國立交通大學

電機學院IC設計產業研發碩士班

碩士論文

助聽器上的低功耗迴音消除器之設計與實現

Ultra low power design for acoustic feedback cancellation in hearing aidS

研究生：石博文

指導教授：張添烜　教授

中華民國九十七年一月

助聽器上的低功耗迴音消除器之設計與實現

Ultra low power design for acoustic feedback cancellation in hearing aids

研 究 生：石博文　　Student：Bo-Wen Shi

指導教授：張添烜　　Advisor：Tian-Sheuan Chang

國 立 交 通 大 學

電機學院 IC 設計產業研發碩士班

碩 士 論 文

A Thesis

Submitted to College of Electrical and Computer Engineering

National Chiao Tung University

in partial Fulfillment of the Requirements

for the Degree of

Master

in

Industrial Technology R & D Master Program on

IC Design

January 2008

Hsinchu, Taiwan, Republic of China

中華民國九十七年一月

助聽器上的低功耗迴音消除器之設計與實現

學生：石博文　　指導教授：張添烜 博士

國立交通大學電機學院產業研發碩士班

摘　　要

對助聽器的使用者而言，助聽器本身會有惱人的迴音干擾出現，且具有電池容量不大和使用者長時間使用的需求。而一般研製在助聽器上的低功耗迴音消除器的設計，都是採用基本的 LMS 或 DLMS 演算法去實現。而主要的低功耗設計方法都還是著重於架構或是電路之上。

基於上述的LMS based演算法，我們將會在更新係數值和濾波行為上使用了乘法運算而導致結構體複雜度始終無法降低。有鑑於此，本論文發展一新式演算法，P^2SPT(Partial & Progressive Signed Power-of-Two)演算法，同時也調整了設計上的結構，而得到遠比目前所知方法還低的功耗情況。

P^2SPT演算法的發展是為了大幅降低使用上的運算複雜度。所以我們利用了人耳對噪音干擾的適應性，以降低迴音到人耳可接受的程度，而不追求完全消除迴音的情況下，來達到超低功耗的目的。

此演算法運用了 sign-sign 演算法和有條件下的 periodic partial update 演算法作為更新的觸發，再將更新係數做 Progressive 的編碼後去驅動濾波器係數。同時針對濾波器運算，我們採用三個 2 的冪次組合成係數。所以功耗自然會比以乘法器為元件的傳統 LMS-based 演算法要來的低得多。

對應的架構設計上，配合我們製程條件為漏電情形嚴重的 90 奈米製程和我們本身運用了P^2SPT演算法之後而簡化結構，再加上希望用register file來取代資料大量移動的shifter register。本論文針對助聽器的低速運轉設計，搭配功耗分析結果，動用了最大折疊架構來更精簡面積及功耗。

總結而言，由本論文發展的P^2SPT演算法和不同於一般迴音消除器的結構設計，我們將可達到所得單一運算功耗為先前最好設計的 1.7%，整體只需 22.26u watt、8.1 K (gate count)，並且消除迴音效果良好的低功耗迴音消除器。

Ultra low power design for acoustic feedback cancellation in hearing aids

Student : B. W. Shi **Advisor : T. S. Chang**

Industrial Technology R & D Master Program of
Electrical and Computer Engineering College
National Chiao Tung University

ABSTRACT

This thesis present a ultra low power design for acoustic feedback cancellation for hearing aids. Unlike traditional designs only focusing on the architecture and circuit level, the presented design exploits the characteristics of hearing aids applications to simplify the algorithm and its associated architecture.

The presented algorithm adopts a partial and progressive signed-power-of-two algorithm. This algorithm simplifies the update step with partial update process and sign only algorithm. Furthermore, we use three power-of-two digits to progressively construct the filter coefficients.

The resulted architecture exploits the low operating frequency of hearing aids such that a fully folded architecture is adopted with low power SRAM. The final implementation with TSMC0.13um only needs 8.16K gate count and 22.26uW power consumption.

誌　　謝

對於張老師的指導，不論是處世或是行事方面，不才都深感受益良多。也誌謝實驗室的同學和學長，同時在此感激各位朋友的鼎力相助。而在個人方面，一直感念於心的則是家人和身旁的她的支持，他們的加油是我很重要的精神依靠。

最後，在下帶著十二萬分的敬意感謝著許許多多的咖啡豆、爵士樂以及我那愛鬧脾氣的電腦，沒有你們，也就沒有這篇論文了。

石博文

Contents

List of Figures

List of Tables

Chapter 1
Introduction

1.1 Motivation

In these years, hearing aids become more and more widely used. In Fig. 1[1], we can find out the future trends of increased requirement for hearing aids.

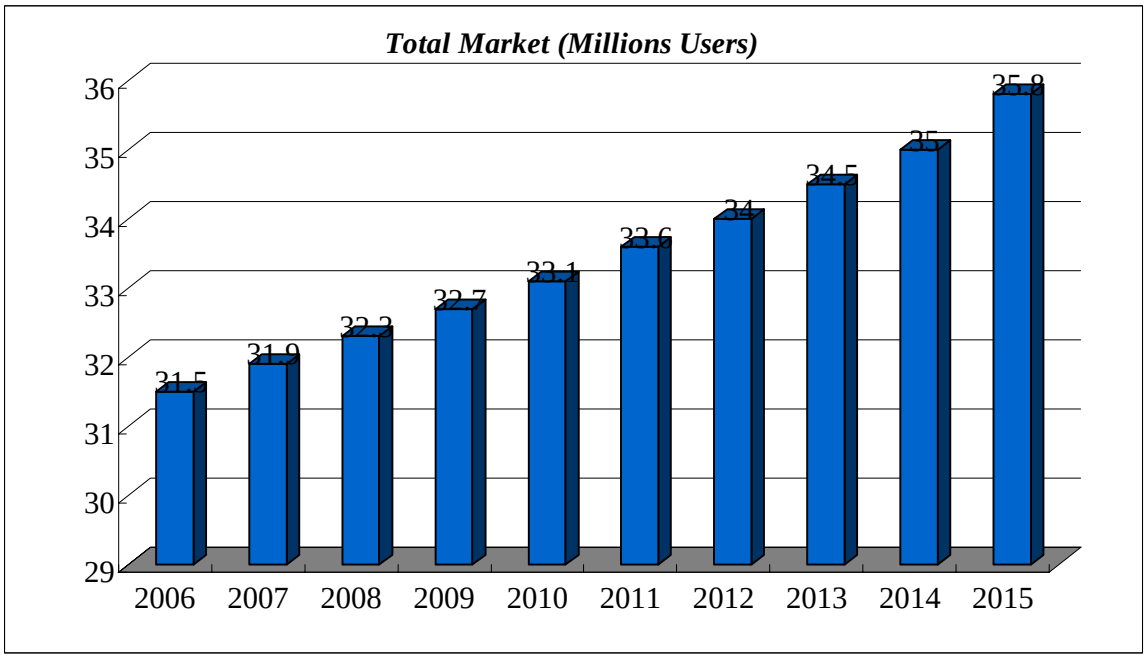


Fig 1. Future trends of hearing aids market

In hearing aids, especially In-The-Ear (ITE) hearing aids, it is needed to design for low power issue. Further, acoustics echo cancellation (AEC) is a major concern in ITE hearing aids, where echo noise is particularly annoying for user.

We will introduce the fundamental problem of acoustic echo noise as follows.

The echo noise is acoustics feedback noise from oneself receiver to the microphone. We show the echo noise interference path in Fig. 2 [2] as follows.

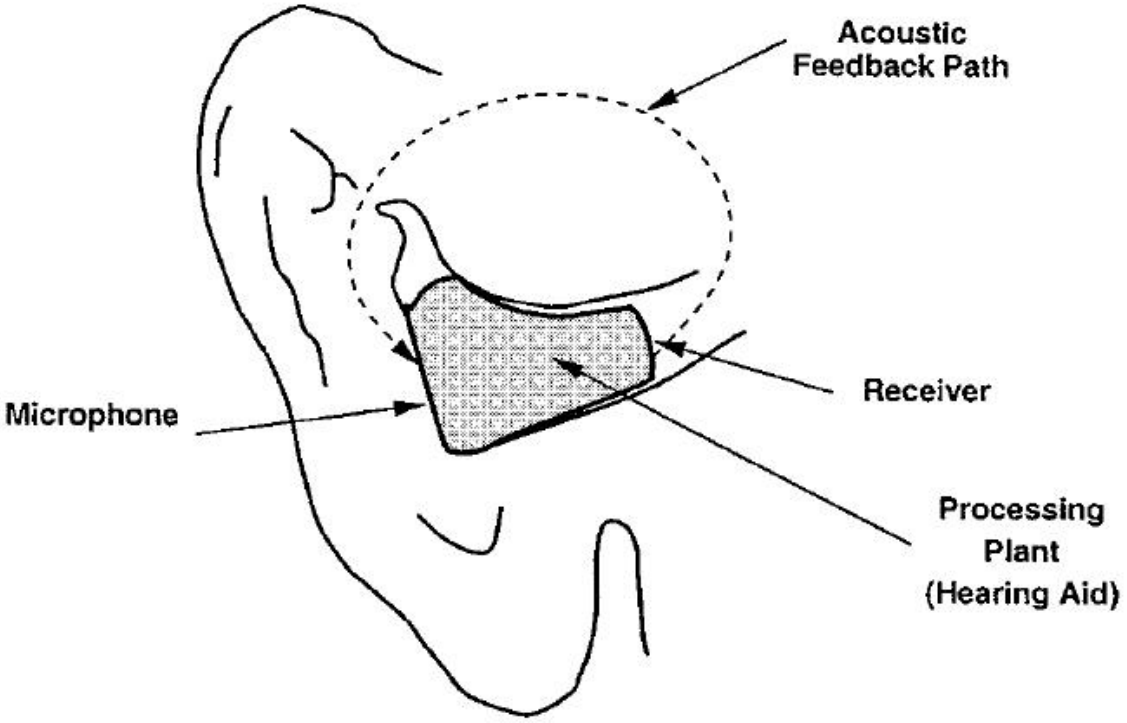


Fig 2. Acoustic feedback in a hearing aid inside of a human ear

The aforementioned issues motivate us to develop an echo canceller with low power consumption. For that reason, this thesis will focus on the low power design of echo canceller in hearing aids.

1.2 Background of low power AEC design in hearing aids

In this section, our background and basis will be presented here. First of all, we start to introduce overview of AEC (Acoustics Echo Cancellation) in hearing aids. Following section 2.1, section 2.2 will begin to introduce related works to you.

1.2.1 Overview of AEC in hearing aids

In section 1.2.1, three parts are included. 1. Types of hearing aids. 2. Echo canceller introduced in hearing aids. 3. LMS algorithm.

1. Types of hearing aids

For the appearance, we have four types of hearing aids [3].

- Body type
- Eyeglass type
- Behind-the-Ear type(BTE)
- In-the-Ear type(ITE)

Different types of hearing aids have different acoustic feedback. If hearing aids have no interference of the echo noise, the acoustic feedback canceller will not be needed anymore. Based on that, the appearance of hearing aids is very important to this thesis. Now that we introduce four types of hearing aids as follows:

Body type:

The body type of hearing aids is shown in Fig. 3.

Fig 3. Body type of hearing aids

The body type shows us that the microphone, Processor and battery are not in the ear. Only the receiver is into the user's ear. In this structure, acoustic feedback noise will not exist. Because of the path from receiver to microphone is too long, the echo noise effect is too weak.

Nevertheless, this type is not convenience to use. This structure is too huge for user's view. By this reason we know, we will go on next type right now, the eyeglass type.

Eyeglass type:

The eyeglass type of hearing aids is shown in Fig. 4.

Fig 4. Eyeglass type of hearing aids

The eyeglass type shows us that microphone, receiver, Processor, and battery are all on the glasses. In this structure, acoustic feedback noise will not exist. That is because the path from receiver to microphone is too long. For that reason, the echo noise effect is weak too.

This type is also not convenience to use.. Users will feel glasses too heavy to carry. People like smaller type as below, the behind-the-ear type and the in-the-ear type.

Behind-the-Ear type:

The behind-the-ear type of hearing aids is shown in Fig. 5.

Fig 5. behind-the-ear type of hearing aids

The BTE type shows us that microphone, processor and battery are all behind the ear and only receiver will be in the ear. In this structure, acoustic feedback noise will begin to exist; the path from the receiver to the microphone is quite near, but echo noise effect is not very serious. In-the-ear type is the type we considered in this thesis. We will introduce this type in next page.

In-the-Ear type:

In-the-ear type is shown in Fig. 6.

Fig 6. in-the-ear type of hearing aids

In-the-ear type shows us that microphone, processor, battery and receiver are all in the ear. That will be smaller, more convenience and more pleasing to appearance, but will have a very serious echo noise to interference user's ear.

ITE (in-the-ear) hearing aid needed echo canceller to eliminate the noise from its own receiver. In this thesis, we assume our hearing aids will be ITE hearing aids. Therefore, the echo noise feedback path will exist.

2. Echo canceller introduced in hearing aids

Fig. 7 show the diagram of hearing aids system to introduce the echo canceller [4].

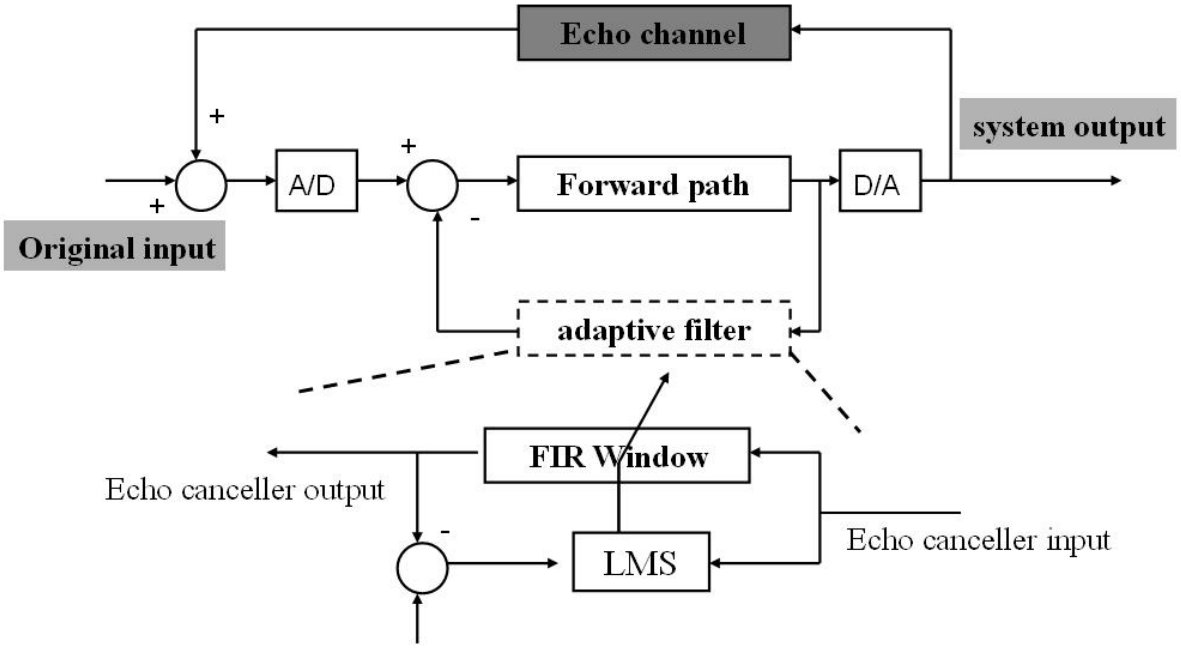


Fig 7. Diagram of hearing aids system

For hearing aids, the echo feedback noise is not welcomed. For that reason, we want to “filter” the echo noise. Besides, the echo channel is changed with time. Therefore, the filter is needed the characteristic of “adaptive”. Thus, the echo canceller is an adaptive filter [5]. Since our main issue is low power design for echo canceller, not all adaptive filter algorithms will be introduced. Other algorithms like NLMS [7], DLMS [8] will not be introduced in this thesis.

3. LMS algorithm

Fig 8 shows the diagram of LMS algorithm.

initialization

$$\vec{x}(0)=\vec{w}(0)=[0,0\ldots0]^T$$

do for $k \geq 0$

$$e(k)=d(k)-\vec{x}^T(k)\vec{w}(k)$$

$$\vec{w}(k+1)=\vec{w}(k)+2ue(k)\vec{x}(k) \qquad (1.2.1)$$

Define:

u: stepsize

d(k): desire signal
e(k): error signal
x(k): input signal vector
w(k): window coefficient vector

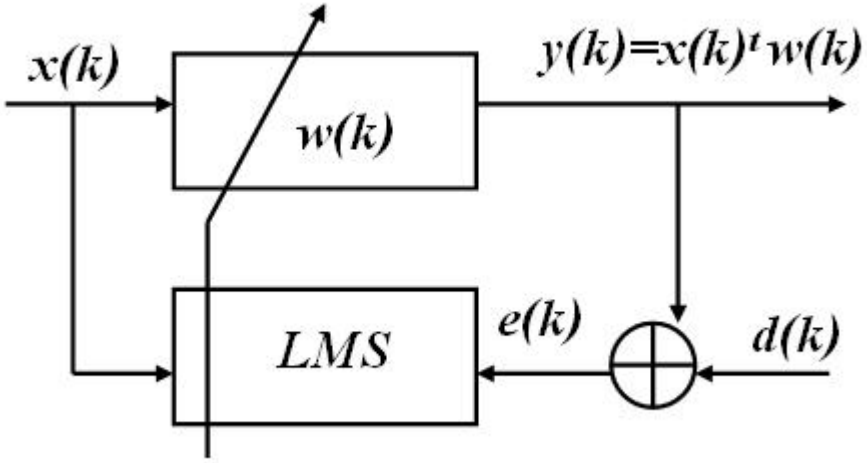


Fig 8. Diagram of LMS algorithm

1.2.2 Related works

In recent years, low-power designs for echo canceller in hearing aids usually focus on the architecture or circuit design. In this section, we will introduce those related work.

1. **Implementation of pipelined LMS adaptive filter for low-power VLSI applications [9].**

The key idea of this design is using power minimization technique, likes Pipelining, Parallel Processing, and Relaxed Look Ahead to have lower frequency. For that reason, this filter can reduce the voltage, and power consumption.

However, its problem is using pipelined LMS algorithm. That means 2 Multiplication for one operation of update & FIR part. Thus, its architecture complexity will be increased greatly (Large area, more switching activity for one operation).

2. **A Low power adaptive filter using dynamic reduced 2's-complement representation [10].**

This design proposed the use of a reduced 2's complement signal representation to conditionally disable the internal signal transitions in the most-significant-bits of a data path. The key idea is to generate the signal representation dynamically according to the signal magnitude.

The drawback of this approach is its high complexity due to LMS algorithm. It needs 2 Multiplication for one operation of update & FIR part.

3. **Ultra-low power DLMS adaptive filter for hearing aid applications[11].**

This echo canceller presents an ultra-low-power, DLMS adaptive filter operating in the subthreshold region for hearing aid applications. In the architecture level, there are using a parallel architecture with pseudo nMOS (for leakage problem) logic style.

The drawback of this approach is its high complexity due to DLMS algorithm.It needs 2 Multiplication for one operation of update & FIR part.

1.3 Thesis Organization

First, we start to focus on our new P^2SPT (Partial & Progressive Signed Power-of-Two) algorithm in Chapter 2. After demonstrating the P^2SPT algorithm, we construct the acoustic feedback model to analyze and verify this new update algorithm's performance. Therefore, we will introduce this model and show the Simulated Results in chapter 3. Then, in chapter 4, architecture designs and power reports will be presented. In the end, conclusion and future work will be given in Chapter 5.

Chapter 2

Partial & Progressive Signed Power-of-Two (P^2SPT) Algorithm Developing

In this chapter, we will introduce the Partial & Progressive Signed Power-of-Two (P^2SPT) algorithm. This algorithm is developed for simplest hardware complexity of echo canceller. The low hardware complexity will give us the low internal signal transition and the low area cost. With those two benefits, we will achieve our goal of low power.

P^2SPT algorithm is assembled by basic LMS algorithm, Sign-Sign algorithm [13], Power-of-Two window coefficient algorithm [14], 2-Staged Periodic Partial Update algorithm [15] and Progressive Update algorithm.

We will introduce those algorithms in section 2.1 Background of P^2SPT Algorithm. After section 2.1, we also have the clearly description in our P^2SPT algorithm in section 2.2.

2.1 Background of P^2SPT Algorithm

The well-know LMS algorithm is introduced in chapter 1. For that reason, we only describe others algorithms one by one in this section. Algorithms as follows:

- (2.1.1) Sign-Sign algorithm.
- (2.1.2) Power-of-Two window coefficient algorithm.
- (2.1.3) 2-Staged Periodic Partial Update algorithm.
- (2.1.4) Progressive Update algorithm.

2.1.1 Sign-Sign Algorithm

The computational complexity of the LMS algorithm is mainly due to multiplications performed in the coefficient updating and in the calculation of

the adaptive filter output. Because we want to minimize hardware complexity, the multiplications of the calculation have to be reduced.
In this section, we focus on the hardware complexity of updating function part. We use the Sign-Sign algorithm to design the simplest updating function part. The coefficient updating in sign-sign algorithm is given by:

$$\vec{w}(k+1) = \vec{w}(k) + 2\mu \operatorname{sgn}[e(k)]\operatorname{sgn}[\vec{x}(k)] \tag{2.1.4}$$

The Sign-sign algorithm is the limit of quantized-data algorithm. We will have the simplest updating hardware cost, if we choose to use this algorithm.

In function (2.1.4), we using function ***sgn(b)*** to change our two inputs. But only considering the Sign-sign algorithm will have the speed problem for updating function part. For example, the every one iteration will give us one Progressive unit step. Therefore, we will need last 1000 times iteration, if our updating coefficient is form 0 to 1000.

For this reason, we will need the Progressive Update algorithm to improve speed of our echo canceller.

2.1.2 Power-of-Two window coefficient algorithm

It is well known that the complexity of a digital filter can be reduced by expressing its coefficients as sums of powers-of-two* (PT) terms [18]. The resulting filter requires no multiplier for coefficient multiplications and enjoys a saving in silicon area when implemented in VLSI [19].

Considering our requirement, we will have to use the Power-of-Two window coefficient algorithm in our echo canceller.

Parameter vector defined and Power-of-Two window coefficient algorithm as shows as follows:

$$\vec{base} = [\text{base}(0), \ldots \text{base}(b\text{-}1)] \tag{2.1.8}$$

$$\vec{\text{update}} = [\text{update}(0), \ldots \text{update}(b\text{-}1)] \tag{2.1.9}$$

$$w_{(j)} = \sum_{i=0}^{b-1} update(i) \cdot 2^{-base(i)} \qquad \textit{for j:1…l} \qquad (2.1.10)$$

$$\vec{W}(k+1) = [W(0), W(1), \ldots W(l)] \qquad (2.1.11)$$

Where ***w(k+1)*** is as same as LMS algorithm defined, the base vector and the update vector are parameter vector by user's requirement. We can unfold the function (2.1.10) to illustrate the base we defined.

$$w(j) = update(0) \cdot 2^{-base(0)} + update(1) \cdot 2^{-base(1)} \ldots\ldots + update(b\text{-}1) \cdot 2^{-base(b-1)} \qquad (2.1.12)$$

Following function (2.1.12), we can define the "base" is $2^{-base\,(i)}$. For example, we will have three bases to use, if we choose the parameter ***b*** is equal to three.

In Fig. 9, we assume there have 4 bases 2^{-3}, 2^{-5}, 2^{-7} and 2^{-9}, and our update parameter only allows 0, 1, 2, and 4 to drive bases form all 0 to all 4. In this case, we can show all window coefficients in Fig. 9.

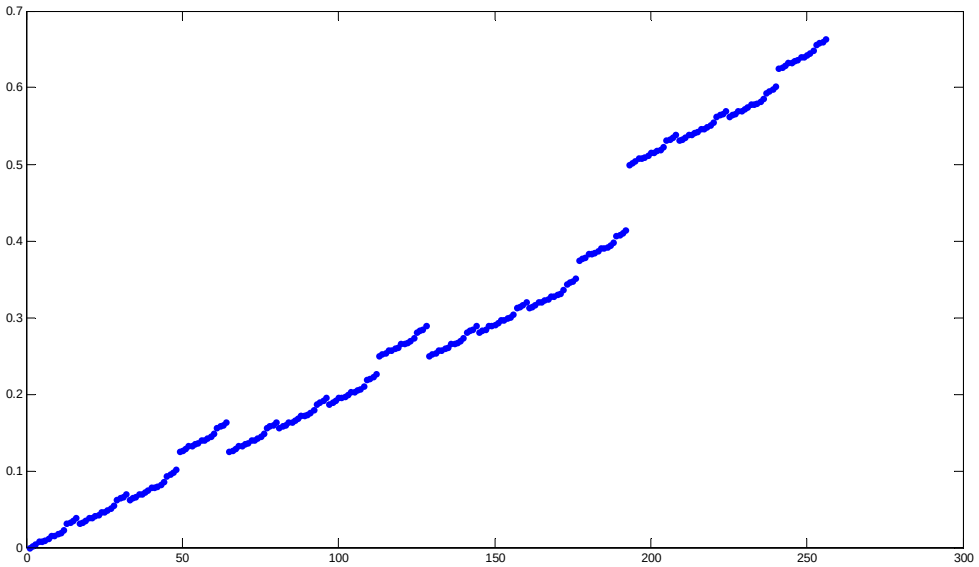


Fig 9. power of two window coefficient

Using Power-of-Two to design the FIR part will lose coefficients precision and those coefficients will be nonlinear. Therefore, we must carefully to choose the “base” in Power-of-Two window coefficient algorithm.

2.1.3 2-Staged Periodic Partial Update algorithm description

Partial updating of the LMS adaptive filter has been proposed to reduce computational costs and power consumption [23]. For this reason, the partial update will be a part of our updating function.

Our 2-Staged Periodic Partial Update algorithm is based on the Periodic partial update LMS algorithm [24] [25] [26]. Therefore, we will introduce this algorithm as follows.

1. Periodic partial update LMS algorithm

The most prevalent type in the literature of selective update scheme is referred to as the periodic partial update LMS algorithm.

To reduce computation needed during the update part of the adaptive filter by a factor of N, the periodic partial update LMS algorithm updates all the filter coefficients every *N* iterations instead of every iteration.

