IMPLEMENTATION OF SPEECH CODING ALGORITHMS ON BF526

A Project Report

submitted by

YELAMATI AVINASH ABHISHEK

in partial fulfilment of the requirements

for the award of the degrees of

BACHELOR OF TECHNOLOGY

AND

MASTER OF TECHNOLOGY

DEPARTMENT OF ELECTRICAL ENGINEERING

INDIAN INSTITUTE OF TECHNOLOGY, MADRAS

CHENNAI 600036

May 2014

THESIS CERTIFICATE

This is to certify that the thesis titled **Implementation of Speech Coding Algorithms on BF526**, submitted by **Yelamati Avinash Abhishek**, to Indian Institute of Technology, Madras, for the award of the degrees of **Bachelor of Technology** and **Master of Technology**, is a bona fide record of the research work done by him under my supervision. The contents of this thesis, in full or in parts, have not been submitted to any other Institute or University for the award of any degree or diploma.

Prof. Dr. R. Aravind

Project Guide

Professor

Dept. of Electrical Engineering

IIT Madras, 600036

Place: Chennai

Date: 6th May 2014

ACKNOWLEDGEMENTS

I would like to express my gratitude to my project guide, Prof. Dr. R. Aravind, who was very helpful throughout the course of this work. He has inspired me to think laterally and I have thoroughly enjoyed implementing novel ideas. He has been a great source of knowledge and inspiration for me in fields not restricted to signal processing, but life in general.

Next, I thank my co-partner M. Abhishek for his continued help and support in helping me complete my project in time. I thank ADI-DSP lab and its members for their workspace and continued support. I wish them all the very best in all their future endeavors.

I owe thanks to all my good friends with whom I shared golden times in IIT. I am grateful to my wing-mates for making my stay at IIT a wonderful experience.

I would like to use this opportunity to thank my parents for the love and care they have bestowed on me. Their support has meant more than anything to me. Lastly I thank each and every one who have been in anyway supportive and ask for their blessings in life.

ABSTRACT

KEYWORDS: BF526 Audio Player, DC removal, Pitch estimation, Reflection coefficients

BF 526 here is used to record, process and playback the audio samples as an audio player. Speech recorded in real conditions from a microphone has a constant component, called the DC component in electronics. This DC component in the signal negatively affects the computation and may cause disturbance, hence to be removed. As a transmission system contains a speech coder which utilizes a pitch detector that is arranged to select a characteristic auxiliary signal portion from the signal to be coded in order to improve the quality of the pitch detection. In order to re-create a minimized bit rate signal with particular perceptual quality, reflection-coefficients form an important parameter.

TABLE OF CONTENTS

ACKNOWLEDGEMENTS **I**

ABSTRACT **Ii**

LIST OF TABLES **Iv**

LIST OF FIGURES **V**

1. INTRODUCTION **1**

2. HARDWARE **3**

2.1 Main Features……..... 3

3. HARDWARE INITIALIZATION FOR AUDIO **8**

3.1 List of Switches to be initialized for Audio ……... 8

3.2 SSM2603 Audio Codec Configuration …….. 10

3.3 Blackfin Audio Interface ……..... 12

3.4 Limitations ……. 13

4. DC REMOVAL **14**

4.1 IIR Filter ……. 14

5. PITCH ESTIMATION **17**

5.1 The Autocorrelation Method ……. 17

5.2 Magnitude Difference Method …….. 21

6. FRACTIONAL PITCH **24**

6.1 The Medan–Yair–Chazan Method ……..... 25

7. LINEAR PREDICTION ANALYSIS 26

7.1 Prediction Schemes ……..... 27

7.2 The Levinson–Durbin Algorithm……... 29

7.3 The Leroux–Gueguen Algorithm …….. 31

7.4 Prediction Gain ……. 34

8. CONCLUSION AND FUTURE SCOPE OF WORK **36**

LIST OF TABLES

5.1 Pitch comparison at 8 kHz and 16 kHz sampling rate …….. 20

5.2 Pitch comparison of two different signals at 8 kHz sampling rate ……..... 20

7.1 Reflection Coefficients for 8 KHz Sampling Rate, Low Pitch ……. 32

7.2 Reflection Coefficients for 16 KHz Sampling Rate, Low Pitch ……... 33

7.3 Reflection Coefficients for 8 KHz Sampling Rate, High Pitch …… 33

LIST OF FIGURES

2.1 Blackfin Platform Overview ……..... 4
2.2 BF526 Architecture ……... 5

3.2 Sport configuration ……... 10
3.3 TWI interface …… 11
3.4 ADC configuration …… 12
3.5 DAC configuration …… 12
3.6 Chart comparing BF526 and BF533 ……. 13

4.1 Input Signal Right and Left Channels ……... 15
4.2 Output Signal Right and Left Channels …… 16

5.1 Autocorrelation plot for 8 KHz sampling rate, Low pitch recording …… 19
5.2 Autocorrelation plot for 16 KHz sampling rate, Low pitch recording …….. 19
5.3 Autocorrelation plot for 8 KHz sampling rate, High pitch recording ……... 20
5.4 MDF plot for 8 KHz sampling rate, Low pitch recording ……... 22
5.5 MDF plot for 16 KHz sampling rate, Low pitch recording ……..... 22
5.6 MDF plot for 8 KHz sampling rate, High pitch recording ……... 23

7.1 Frame wise calculation of LPCs ……... 27
7.2 Plot of Prediction Gain vs Order for Voiced and Unvoiced signal ……... 34

CHAPTER 1

INTRODUCTION

Speech is the most desirable medium of communication between humans. There are several ways of characterizing the communications potential of speech. Most of the speech transmission happens over mobile telephony and Voice over IP. For effective transmission this involves transmitting large number of bits, which requires serious hardware and huge complexities. The main goal of speech coding is either to maximize the perceived quality at a particular bit-rate, or to minimize the bit-rate for a particular perceptual quality. In general, speech coding can be considered to be a particular specialty in the broader field of speech processing.

Speech processing is currently experiencing a revolution that has been brought about by the advancements of digital signal processing (DSP) techniques and systems. Conversely, it is fair to say that speech processing has been a cradle for DSP, in the sense that many of the most widely used algorithms in DSP were developed or first brought into practice by people working on speech processing.

The primary purpose of a speech coding is to convert an analogue speech signal into digital form for efficient transmission or storage and to convert a received digital signal back to analogue. Speech coding uses specific speech parameter estimation using audio signal processing techniques to model the speech signal, combined with generic data compression algorithms to represent the resulting modeled parameters in a compact bit-stream.

For the decoding part, determining if a segment is a voiced or unvoiced sound is not all of the information that is needed by the LPC decoder to accurately reproduce a speech signal. In order to produce an input signal for the LPC filter the decoder also needs another attribute of the current speech segment known as the pitch period.

We require an efficient platform like Blackfin where can perform High Performance for real time audio signal processing. High performance, low power and ease to use are the key factors which makes Blackfin apt for DSP applications. Here we provide results on simulation of techniques used for speech coding.

The rest of the thesis is organized as follows: Chapter 2 describes about the Blackfin platform and its features. Chapter 3 involves Hardware initialization for audio applications. In chapter 4, we discuss how to eliminate DC using IIR filter. Chapter 5&6 involves integer and fractional pitch estimation. Chapter 7 is about algorithms to calculate reflection coefficients.

CHAPTER 2

HARDWARE

The BF-526 is a fixed-point digital signal processor developed by Analog Devices. It's a member of the Blackfin family but offers much better performance with a low power consumption compare with the previous Blackfin processors.

2.1 Main Features

• Clock Speed of 400MHz

• Dual 16-Bit multiplication-accumulation (MAC)

• Two 40-Bit Arithmetic Logic units (ALU)

• Four 8-Bit Video ALUs

• 40-Bit shifter

• RISC like Register and Instruction Model

The BF-526 processor system also includes peripherals. These peripherals are connected to the core of the processor via several high bandwidth buses. Some of these important peripherals are described below.

2.1.1 Watchdog Timer:

This is a 32-bit timer that can be used to implement a software watchdog function.

A watchdog timer is a piece of hardware built into the processor that causes the processor i.e. reset or interrupt when it judges that the system has hung, or is no longer executing the correct sequence of code. The watchdog timer is initialized and enabled by the programmer.

