUP V2P board is 100 MHz, which is used for both architectures, while the processor frequency is clocked at 300 MHz. The FPGA-based implementation is then compared with other embedded platform such as pure software implementation and DSP-based implementations.

The resource usage of all the cores used for implementing the FPGA-based architecture and pure software architecture on a Xilinx XC2VP30 FPGA chip are shown in Table 4. The hardware accelerator consumes a considerable amount of resources compared to the other cores due to the 24-bit FFT implementation.

Estimation has been made to evaluate the performance of the FPGA-based echo canceller in real-time and compare with other implementations and the comparisons are shown in Table 5. A 21 s wave data sampling at 8 kHz is used as the input source. The FPGA-based implementation takes an average of 8 s to complete the entire processing. Therefore, the echo canceller can perform one step of 128 samples echo cancelling in 6.1 ms, or equivalently 21 000 samples per second. The equivalent software implementation is developed based on a MATLAB model and compile to PowerPC processor, which takes an average of 11 min or 660 s to finish the calculations. Therefore, the software performance is 254 samples per second. It shows that with the support of hardware accelerator, echo canceller can achieve 82.5 times speedup when compared with the pure software running on a 300 MHz PowerPC.

We have also implemented the echo canceller on a DSP platform for comparison. A DSP platform emulator CCStudio v3.0 is used in measuring the execution time of the echo canceller. The emulator can report cycle accurate timing results. A DSP processor TMS320C6410 [22] is used for the comparison. The DSP processor is clocked at 400 MHz and a 64 MB external SDRAM memory is attached to the processor. We use the same C source code for PowerPC to implement the echo canceller on the DSP, while some operations, such as FFT/IFFT and fixed point operations, are replaced by architecture specific routine as suggested by the vendor to optimise the execution time. We further assume the data is stored in SDRAM in advance. The emulation results show that the DSP platform can process a 21 s wave data in 53.2 s. Although the DSP platform can achieve 12 times speedup, it cannot deliver real-time performance.

7. Conclusions

In this paper, an FPGA-based architecture for a novel switching two-path frequency domain echo canceller has been proposed.

The proposed echo cancellation algorithm is robust against double-talk situation and is efficient in tracking echo path variation. Unlike other popular approaches, it does not require the adaptation of two filters at the same time even during double-talks. In hardware implementation, we exploit the reconfigurability of FPGA and different techniques have been applied such as bitwidth analysis to reduce the circuit size while maintaining the quality of the results. In addition, the algorithm has been profiled and the most computation intensive part has been extracted and implemented as hardware accelerator to speed up the overall computation. A comparison with pure software implementation and DSP-based implementation has been made and the results show that using a hardware accelerator coupled with a PowerPC processor in a co-design configuration reduces the number of cycles required to perform the most critical operation by about 90% with a total speedup of 82.5 times. Real-time performance can be achieved on FPGA-based platform while it is not possible on pure software implementation and DSP-based implementation.

Acknowledgments

This paper is supported by the Research Grants Council of HKSAR (PolyU 7191/06E) and the research committee of the Hong Kong Polytechnic University.