In addition, the coefficient updates for this algorithm are regular, as only one coefficient is changed at one iteration. With this concept, the coefficient update is given by:

$$\vec{w}_j(k+1)=\begin{cases}\vec{w}_j(k)+2\mu e(l)\vec{x}(l), & if\ j=(k \bmod \mathrm{N})+1,\ l=\mathrm{N}\left\lfloor\frac{k}{\mathrm{N}}\right\rfloor \\ \vec{w}_j(k), & otherwise.\end{cases} \tag{2.1.13}$$

Where $\lfloor\cdot\rfloor$ denotes the truncation operation, ***k*** mod N denotes iteration ***k***

modulo N . By considering N iterations of the updates in function (2.1.13), it can be shown that this algorithm is equivalent to the following N-fold coefficient vector update:

$$\vec{w}(k+N) = \vec{w}(k) + 2\mu e(k)\vec{x}(k) \tag{2.1.14}$$

It describes a modified version of the LMS adaptive algorithm that uses every N-th instantaneous gradient to update the filter coefficients.

2. 2-Staged Periodic partial update LMS algorithm

Following Periodic partial update LMS algorithm, the 2-Staged Periodic partial update LMS algorithm is proposed. In fact, we just try to add one condition to change the update period of partial update. The 2-staged periodic partial update LMS algorithm as shows as follows:

$$P = \begin{cases} 2N, & if \ \text{condition} == \text{true} \\ N, & otherwise. \end{cases} \tag{2.1.15}$$

$$\vec{w}_j(k+1) = \begin{cases} \vec{w}_j(k) + 2\mu e(l)\vec{x}(l), & if\ j = (k \bmod \mathrm{P}) + 1,\ l = \mathrm{P}\left\lfloor \frac{k}{\mathrm{P}} \right\rfloor \\ \vec{w}_j(k), & otherwise. \end{cases} \tag{2.1.16}$$

In consequence, we double the period of partial update when our condition will be true. We use this 2-stage to control the partial update frequency. Therefore, we can have high frequency update when we have to match the echo channel, and have low update frequency if there are not necessary for our echo canceller.

2.1.4 Progressive Update algorithm description

Following section 2.1.1 to section 2.1.3, we can figure out the sign-sign algorithm means our update trigger will use the sign of input vector and the sign of error signal, the power-of-two window coefficient algorithm told us which the window coefficients only support power-of-two coefficients in the FIR part. In addition, we have 2-staged partial updating to reduce the computational cost.

Besides, the most important part we are not discussed. Our encoder, it is to receive the update information and to transform this information to real window coefficients that we just need.

Consequently, the encoder needs to have simplest hardware cost and also have enough updating speed to catch echo noise. Considering this two reason, we present this Progressive Update algorithm here. The Progressive Update algorithm as shows as follows:

$$p(i) = \left\lfloor \frac{\text{Coeff}}{4^i} \right\rfloor \% 4 \qquad for\ i=0 \cdots b-1 \tag{2.1.17}$$

$$\vec{p} = [p(o), p(1), \ldots p(b-1)] \tag{2.1.18}$$

We define the parameter ***coeff*** is a coefficient for this example. Next, we encode this parameter to produce the Vector ***p*** of ***b*** elements.

Finally, we use those elements in the vector ***p*** to drive the “base” of FIR part. The example as shows follows:

$$w(j) = p(0).\ 2^{-base(0)} + p(1).\ 2^{-base(1)} \ldots\ldots + p(b\text{-}1).\ 2^{-base(b-1)} \tag{2.1.19}$$

In section 2.2, we propose P^{2}SPT algorithm after those basic algorithms integration.

2.2 P^{2}SPT Algorithm description

In this thesis, we use the Partial & Progressive Signed Power-of-Two (P^2SPT) algorithm to design our echo canceller. Therefore, we present this algorithm here.

First of all, we will define parameters for this algorithm. What is more, functions in this algorithm will be introduced on next page. Moreover, we will describe our P^2SPT algorithm to you in the end. In the first, parameters and vectors as defines as follows:

y_n : ouput signal of n iteration for LMS algorithm.
d_n : desire signal of n iteration for LMS algorithm.
e_n : error signal of n iteration for LMS algorithm.

N : window tap number.
b : bit group coefficient. (the odd number of not zero value)

Δ_l : length of partial update.
δ_b : bound of partial update decision.
δ_s : start tap number of partial update decision.
δl : length of partial update decision.
α : iteration of partial update first.
β : iteration of partial update second.

$\vec{\mathrm{x}}_n$: N numbers input signal vector of n iteration . $\{x_n,\ldots,x_{n-N}\}$
$\vec{\mathrm{w}}_n$: N numbers window coefficient vector of n iteration. $\{w_0,\ldots,w_{N-1}\}$
$\vec{\mathrm{c}}_n$: N numbers counter coefficient vector of n iteration. $\{c_0,\ldots,c_{N-1}\}$
$\vec{\gamma}$: b numbers vector of power-of-two parameter. $\{r_0,\ldots,r_{b-1}\}$

P : N×b matrix of pregresstive coefficient.

Bound function is shown as follows. We will use this function to limit the counter coefficient vector for our hardware design.

$$\text{bound}\{\vec{x}\} = \begin{cases} 2^{2\delta} - 1 & , X(i) > 2^{2\delta} - 1 \quad \text{for } i = 0 \cdots N\text{-}1 \\ 2^{-2\delta} + 1 & , X(i) > 2^{-2\delta} + 1 \quad \text{for } i = 0 \cdots N\text{-}1 \end{cases} \tag{2.2.1}$$

Sign function is shown as follows; this function will be defined to process the vector ***X***.

$$\text{sign}\{\vec{x}\} = \begin{cases} -1 & , X(i) < 0 \quad \text{for } i = 0 \cdots N\text{-}1 \\ 0 & , X(i) = 0 \quad \text{for } i = 0 \cdots N\text{-}1 \\ 1 & , X(i) > 0 \quad \text{for } i = 0 \cdots N\text{-}1 \end{cases} \tag{2.2.2}$$

Binary function is shown as follows; we have to define this function for care about the zero-state occurred.

$$\text{binary}\{x\} = \begin{cases} 0 & , X = 0 \\ 2^{x-1} & , X \neq 0 \end{cases} \tag{2.2.3}$$

Partial function is shown as follows; the update trigger will impact our window coefficients when the partial function is allowed.

$$partial\{\vec{x}_n\} =$$

$$\begin{cases} \vec{x}_n & ,\text{if } (n \,\%\, \alpha = 0) \,\&\, ((\max[\vec{c}(\delta_s : \delta_s + \delta_l - 1)] \leq \delta_b) \,|\, ((\max[\vec{c}(\delta_s : \delta_s + \delta_l - 1)] \geq \delta_b) \,\&\, (n \,\%\, \beta = 0))) \\ \vec{0} & ,otherwise. \end{cases}$$

(2.2.4)

Following parameters and functions we defined, the P^{2}SPT algorithm is presented here. Just like the traditional LMS algorithm, function (2.2.5) and (2.2.6) will produce the error signal $\boldsymbol{e_n}$.

$$y_n = \vec{w}_n^T \cdot \vec{x}_n \tag{2.2.5}$$

$$e_n = d_n - y_n \tag{2.2.6}$$

Than, we fresh and decompose our counter coefficient vector $\boldsymbol{c_n}$.

$$\vec{c}_n = \text{bound}\{\vec{c}_n + partial(\text{sign}\{e_n \cdot \vec{x}_n\})\} \tag{2.2.7}$$

$$\mathbf{P}(k,i) = \left\lfloor \frac{\vec{c}_n(k)}{4^i} \right\rfloor \% 4 \qquad for\ i=0\cdots b-1\ \&\ k=0\cdots N-1 \tag{2.2.8}$$

We will have new window coefficients in next iteration, when the updated operation is finished by the progressive coefficient matrix $\mathbf{P}(k,i)$ drive the base.

$$\vec{w}_{n+1}(k) = \sum_{i=0}^{b-1} \text{binary}\{\mathbf{P}(k,i)\} \cdot 2^{-F(i)} \qquad for\ k=0\cdots N-1 \tag{2.2.9}$$

Considering our Architecture design, we support window coefficients as shows as Fig. 11 P^{2}SPT algorithm window coefficients in next page.

In Fig. 10, we have 3 bases, and our update parameter only allows 0, 1, 2, and 4 to drive our bases form all 0 to all 4. In this case, window coefficients will be showed in Fig. 10.

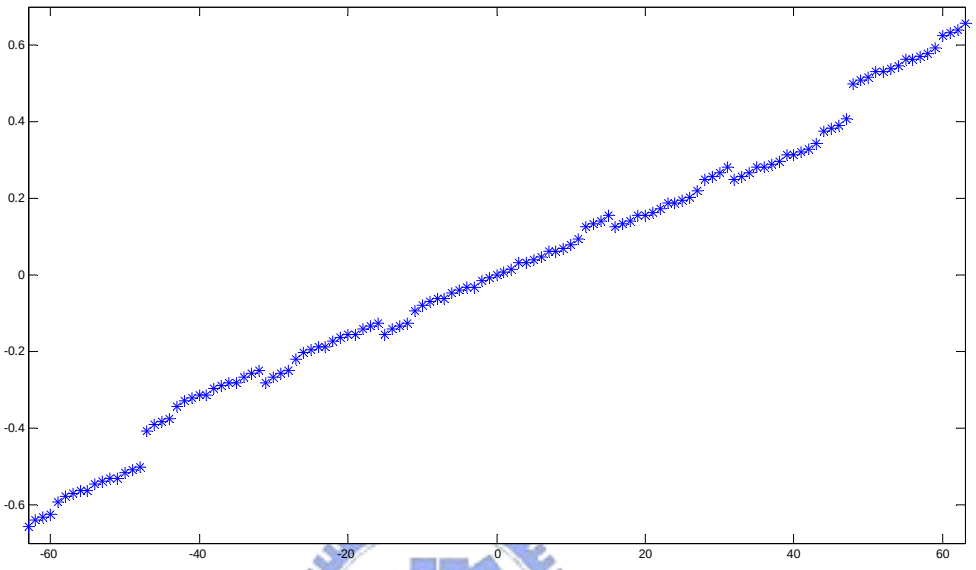


Fig 10. P^2SPT algorithm window coefficients

In order to have the minimum power consumption of echo canceller, we have developed this algorithm to obtain our goal of simplest architecture cost. In next chapter Acoustic Feedback Model Construction & Simulated Results, we will introduce the acoustic feedback model and simulate this algorithm to verify its performance.

Accordingly, we will prove our algorithm’s performance as same as others, and based on our algorithm we can construct to simplest architecture.

Chapter 3
Acoustic Feedback Model Construction & Simulated Results

In chapter 3 we will build a system model for realized acoustic feedback characteristics and help us to adjust the design of echo canceller. Moreover, we will also verify our P^2SPT algorithm's performance by this model.

In section 3.1, we will start to construct our acoustic feedback model. For explain the operation of our model, a simple example will be presented in the end of this section. Following section 3.1, the section 3.2 will shows the simulated results for you.

3.1 Model Construction

We will present section 3.1.1 model introduction to introduce our acoustic feedback model. Accordingly section 3.1.1, the example for model operation is showed in section 3.1.2.

3.1.1 Model introduction

Our model can be separated out of three parts in Fig. 12; the echo channel, the forward path and the echo canceller. There also have a two domain need to consider, the analog domain and the digital domain. We describe this model in next page.

We show Fig. 11 Diagram of Acoustic feedback model as follows.

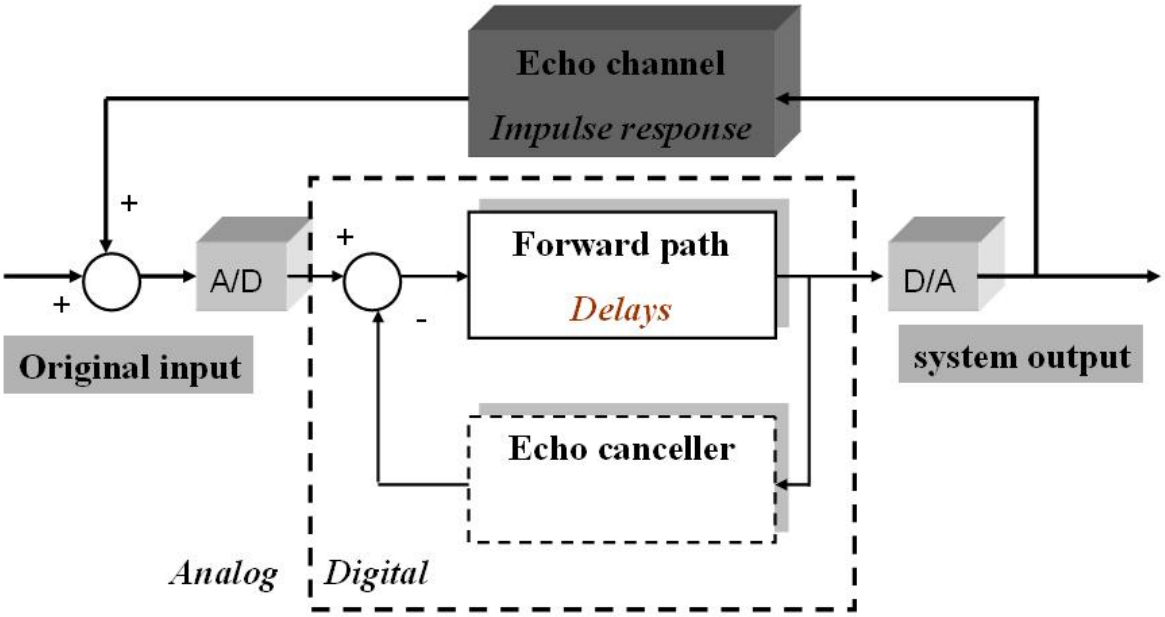


Fig 11. Diagram of Acoustic feedback model

First of all, let us consider the truncation error in digital domain. We will choose 12 bit to be one word. When the echo canceller has signal processing, this condition will be adopted. Furthermore, we are setting the 16K Hz sampling to the analog-to-digital converter.

In Fig. 11, the model we built is including three parts.

1. Echo channel.
2. Forward path.
3. Echo canceller.

Now let us give a clear explanation for those three main parts as follows.

1. Echo channel

According 1.2.1 Overview of AEC in hearing aids, we know ITE hearing aids will have echo noise exist. Therefore, we show the path of acoustic feedback in Fig. 12 [2].

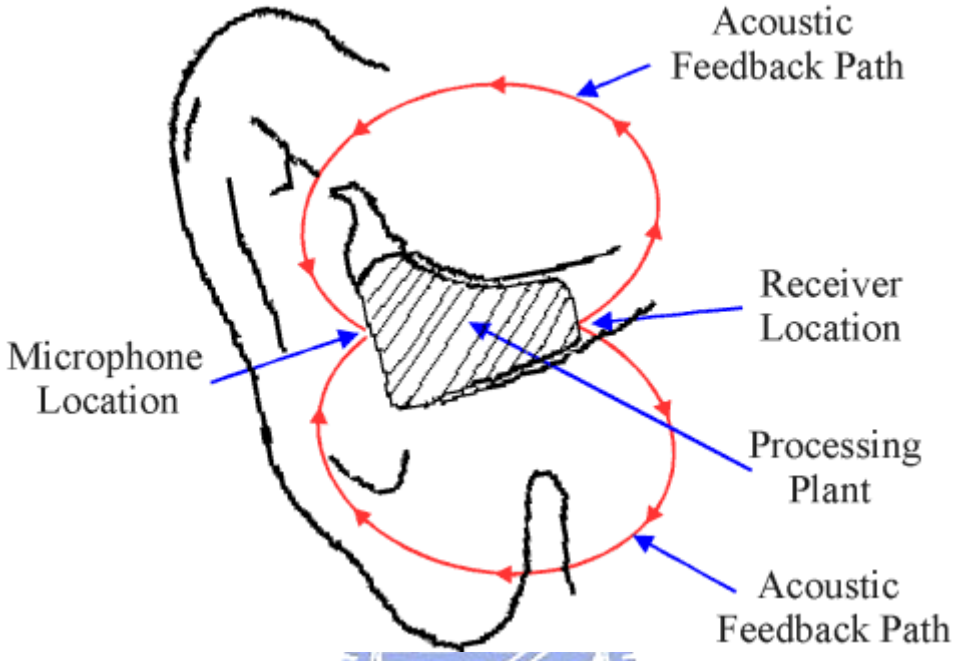


Fig 12. acoustic feedback path in the ear

In Fig. 12, we can find out the feedback channel. In consequence, the acoustic feedback channel means all the noise come from the receiver. For this reason, we can say “the path” is a response from the receiver’s signal.

The acoustic feedback channel modeling [27] [28] [29], usually using the dummy head to describe the characteristic of echo path. For example, we according to the data form [30] and show you how to catch the characteristic of acoustic feedback path.

First of all, the testing platform of dummy head will be setup already. Than, using the Sweep Stimulus method and the White Noise method to trigger the echo channel. Actually, we can also to say that it is tried to find out the impulse response in the time domain or the frequency response in the frequency domain.

We show testing platform of the dummy head in Fig. 13.

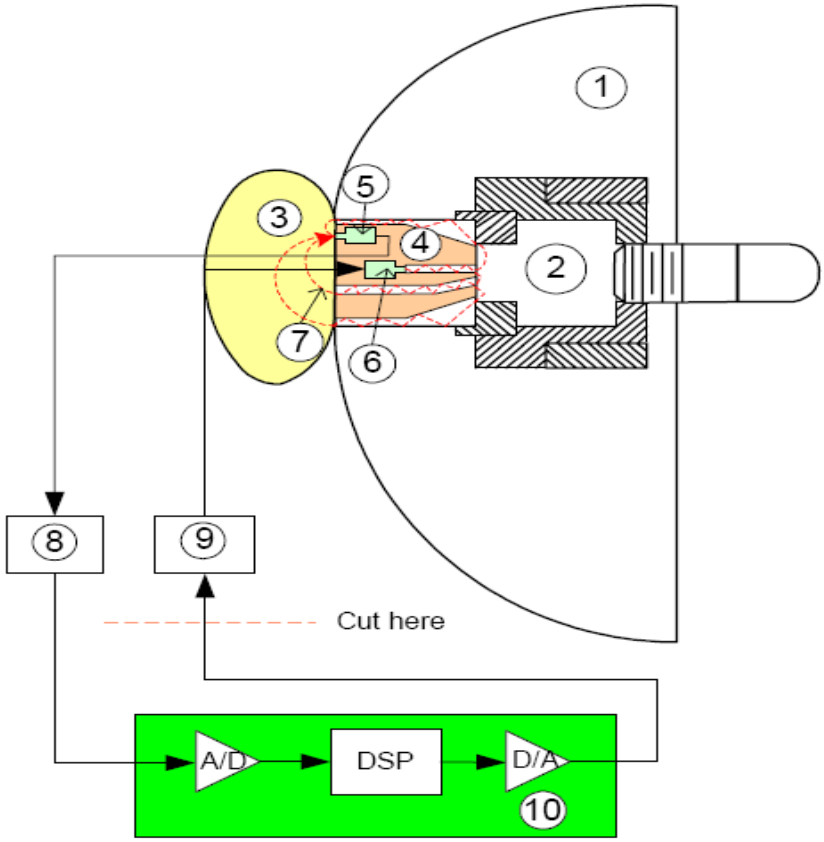


Fig 13. testing platform of dummy head

Legend for Fig. 13:

(1) B&K head and torso simulator type 4128C
(2) B&K right ear simulator type 4158C
(3) Pinna simulator
(4) ITE hearing instrument shell with a microphone and receiver
(5) Knowles EA-1843 microphone
(6) Knowles BK-1604 receiver
(7) Acoustical feedback path
(8) Interface between microphone and ADC
(9) Interface between DAC and receiver
(10) EZ-KIT LITE ADSP 2189M Evaluation Board, including a codec AD73322 and a DSP 2189

Based on this platform, we can describe the frequency response of Acoustic feedback channel like Fig. 14.

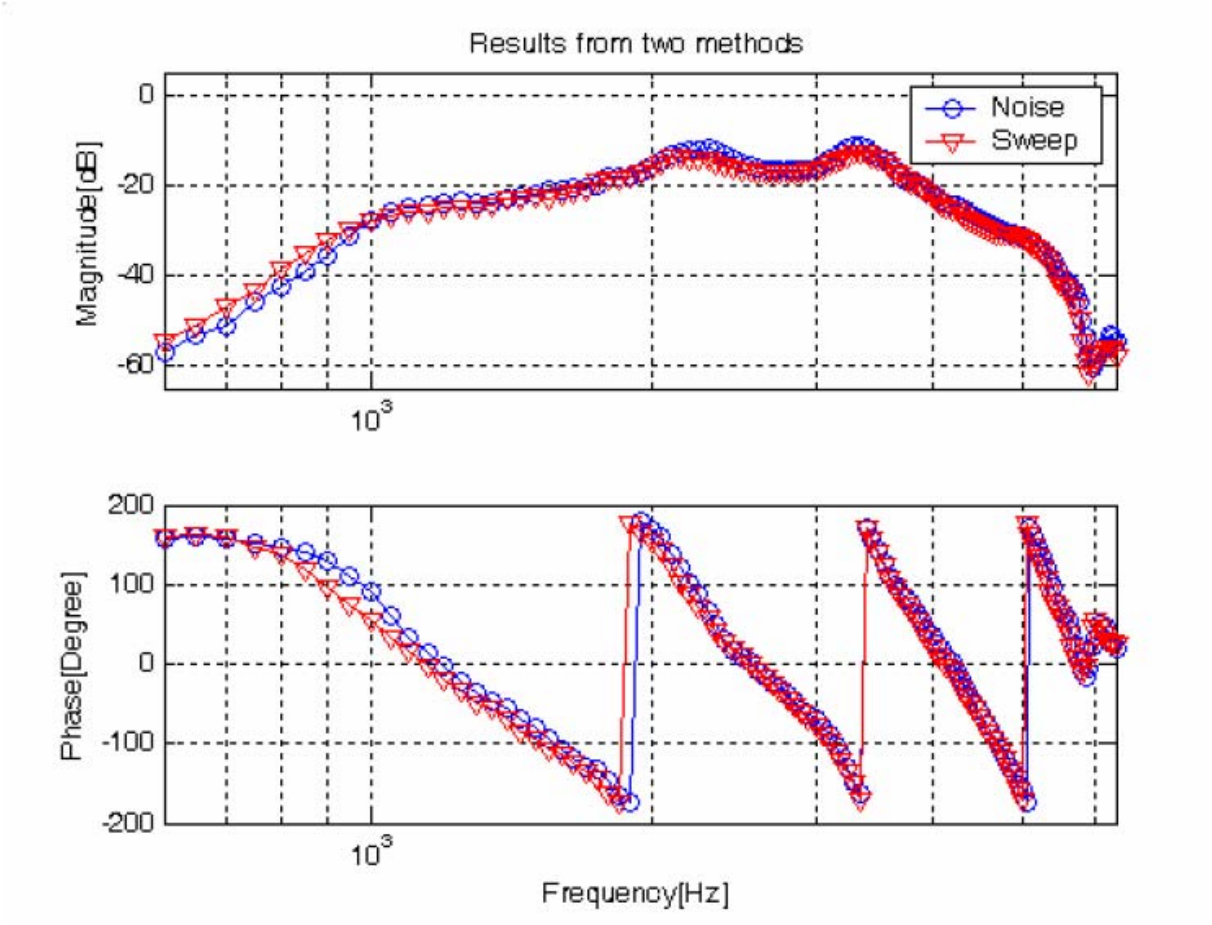


Fig 14. frequency response of Acoustic feedback channel

In Fig. 14, there means Sweep Stimulus method and White Noise method will have the same result. We can find out the similarly response. According to this result, we can identity frequency response of acoustic feedback channel is just like Fig. 14. We can describe frequency response of our echo channel in Fig. 16. We show frequency response of our echo channel in Fig. 15.

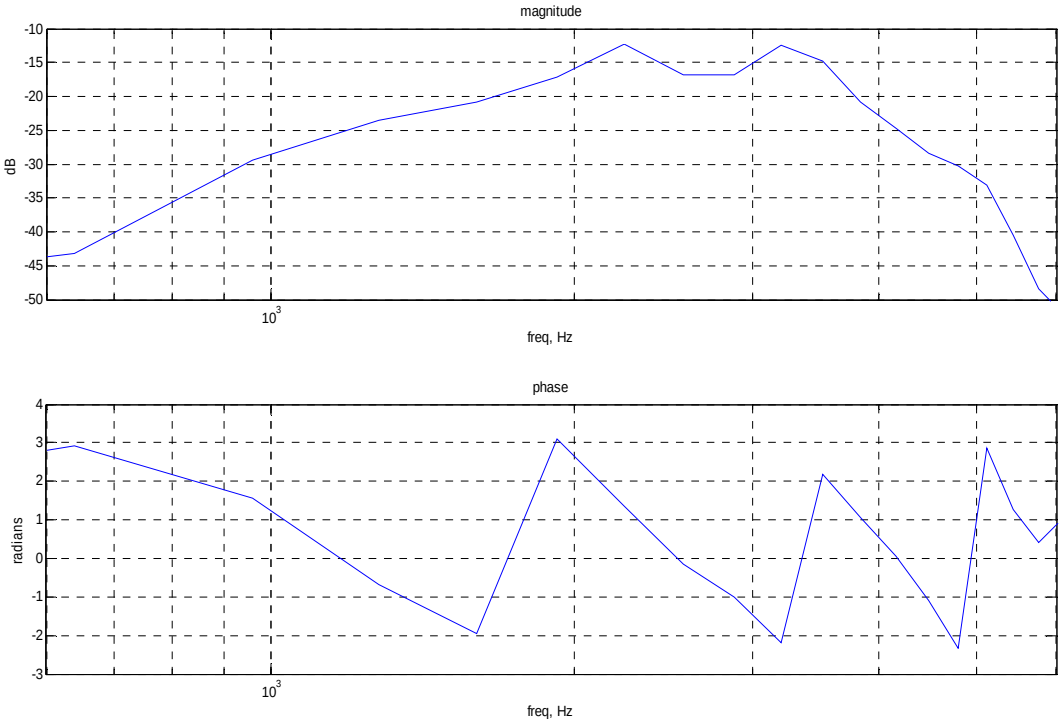


Fig 15. frequency response of our echo channel

In order to simulate echo noise effect, we need to do “convolution”. When system output signal feedback to microphone by echo path, the real echo noise is the convolution by output signal and echo channel.

In a word, we will simulate our model in time domain. Consequently, we will need this echo channel impulse response.The echo channel will be changed to the 100 taps Finite Impulse Response (FIR). The result is on Fig. 16. Fig. 16 shows the Finite-Impulse-Response (FIR) of our model’s echo channel.