2.1.2 Real-Time Clock:

The Real-Time Clock (RTC) provides a robust set of digital watch features such as: stopwatch, current time and alarm and is clocked by a 32.768 kHz crystal external to the processor. The RTC has a separate power supply that is isolated from the rest of the DSP.

This means that even when the DSP is in low power state, the RTC remain powered to preserve the current time and calendar information. The RTC can wake up the processor from Sleep mode or Deep Sleep mode by generating an RTC wake-up event.

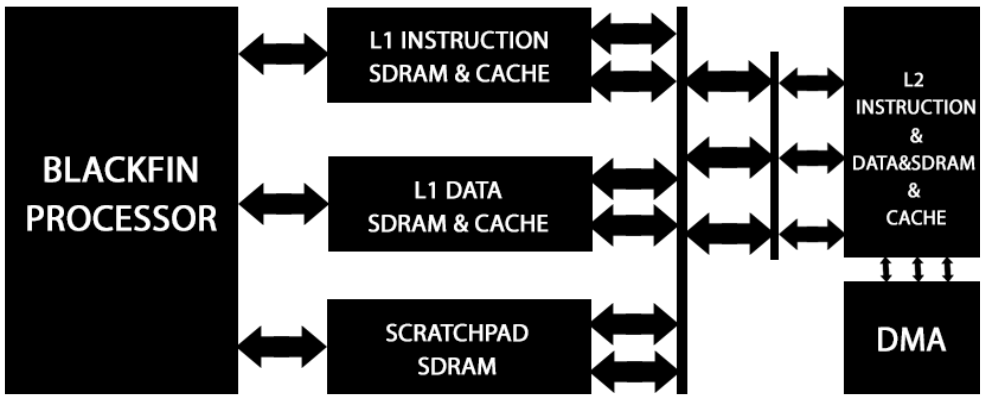

Figure 2.1: Blackfin Platform Overview

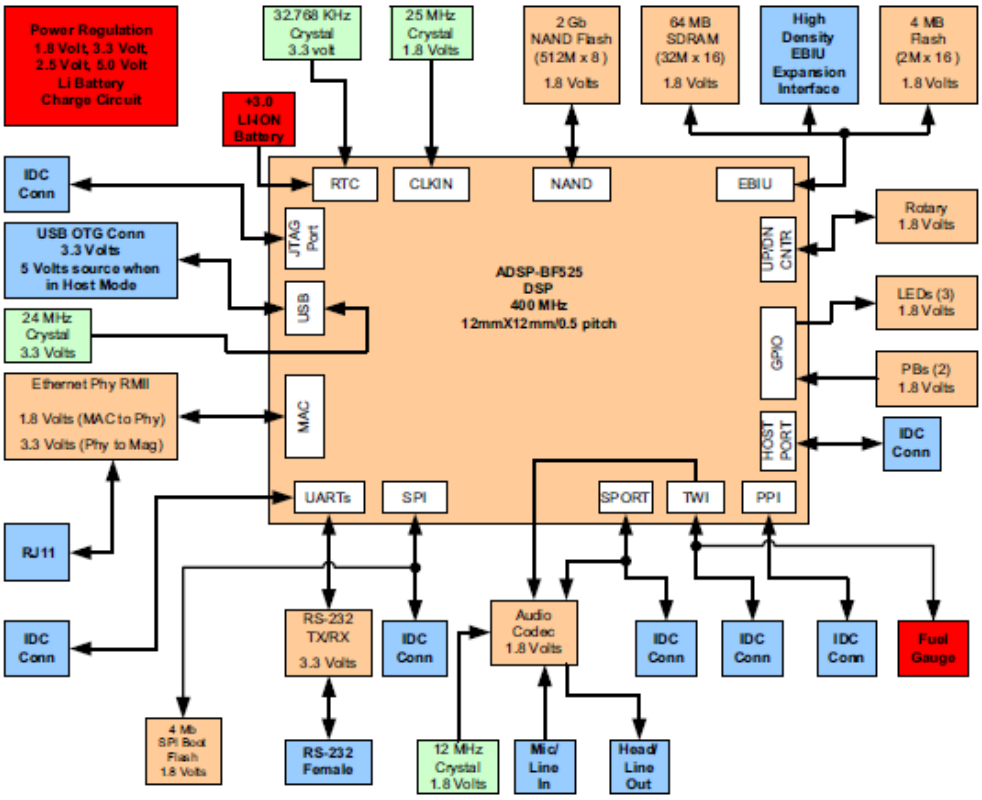

Figure 2.2: BF526 Architecture

2.1.3 Serial Ports:

The processor has two dual-channel synchronous serial ports (SPORT0 and SPORT1) for serial and multiprocessor communications. Some main features of these SPORTs are:

• Capable to operate as a bidirectional I2S serial bus. Each SPORT has two sets of independent transmit and receive pins, enabling eight channels of I2S stereo audio.

• Each transmit and receive port can used either as an external serial clock or can generate its own clock in a wide range of frequencies.

• Each SPORT can be set to data word lengths from 3 to 32 bits. And can be set to transfer the most significant bit (MSB) first or the least significant bit (LSB) first transferred in most significant bit first or least significant bit first format.

• The SPORTs can generate an interrupt once completing the transfer of a data word or after transferring an entire data buffer or buffers through DMA.

• Each SPORT can operate according to ITU recommendation.

2.1.4 Serial Peripheral Interface (SPI) Port:

Serial peripheral interface (SPI) is an interface that enables the serial (one bit at a time) exchange of data between two devices, one device called a master and the other device called a slave. The SPI operates in full duplex mode which means that data can be transferred in both directions at the same time. The select input pin (SPISS) lets other SPI devices select the processor, and seven SPI (SPISEL 1-7) select output pins let the processor select other SPI devices.

2.1.5 Direct Memory Access Controller (DMA):

Direct memory access (DMA) is a means of having a peripheral device control a processor’s memory bus directly. DMA permits the peripheral, such as a UART, to transfer data directly to or from memory without having each byte (or word) handled by the processor. DMA transfers can occur between the internal memories and any of processor’s DMA-capable peripherals. In addition DMA transfer can happen between any DMA-capable peripheral and external devices which are connected to the external memory interfaces such as SDRAM controller and the asynchronous memory controller. DMA capable peripherals include the SPORTs, SPI port, UART, and PPI. Each DMA-capable peripheral has at least one dedicated DMA channel.

The DMA controller is a device that controls these DMA transfers. Upon a request for DMA transfer from any DMA-capable peripheral, it initiates a DMA request signal to the processor asking its permission to use its bus, once permission from the processor was granted, the DMA controller reads and writes one or more memory bytes, driving the address, data, and control signals as if it were itself the processor. When the transfer is complete, the DMA controller de-asserts the DMA request signal and the processor resumes its control of the bus.

2.1.6 L1 Instruction and Data Memory:

L1 instruction Memory and L1 data memory are both part of the internal memory and are accessed at full processor speed. The instruction memory consist of up to 80K bytes SRAM, of which 16K bytes can be configured as a four-way set-associative cache. The data memory consist of up to two banks of up to 32K bytes each. Each memory bank is configurable, offering both Cache and SRAM functionality.

All of the peripherals, except for general-purpose I/O, Real-Time Clock, and Timers, are supported by the DMA. In addition, the processor core can operate in full speed even when all peripherals are used due to the multi on-chip busses.

CHAPTER 3

HARDWARE INITIALIZATION FOR AUDIO

3.1 List of Switches to be initialized for Audio:

- SPORT0A ENBL Switches/I2C ENBL (SW2 and SW7)
- MIC Gain Switch (SW9)
- Audio LPBK (Loopback) Switch (SW10)
- GPIO Enable Switch (SW20)

3.1.1 SPORT0A ENBL Switches/I2C ENBL (SW2 and SW7):

The SPORT0A enable switches (SW2 and SW7) connect the SPORT0A and TWI interfaces of the processor to the audio codec, SSM2603 (U31). When the SPORT0A interface is used on the expansion interface II, turn SW2 and SW7 positions 1 and 2 all OFF. To disconnect TWI to the audio codec turn SW2 positions 3 and 4 OFF. By default, SW2 and SW7 are set to all ON.