References

[1] J.L. Gauvain, J.J. Gangolf, L. Lamel, Speech recognition for an information kiosk, Int. Conf. Spoken Lang. 2 (1996) 849–852.
[2] A. Burstein, A. Stolzle, R.W. Brodersen, Using speech recognition in a personal communications system, IEEE Int. Conf. Commun. 3 (1992) 1717–1721.
[3] T. Isobe, M. Morishima, F. Yoshitani, N. Koizumi, K. Murakami, Voice-activated home banking system and its field trial, Int. Conf. Spoken Lang. 3 (1996) 1688–1691.
[4] A. Hagen, B. Pellom, R. Cole, Children's speech recognition with application to interactive books and tutors, in: IEEE Workshop on Automatic Speech Recognition and Understanding, 2003, pp. 186–191.
[5] S.L. Gay, An introduction to acoustic echo and noise control, in: S.L. Gay, J. Benesty (Eds.), Acoustic Signal Processing for Telecommunication, Kluwer Academic Publishers, 2000, Chapter 1.
[6] T. Gänsler, S.L. Gay, M.M. Sondhi, J. Benesty, Double-talk robust fast converging algorithms for network echo cancellation, IEEE Trans. Speech Audio Process. 6 (2000) 656–663.
[7] K. Ochiai, T. Araseki, T. Ogihara, Echo canceler with two echo path models, IEEE Trans. Commun. 25 (6) (1977) 589–595.
[8] Y. Haneda, S. Makino, J. Kojima, S. Shimauchi, Implementation and evaluation of an acoustic echo canceller using duo-filter control system, in: Proc. EUSIPCO96 (European Signal Processing Conference), Sept. 1996, pp. 1115–1118.
[9] W.C. Chew, B. Farhang-Boroujeny, FPGA implementation of acoustic echo cancelling, in: IEEE TENCON, 1999, pp. 263–266.
[10] S.A. Jang, Y.J. Lee, D.T. Moon, Design and implementation of an acoustic echo canceller, in: IEEE Asia–Pacific Conference on ASIC Proceedings, 2002, pp. 299–302.
[11] K. Ghose, V. Reddy, A double-talk detector for acoustic echo cancellation applications, Signal Process. 80 (2000) 1459–1467.
[12] P. Huber, Robust Statistics, John Wiley & Sons, 1981.
[13] J. Huo, S. Nordholm, Z. Zang, New weight transform schemes for delayless subband adaptive filters, in: Globecom 2001, 2001.
[14] J. Noseworthy, M. Leeser, Efficient use of communications between an FPGA's embedded processor and its reconfigurable logic, in: International Symposium on Field Programmable Gate Arrays, 2006.
[15] G.A. Constantinides, P.Y.K. Cheung, W. Luk, Synthesis of saturation arithmetic architectures, ACM Trans. Des. Automat. Electron. Syst. 8 (3) (2003) 334–354.
[16] G.A. Constantinides, P.Y.K. Cheung, W. Luk, Wordlength optimization for linear digital signal processing, IEEE Trans. Comput.-Aided Des. 22 (10) (2003) 1432–1442.
[17] Xilinx Inc. Fast Fourier Transform v6.0, Product Specification, 2008.
[18] J. Benesty, D. Morgan, J. Cho, A new class of doubletalk detectors based on cross-correlation, IEEE Trans. SAP 8 (2) (2000) 168–172.
[19] Éric Moulines, O.A. Amrane, Y. Grenier, The generalized multidelay adaptive filters: structure and convergence analysis, IEEE Trans. SP 43 (1) (1995) 14–28.
[20] T. Gänsler, A robust frequency-domain echo canceller, in: Proc. ICASSP'97, 1997, pp. 2317–2320.
[21] E. Hänsler, G. Schmidt, Acoustic Echo and Noise Control – A Practical Approach, John Wiley & Sons, 2004.
[22] Texas Instrument Inc. TMS320C6413, TMS320C6410 Fixed-Point Digital Signal Processors, Data Manual, January, 2006.

Ka Fai Cedric Yiu received his M.Sc. from University of Dundee and University of London, and D.Phil. from University of Oxford. He is an Associate Professor with Department of Applied Mathematics, the Hong Kong Polytechnic University, Hong Kong. His current research interests include optimization and optimal control, signal processing, sensor array processing, FPGA and algorithm designs.

Yao Lu received the B.E. degree in Electrical Engineering from the Nanjing University of Aeronautics and Astronautics, the M.Sc. degree in Electronics from Queen's University, Belfast in Northern Ireland. His recent interests include multi-touch technology and iPhone apps development.

Chun Hok Ho received the B.Eng. degree (Honors) in computer engineering, the M.Phil. degree in computer science and engineering from the Chinese University of Hong Kong, Hong Kong, and Ph.D. degree in the Custom Computing Group, Department of Computing, Imperial College, London, UK. His research interests include computer arithmetic, computer architecture, design automation, and optimization.

Wayne Luk received the M.A., M.Sc., and D.Phil. degrees in engineering and computing science from the University of Oxford, Oxford, UK. He is a Professor of computer engineering with the Department of Computing, Imperial College London, London, UK, and a Visiting Professor with Stanford University, Stanford, CA, and with Queen's University Belfast, Belfast, UK. His research interests include theory and practice of customising hardware and software for specific application domains, such as multimedia, communications, and finance. Much of his current work involves high-level compilation techniques and tools for parallel computers and embedded systems, particularly those containing reconfigurable devices such as field-programmable gate arrays.

Jiaquan Huo received his B.E. in Electronic Engineering from the South China University of Technology, and hist M.Eng. and Ph.D. from Curtin University of Technology. He is currently a staff engineer with Dolby Laboratories, Australia.

Sven Nordholm received his MscEE (Civilingenjör), Licentiate of engineering and Ph.D. in Signal Processing from Lund University. He was one of the founders of the Department of Signal Processing, Blekinge Institute of Technology in Ronneby in 1990. At BTH he held positions as Lecturer, Senior Lecturer, Associate Professor and Professor. Since 1999 he has been at Curtin University in Perth, Western Australia. From 1999–2002 he was director of ATRI and Professor at Curtin University. From 2002 to 2009 he was director Signal Processing laboratory in WATRI. From 2009 he is professor of Signal Processing with Curtin University. He is also Chief Scientist and co-founder of a start-up company Sensear. His main research efforts have been spent in the fields of Speech Enhancement, Adaptive and Optimum Microphone Arrays, Acoustic Echo Cancellation, Adaptive Signal Processing, Sub-band Adaptive Filtering and Filter Design.