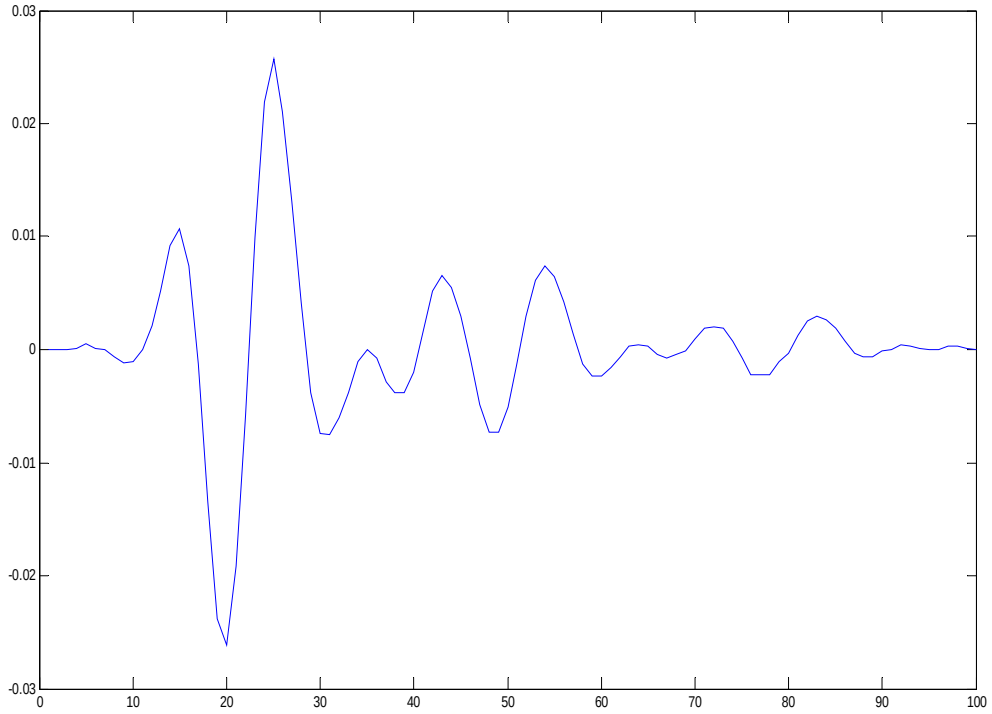


Fig 16. FIR of our echo channel

On other hand, the real echo feedback path is not steady. It will change with time. According to information from [31], we can list some characteristics for the echo feedback path in next page.

The variation is depending on user's physiological structure of ear and head in the feedback path of ITE hearing aids. Fig.17 will show the ear is a complicated structure.

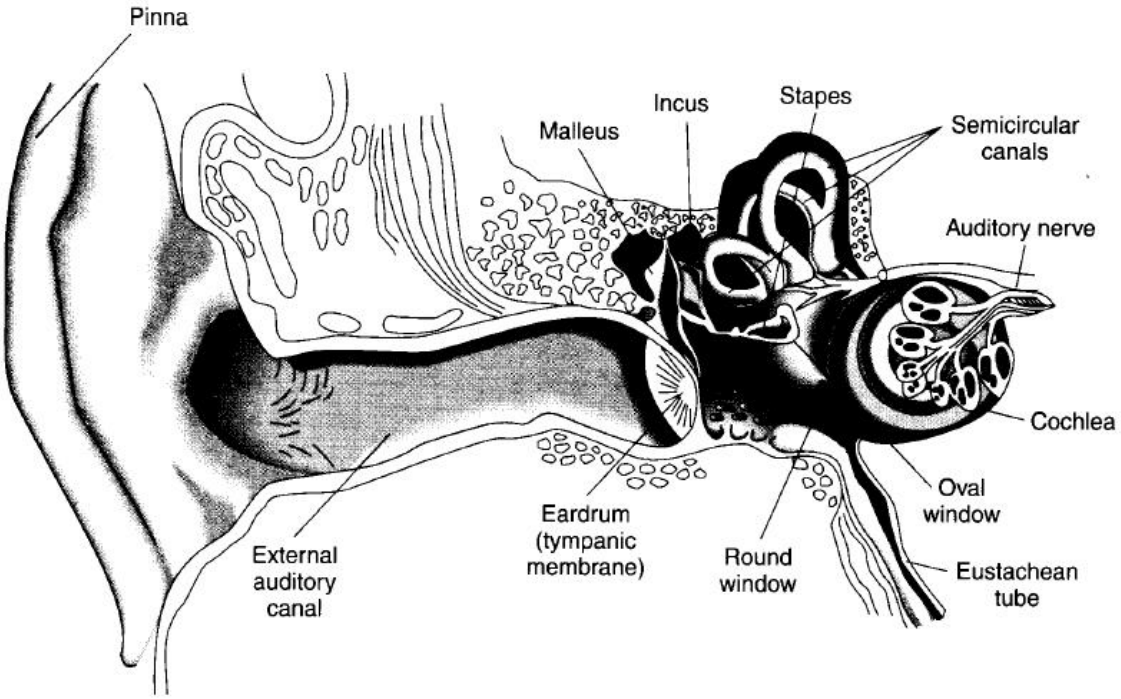


Fig 17. Sketch of human ears

According to data we know, the variation statistics in the feedback path as shows as follows:

- The feedback path will have a ten percent of variation in the amplitude.
- The feedback path will change of 3~5ms.

Consider the variation of feedback echo path. We assume our echo channel will be changed by every 5ms, and the value of echo will have 10% variation with gauss random probability. Moreover, we also have two iteration delays variation of the echo channel with gauss random probability.

2. Forward path

This part is to simulate other function of hearing aids. In fact, forward path will include like compression part and band filters. But in this thesis, we only consider the effect of echo canceller. Therefore, we will use delays to

replace all functions in the forward path. In another word, the forward path just delays the input signal. This result will let us easy to know the performance of echo canceller by Mean-Square-Error (MSE) or human's ear verification. Besides, the forward path delays we choose are 50 iteration delays. Fig. 18 shows Diagram of forward path.

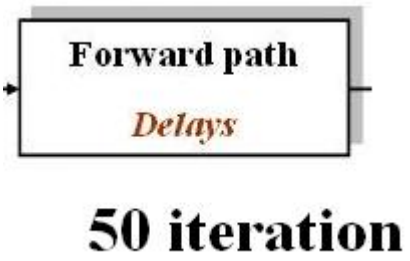


Fig 18. Diagram of forward path

3. Echo canceller

Echo canceller part can be any echo canceller we like, if the echo canceller is adaptive filter form. Therefore, we draw the diagram of dotted line. Furthermore, we will use a basic adaptive echo canceller (LMS filter) to show our model's operation, and use our algorithm to prove the performance in section 3.2. Fig. 19 shows Diagram of echo canceller.

Fig 19. Diagram of echo canceller

3.1.2 Example for Model's Operation

We will describe our model's operation by an example. The model will be introduced in section 3.1.1. Therefore, the echo canceller is a LMS filter and work on the 12 bits and 16 KHz sampling rate.

First of all, we assume that the original signal is like Fig. 20. The Y-axis is from -0.8 to 0.8 and X-axis from 0 to 40000 (unit: iterations). This signal is the voice by man. The signal length as longs as one second.

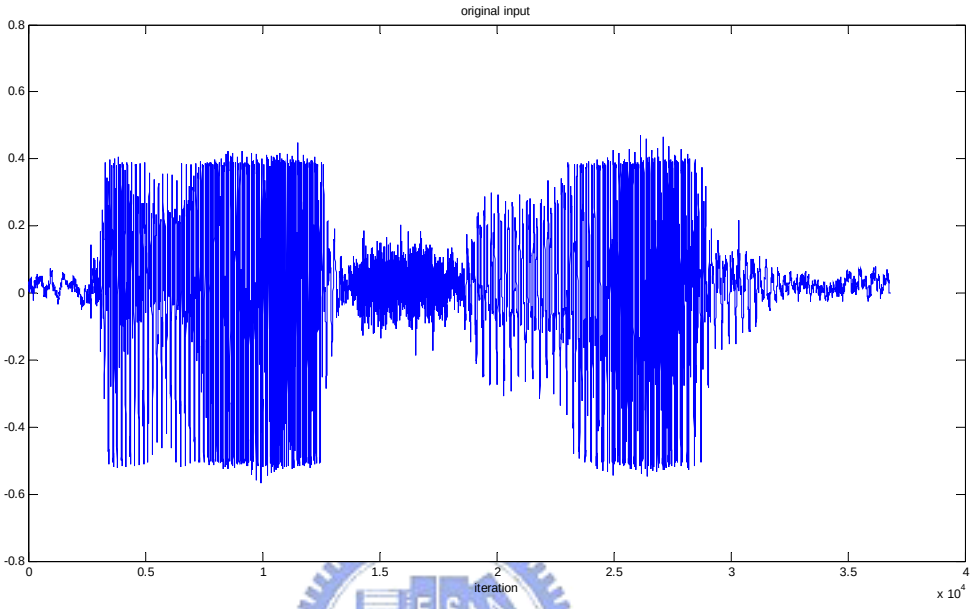


Fig 20. Waveform of the original signal

This operation of our model is input this original signal into the forward path, there means delay by 50 iterations than the signal will output to be the system output. In consequence, this system output will start feedback to interfere original signal by echo path. Furthermore, the echo canceller will use this system output to make the echo canceller's output.

Those results will show in the Fig. 21 Waveform of the echo canceller output signal and Fig. 22Waveform of the system output signal.

Fig. 21 shows us the echo canceller output signal.

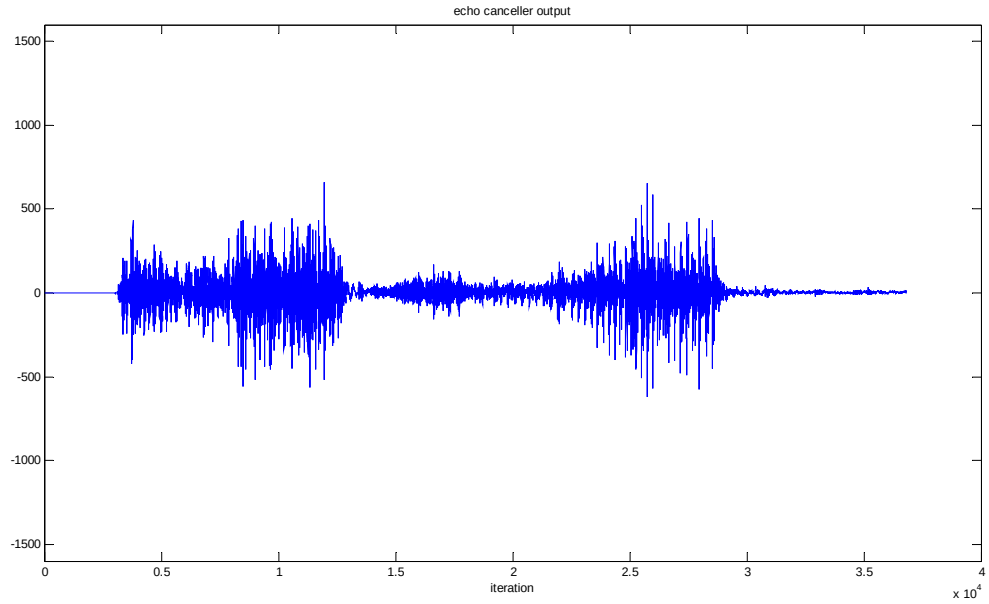


Fig 21. Waveform of the echo canceller output signal

In Fig. 22, the system output signal with echo canceller is presented. Because echo canceller's output will cancel real echo noise in our model, this output signal is to be similar with original voice.

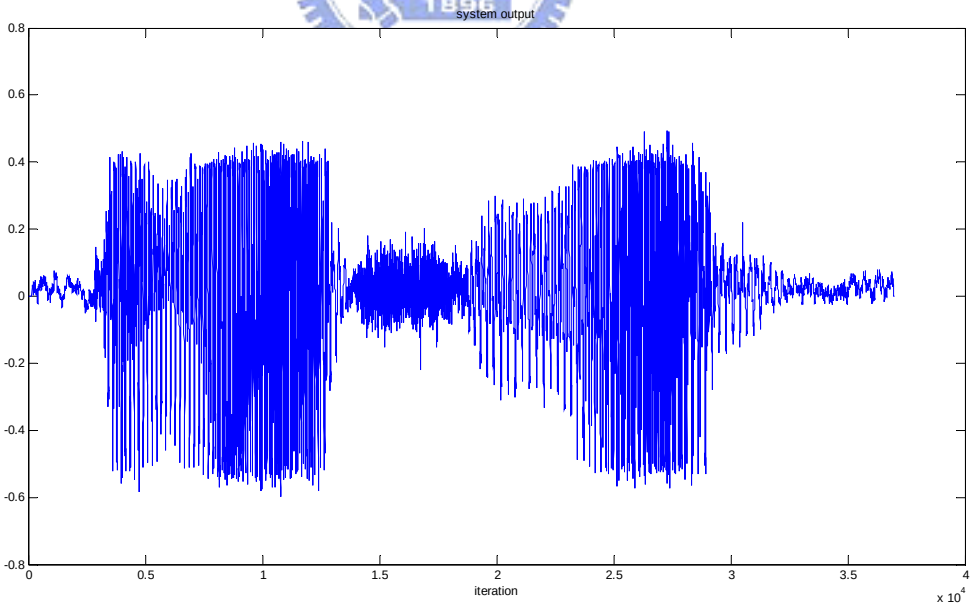


Fig 22. Waveform of the system output signal

Now, if we turn off this echo canceller. The system output will be look

like Fig. 23. In this case, the sound of this signal will bring a bleep that we called echo noise.

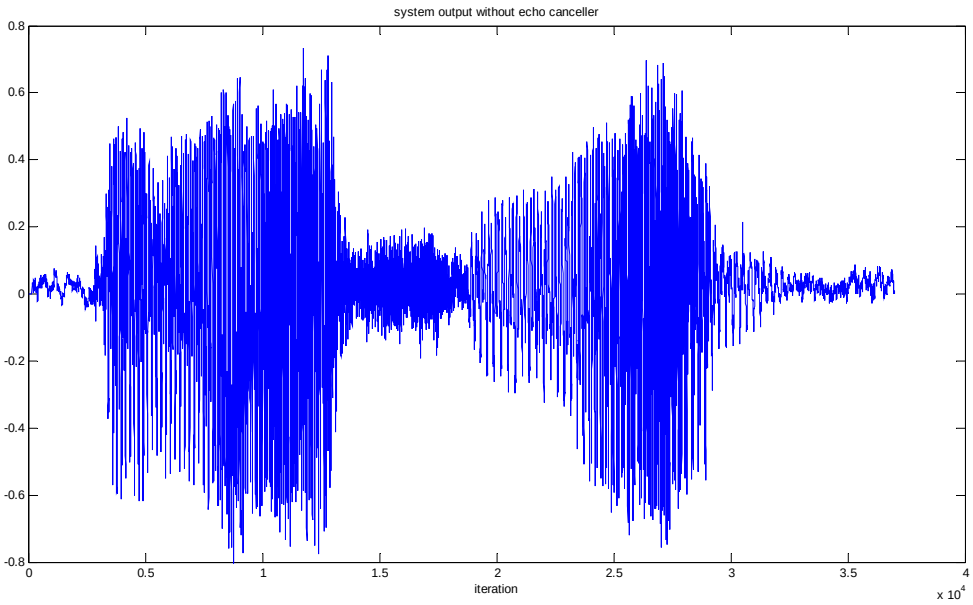


Fig 23. Waveform of the system output signal without echo canceller

In this example, we try to show the performance of system output between enable or disable this echo canceller. Besides, we will show MSE (Mean Square Error) of system output to original input signal in Fig 24 MSE with echo canceller and Fig 25 MSE without echo canceller.

There two can give us the data of voice's performance compared. Because we want to find out the performance of human's hearing experience, not the filter speed. We use this model to verify the performance that we want to know like this example.

Consequently, we will use the same data like this example to show you that our algorithm's performance and others in next section. Moreover, we will show the (Signal Noise Ratio) SNR and the voice's Spectrogram to prove that our algorithm has the same performance to other LMS-based algorithms.Fig 24 shows the MSE (with echo canceller) of system output to original input.

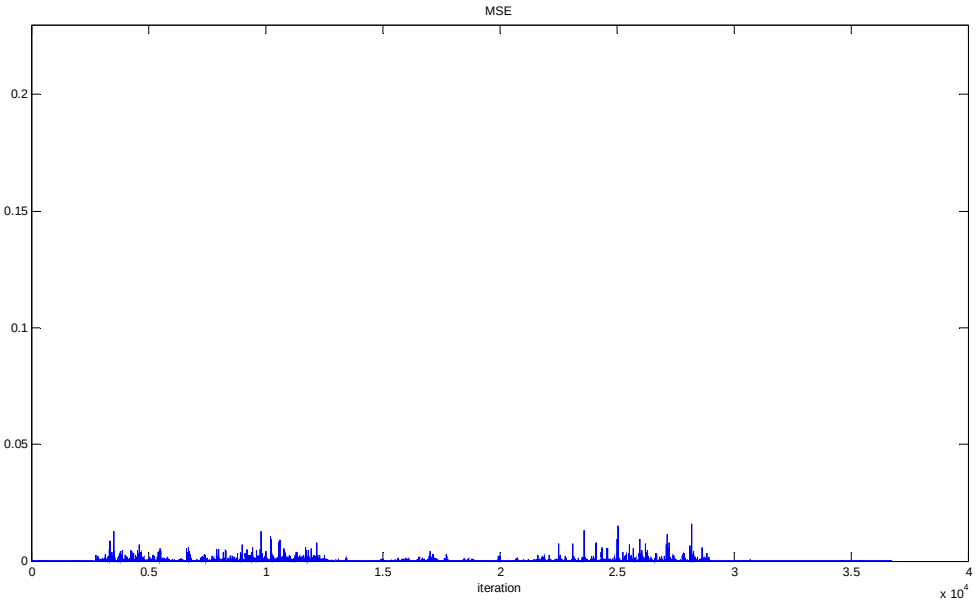


Fig 24. MSE (with echo canceller) of system output to original input

Fig 25 shows the MSE (without echo canceller) of system output to original input. We can see the difference between Fig. 24 and Fig. 25, if we try to compare these two figures.

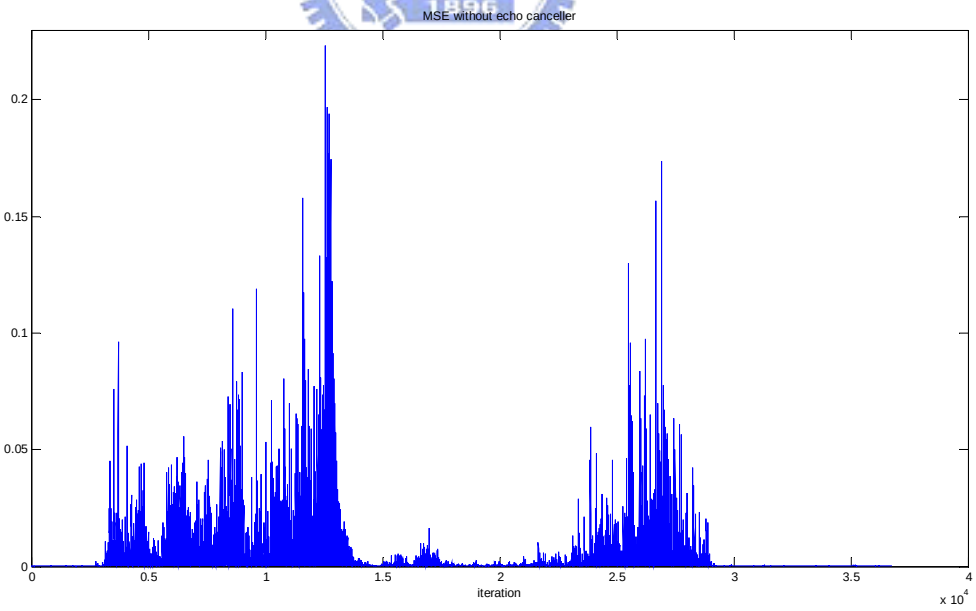


Fig 25. MSE (without echo canceller) of system output to original input

3.2 Simulated Results

In this section, we will simulate our P^{2}SPT algorithm on the MATLAB. According to section 3.1, our model's Specification will be ordered in section 3.2.1.

In section 3.2.2, we will compare our algorithm with other well-know LMS-based algorithms. In section 3.2.3, we will show the performance on human's voice test. We will poof our algorithm has good performance as same as other algorithms in the performance of echo cancellation. Besides, we will have simplest architecture in our hardware design that we need.

3.2.1 Simulation model Specification

Fig 26as shows as follows; we will have a clearly description of this model in next page.

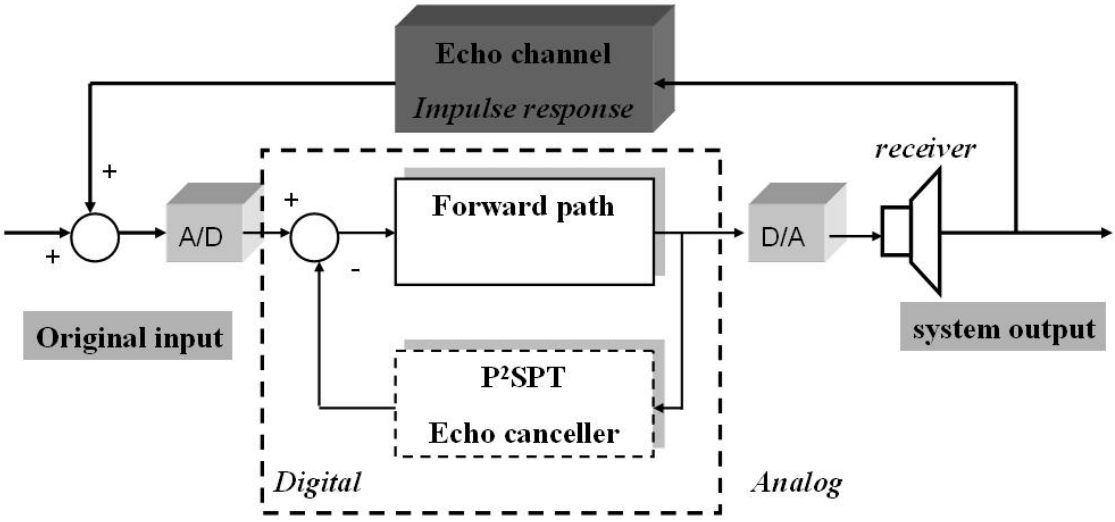


Fig 26. model of simulation

In Fig 26, we will have two domain and four main units in our model. First, we define sample rate will be 16 KHz, as same as our hardware specification.

In the next, we start to discuss those two domains. Our design will work on the digital domain. Therefore, we will have the truncation error problem that we have to consider about.

In addition, the unwelcome really echo noise is an analog signal. Considering this reason, when the signal is processed in the digital domain we will transform our signal to 12 bit precision.

Considering four main units in Fig 26, we describe the forward path first. In reality, the forward path means the compression and other functions in the hearing aid. But we are only considering the echo cancellation in this thesis. For this reason, we just assume our forward path is 50 iteration delays.

In the echo canceller unit, we design this echo canceller using the P^2SPT algorithm. Moreover, we describe this unit as same as our hardware design. In consequence, we can verify our hardware design of compared the output signal for this simulation model and our real design.

Echo channel is a matrix. This matrix saves the echo impulse response which introduced in section 3.1. Furthermore, we change this channel with every 3ms on the variation condition that also introduced in section 3.1 too.

In the last part, the receiver effect will be considered. In practice, our receiver will amplify the system output. Accordingly, we have to assume the worst case in our echo model. We will have to amplify our system output to 12db.In next page, we form that information of this model in table 1 information of simulation model.

Table 1 information of simulation model shows simulation model's information as follows:

table 1. information of simulation model

Specification	**Description.**
Digital sample rate	16K (Hz)
Digital Bit number	12 (bit)
forward path	50 delays (iterations)

Echo canceller	P^2SPT (using our algorithm) Also can change to other algorithms.
Echo channel	Matrix (1x100) Channel change every 3ms. (Variation with Gauss random) 90% ~ 110% of amplitude. -1 and 1 delay iteration will plus to channel.
Receiver effect	Our system output Amplify to 12db.

In next section, we will start compare our algorithm with other well-know LMS based algorithms in this model. Actually, we are not proved that our algorithm is faster or more accurately than others. The result we want to show you that is our algorithm has the same performance in human's hearing experience. As we can know, our user will feel nothing of echo noise when they wear the hearing aid, if the feedback echo noise is under the allowed range [32].

For that reason, we can have the simplest hardware design for low power issue and our user also not interfere by echo noise.

3.2.2 Algorithms compared

In this section, we will prove our algorithm's performance will be as same as other algorithm's performance in echo cancellation of the model we introduced in section 3.2.1.

In Fig. 27 learning curves of four algorithms, we will show the LMS, DLMS, NLMS and our P^2SPT algorithm's learning curve.

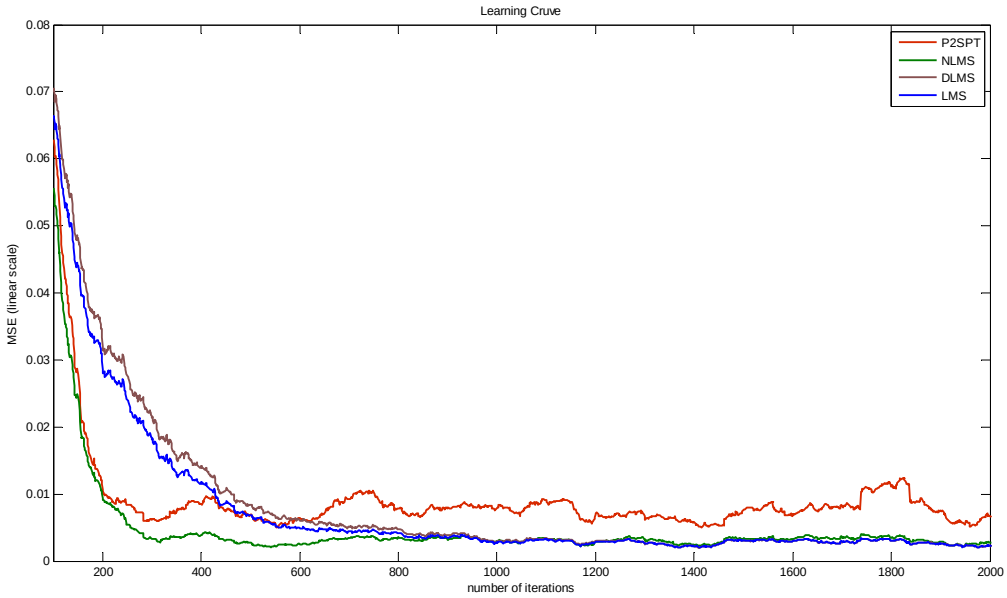


Fig 27. learning curves of four algorithms

Our input signal is a random white noise signal between 1 and -1. Moreover, we also put in 10% noise to the input signal. All algorithms will have 32 taps and LMS algorithm's step-size is 0.1, DLMS algorithm's step-size is 0.1 too. Considering the model we built, the channel of we learning is the echo channel that we introduced in section 3.1. in Fig 27, if we compare speed of these algorithms, we have the order from fast to slow will be:

- **NLMS > P^{2}SPT > LMS > DLMS.**

If we compare precision of these algorithms, we can find out the order from batter to bad precision will be:

- **NLMS > LMS > DLMS > P^{2}SPT.**

In fact, we want to compare the performance of "echo canceller's algorithms", not the filter speed or precision. Therefore, we will use the model (introduced in section 3.2.1) to verify all of algorithm's performance of echo cancellation.