3.1.2 MIC Gain Switch (SW9):

The microphone gain switch (SW9) sets the gain of the MIC signal, which is connected to the top 3.5 mm jack (J4). The gain can be set to 14 dB, 0 dB, or –6 dB by turning ON position 1, 2, or 3 of SW9.

When the corresponding position for the desired gain is ON, the remaining positions must be OFF. Here we have used the default value of -6dB.

3.1.3 Audio LPBK (Loopback) Switch (SW10):

The SW10 switch places the EZ-Board in a loopback to test the board for Signal/circuit continuity and functionality. SW10 positions 1 and 2 connect the MICIN signal to the headphone's left and right outputs for audio loopback. Do not turn SW10 positions 1 and 2 ON at the same time.

3.1.4 GPIO Enable Switch (SW20):

The general-purpose input/output switch (SW20) disconnects the associated push buttons and LED circuits from the GPIO pins of the processor and allows the signals to be used for other purposes. Depending on the switch configuration, the signals can be used as push buttons, one-time-programmable memory (OTP) flag for writing, or on-the-go (OTG) host mode 5V.

3.2 SSM2603 Audio Codec Configuration:

- Auto SPORT
- TWI
- Sampling Rate
- Clock

3.2.1 Auto Sport:

A synchronous, high speed serial port that can support TDM, I2S and a number of other configurable framing modes for connection to ADCs, DACs, other processors, FPGAs, etc. SPORT is used as a data channel for audio data. It connects TX to audio DAC. And further RX to audio ADC. Full-duplex SPORT RX/TX connects to audio codec. The driver can sense changes to SSM2603 Digital Audio Interface Format Register and automatically updates the corresponding SPORT device configuration registers to support the present interface mode. This feature is termed as Auto-SPORT configuration support. SSM2603 Audio Codec doesn't support DMA, but supports indirectly via SPORT. It also handles audio dataflow between Blackfin and SSM2603

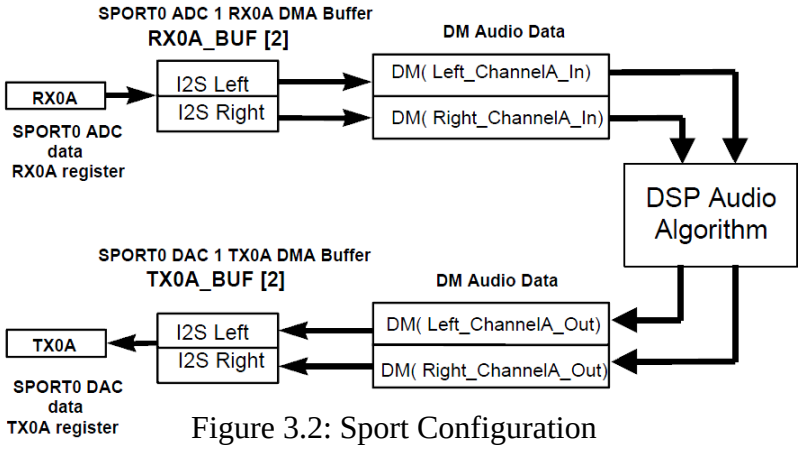

Figure 3.2: Sport Configuration

3.2.2 TWI:

Another popular serial peripheral interface bus

- More flexible than SPI
- Master and slave modes supported
- 7-bit slave address
- 400khz data transfer speed
- Bidirectional, open-drain bus (device pulls down, resistors pull up)
- Two wires, SCL (clock) and SDA (data)

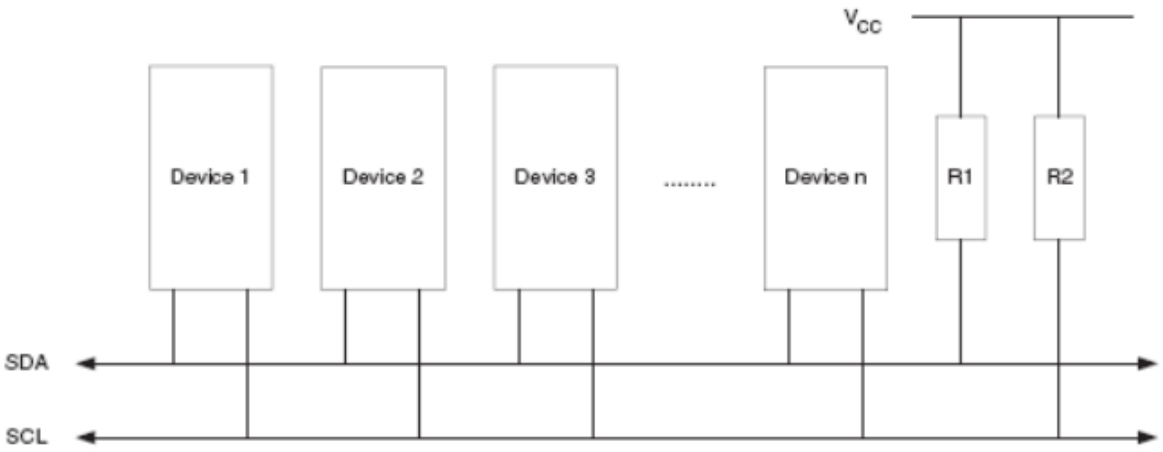

Figure 3.3: TWI interface

3.2.3 Sampling Rate:

The present Blackfin board supports sampling rates of 8K, 16k, 48K and 88.2k. As we are dealing with speech, sampling rates of 8k and 16k are sufficient. So, in SSM2603 we initialized ADC and DAC at two different sampling rates i.e. 8k and 16k. The sampling rates should be in accordance with Master clock.

3.3 BLACKFIN AUDIO INTERFACE:

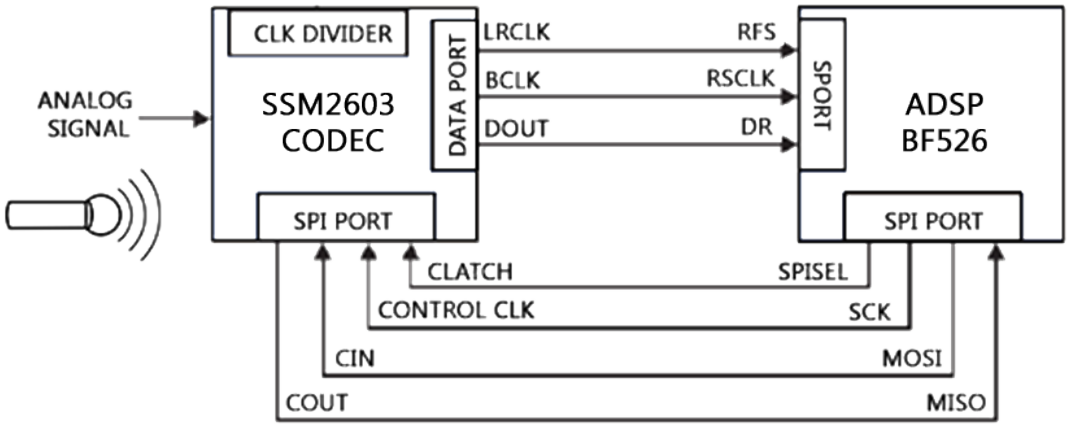

Figure 3.4: ADC configuration

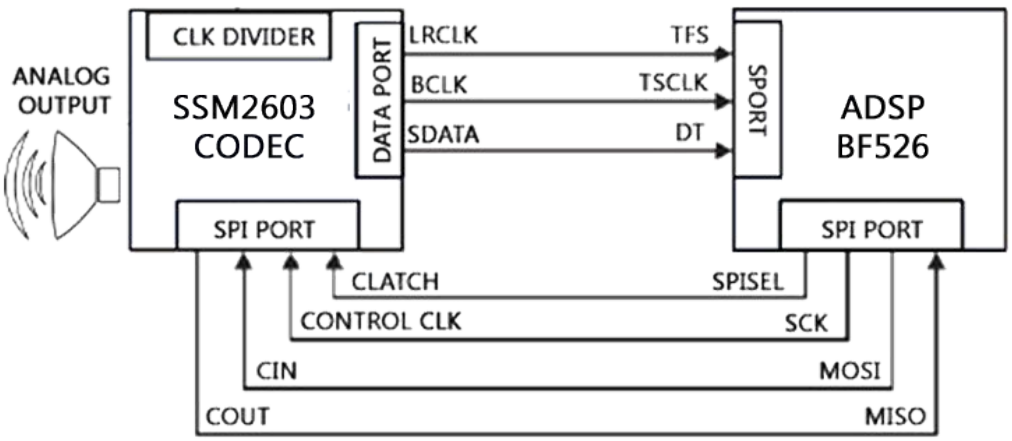

Figure 3.5: DAC configuration

3.4 LIMITATIONS

We first started working with Blackfin BF533 board. It was the most popular board with huge technical support. The only limitation we found with the BF533 board is that it has sampling rates of only 48 kHz and 96 kHz. As this would result in huge number of samples and moreover as we were dealing with voice such high sampling rates were not necessary. The next possible option was to work on BF526 which supported wide range of sampling rates like 8 kHz, 16 kHz and 48 kHz. We chose to stick to 8 kHz and 16 kHz sampling rates.