Consequently, to compare the speed or the precision of those filters can not really find out the result we want. Only to compare those algorithms by real voice's signal and have real echo noise occurred, the result will mean real performance of echo cancellation.

First of all, we will show the LMS algorithm. Next, we will have the DLMS algorithm and the NLMS algorithm. Finally, we will show our P^2SPT algorithm's performance and the performance of not using echo canceller to you.

LMS algorithm:

In next page, we will have the information of four types in Fig 29 LMS algorithm's performance. We have the original input signal in lower left-hand corner, and system output in lower right-hand corner. In upper left, we have the echo canceller output. In upper right, we show the MSE of original input and system output.

Accordingly, Fig. 29 Fig 30, Fig. 31 and Fig. 32 are following this order. Besides, we will show the MSE result of those algorithms in the end.

Fig. 28 as shows as follows:

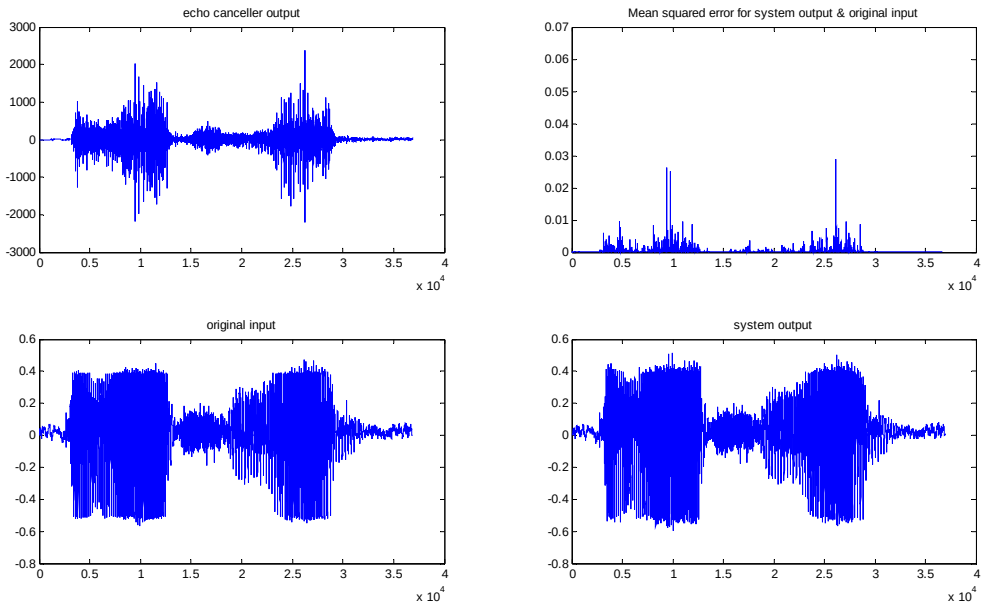


Fig 28. LMS algorithm's performance

DLMS algorithm:

Fig. 29 as shows as follows, our delay parameter is four iterations.

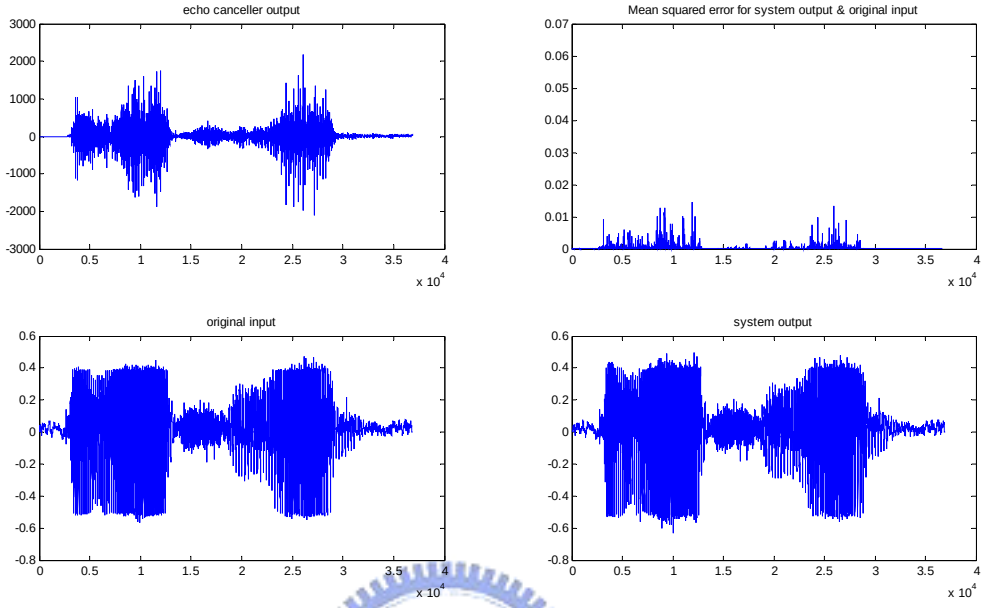


Fig 29. DLMS algorithm's performance

NLMS algorithm:

Fig. 30 NLMS algorithm's performance as shows as follows:

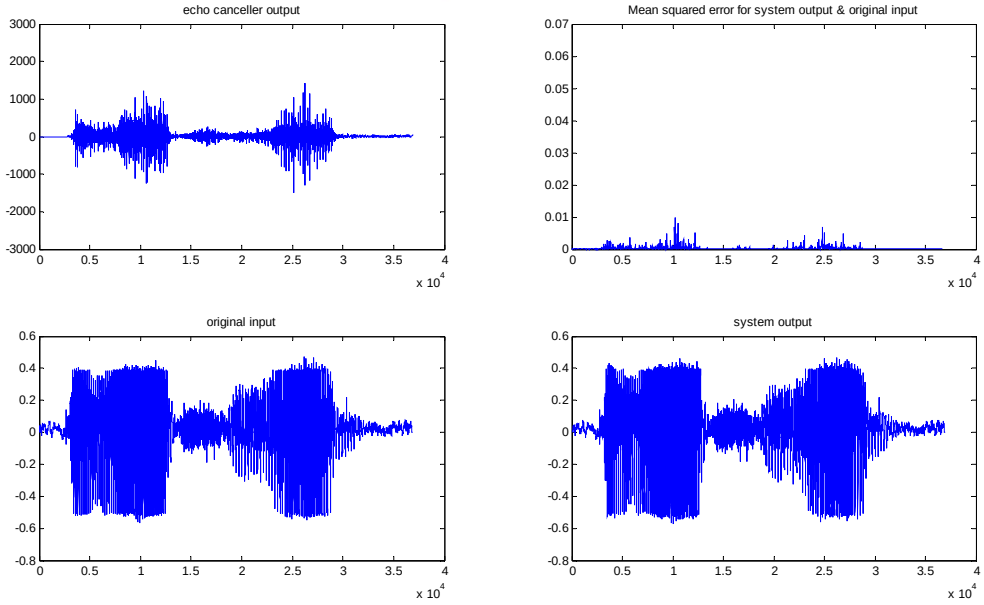


Fig 30. NLMS algorithm's performance

P^{2}SPT algorithm:

Fig. 31 P^{2}SPT algorithm's performance as shows as follows:

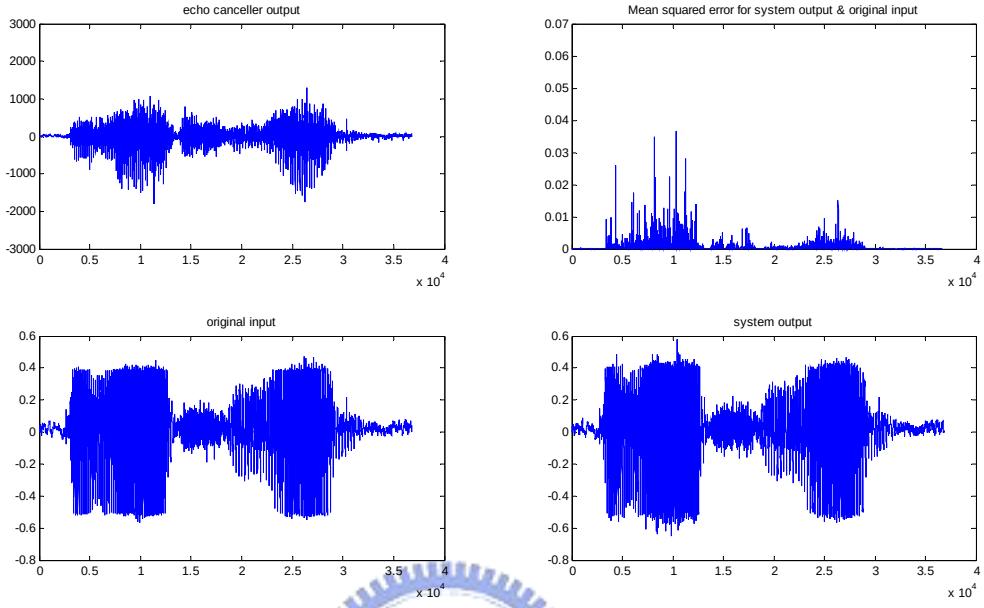


Fig 31. P^{2}SPT algorithm's performance

Using no echo canceller:

Fig. 32 no echo canceller's performance as shows as follows:

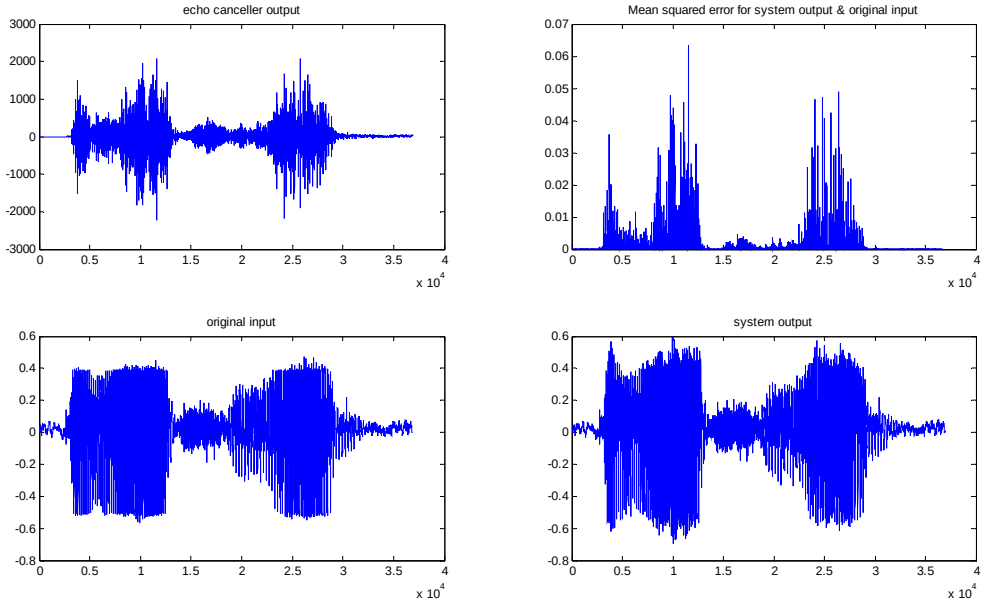


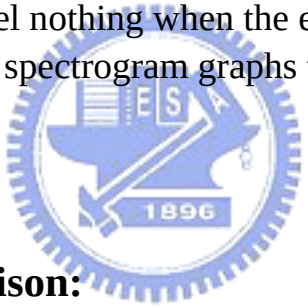
Fig 32. no echo canceller's performance

We use table 2 MSE of algorithms to show the performance of those algorithms.

table 2. the performance of algorithms

Algorithm Type	**NLMS**	**LMS**	**DLMS**	**P^{2}SPT**	**Using NO canceller**
MSE (unit: 10^-4)	0.2297	0.421	0.4174	1.93	13.36
SNR (db)	32.79	30.16	30.19	23.55	15.14

Our algorithm's performance look likes not better than others. But in the users hear, all of algorithms can cancel the echo except using no echo canceller. In fact, human's hearing feel nothing when the echo noise under the allowed range [32]. We will use the spectrogram graphs to prove this point in next page.



Spectrograms comparison:

We have the original input wave in upper left-hand corner, and the spectrograms of original input in upper right -hand corner. In lower left, we have the spectrograms of system output without echo canceller. In lower right, we show the spectrograms of P^{2}SPT system output. Fig. 33 as shows as follows:

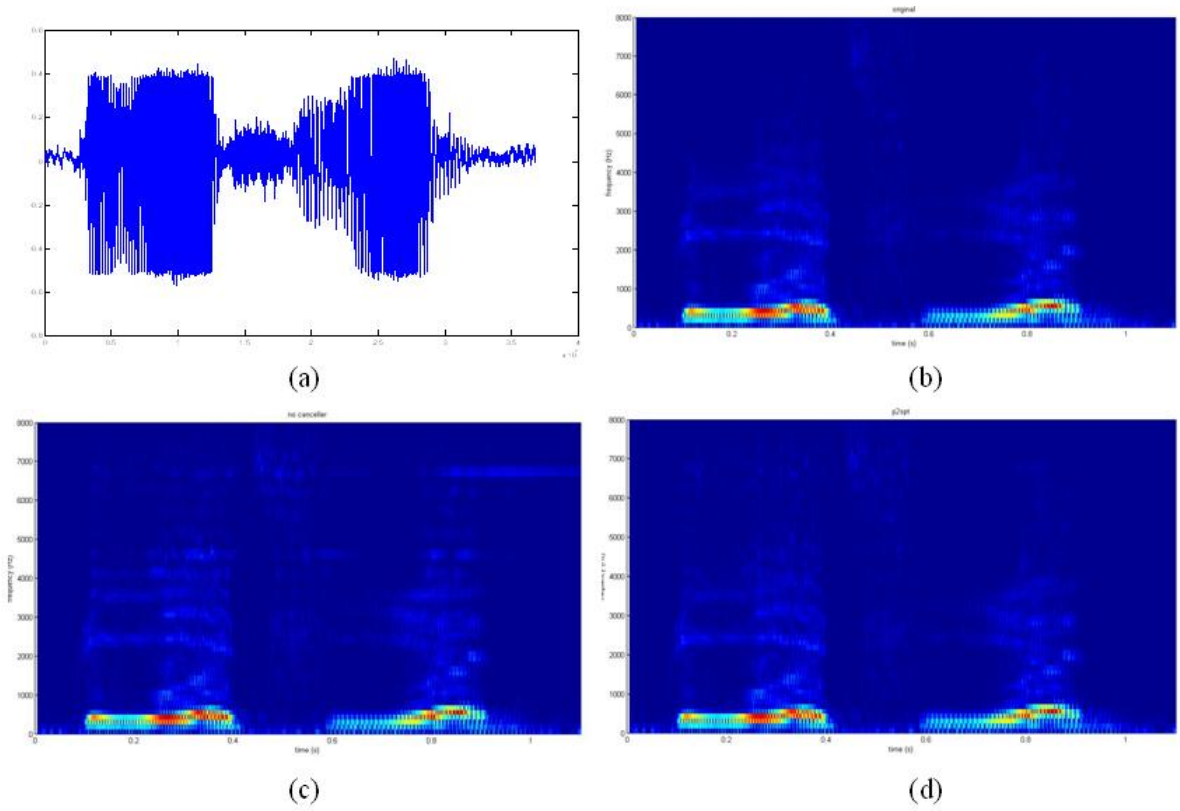


Fig 33. (a) original input wave. The spectrograms of (b) original input (c) system output without echo canceller (d) P^2SPT system output

To compare with original input, we can find out the different between system output without echo canceller and P^2SPT system output. The high frequency energy occurs in system output without echo canceller. We will compare our P^2SPT system output with others in next page. Therefore, we can analyze those spectrograms plots to prove there no different in user's experience.

We have the spectrograms of NLMS system output in upper left-hand corner, and the spectrograms of LMS system output in upper right -hand corner. In lower left, we have the spectrograms of DLMS system output. In lower right, we show the spectrograms of P^2SPT system output. Fig. 34 as shows as follows:

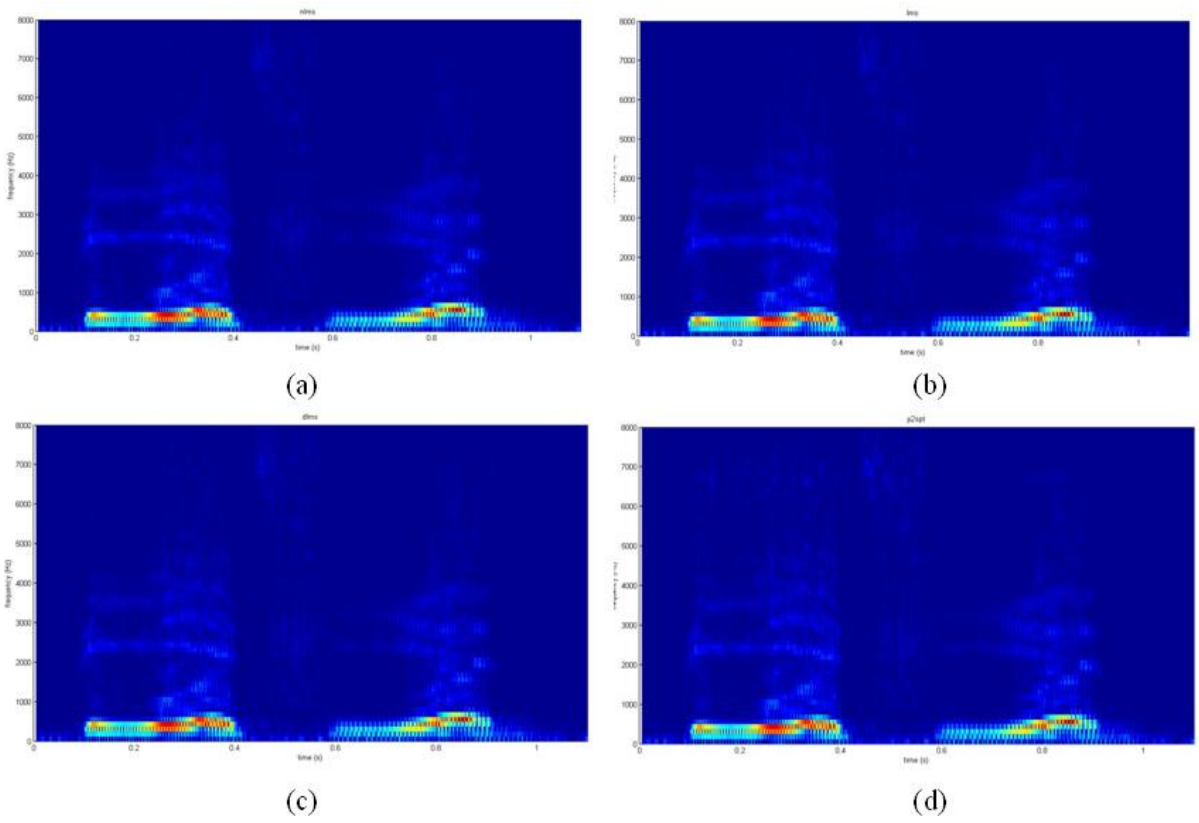


Fig 34. The spectrograms of (a) NLMS system output (b) LMS system output (c) DLMS system output (d) P2SPT system output

In Fig. 34, there are not evidence that our algorithm's performance is bad than others. In fact, our algorithm only increases a little bit high frequency energy and there are not occur any effect in user's hearing experience.

3.2.3 Performance on human's voice

In this section, we will show you that our algorithm has good echo cancellation performance in human's voice input. In consequence, we will use the same model to verify algorithm's performance on human's voice testing.

In fact, all we really want is reduced the power consumption of echo canceller. Therefore, to compare the speed or the precision is not the point we considered. For that reason, we only to show you that the echo noise will be eliminated in spectrogram and waveform graphs.

First, we will show the test of 4 types human's voice to you. Moreover, considering the testing of long voice's stability and forward path changed, we will verify those two parts in the end.

In human's voice testing, we present four voice's types as follows:

- Man's voice.
- Woman's voice.
- Boy's voice.
- Girl's voice.

For those different types, we will show you that the echo cancellation performance of our P^2SPT algorithm in spectrogram and waveform graphs.

Accordingly, those four types will have different echo noise response. Consequently, Boy's voice will be lowest frequency voice and woman's voice is highest frequency voice in this test. Therefore, we can prove to you that our algorithm is real useful of echo cancellation in hearing aid.

Man's voice:

First of all, we assume that man's original signal is like Fig. 35. The Y-axis is from -1 to 1 and X-axis from 0 to 12000 (unit: iterations).

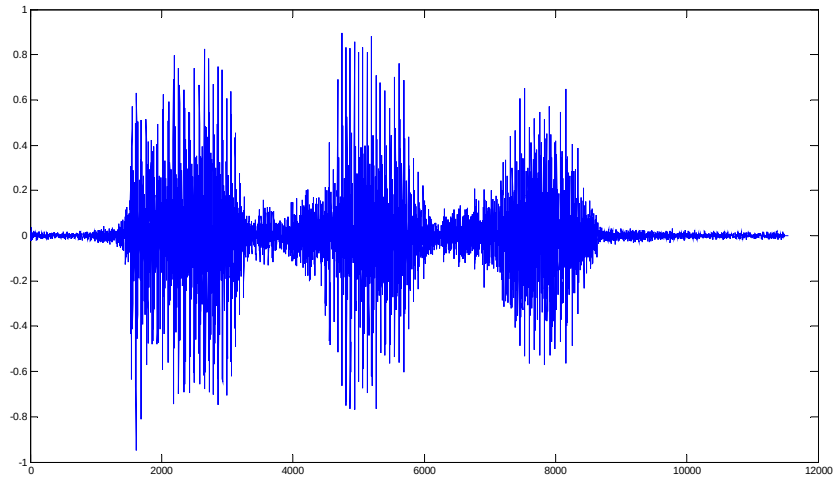


Fig 35. Waveform of man's original signal

In addition, we show echo canceller output in Fig. 36 Waveform of man's echo canceller output signal. (12bit: value 2048 means 1)

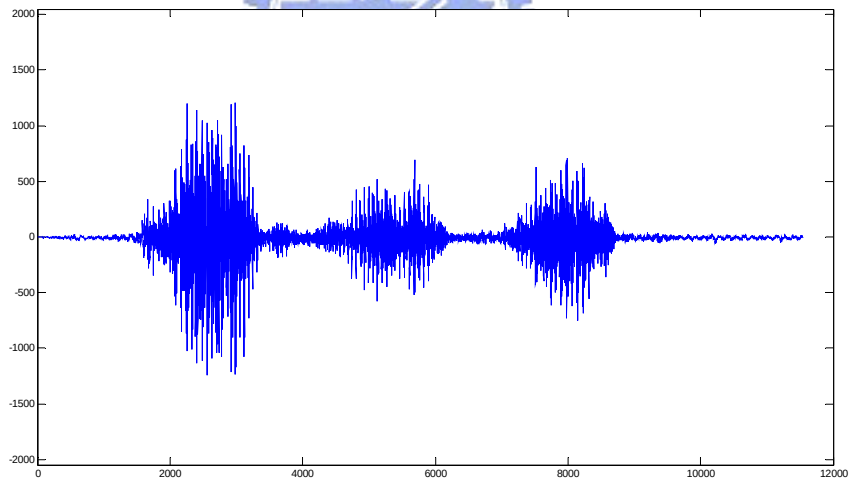


Fig 36. Waveform of man's echo canceller output signal

Our system output will as shows as follows; under the echo canceller work.

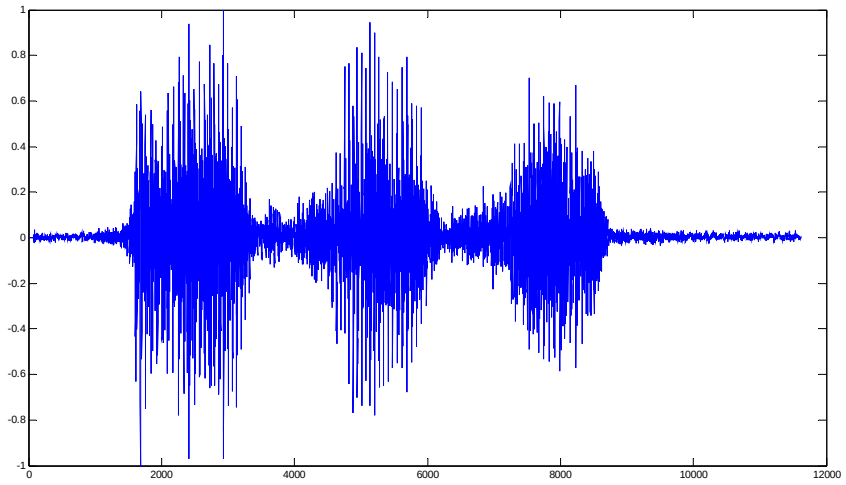


Fig 37. Waveform of man's system output signal

Now, if we turn off the echo canceller. The system output will be look like Fig. 38. In this case, the sound of this signal will bring a bleep that we call echo noise. (Y-axis also from -1 to 1)

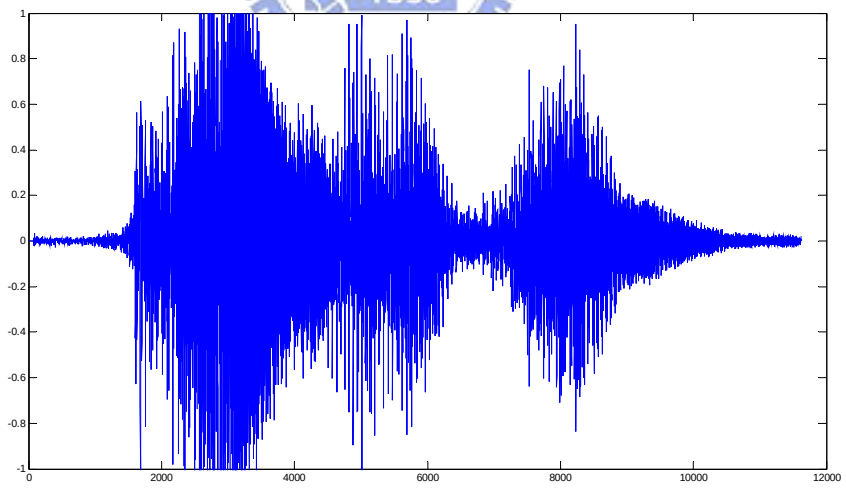


Fig 38. Waveform of man's system output signal without echo canceller

We will show MSE of system output to original input. Also, we will show MSE without echo canceller too. Fig. 39 shows MSE (with echo canceller) of

system output to original input. (Y-axis is from 0 to 2)

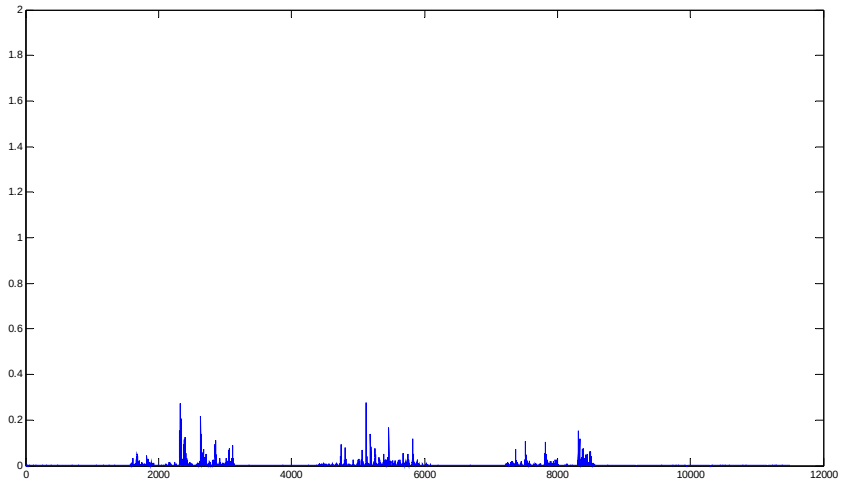


Fig 39. MSE (with echo canceller) of man's testing

Fig. 40 shows MSE (without echo canceller) of system output to original input. (Y-axis is from 0 to 2)

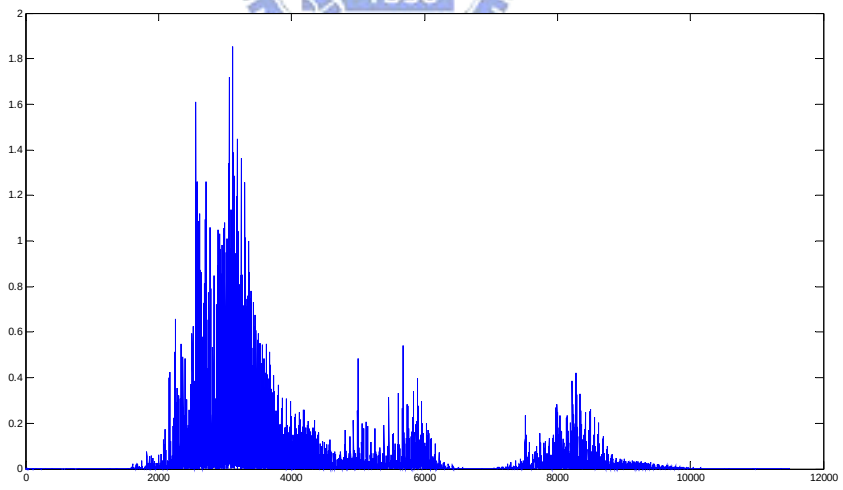


Fig 40. MSE (without echo canceller) of man's testing

Man's spectrograms results as shows as follows. In Fig. 43, we can see the energy distribution is different with Fig. 41. On other hand, the

spectrograms result of Fig. 42 is similar to Fig. 41.