	BF526	BF533
AUDIO CODEC	SSM2603	AD1836
SAMPLING RATES	8K, 16K, 48K, 96K	48K, 96K
ON CHIP RAM	64MB	32MB

Figure 3.6: Chart comparing BF526 and BF533

Besides this we also faced few more practical limitations of direct real-time processing as the data had to be first written into SDRAM and then to be used for computation which made it offline computation. Further there was noise addition into playback due to complex computations on sample data.

CHAPTER 4

DC REMOVAL

4.1 IIR FILTER

IIR filters are digital filters with infinite impulse response. Unlike FIR filters, they have a feedback (a recursive part of a filter) and are known as recursive digital filters. For this reason IIR filters have much better frequency response than FIR filters of the same order.

In audio recording, a DC offset is an undesirable characteristic of a recording sound. It occurs in the capturing of sound, before it reaches the recorder, and is normally caused by defective or low-quality equipment. The offset causes the center of the recording waveform to not be at 0, but at a higher value, for example, +1. This can cause two main problems. Either the loudest parts of the signal will be clipped prematurely, since the base of the waveform has been moved up, or inaudible low-frequency distortion will occur.

4.1.1 Recording

- Sample format used unsigned 16-bit
- Total samples = sampling rate(Hz) * time(s)

In our case:

At 8 KHz

Total samples = 8000 * 30 = 240000

Transfer Function: $H(z) = \frac{1-z^{-1}}{1-\alpha z^{-1}}$ here **α** is close to **1**

Difference equation:

$$y[n] = x[n] - x[n-1] + \alpha\, y[n-1] \tag{4.1}$$

Here, $x[n]$ are the input samples and $y[n]$ is filtered output. We calculate the filtered output for various values of **α**.

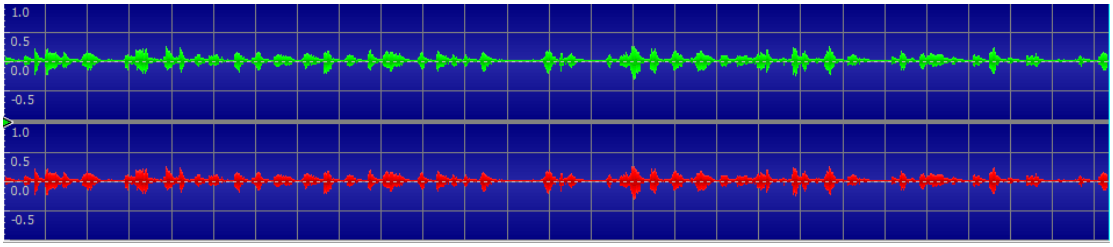

Figure 4.1: Input Signal Right and Left Channels

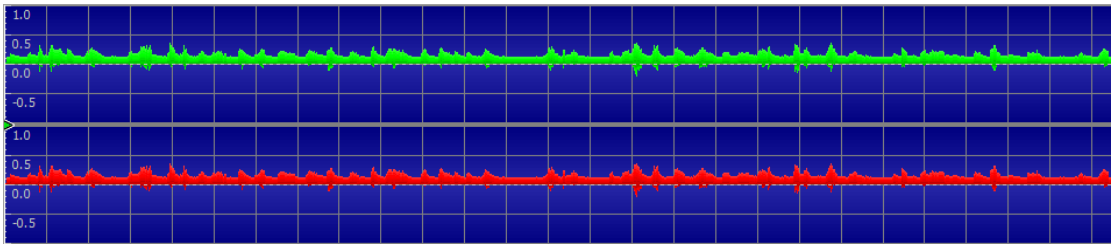

Figure 4.2: Filtered Output of Input signal of Right and Left Channels

4.1.2 Implementation and Complexity:

By using the difference equation of IIR filter, for **α** varying between 0.9 to 0.99 output samples are calculated. These output samples are run to hear the filtered voice without DC. Due to the multiplication factor **α** samples are represented in fractional point. As the platform we are dealing with is a fixed-point DSP, during cross-compiling these samples are converted back to integers which result in bit overflow. An efficient way to implement was to shift the samples and reduce the sample bit size.

CHAPTER 5

PITCH PERIOD ESTIMATION

One of the most important parameters in speech analysis, synthesis, and coding applications is the fundamental frequency, or pitch, of voiced speech. Pitch frequency is directly related to the speaker and sets the unique characteristic of a person. Voicing is generated when the airflow from the lungs is periodically interrupted by movements of the vocal cords. The time between successive vocal cord openings is called the fundamental period, or pitch period. For men, the possible pitch frequency range is usually found somewhere between 50 and 250 Hz, while for women the range usually falls between 120 and 500 Hz. In terms of period, the range for a male is 4 to 20 milliseconds, while for a female it is 2 to 8 milliseconds. The following are the two methods to find the pitch period of a voice signal.

5.1 THE AUTOCORRELATION METHOD:

Here we assume that we want to perform the estimation on the signal s[n], with n being the time index. We consider the frame that ends at time instant m, where the length of the frame is equal to N (i.e., from n = m-N+1 to m) reflects the similarity between the frame s[n], n = m-N+1 to m, with respect to the time-shifted version s [n-l], where l is a positive integer representing a time lag. The range of lag is selected so that it covers a wide range of pitch period values.

Autocorrelation Equation:

$$\mathrm{R}\,[\mathrm{l},\mathrm{m}] = \sum_{n=m-N+1}^{m} s[n] * s[n-l] \qquad (5.1)$$

By calculating the autocorrelation values for the entire range of lag, it is possible to find the value of lag associated with the highest autocorrelation representing the pitch period estimate, since, in theory, autocorrelation is maximized when the lag is equal to the pitch period.

The pseudo code for this method:

PITCH (m, N)

1. peak 0
2. for l ←20 to 150
3. autoc ←0
4. for n ←m-N+1 to m
5. autoc← autoc + s[n]*s[n - l]
6. if autoc >peak
7. peak ←autoc
8. lag← l
9. return lag

The source of the samples is a constant pitch voice recorded at our lab. As the samples we are handling are of male voice the lag would be between 40 and 90. In order to detect the peak we first ran the code by fixing the lag in range of 20 to 100. The lag value which we got in this range was 78. Once we found that peak we then

ran the code by changing the lag in range of 100 to 200. As the lag now detected was twice the lag detected in the previous range this was our principle pitch. We then changed the range of the lag in range of 200 to 300 to confirm if the peak now detected was thrice the principle pitch.