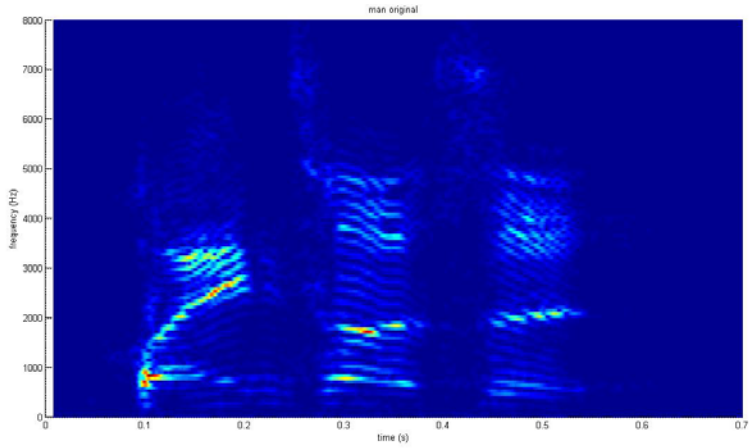


Fig 41. Spectrograms of man's original signal

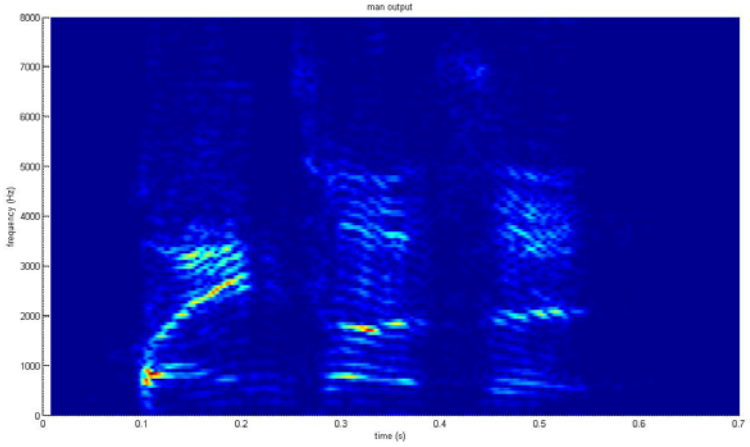


Fig 42. Spectrograms of man's system output signal

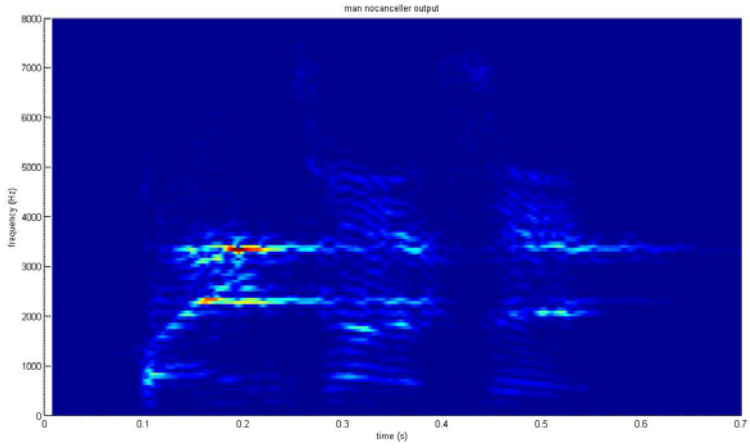


Fig 43. Spectrograms of man's system output signal without echo canceller

Woman’s voice:

As same as man’s voice testing, woman’s original signal is like Fig. 44. The Y-axis is from -1 to 1 and X-axis from 0 to 18000 (unit: iterations).

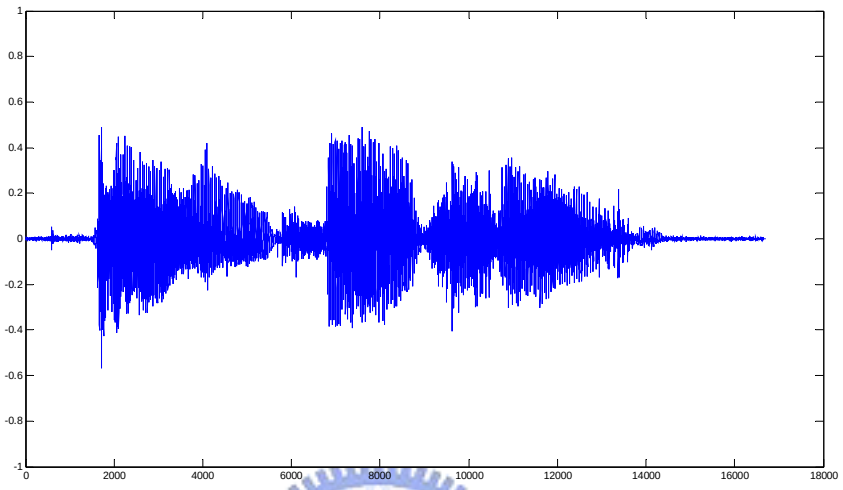


Fig 44. Waveform of woman’s original signal

Besides, we will show the echo canceller output in Fig. 45 Waveform of woman’s echo canceller output signal. (12bit: value 2048 means 1)

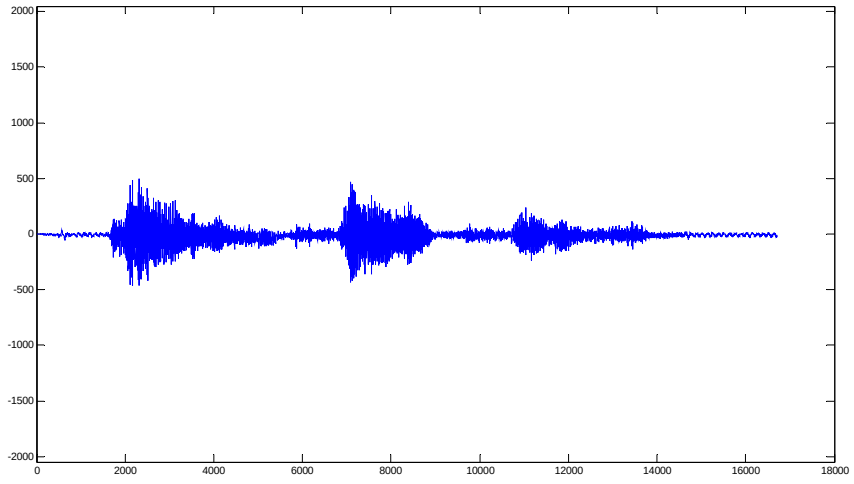


Fig 45. Waveform of woman’s echo canceller output signal

Our system output will as shows as follows; with echo canceller work.

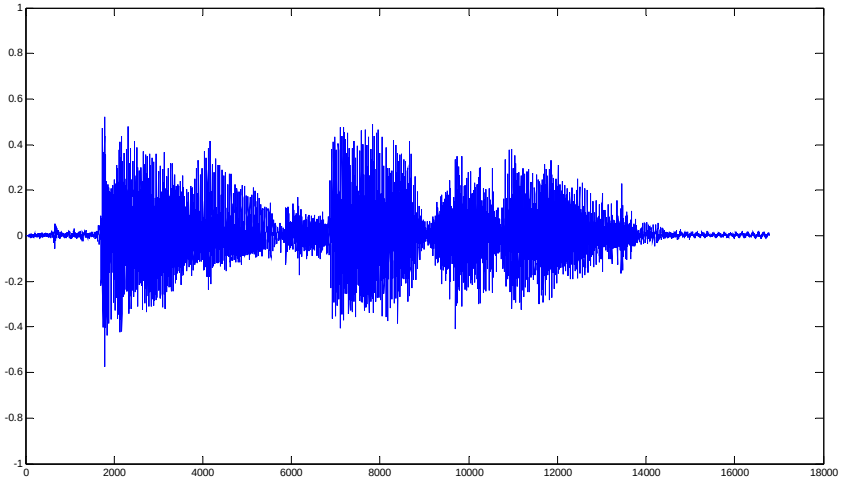


Fig 46. Waveform of woman's system output signal

As well, we also turn off the echo canceller. The system output will be look like Fig. 47. (Y-axis also from -1 to 1)

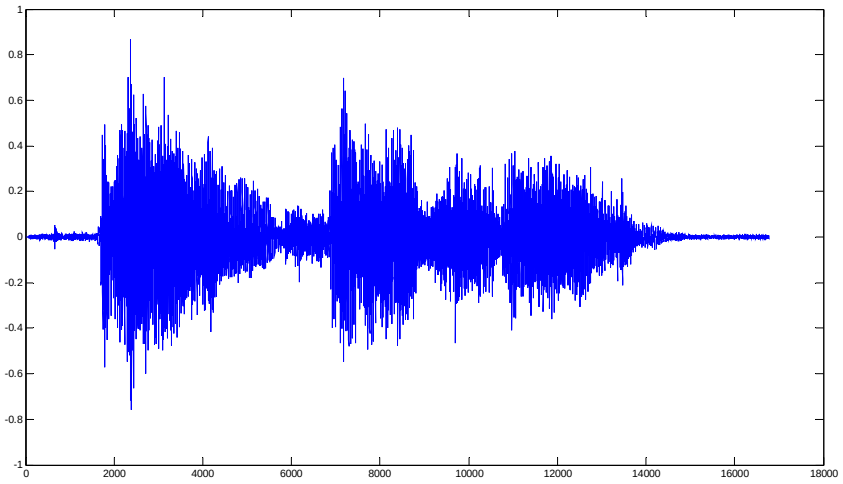


Fig 47. Waveform of woman's system output signal without echo canceller

Fig. 48 shows MSE (with echo canceller) of system output to original input. (Y-axis is from 0 to 0.4)

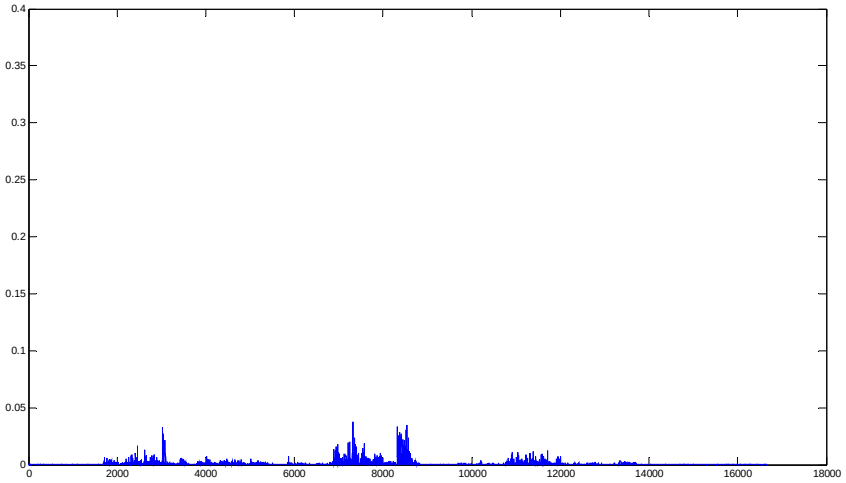


Fig 48. MSE (with echo canceller) of woman's testing

Fig. 49 shows MSE (without echo canceller) of system output to original input. (Y-axis is from 0 to 0.4)

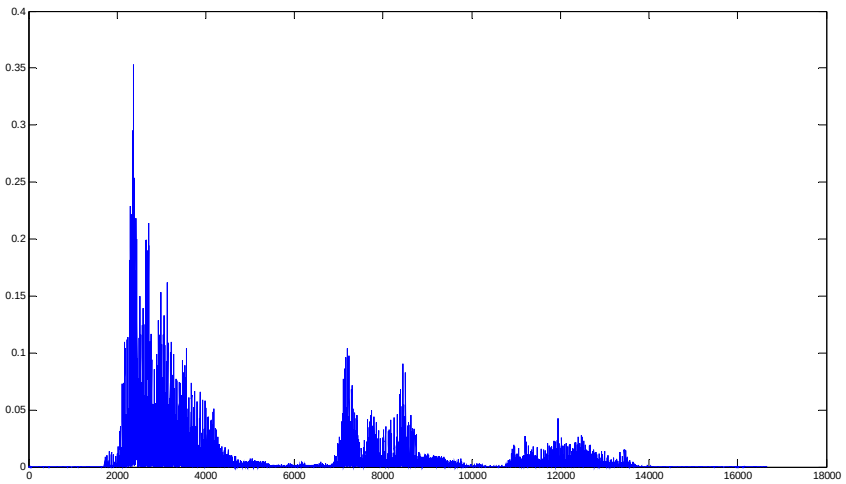


Fig 49. MSE (without echo canceller) of woman's testing

Woman’s spectrograms results as shows as follows. In Fig. 52, we can see the energy distribution is also different with Fig. 50 and the spectrograms result of Fig. 51 is similar to Fig. 50.

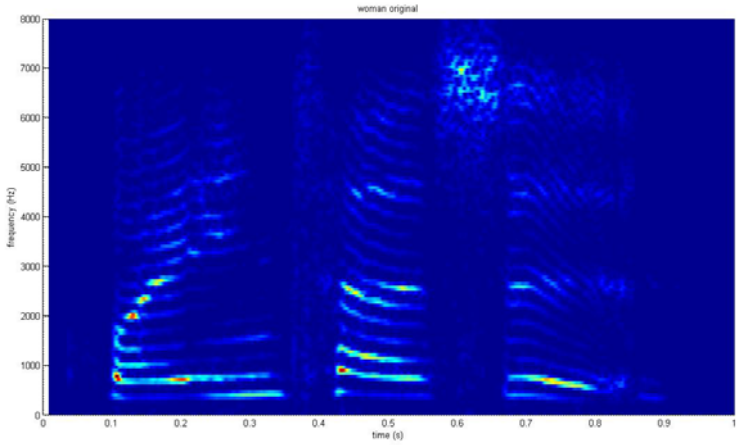


Fig 50. Spectrograms of woman’s original signal

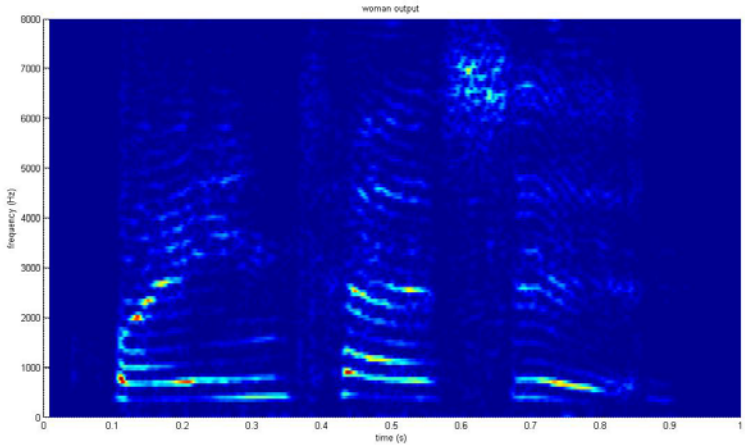


Fig 51. Spectrograms of woman’s system output signal

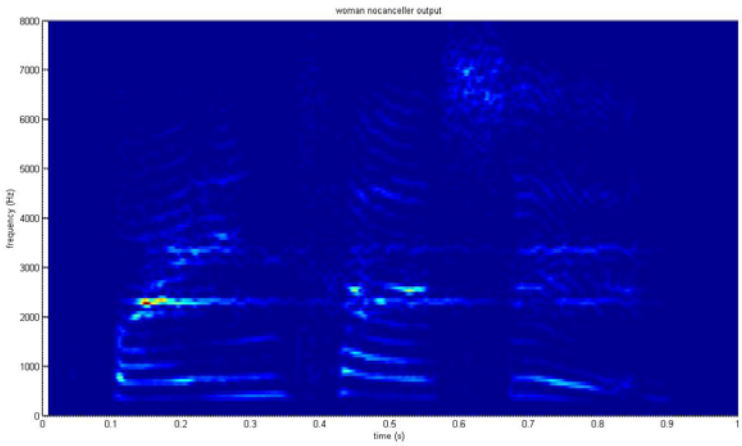


Fig 52. Spectrograms of woman’s system output signal without echo canceller

Boy’s voice:

The boy’s original signal is like Fig. 53. The Y-axis is from -1 to 1 and X-axis from 0 to 20000 (unit: iterations).

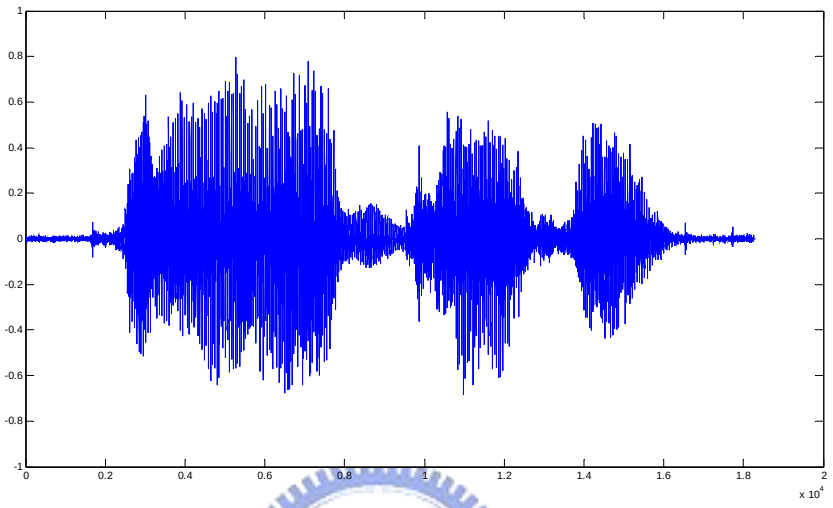


Fig 53. Waveform of boy’s original signal

We will show the echo canceller output in Fig. 54 Waveform of boy’s echo canceller output signal. (12bit: value 2048 means 1)

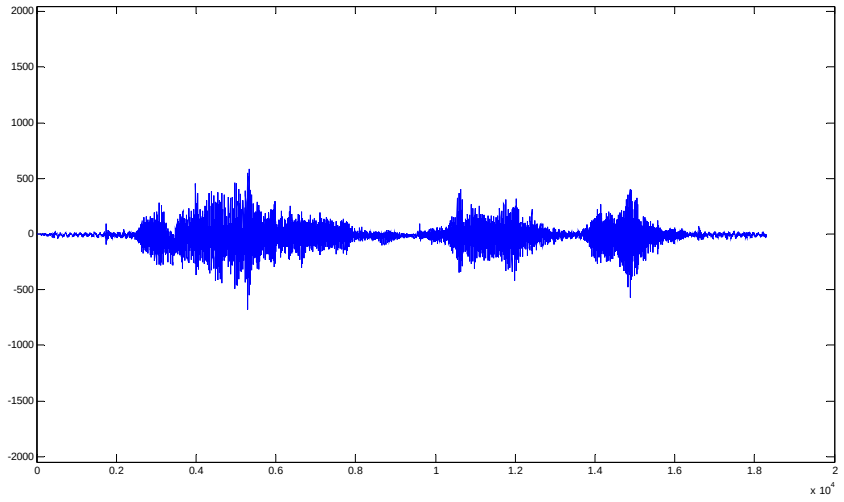


Fig 54. Waveform of boy’s echo canceller output signal

Our system output will as shows as follows; under the echo canceller work.

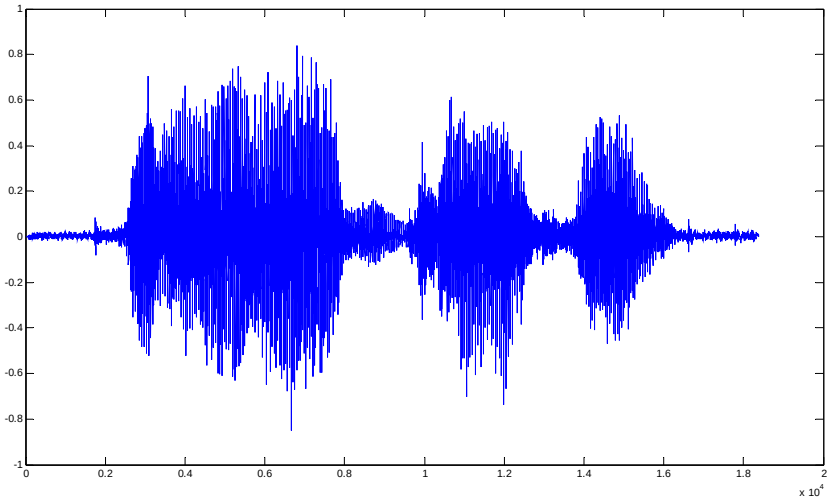


Fig 55. Waveform of boy's system output signal

In next graph, if we also turn off the echo canceller. The system output will be look like Fig. 56. (Y-axis also from -1 to 1)

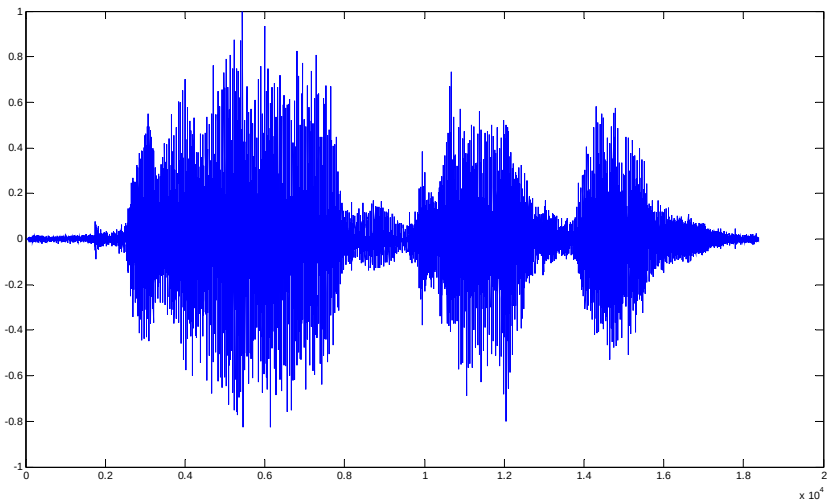


Fig 56. Waveform of boy's system output signal without echo canceller

Fig. 57 shows MSE (with echo canceller) of system output to original input. (Y-axis is from 0 to 0.35)

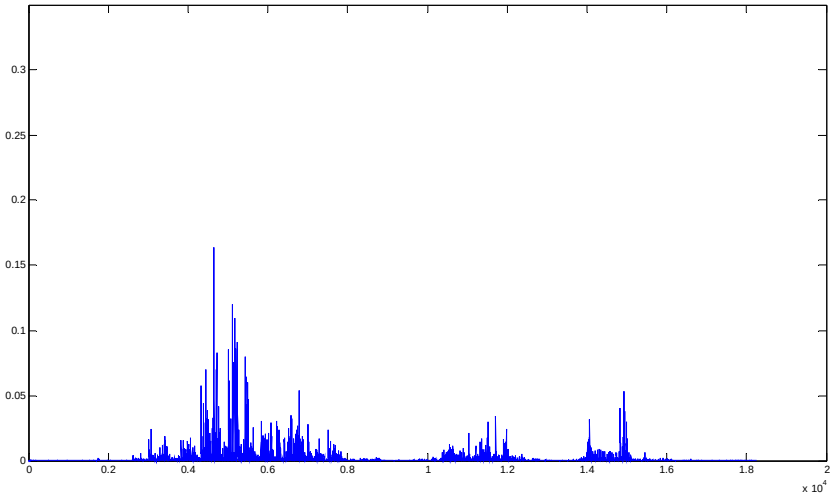


Fig 57. MSE (with echo canceller) of boy's testing

Fig 58shows MSE (without echo canceller) of system output to original input. (Y-axis is from 0 to 0.35)

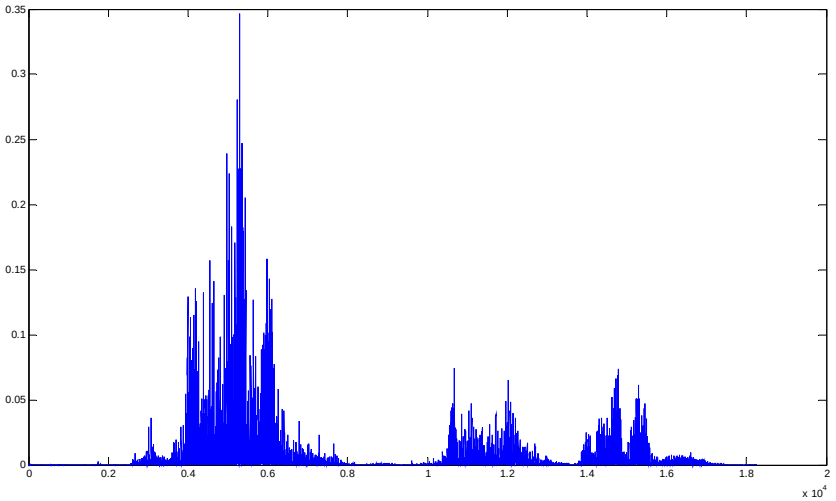


Fig 58. MSE (without echo canceller) of boy's testing

Boy’s spectrograms results as shows as follows. In Fig. 61, we can see the energy distribution is different with Fig. 59 and the spectrograms result of Fig. 60 is similar to Fig. 61.

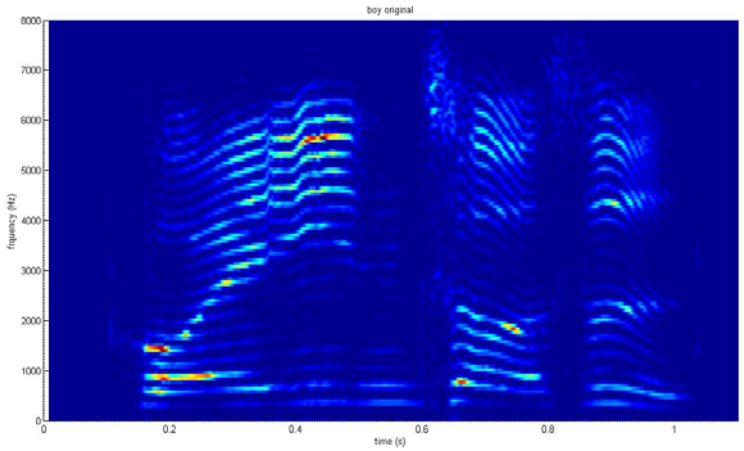


Fig 59. Spectrograms of boy’s original signal

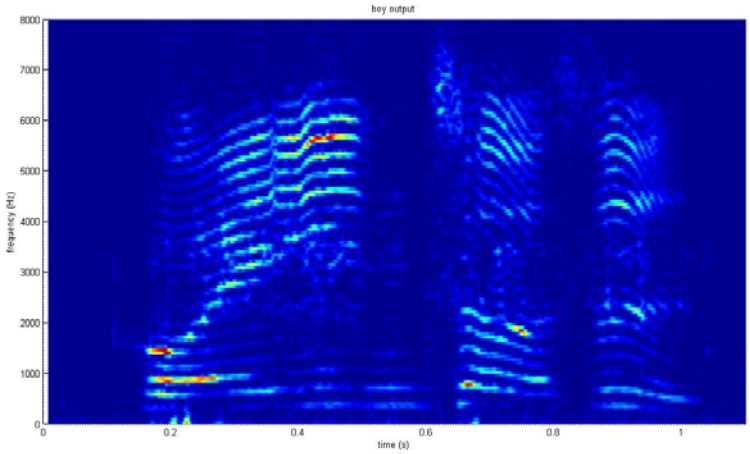


Fig 60. Spectrograms of boy’s system output signal

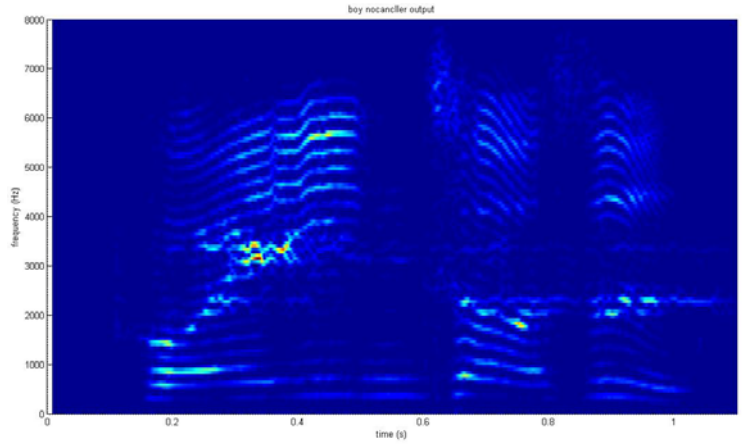


Fig 61. Spectrograms of boy’s system output signal without echo canceller

Girl’s voice:

The girl’s original signal is like Fig. 62. The Y-axis is from -1 to 1 and X-axis from 0 to 15000 (unit: iterations).

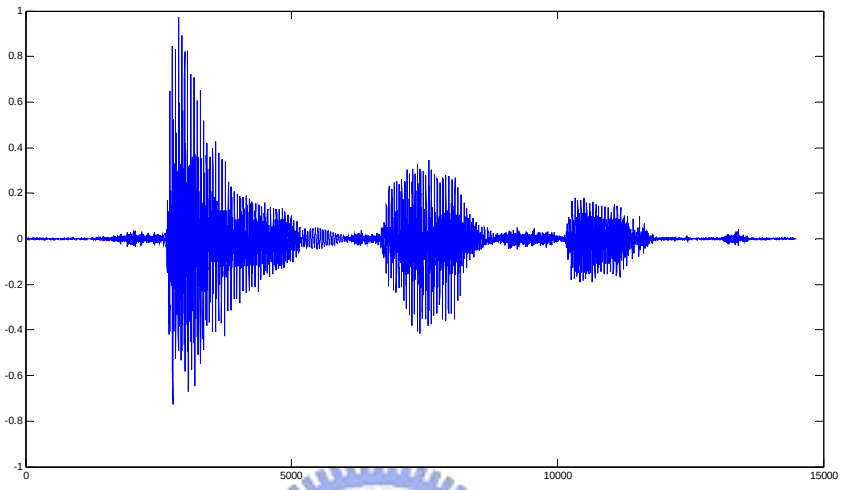


Fig 62. Waveform of girl’s original signal

The echo canceller output in Fig 63 Waveform of girl’s echo canceller output signal, as shows as follows. (12bit: value 2048 means 1)

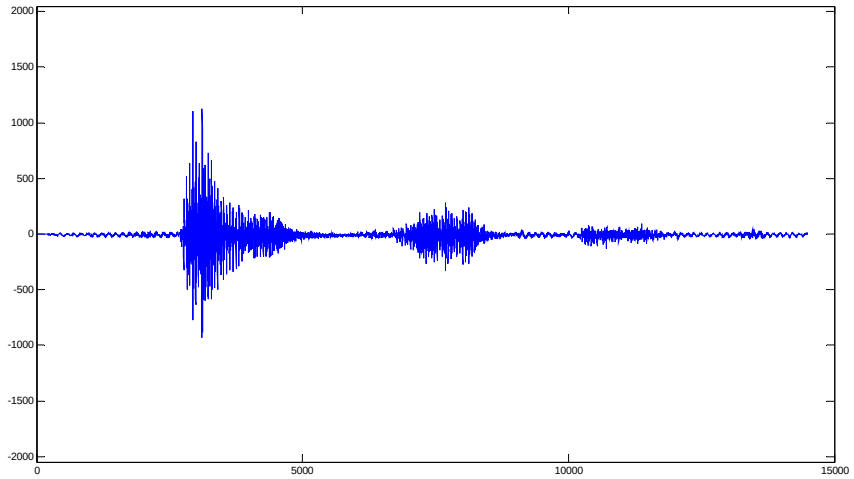


Fig 63. Waveform of girl’s echo canceller output signal

Our system output will as shows as follows; under the echo canceller work.

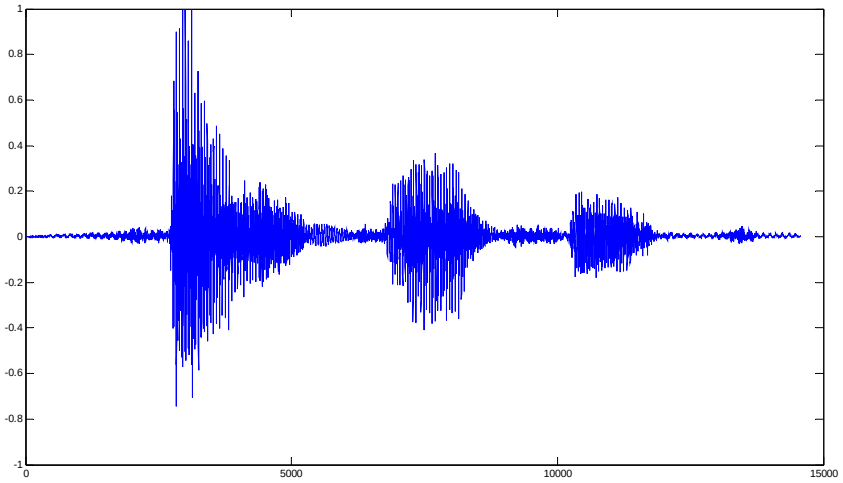


Fig 64. Waveform of girl's system output signal

If we also turn off the echo canceller, the system output will be look like Fig. 65. (Y-axis also from -1 to 1)

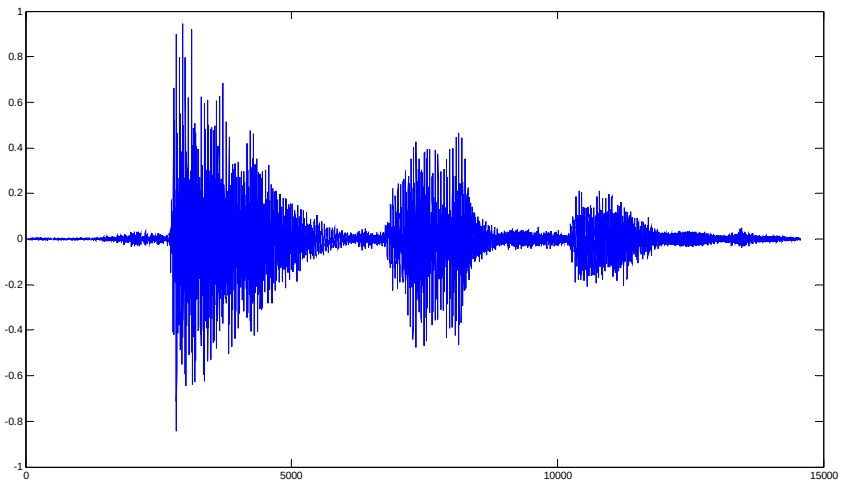


Fig 65. Waveform of girl's system output signal without echo canceller

Fig. 66 shows MSE (with echo canceller) of system output to original input. (Y-axis is from 0 to 0.25)

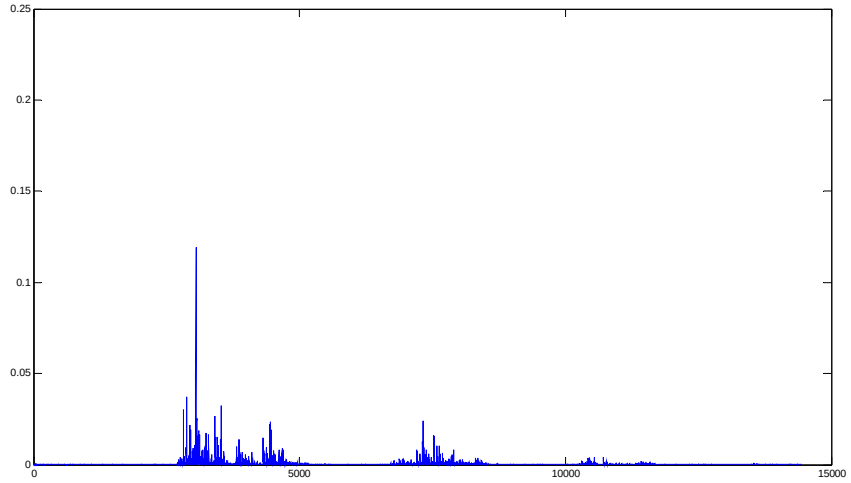


Fig 66. MSE (with echo canceller) of girl's testing

Fig. 67 shows MSE (without echo canceller) of system output to original input. (Y-axis is from 0 to 0.25)

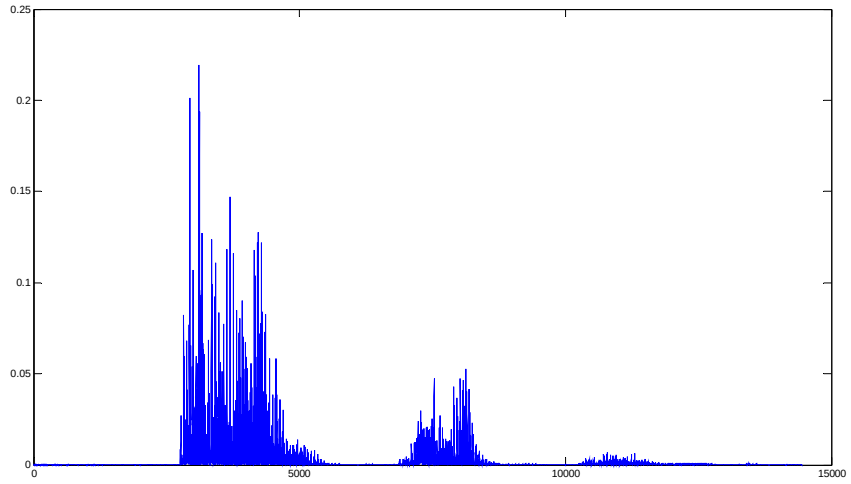


Fig 67. MSE (without echo canceller) of girl's testing

Girl’s spectrograms results as shows as follows. In Fig. 70, we can see the energy distribution is different with Fig. 68 and the spectrograms result of Fig. 69 is similar to Fig. 68.

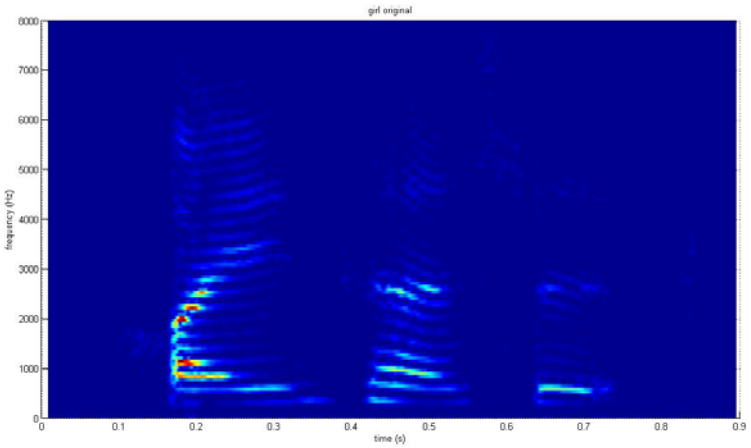


Fig 68. Spectrograms of girl’s original signal

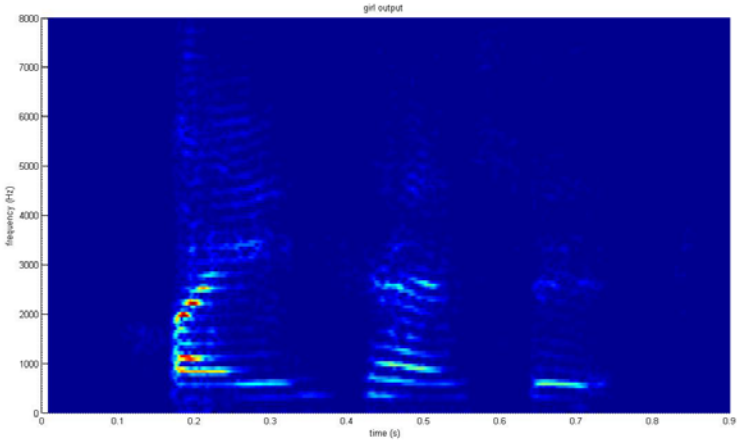


Fig 69. Spectrograms of girl’s system output signal

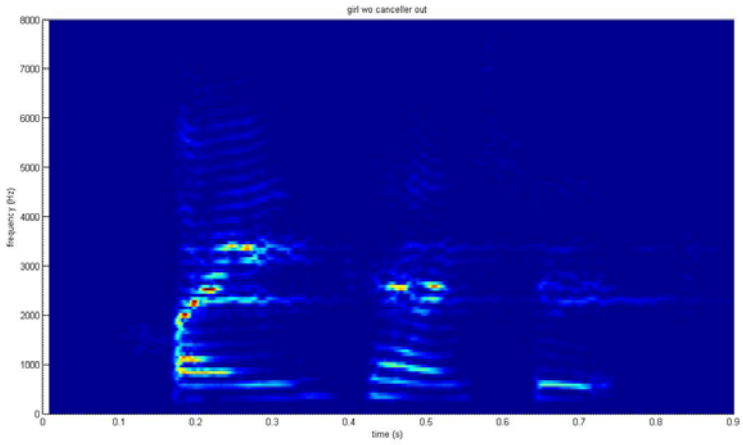


Fig 70. Spectrograms of girl’s system output signal without echo canceller

We will form all MSE and SNR of voice test in table 4.3.1 MSE of voice testing.

table 3. performance of voice testing

	man	***woman***	***boy***	***girl***
MSE with Echo canceller	0.0026	0.00058188	0.0013	0.00042373
MSE w/o Echo canceller	0.0422	.006	0.0059	0.0031
SNR (db) with Echo canceller	18.43	26.153	17.78	16.23
SNR (db) w/o Echo canceller	6.334	6.02	11.21	7.589

In table 3, we can discover our echo canceller can reduced the most echo noise in MSE and SNR data results. Moreover, we also use the spectrograms to prove our algorithm has good performance in echo cancellation result. Therefore, the echo noise will not influence our user's hearing experience when our algorithm worked.

In next topic, we will discuss that the long voice testing and forward path delays changed effect for our algorithm.

Forward path delays changed:

We consider the forward path delays effect in this model. We will show you that the performance of echo cancellation is not depended on the forward path delays.

We use MSE to show you that the performance of echo cancellation is not depended on the forward path delays in table 4 MSE of forward path delays changed.

We use man's voice signal as shows as follows. The result of table 4 is not depended on forward path iteration delays. Therefore, other components of hearing aid will have more design adaptability.

table 4. MSE of forward path delays changed

Iteration delays	50	100	150	300
MSE	0.0026	0.00255	0.00252	0.00264

Long voice's testing:

In order to test the stability of our algorithm, we use the voice's signal of 700000 iterations (almost 44 seconds) to prove our algorithm will adaptive the window coefficient for control the echo noise.

In Fig. 71 original long voice's signal, we have the system input for our model.

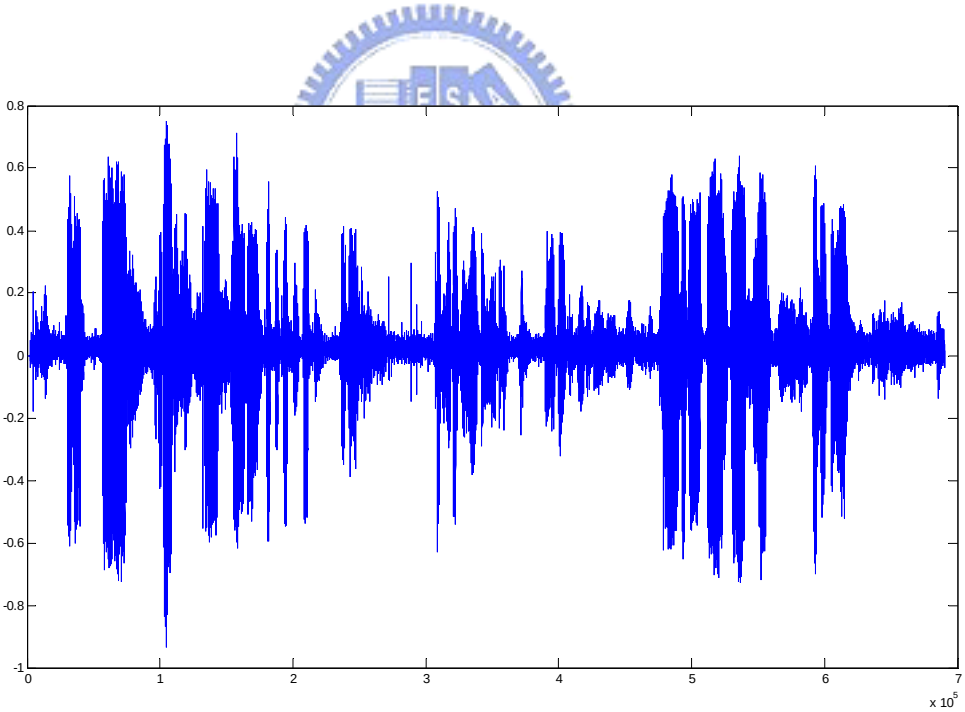


Fig 71. original long voice's signal

In Fig. 72 long voice's system output signal, we have the system output for our model.

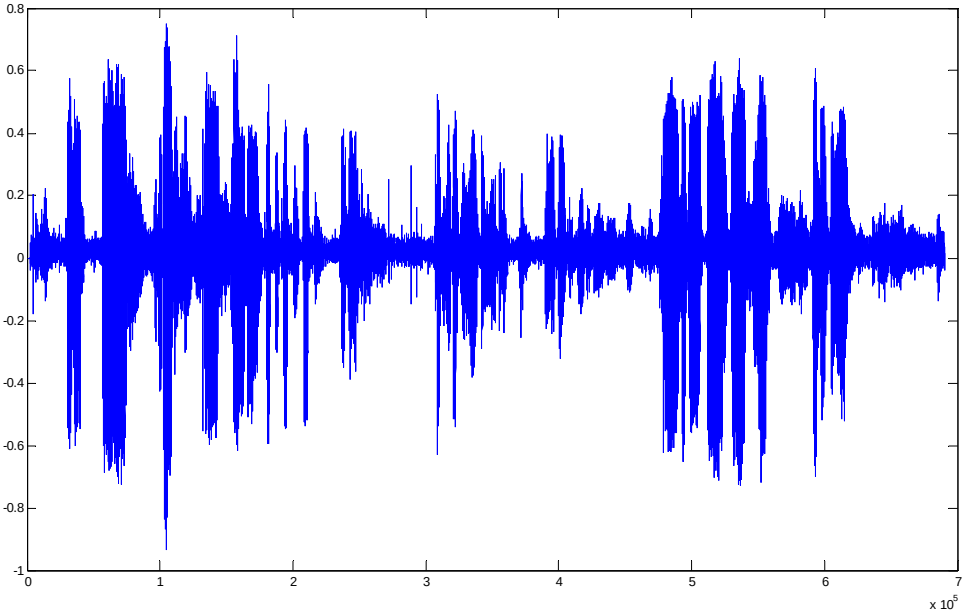


Fig 72. long voice's system output signal

In Fig. 72, we can find out the system output will sounds like the original input signal in the human's hearing experience.

Actually, the key point of this thesis is low power design. So we just achieved the goal of cancelled echo noise and will focus on real design's power consumption. Accordingly, we just prove our algorithm is achieved the goal of cancelled echo noise in human's hearing experience.

Therefore, we will have the simplest architecture design when we using our P^2SPT algorithm in next chapter, Chapter 4 Architecture Designs & Power Reports.

Chapter 4
Architecture Designs & Power Reports

Considering the power consumption of echo canceller, we have to do more works in architecture design. Based on the P^2SPT algorithm, we fold the new architecture for our low power issue. We will discuss the reason of folding in section 4.1. Furthermore, we will show the power report in section 4.2, when we finish introduced our design's architecture.

4.1 Architecture Designs

In this section, we start to introduce our design's architecture. For the reason of us folding our design, we will to explain that in section 4.1.1. After section 4.1.1, we start introduced our echo canceller architecture design in section 4.1.2. We will have a clear description of our architecture design in this section.

4.1.1 Architecture design's description

The "folding" [34] is key-point architecture technique in this thesis. Based on our specification and work's conditions, folding our design will minimize design's power consumption. We will illustrate the reason of folding, and give estimated results in the end of this section.

In next page, we show the reason of folding in Fig. 73. We use the folding architecture technique for 1. Reduce area for efficient power consumption, and 2. Using SRAM-types register file. In fact, all of the architecture designs will only for the one reason; minimizes the power consumption of our echo canceller design.

Considering our work's conditions as shows as follows, we folding our design for minimize power consumption.

- Using standard library.
- Fixed voltage.
- Work frequency too low.
- Leakage problem in 90 processes.
- Simplest architecture.
- Replace shifter register.

The key point for using folding as shows as follows:

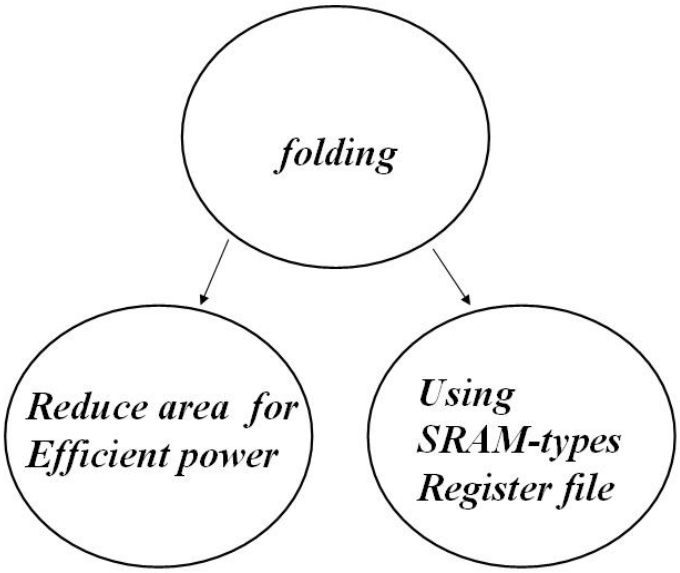


Fig 73. key point for using folding skill

In Fig. 73, we want to use the "registers file" to replace "shifter registers", and using the SRAM type register file to replace D flip-flop. Therefore, the clock loading and power consumption will be minimized. In our specification, the sampling rate will be 16K Hz. So we should to "folding" for smaller area, if our design has simplest architecture (to reduce the leakage power, especially in the advanced processes [35].).

1. Reason of registers file to replace shifter registers:

First of all, we start explained the reason of use the “registers file” to replace “shifter registers”.