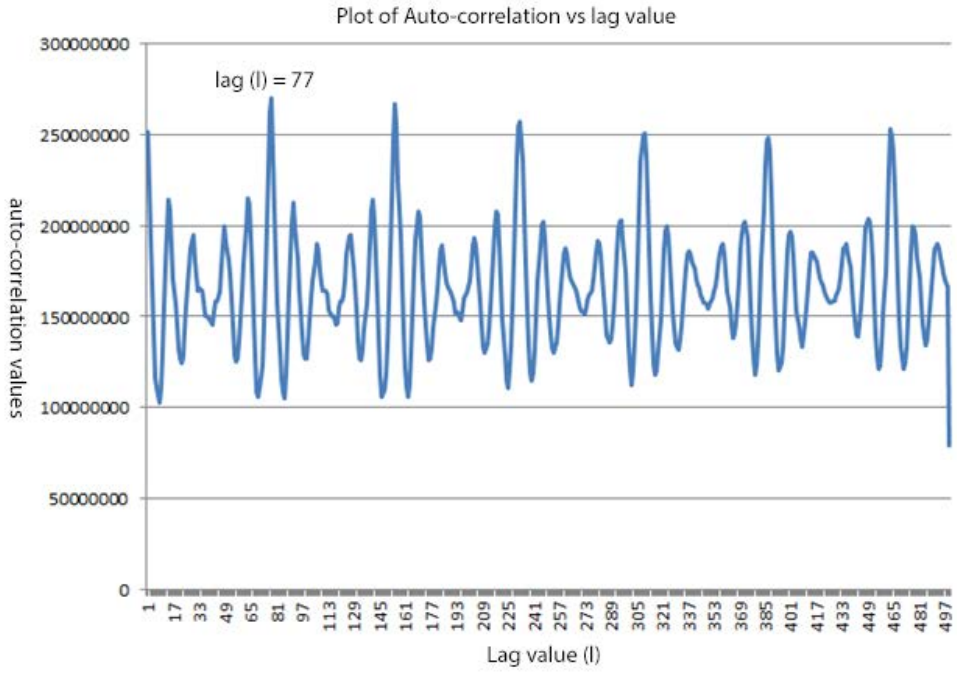

Figure 5.1: Autocorrelation plot for 8 KHz sampling rate, Low pitch recording

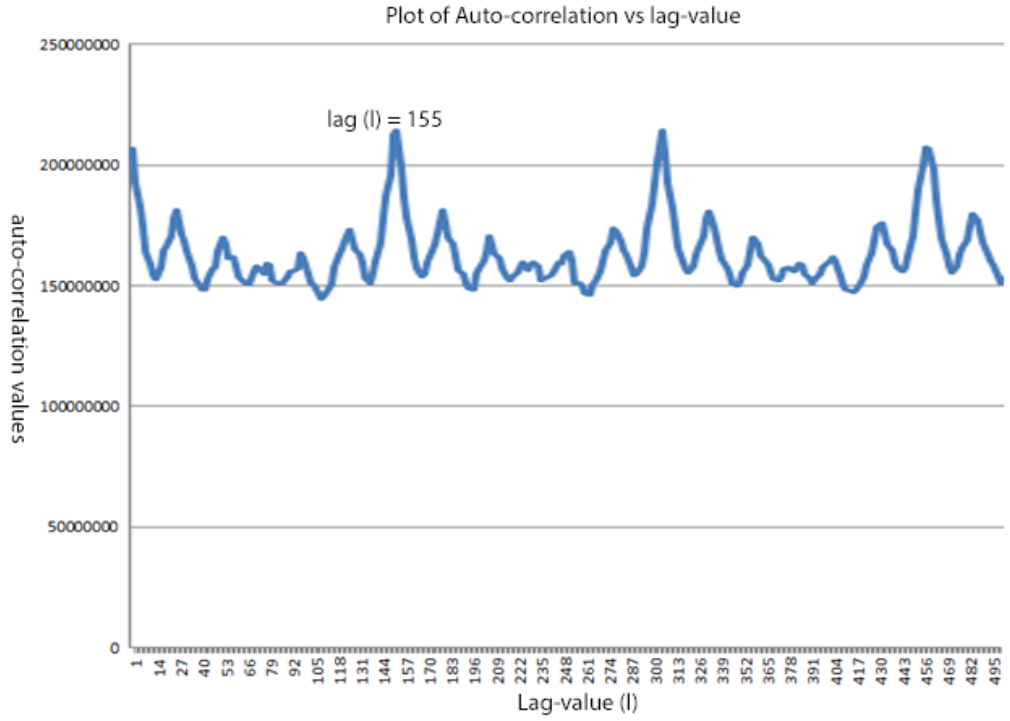

Figure 5.2 Autocorrelation plot for 16 KHz sampling rate, Low pitch recording

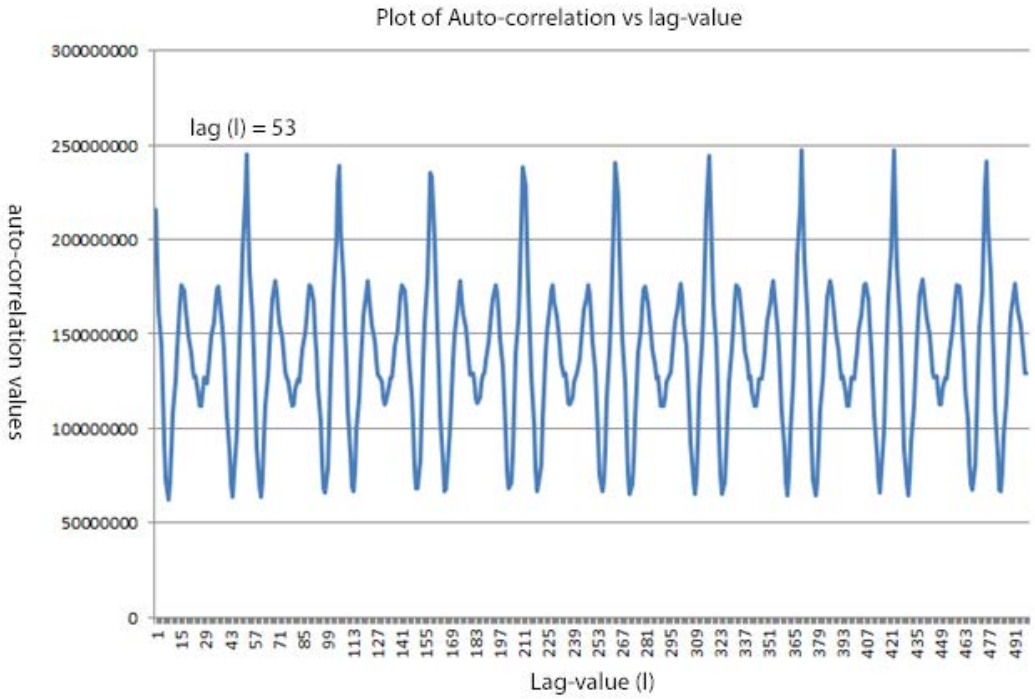

Figure 5.3: Autocorrelation plot for 8 KHz sampling rate, high pitch recording

5.1.1 Observations:

Table 5.1: Pitch comparison at 8 kHz and 16 kHz sampling rate

8k Sampling Rate	16k Sampling Rate
Pitch – Lag (L) = 78	Pitch – Lag (L) = 155
Pitch period (T) = 78/8000 = 9.75ms	Pitch period (T) = 155/16000 = 9.69ms
Fundamental frequency = 102 Hz	Fundamental frequency = 103 Hz

Table 5.2: Pitch comparison of two different signals at 8 kHz sampling rate

8k Sampling Rate (Low Pitch)	8k Sampling Rate (High Pitch)
Pitch – Lag (L) = 78	Pitch – Lag (L) = 53
Pitch period (T) = 78/8000 = 9.75ms	Pitch period (T) = 53/8000 = 6.625ms
Fundamental frequency = 102 Hz	Fundamental frequency = 151Hz

Sampling rate doesn't affect the fundamental pitch frequency.

The high pitch signal is **1.5 times** the low pitch signal.

5.2: MAGNITUDE DIFFERENCE FUNCTION:

The main drawback of the autocorrelation method is the need for multiplication, which is relatively expensive for implementation, especially in those processors with limited functionality. To overcome this problem, the magnitude difference function is used.

$$MDF[l,m] = \sum_{n=m-N+1}^{m} |s[n] - s[n-l]| \quad (5.2)$$

For short segments of voiced speech it is reasonable to expect that s [n] – s [n-l] is small for l = 0, ±T, ±2T. . . with T being the signal's period. Thus, by computing the magnitude difference function for the lag range of interest, one can estimate the period by locating the lag value associated with the minimum magnitude difference. Note that no products are needed for the implementation of the present method.

Pseudo code for this method:

```
PITCH_MD (m, N)
1. min ← ∞
2. for l ←20 to 150
3.      mdf ←0
4.      for n =m-N+1 to m
5. mdf ←mdf + |s[n] - s[n - l]|
6. if mdf < min
7.      min ← mdf
8.      lag l
9. return lag
```

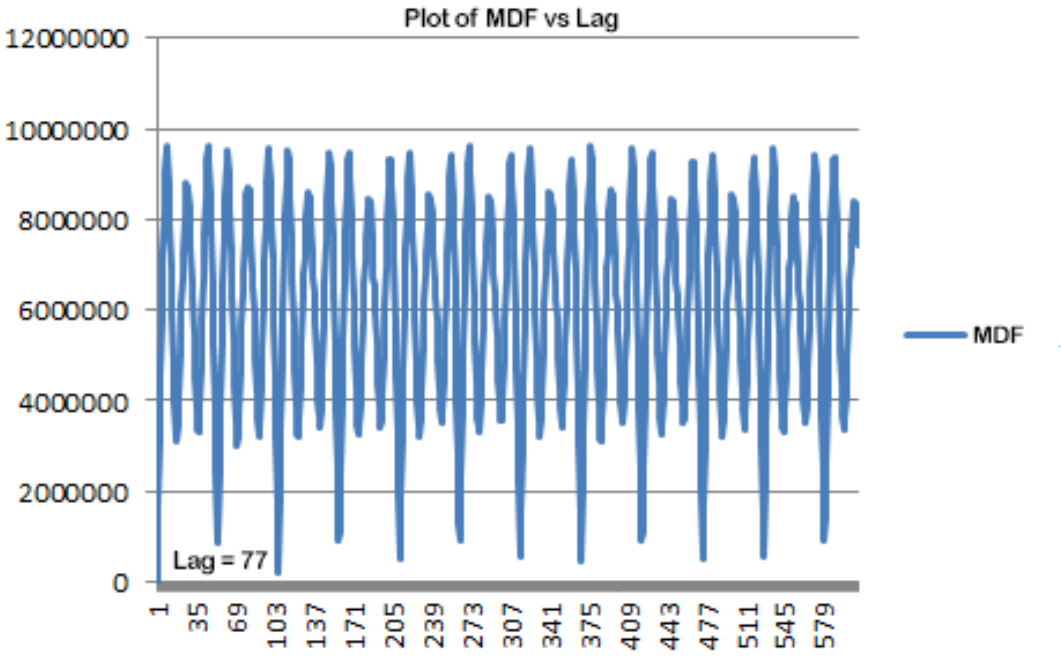

Figure 5.4: MDF plot for 8 KHz sampling rate, Low pitch recording

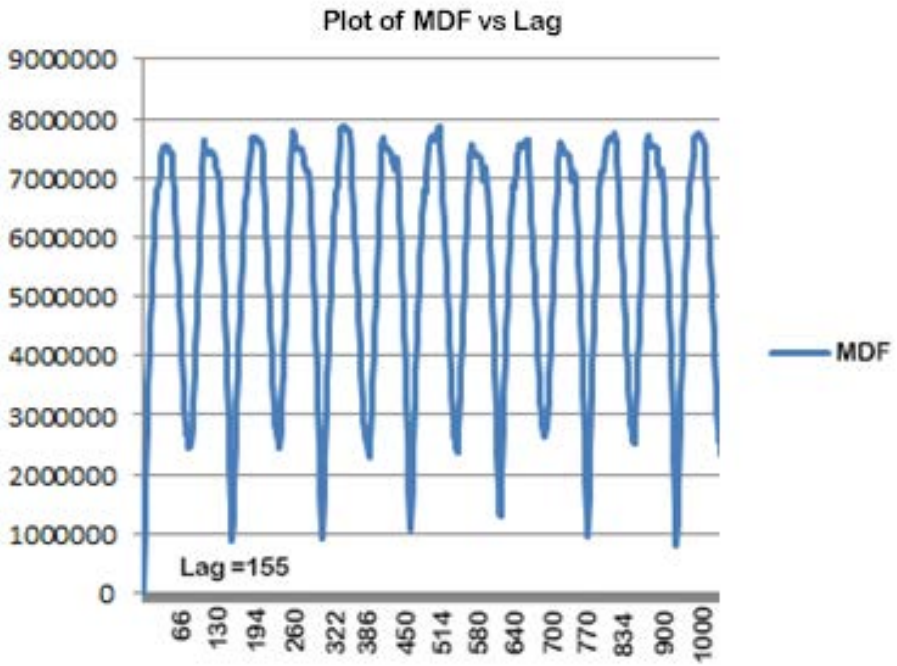

Figure 5.5: MDF plot for 16 KHz sampling rate, Low pitch recording

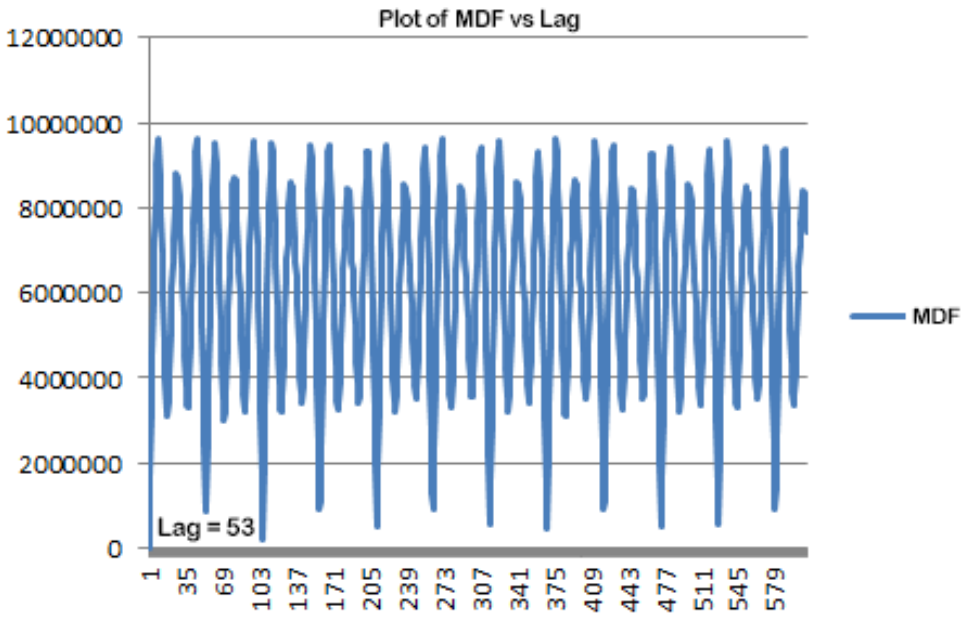

Figure 5.6: Autocorrelation plot for 8 KHz sampling rate, High pitch recording

5.2.1 Observations

As we observe from the plots, the pitch lag values are close to that in auto-correlation method. In auto correlation we use multiplication which results in register overflow, whereas in MDF method subtraction of two registers is well within range. Moreover, multiplicative-sum of registers involves a time factor compared to difference method. We prefer this method since it is computationally less complex compared to that of auto-correlation method.

CHAPTER 6

FRACTIONAL PITCH

Higher resolution is necessary to achieve good performance. In fact, pitch period of the original continuous-time (before sampling) signal is a real number; thus, integer periods are only approximations introducing errors that might have negative impact on system performance.

Medan, Yair, and Chazan published an algorithm for pitch period determination, which is based on a simple linear interpolation technique. The method allows the finding of a real-valued pitch period and can be implemented efficiently in practice.

Here we consider the continuous-valued pitch period T_0 defined by where T_s is the sampling period $(8\text{ kHz})^{-1} = 0.125\text{ ms}$ and η_0 is the fractional pitch period.

Here we assume $0 \leq \eta_0 < 1$ so that

$$\frac{T_o}{T_s} - 1 < N_o \leq \frac{T_o}{T_s}$$

Or

$$N_o = \left\lfloor \frac{T_o}{T_s} \right\rfloor$$

6.1 THE MEDAN–YAIR–CHAZAN METHOD:

Step 1: For $N = N_{min}$ to N_{max}, compute the normalized autocorrelation from

$$R[m,N] = \frac{\sum_{n=m-N+1}^{m} s[n] * s[n-N]}{\sqrt{\sum_{n=m-N+1}^{m} s^2[n] \sum_{n=m-N+1}^{m} s^2[n-N]}} \tag{6.1}$$

Step 2: Find the maximum value of $R[m,N]$ the corresponding value of N is the optimal integer pitch period N_0.