In Fig. 74 Diagram of shifter registers, we can find out that all input data will shift in the every iteration. Shifter registers will send the data to the next register in the every iteration, there means a new input will force others moved forward. For instance, if our filter has 32 taps, not necessary move will occur for 31 times.

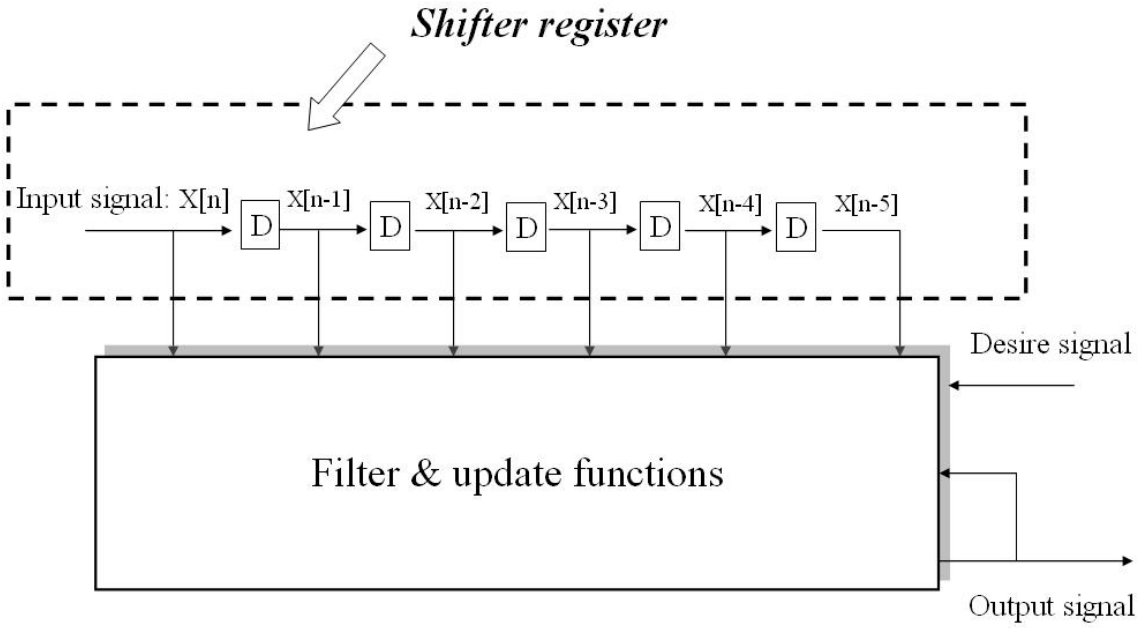


Fig 74. Diagram of shifter registers

In our application, the speech’s input signal always swing between 1 and -1. Therefore, lot not necessary power consumption will occur in the shifter registers; especially we use 2’s complement to represent input data in our hardware design.

In the past, shifter register was popular form in the filter design. But considering the power issue, we have to find out some way do not real “shift” all input data, but can also catch all data’s shifter information that just we need.

We choose register file to replace shifter register, because every input data will “replace” the oldest input data we do not used.

In Fig. 75 Diagram of register file, we can figure out the new input data will replace the oldest one, if we give the suitable address.

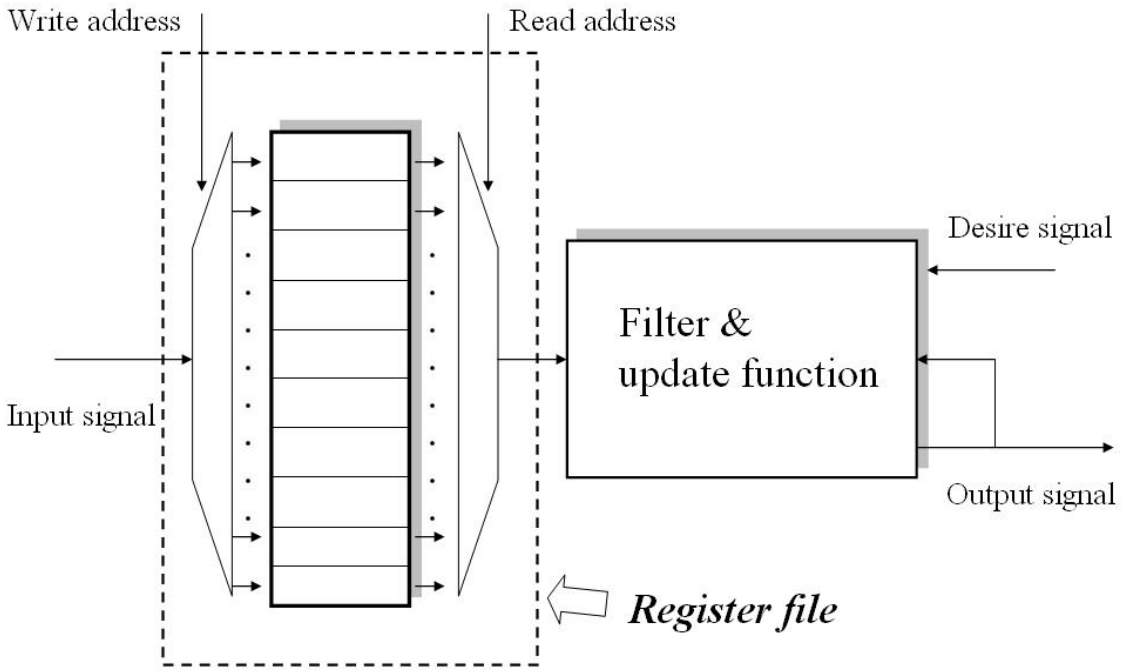


Fig 75. Diagram of register file

Considering the register file in filter design, we have to start used the technique we call “folding”. Accordingly, we can use the register file of SRAM type to reduce the power consummation and layout area; the shifter register only can compose by D flip-flop.

But in fact, we will have to design the unit of control those addresses and data follows. Therefore, the address scheduling [36] will be considered.

In our research, if shifter register will replace by register file, the power consumptions will reduce to 10% in the register part. But the control unit design is necessary. In consequence, some power consumption will transform to this control unit.

2. Estimation of Power to Folding:

We estimate the power consumption for different folding way of our design, because we want to know the minimize power consumption of folding. First of all, we start introduced our work's condition and show the estimated results in the next. In the main, our clock rate is 16K Hz, if our sampling rate is defined to 16K Hz. So if we folding the architecture, than the clock rate will higher than 16K Hz.

In the low power design we know [37]; the speed Accelerated will increase the power consumption. But in this case, our works clock rate is too slow to use. Especially our echo canceller is design for the simplest structure. Accordingly, we folding will increase the power consumption of unit area, but also reduce the total area of our design. In a ward, In order to realize relationship of folding and power consumption, we assume 3 conditions as follows:

1. The power consumption of Registers part as defines as follows:

table 5. defines of registers part's power consumption

Fold	**u W**	**Estimation function**
32	68	(AC*freq + DC)*1.3v freq:512K Hz
16	68	(AC*freq + DC)*2)*1.3v freq:256K Hz, 2 register file
8	68	(AC*freq + DC)*4)*1.3v freq:128K Hz, 4 register file
4	68x2	Double for using D flip-flop (8 register file)
2	68x4	Double for using D flip-flop (16 register file)
1	68x8	Double for using D flip-flop (shifter register)

The SRAM-types register file only supply to folding 8 times. We assume the power will double increase for using D flip-flop and return to the shifter register. In next page, we will show the data ship of TSMC_013 SRAM-types register file to you. TSMC_013 SRAM-types register file as follows:

table 6. defines of registers part's power information

TSMC_013 register file	AC current	DC current
SYHD130_8X12X1CM2	0.004 mA/MHz	1.521 uA
SYHD130_16X12X1CM2	0.004 mA/MHz	1.665 uA
SYHD130_32X12X1CM2	0.004 mA/MHz	1.950 uA

2. Tap part will increase 10% power consumption for folding.

For instance, if we folding 32 times for using one tap unit and try to no folding for 32 tap units, the folding structure will increase power consumption with higher clock frequency.

3. Other part will increase 10% power consumption for folding.

There will only one other part, but design complexity will increase by folding. Based on these 3 conditions, we will have a clear calculation of our estimation. In this analysis, we try to describe the relationship of the folding and the power consumption. Therefore, we estimated design's power to you in Table 7. In next page, we will show the estimation of fold to power consumption likes Fig 76.

Table 7 as shows as follows:

table 7. clear calculation of folding to power estimation

Fold	**u W**	**Tap + other + register**
1	1014	32*(24*0.59) + 28*0.59 + 68*8
2	542	16*(24*0.6561) + 28*0.6561 + 68*4
4	296	8*(24*0.729) + 28*0.729 + 68*2
8	168	4*(24*0.81) + 28*0.81 + 68
16	136	2*(24*0.9) + 28*0.9 + 68
32	128	24 + 28 + 68

In Fig. 76 estimation of folding to power consumption, we can see the relationship of folding and power consumption.

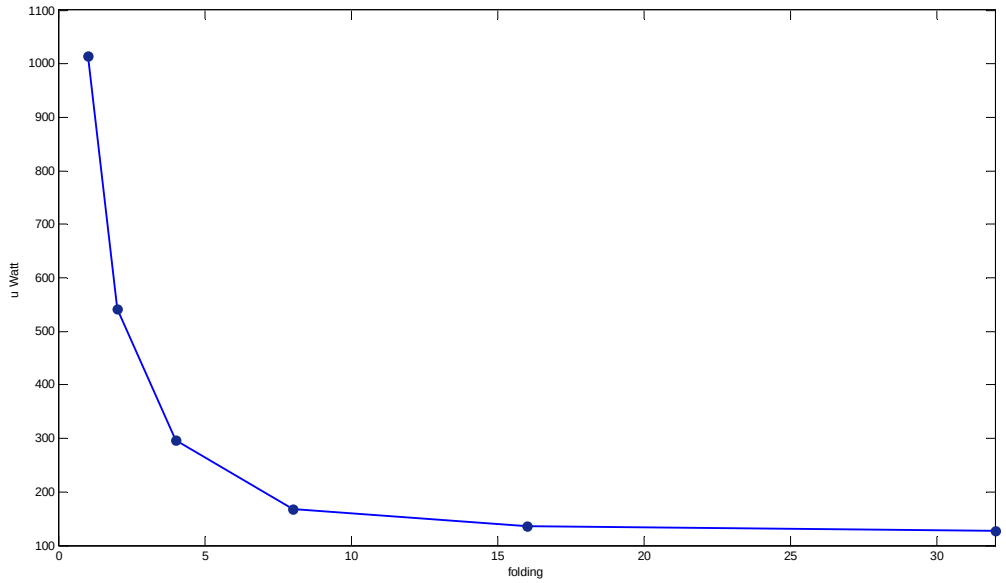


Fig 76. estimation of folding to power consumption

As we can see in Fig. 76 estimation of folding to power consumption, we will have minimized power consumption when we folding 32 times of our design.

Based on this result, we will design our architecture for folding 32 times. We will show our real architecture design in next section, section 4.1.2 Design's Architecture.

4.1.2 Design's Architecture

In this section, we will introduce our architecture designs. First of all, we show Fig. 77 Diagram of design's architecture to you.

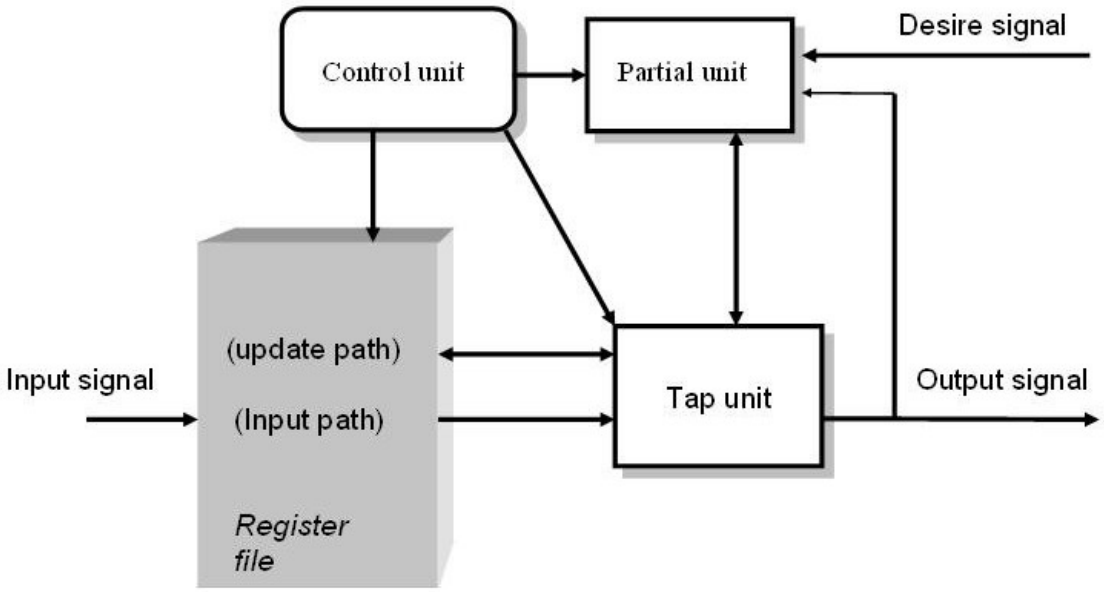


Fig 77. Diagram of design's architecture

Accordingly, there are four main blocks in Fig. 77.Simple functions illustration of these blocks as shows as follows, and the clear explanation will in next page.

- Control unit: to produce addresses & flow control signals.
- Partial unit: to handle the partial update function.
- Tap unit: to execute the tap function & to produce the output data.
- Register file: to save the input data & update coefficients.

1. Control unit

In order to explain our control unit design, we show the architecture of control unit in Fig. 78.

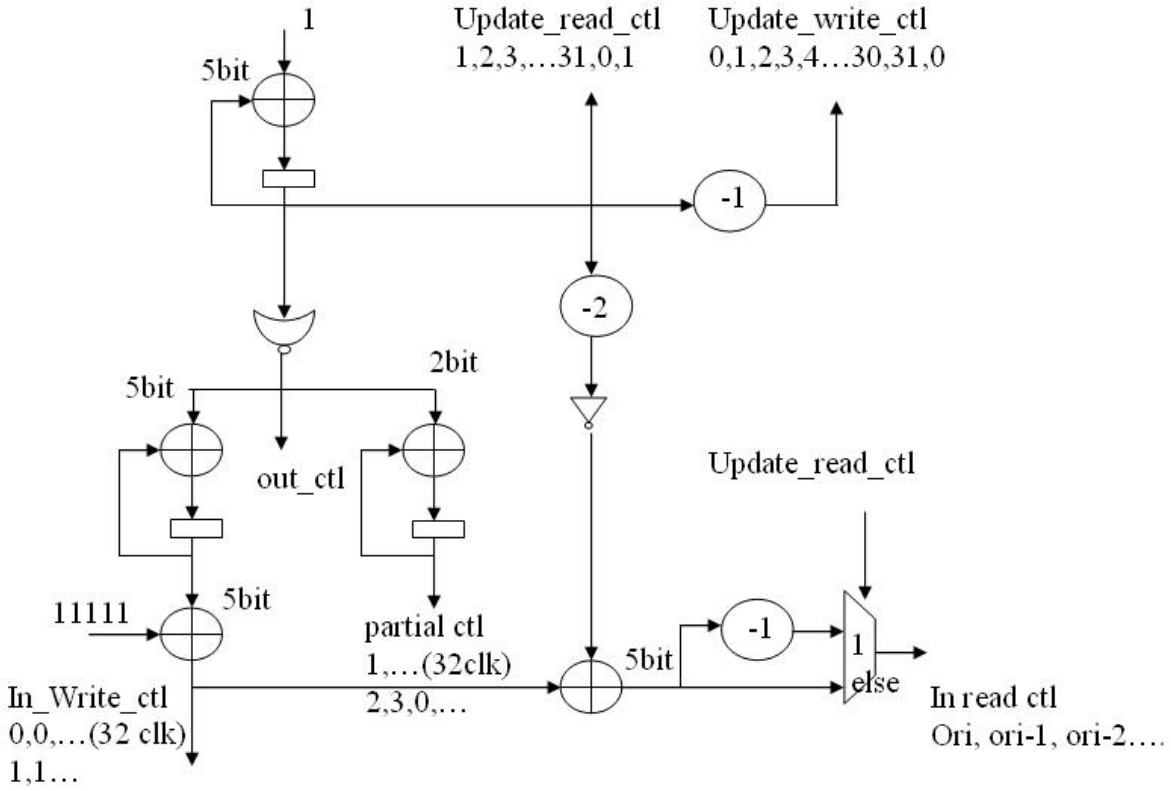


Fig 78. architecture of control unit

As we can understand, this unit's key-point is figured out the relationship between all the control and address flows. Besides, we try to share the counter and adder as we can. In consequence, we can have the architecture as likes as Fig. 78.

Our design has 32 taps, and folding 32 times (to 1 tap.). Furthermore, our control unit should to take care the "shift feeling" in the address signal that we call "in_read_ctl". In that part, we use "in_write_ctl" to move our start position in the register file. Altogether, the control unit will control the data flows of our designs.

2. Partial unit

We show the architecture of partial unit in Fig. 79.

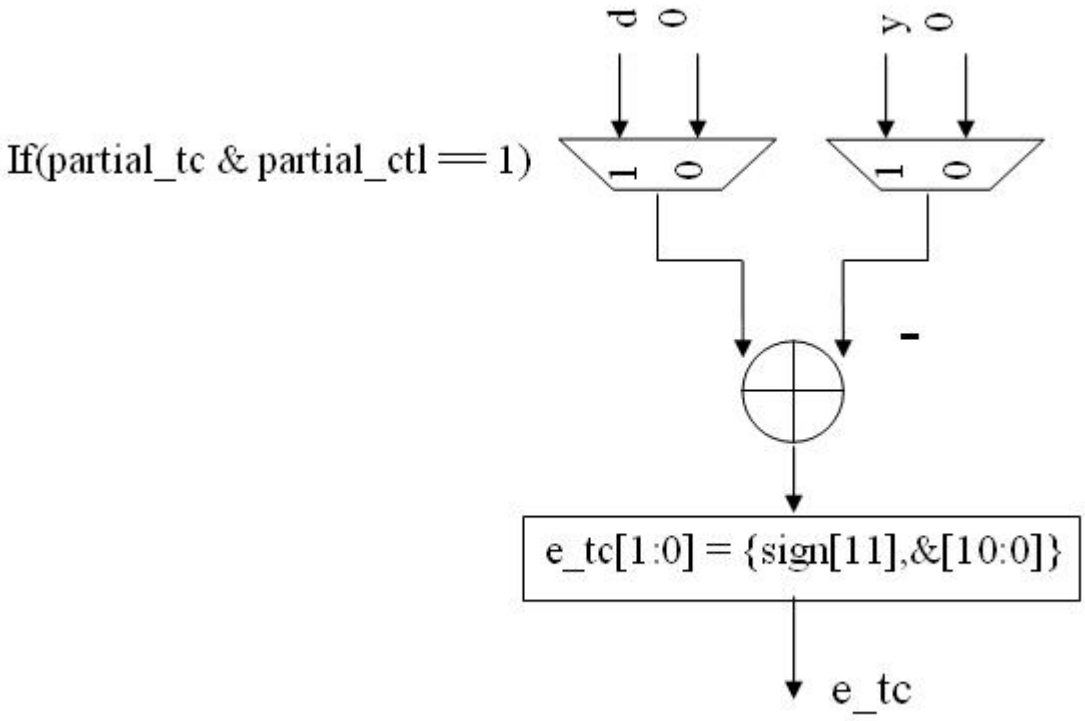


Fig 79. architecture of partial unit

In the main, this unit will try to reduce the update operation. Therefore, to reduce the update operation will reduce the design's power consumption.Based on our P^2SPT algorithm, we can design the simplest architecture of our partial update function.

For example, if our partial update condition will not be true, the update operation is n/2. On the other hand, if the condition will be true, the update operation is n/4 (n: iterations; the complete explanation in the chapter 2).

3. Tap unit

The architecture of tap unit is shown in Fig. 80.

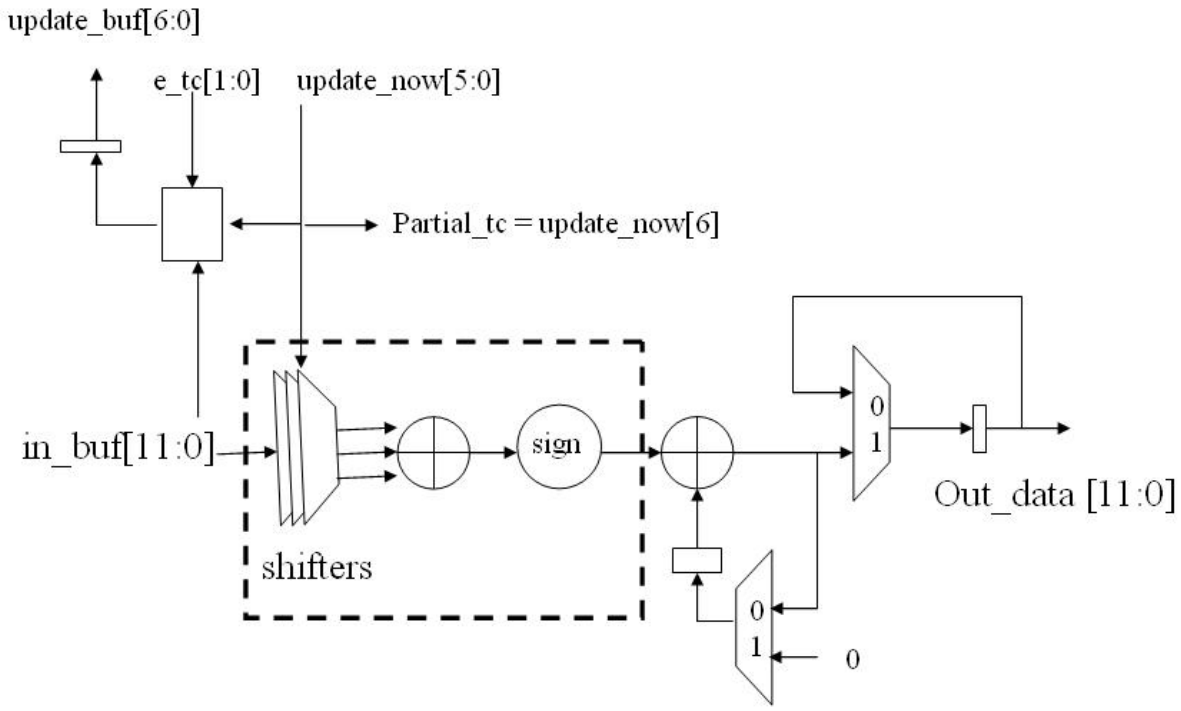


Fig 80. architecture of tap unit

Tap unit will compute the output data and update coefficients in next iteration. Based on the folding, the tap unit must to add up the output data for 32 times. Furthermore, new update coefficients will be saved in the register file and take update coefficients out to the register file for operation in the same time, but not same position.

Besides, we do not want to have the multiplier in our tap unit. Consequently, our algorithm includes the Power-of-Two window coefficient algorithm. In the P^2SPT algorithm, we must choose the based to combine coefficients of update window. In this tap unit, this function is shifters. Therefore, we will have the simplest structure of our tap unit.

4. Register file

In this block, we define two register file to save the data.

1. Input data register file.
2. Update coefficient register file.

Based on our specifications, bit numbers of input data is 12 bits, and we need to save 7 bits for update coefficient. For example, if we design on the TSMC_013 library. We need to use the 2 ports SRAM register file likes SYHD130_32X12X1CM2 for input data, and SYHD130_32X7X1CM2 for update coefficient.

We have an issue for our input data register file. For every start access of our folding structure, the input data will save to the register file and be taking in the same time. There are not allowed of 2 ports SRAM register file. For that reason, we design the register file start unit to process this issue.

We show Fig. 28 Diagram of register file start unit as follows:

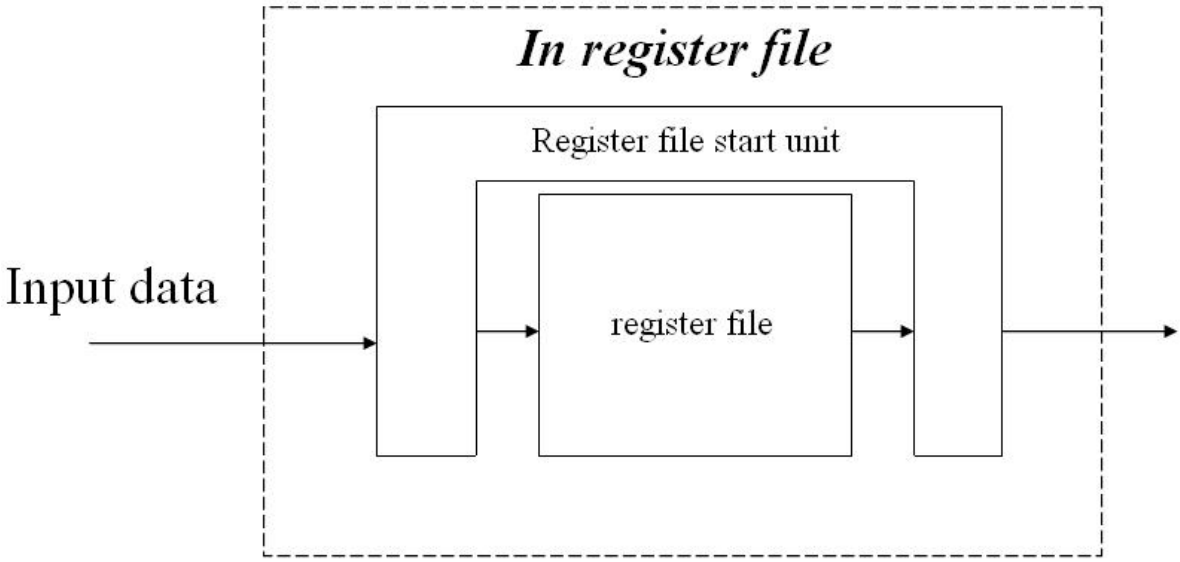


Fig 81. Diagram of register file start unit

The architecture of register file start unit is shown in Fig. 82.