Step 3: Compute η_0 from

$$\alpha_1[m,N] = \sum_{n=m-N+1}^{m} s[n] * s[n-N_0]$$

$$\alpha_2[m,N] = \sum_{n=m-N+1}^{m} s[n] * s[n-N_0-1]$$

$$\alpha_3[m,N] = \sum_{n=m-N+1}^{m} s^2[n]$$

$$\alpha_4[m,N] = \sum_{n=m-N+1}^{m} s^2[n-N_0]$$

$$\alpha_5[m,N] = \sum_{n=m-N+1}^{m} s^2[n-N_0-1]$$

$$\alpha_6[m,N] = \sum_{n=m-N+1}^{m} s[n] * s[n-N_0-1]$$

$$\eta_0[m,N] = \frac{\alpha_2[m,N]\alpha_4[m,N] - \alpha_1[m,N]\alpha_6[m,N]}{\alpha_2[m,N](\alpha_4[m,N] - \alpha_6[m,N]) + \alpha_1[m,N](\alpha_5[m,N] - \alpha_6[m,N]}$$

Step 4: If $\eta_0 < 0$, $N_0 \leftarrow N_0 - 1$, go back to Step 3.

Step 5: If $\eta_0 \geq 1$, $N_0 \leftarrow N_0 + 1$, go back to Step 3.

Here N_0 calculated is same as the integer pitch of auto-correlation method. Pitch is the sum of integer part and fractional part. As per the algorithm if fractional pitch is less than zero then integer part is reduced by one and if the fractional part is greater than or equal to 1 then integer part is increased by one. Hence the final pitch is either less than or greater than integer part.

For 8 KHz sampling rate, total pitch:

Integer Pitch = 77

Pitch = 74.389198

For 16 KHz sampling rate, total pitch:

Integer Pitch = 155

Pitch = 157.40164

For 8 KHz (High pitch) sampling rate, total pitch:

Integer Pitch = 53

Pitch = 51.43732

6.1.1 Observations:

Here we observe that the pitch frequency calculated through total pitch is almost the same as that of integer pitch. The pitch frequency of 8 KHz and 16 KHz is the same.

He we also observe that the ratio of **1.5 times** is maintained across 8 KHz (High pitch) and 8 KHz (Low pitch).

CHAPTER 7

LINEAR PREDICTION ANALYSIS

Due to the dynamic nature of a speech signal, the LPCs must be calculated for every signal frame. Within a frame, one set of LPCs is determined and used to represent the signal's properties in that particular interval, with the underlying assumption that the statistics of the signal remain unchanged within the frame. The process of calculating the LPCs from signal data is called linear prediction analysis. LP analysis is performed for every signal frame ending at time m. The autocorrelation values $R[l,m]$are estimated for each frame and the normal equation is solved to yield the set of LPCs associated with the particular frame.

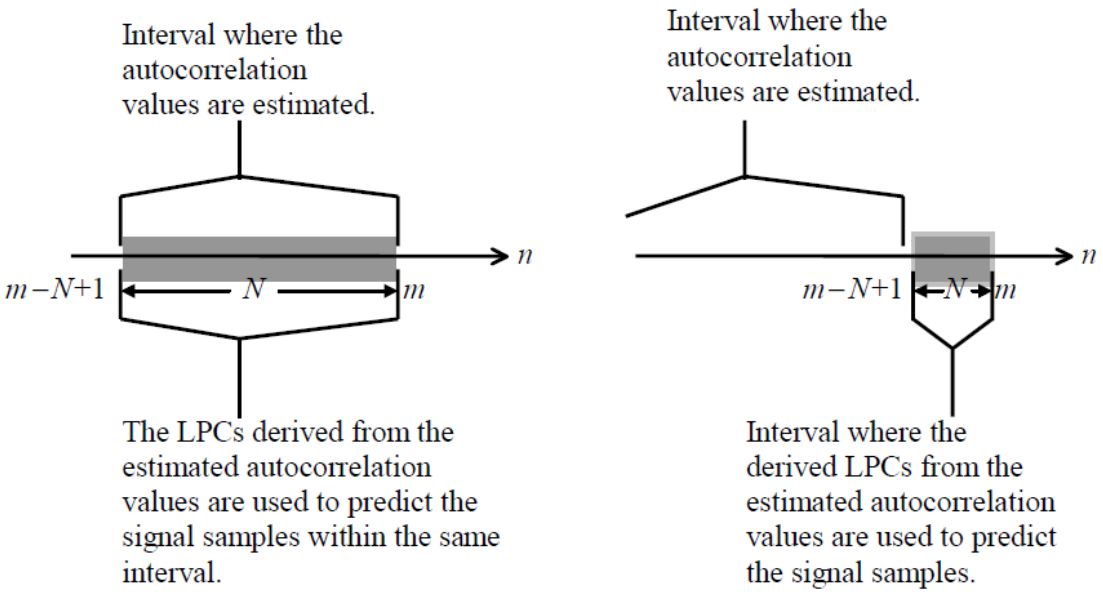

Figure 7.1: Frame wise calculation of LPCs

7.1 PREDICTION SCHEMES:

Two main techniques are applied in speech coding: internal prediction and external prediction. For internal prediction, the LPCs derived from the estimated autocorrelation values using the frame's data are applied to process the frame's data themselves. In external prediction, however, the derived LPCs are used in a future frame; that is, the LPCs associated with the frame are not derived from the data residing within the frame, but from the signal's past. The reason why external prediction can be used is because the signal statistics change slowly with time. If the frame is not excessively long, its properties can be derived from the not so distant past.

Many speech coding algorithms use internal prediction, where the LPCs of a given frame are derived from the data pertaining to the frame. Thus, the resultant LPCs captures the statistics of the frame accurately. Typical length of the frame varies from 160 to 240 samples. A longer frame has the advantage of less computational complexity and lower bit-rate, since calculation and transmission of LPCs are done less frequently. However, a longer coding delay results from the fact that the system has to wait longer for sample collection. Also, due to the changing nature of a non-stationary environment, the LPCs derived from a long frame might not be able to produce good prediction gain. On the other hand, a shorter frame requires more frequent update of the LPCs, resulting in a more accurate representation of the signal statistics. Drawbacks include higher computational load and bit-rate.

The two popular methods of finding reflection coefficients (internal prediction) are described below:

7.2 THE LEVINSON–DURBIN ALGORITHM:

We use this algorithm in order to calculate LPCs from the equation

$$a = R_s^{-1} r_s \tag{7.1}$$

This doesn't involve complex computations for matrix inversion.

$$\begin{bmatrix} R[0] & R[1] & \cdots & \dots & R[M] \\ R[1] & R[0] & & \dots & R[M-1] \\ \vdots & & \ddots & & \vdots \\ \vdots & \vdots & & \vdots & R[1] \\ R[M] & R[M-1] & \cdots & \vdots & R[0] \end{bmatrix} \begin{bmatrix} 1 \\ a_1 \\ \vdots \\ \vdots \\ a_m \end{bmatrix} = \begin{bmatrix} J \\ 0 \\ \vdots \\ 0 \\ 0 \end{bmatrix} \tag{7.2}$$

The Levinson–Durbin algorithm is summarized as follows. Inputs to the algorithm are the autocorrelation coefficients $l = 0$; with the LPCs and RCs.

- Initialization: $l = 0$, set

$$J_0 = R[0]$$

- Recursion: for $l = 1,2,3 \dots . M$

  Step 1: Compute the $l\ th$ RC

$$k_l = \frac{1}{J_l}\left(R[l] + \sum_{i=1}^{l-1} a_i{}^{l-1} R[l-i]\right) \tag{7.3}$$

  Step 2: Calculate LPCs for the $l\ th$ - order predictor:

$$a_l{}^{l} = -k_l$$

$$a_i{}^{l} = a_i{}^{l-1} - k_l a_{l-i}^{l-1} \; ; i = 1,2,3 \dots, l-1 \tag{7.4}$$

  Stop if $l = M$.