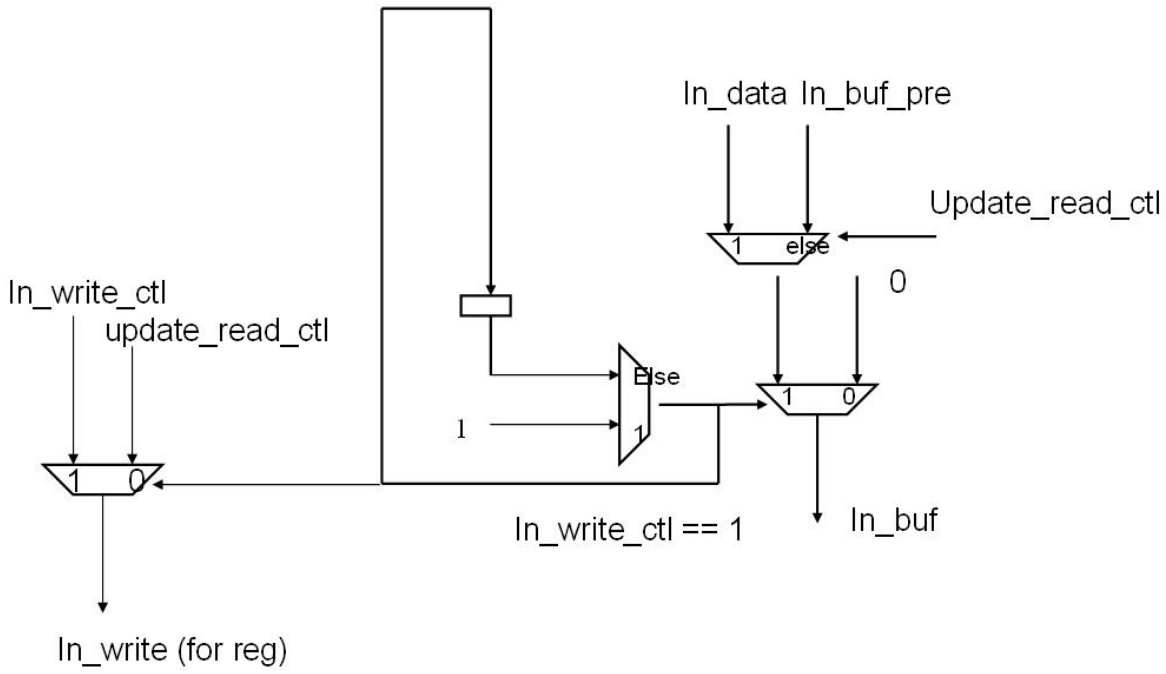


Fig 82. architecture of register file start unit

Accordingly, this unit's works is handled the start issue of our folding designs. The key-point idea of this unit is the "bypass path". We design the bypass path for input data, and also save the data in to the register file too. Therefore, we can use the SRAM register file and fix the problem of start issue.

In the next, Architecture of Echo Canceller will present to you that overall architecture of our designs. In addition, our design is combined by those four sub unit. Therefore we only need to show the block connect to you, and to define signals we used.

Architecture of echo canceller

We will present our overall design of architecture to you. First of all, we show Fig. 83.

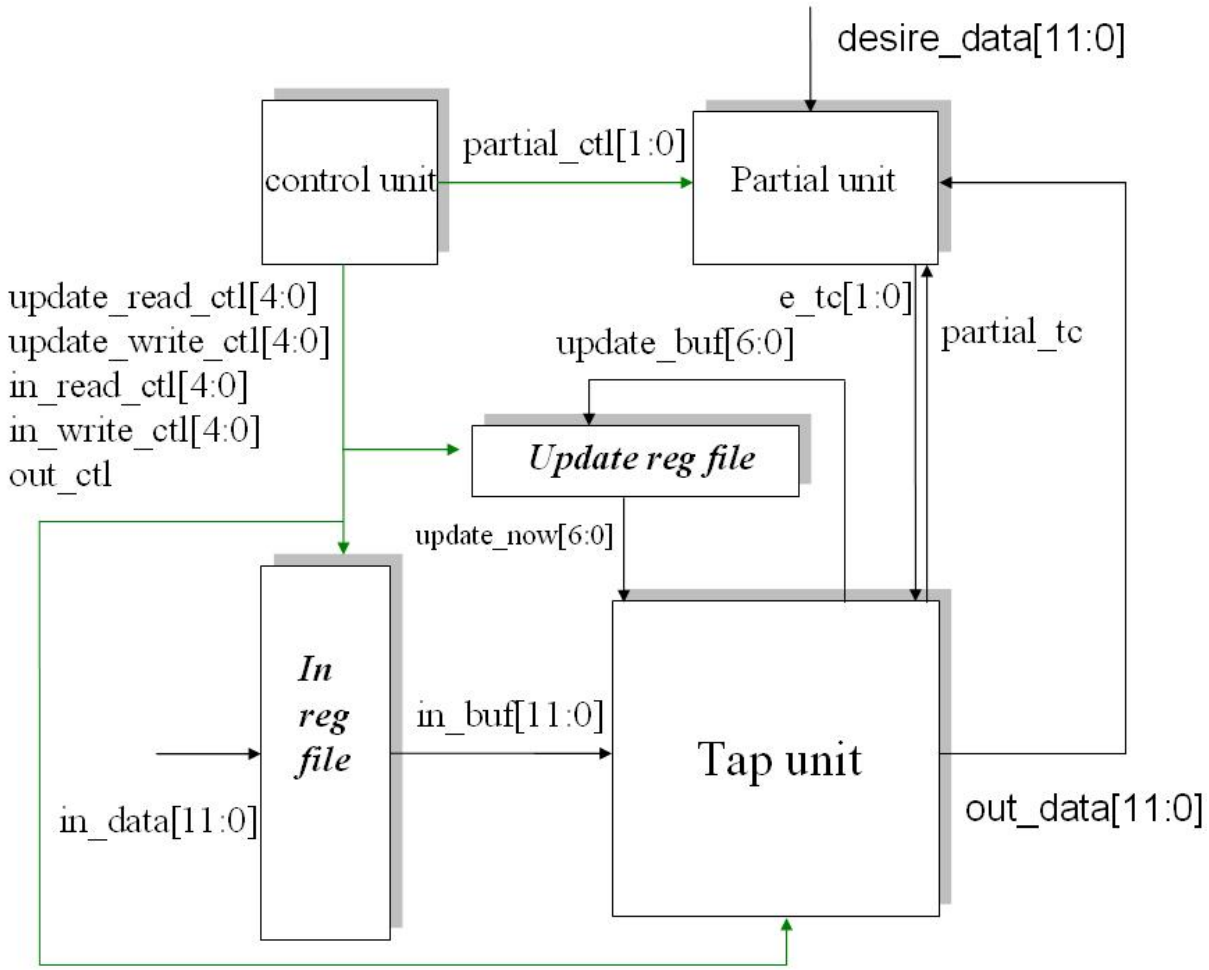


Fig 83. architecture of echo canceller

Consequently, this topic's point is connected those four blocks, and showed this overall architecture to you. After hard work of design our echo canceller, we will present our power report to you in next section, section 4.2 power reports. We will propose the power data in this section.

4.2 Power Reports

In power reports, we have proposed two sections to you. First of all, section 4.2.1 will report our design's power information and other information of our design. In section 4.2.2, we will compare our design's power to three related works. Those related works will be introduced in chapter 1. Now, let's start section 4.2.1.

4.2.1 Design's Power Report

We have two processes (TSMC 013, and UMC 90.) to report. The data of our design as shows as follows. We will have the same clock frequency of 512K Hz (folding: 16K→512K). Fig. 84 will shows the power consumption of these two processes.

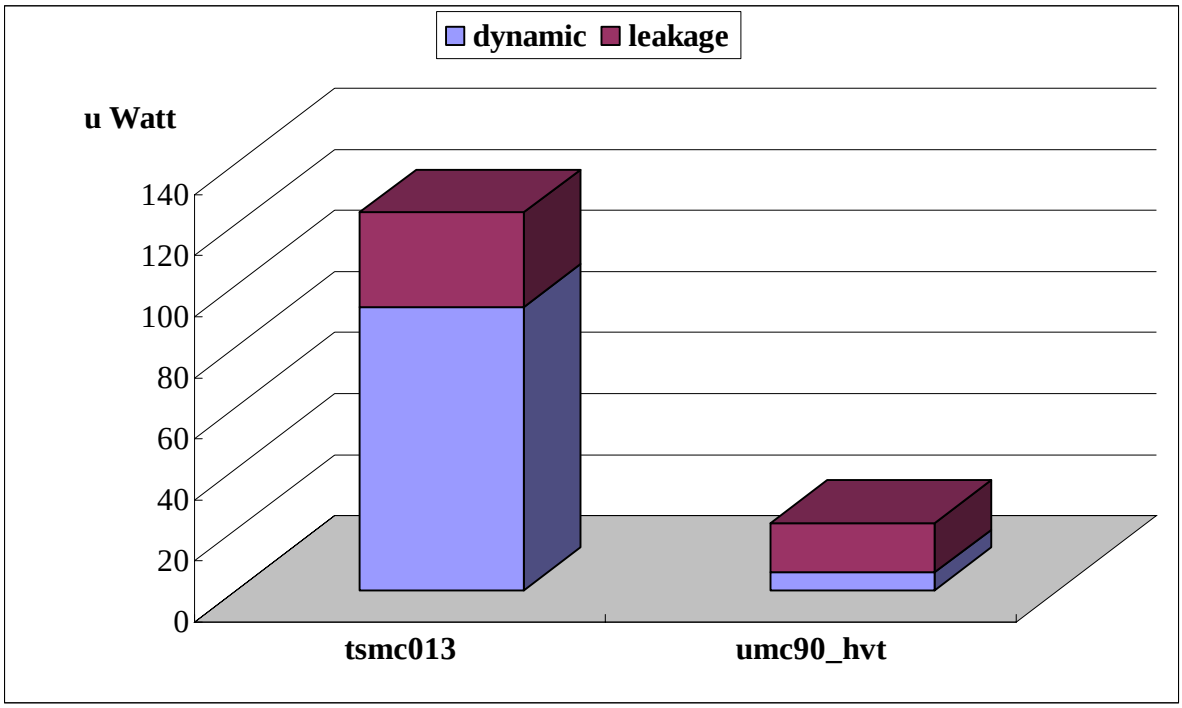


Fig 84. power consumption of TSMC_013 and UMC_90

In Fig. 84, we have the power consumption of 123.8u Watt for process TSMC_013. Moreover, process UMC_90's power report is 22.26u Watt. Considering we need the more information of our design, table 8 will have all the data we need. Table 8 as shows as follows:

table 8. data of echo canceller's report

Process	**Library**	**voltage**	**Gate count**	**Power report (u Watt)**		
TSMC013	tsmc_013	1.3 v	6.5K	Total:	***123.8***	
				Dynamic:	93.06	(75.17%)
				Leakage:	30.74	(24.83%)
UMC90	l90sphvt	0.9 v	8.1K	Total:	***22.26***	
				Dynamic:	5.88	(26.42%)
				Leakage:	16.38	(73.58%)

In table 8, we can find out the leakage power consumption will more than dynamic power consumption in process of UMC 90. For that reason, to use folding to reduce the design's area is necessary. We show our echo canceller's layout in Fig. 85.

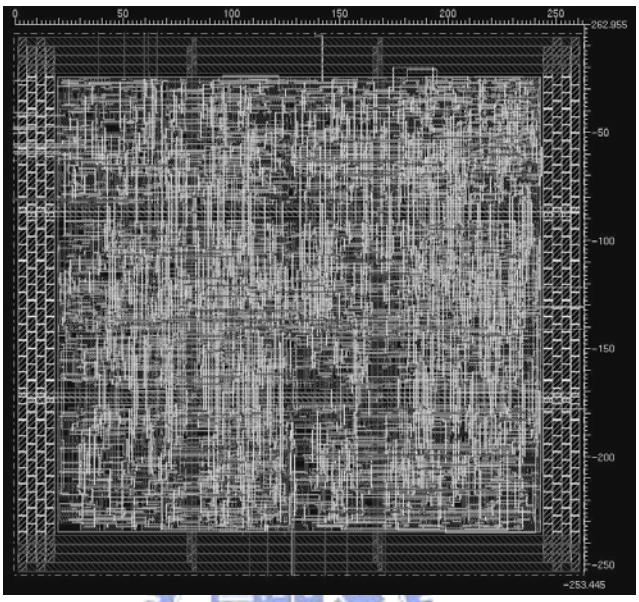


Fig 85. design's layout

Based on TSMC_013 process, we show our design's power consumption of four units in table 9.

table 9. data of echo canceller's report

Unit's name	REG_FILE	CONTROL_UNIT	TAP_UNIT	PARTIAL_UNIT
u Watt	69.208	29.07	24.41	1.112

We can show the percentage of unit's power consumption in Fig. 86.

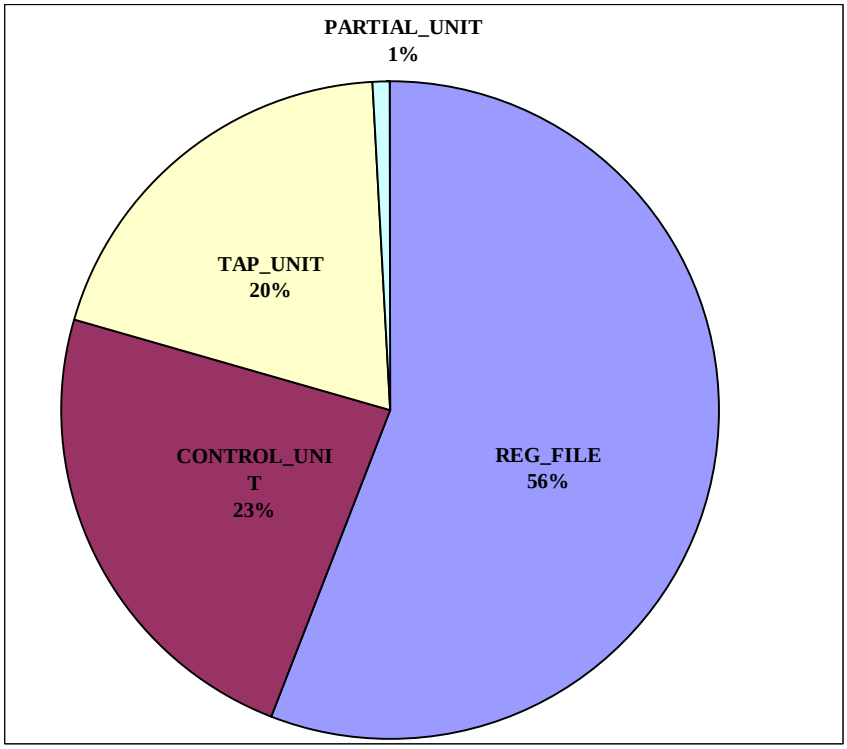


Fig 86. percentage of unit's power consumption

We can find out the REG_FILE has 56%, TAP_UNIT has 20% and CONTROL_UNIT has 23% power consumption in our design. Moreover, the PARTIAL_UNIT only consumed 1% power consumption in this case. In section 4.2.2, we will start compared our design with other related works.

4.2.2 Compared with related works

We start to compare the power consumption with other related works. Accordingly, we choose the data of UMC_90 to compare others. Because we have different taps, bits, and sampling frequency, the comparison will use the energy per operation expression. The energy per operation expression functions as defines as follows:

Energy per operation = power consumption / (taps*bits*sampling frequency) (4.2.1)

In table 10, we have the comparison of our design, and other related works.

table 10. table of compared with related works

designs	V_{DD} (v)	Process (um)	Sampling frequency	taps	bits	Total power	Energy per operation
1.	2.5	1.2	32K (Hz)	32	16	44.55m (W)	2720p (J)
2.	2.5	0.25	64K (Hz)	120	10	200m (W)	2604p (J)
3.	0.4	0.35	22K (Hz)	34	10	1.533m (W)	205p (J)
Our designs	0.9	0.09	16K (Hz)	32	12	22.26u (W)	3.623p (J)

We have introduced these related works in chapter 1, so we do not describe these designs in this section. Nevertheless, we will show the name of those three related works again. Designs of related works as shows as follows:

1: Implementation of pipelined LMS adaptive filter for low-power VLSI applications [9].

2: A Low power adaptive filter using dynamic reduced 2's-complement representation [10].

3: Ultra-low power DLMS adaptive filter for hearing aid applications [11].

We have the comparison of energy per operation in Fig. 88. In the main, we can find out our designs is ultra-low power designs with other related works. Because of we develop the new update algorithm and use the different architecture design skills for our process, so our design's power consumption will be minimized.

We show the comparison of energy per operation in Fig. 87.

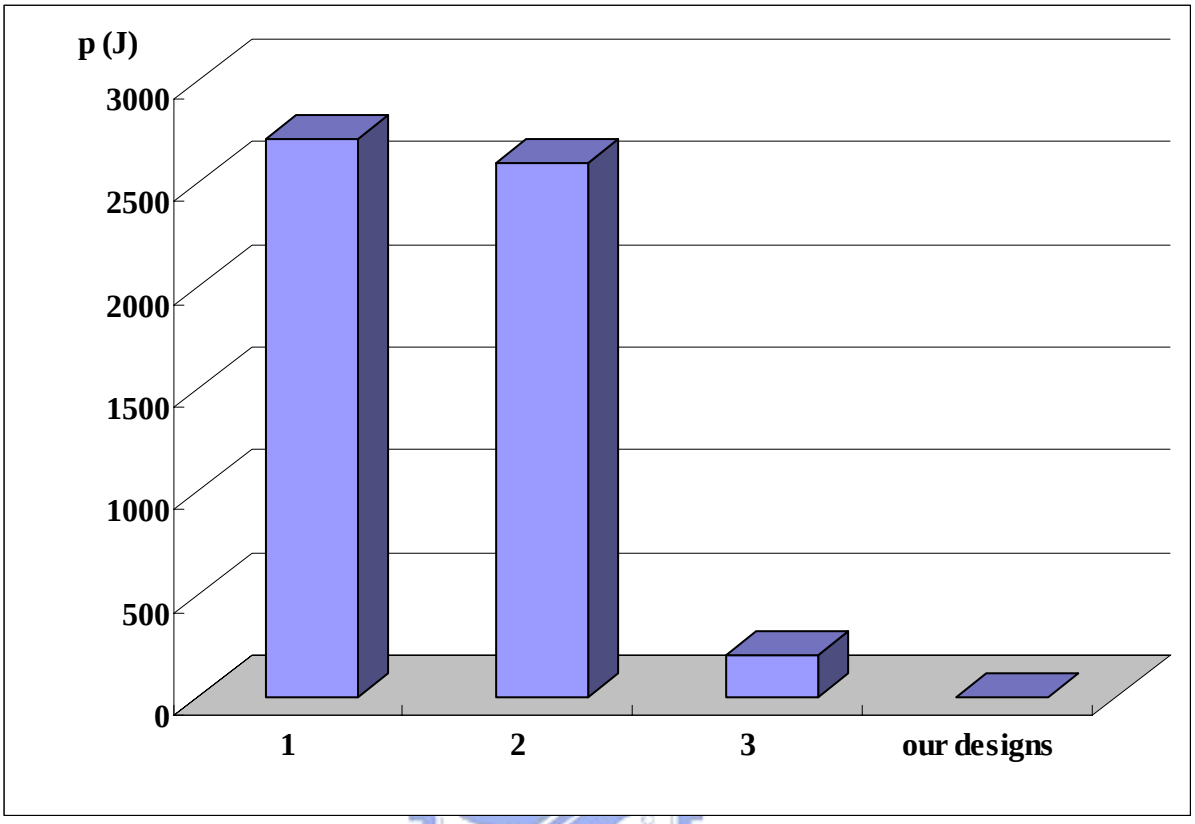


Fig 87. comparison of energy per operation

Chapter 5
Conclusions & Future Works

We develop the acoustic feedback model for simulation and the P^2SPT (partial & progressive signed power-of-two) algorithm for simplest hardware structure. For minimized power consumption, we have developed new architecture by using folding and SRAM-types register file.

We also have showed the performance of P^2SPT algorithm, the test included:

- Compared with other algorithms.
- Human's voice of man, woman, boy, and girl.
- Test of long input data and forward path changed.

Finally, we have presented the data of power consumption, the report included:

- Two processes power report of TSMC_013 and UMC_90.
- Ratio of dynamic and leakage power consumption.
- Ratio of power consumption for each component.
- Compared with other low-power designs.

The future work can be: (1) considered the real forward path effect in this acoustic feedback model, (2) considered the A/D converter delays in this acoustic feedback model, (3) developed the real echo feedback channel by dummy head experimentation, (4) adapted the P^2SPT algorithm's coefficients and it's partial update condition when the model constructed, (5) real time prototyping on FPGA platform, (6) designed the library of low leakage power consumption and low voltage for hardware design.

Bibliography

[1] S. Kotchkin, Semi-Annual Hearing Aid Market Survey, Better Hearing Institute Marke Trak VII™

[2] M. G. Siqueira, and A. Alwan, "Steady-State Analysis of Continuous Adaptation in Acoustic Feedback Reduction Systems for Hearing-Aids," *IEEE Trans. Signal Processing*, vol. 8, no. 4, July 2000

[3] H. Dillon, Hearing Aids, *Boomerang*, Turramurra, Australia, 2000

[4] J. Hellgren, "Variations in the feedback of hearing aids," *J. Acoust. Soc. Am*. Vol. 106, no. 2821, 1999

[5] W. Hsu, F. Chui, and D. A. Hodges, "An acoustic echo canceller," *IEEE Journal Solid State Circuits* vol. 24, no. 6, pp. 1639-1646, Dec. 1989.

[6] J. R. Treichler, C. R. Johnson, Jr., and M. G. Larimore, *Theory and Design of Adaptive Filters*. New York: Wiley-Intersci., 1987.

[7] R. Aichner, W. Herbordt, H. Buchner, and W. Kellermann. *Least-squares error beamforming using minimum statistics and multichannel frequencydomain adaptive filtering*. In Proc. Int. Workshop on Acoustic Echo and Noise Control (IWAENC), pages 223–226, Kyoto, Japan, Sep. 2003.

[8] K. Roy, Ultra-Low-Power DLMS Adaptive Filter for Hearing Aid Applications, *IEEE Trans. VLSI Systems*, vol. 11, No. 6, pp. 1058-1067, Dec. 2003

[9] B. Dukel, "Implementation of pipelined LMS adaptive filter for low-power VLSI applications," *IEEE CUSTOM INTEGRATED CIRCUITS CONFERENCE*, pp.II533-II536, 2002

[10] Z. Yu and M. L. Yus,"Low Power Adaptive Filter Using Dynamic Reduced 2's-Complement Representation," *IEEE CUSTOM INTEGRATED CIRCUITS CONFERENCE*, pp.141-144, 2002

[11] H. Kim and K. Roy, "Ultra-Low Power DLMS Adaptive Filter For Hearing Aid Applications" *IEEE Trans. VLSI Systems*, Vol. 11, No. 6, Dec. 2003

[12] S. Haykin, *Adaptive filter theory*, 4th ed. Prentice-Hall, 2002.

[13] K. K. Parhi, VLSI Digital Signal Processing Systems Design and Implementation, pp. 645-692, New York, 1999

[14] V. J. Mathews, "Performance analysis of adaptive filters equipped with dual sign algorithm" *IEEE Trans. on Signal Processing*, vol. 39, pp. 85-91, Jan. 1991.

[15] K. H. Chen, C. N. Chen, and T. D. Chiueh, "Grouped Signed Power-of-Two Algorithms for Low-Complexity Adaptive Equalization" *IEEE Trans. on Circuits and Systems*, vol. 52, no. 12, Dec. 2005

[16] K. Do˘gança, and O Tanrıkulu, "Adaptive Filtering Algorithms With Selective Partial Updates" *IEEE Trans. on Circuits and Systems*, vol. 48, no. 8, August 2001

[17] P. S. Diniz, Adaptive Filtering Algorithms and Practical Implementation, pp. 141-146, Netherlands, 2002

[18] P. S. Diniz, Adaptive Filtering Algorithms and Practical Implementation, pp. 152-153, Netherlands, 2002

[19] Y. C. Lim and S. Parker, "FIR filter design over a discrete powers-of-two coefficient space," *IEEE Trans. Acoust., Speech, Signal Processing*, vol. ASSP-31, pp. 583-591, June 1983.

[20] *G*, Pirani and V. Zingarelli, "Adaptive multiplication-free transversal equalizers with application to digital radio systems," *IEEE Trans. on Communications*, vol. COM-32, pp. 1025-1033, Sept. 1984.

[21] P. S. Diniz, Adaptive Filtering Algorithms and Practical Implementation, pp. 151-152, Netherlands, 2002

[22] P. Xue and B. Liu, "Adaptive equalizer using finite-bit power-of-two quantizer" *IEEE Trans. on Acoust, Speech, and Signal Processing*, vol. ASSP-34, pp. 1603-1611, Dec. 1986

[23] E. Eweda, "Convergence analysis and design of an adaptive filter with finite-bit power-of-two quantizer error" *IEEE Trans. on Circuits and Systems II*, vol. 39, pp. 113-115, Feb. 1992

[24] M. Godavarti, "Implementation of a G.165 line echo canceller on Texas Instruments' TMS320C3x and TMS320C54x chips," in *Proc. ICSPAT Conf.*, pp. 65–69, Sep. 1997.

[25] J. R. Treichler, C. R. Johnson, Jr., and M. G. Larimore, *Theory and Design of Adaptive Filters*. New York: Wiley-Intersci., 1987.

[26] S.C. Douglas, "Adaptive filters employing partial updates," *IEEE Trans. Circuits Syst.*, vol. CAS-II, pp. 209–216, Mar. 1997.

[27] J. Hellgren, T. Lunner, and S. Arlinger, "System identification of feedback in hearing aids," *J. Acoust. Soc. Am.*, vol. 105, no. 6, June 1999

[28] H. C. Howell, K. A. Weaver, and D. S. Barker, "The hearing aid feedback path: Mathematical simulations and experimental verification," *J. Acoust. Soc. Am.*, vol. 78, no. 5, Nov. 1985

[29] J. Hellgren, T. Lunner, and S. Arlinger, "Variations in the feedback of hearing aids," *J. Acoust. Soc. Am.*, vol. 106, no. 5, Nov. 1999

[30] J. Yang, M. T. Tan and J. S. Chang "Modeling External Feedback Path of an ITE Digital Hearing Instrument for Acoustic Feedback Cancellation" *IEEE Trans. Signal Processing*, vol. 53, no. 3, pp. 1326-1329, July 2005

[31] Hero, A.O., "Partial update LMS algorithms," *IEEE Trans. Signal Processing*, vol. 53, issue 7, pp. 2382-2399, July 2005

[32] A. H. Sayed, Fundamentals of Adaptive Filtering, pp. 225-229, New Jersey, 2003

[33] P. S. Diniz, Adaptive Filtering Algorithms and Practical Implementation, pp. 157-159, Netherlands, 2002

[34] P. S. Diniz, Adaptive Filtering Algorithms and Practical Implementation, pp. 166-169, Netherlands, 2002

[35] K. K. Parhi, VLSI Digital Signal Processing Systems Design and Implementation, pp. 149-186, New-York, 1999

[36] K. K. Parhi, VLSI Digital Signal Processing Systems Design and Implementation, pp. 645-692, New-York, 1999

[37] J. L. Hennessy & D. A. Patterson, Computer Architecture a Quantitative Approach, third edition, pp. 111-117 & pp. 181-189, San Francisco, 2003