Step 3: Compute the minimum mean-squared prediction error associated with the $l\,th$ -order solution

$$J_l = J_{l-1}(1 - k^2) \tag{7.5}$$

Set $l \leftarrow l + 1$; return to Step 1.

The final LPCs are

$$a_i = a_i^{(M)};\ i = 1,2,3 \ldots, M \tag{7.6}$$

Note that in the process of solving the LPCs, the set of RCs ($k_i, i = 1,2 \ldots, M$) is also found. A virtue of the Levinson–Durbin algorithm lies in its computational efficiency. Its use results in a huge saving in the number of operations and storage locations compared to standard methods for matrix inversion. Another benefit of its use is in the set of RCs, which can be used for the verification of the minimum phase property of the resultant prediction-error filter. A system is minimum phase when its poles and zeroes are inside the unit circle.

In our case the system is a minimum phase system as the values of the coefficients lie in range of -1 to +1.

7.3 THE LEROUX–GUEGUEN ALGORITHM:

A potential problem with the Levinson–Durbin algorithm lies in the values of the LPCs, since they possess a large dynamic range and a bound on their magnitudes cannot be found on a theoretical basis. The issue is of little concern if the algorithm is implemented under a floating-point environment. However, it could present some difficulties for fixed-point implementation.

Leroux and Gueguen proposed a method to compute the RCs from the Auto-correlation values without dealing directly with the LPCs. Hence, problems related to dynamic range in a fixed-point environment are eliminated.

Consider:

$$\varepsilon^{l}[k] = E\{e^{l}[n]s[n-k]\} = \sum_{i=0}^{l} a_{l}{}^{(l)} R[i-k] \qquad (7.7)$$

Where

$e^{l}[n] =$ Prediction error using an lth order prediction error filter,

$a_i^{(l)} =$ LPC of an lth order predictor,

$R[k] =$ Auto-correlation value of signal $s[n]$

A fixed-point implementation arises from the fact that the parameters $\varepsilon,$ have a fixed dynamic range.

Pseudo Code:

- Initialization: l = 0, set $\varepsilon^{(0)}[k] = R[k],\ k = -M+1, \ldots, M.$
- Recursion : for $l = 1,2,3 \ldots M$

  Step 1: Compute the lth RC

$$k_l = \frac{\varepsilon^{(l-1)}[l]}{\varepsilon^{(l-1)}[0]}$$

Stop if l = M.

Step 2: Calculate the ε parameters:

$\varepsilon^{(l)}[k] = \varepsilon^{(l-1)}[k] - k_l \varepsilon^{(l-1)}[l-k];\ \ k = -M + l + 1, .,0, l + 1, .M$

Set $l \leftarrow l + 1; return$ to Step 1.

Table 7.1: Reflection Coefficients for 8 KHz Sampling Rate, Low Pitch

Levinson Durbin Algorithm	**Leroux-Guegen Algorithm**
-0.93583286	-0.93583286
-0.27649468	-0.27649468
-0.067862764	-0.067862555
-0.31291151	-0.31291157
0.36889407	0.36889416
-0.25631648	-0.25631621
-0.43616316	-0.4361629
0.2384799	0.23847909
-0.43663645	-0.4366343
-0.22851901	-0.22851695

Table 7.2: Reflection Coefficients for 16 KHz Sampling Rate, Low pitch

Levinson Durbin Algorithm	**Leroux-Guegen Algorithm**
-0.89918661	-0.89918661
0.040188871	0.04018886
0.02616393	0.026164062
0.10910588	0.1091055
-0.1402048	-0.14020436
0.037483968	0.037483659
-0.049390748	-0.049390711
-0.19504443	-0.19504438
-0.34542409	-0.34542418
-0.034607828	-0.034607839

Table 7.3: Reflection Coefficients for 8 KHz Sampling Rate, High Pitch

Levinson Durbin Algorithm	**Leroux-Guegen Algorithm**
-0.89918661	-0.89918661
0.040188871	0.04018886
0.02616393	0.026164062
0.10910588	0.1091055
-0.1402048	-0.14020436
0.037483968	0.037483659
-0.049390748	-0.049390711
-0.19504443	-0.19504438
-0.34542409	-0.34542418
-0.034607828	-0.034607839

7.4 PREDICTION GAIN:

Prediction Gain of a predictor is the ratio between the variance of the input signal and the variance of the prediction error in decibels (dB).

$$PG[m] = 10\log_{10}\left(\frac{\sum_{n=m-N+1}^{m} s^2[n]}{\sum_{n=m-N+1}^{m} e^2[n]}\right) \quad (7.8)$$

Where,

$$e[n] = s[n] - \hat{s}[n] = s[n] + \sum_{i=1}^{M} a_i[m]s[n-i] \; ; n = m - N + 1, \ldots, m \quad (7.9)$$

The LPCs are found from the samples inside the interval [m – N + 1, m] for internal prediction, and n < m – N+ 1 for external prediction. Note that the prediction gain is a function of the time variable m.

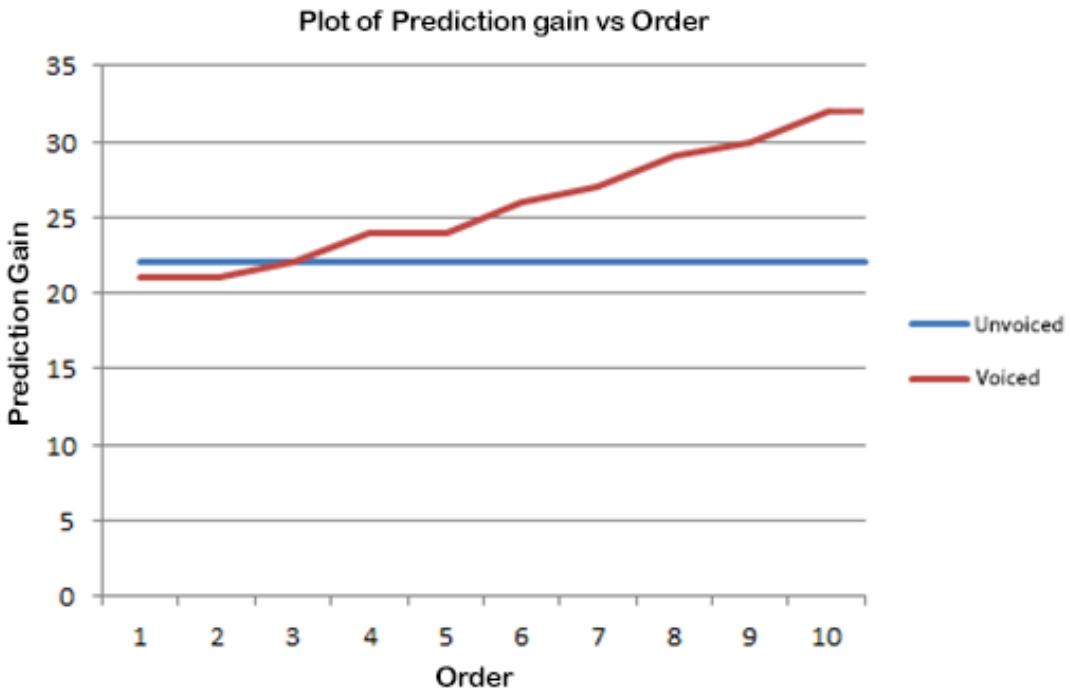

Figure 7.2 Plot of Prediction Gain vs Order for Voiced and Unvoiced signal

Prediction gain is a measure of the predictor's performance. A better predictor is capable of generating lower prediction error, leading to a higher gain. Order is an important parameter to recreate the exact copy of the original signal. Prediction gain helps us to find the optimal order to calculate the reflection coefficients. As we can observe from the plot, the prediction gain for voiced signal saturates after certain order which is optimal. For a given prediction order, the average prediction gain obtainable for voiced frames is higher than for unvoiced frames. Prediction gain is the highest for short frames, which is expected, since as the frame's length increases, the statistics derived from the signal's past become less and less accurate for representing the frame itself; therefore prediction gain drops.

CHAPTER 8

CONCLUSION AND FUTURE SCOPE OF WORK

A real time implementation of speech coding was successfully developed on Blackfin 526 DSP. Pitch estimation and reflection coefficients were successfully computed and will be used for linear predictive coding model.

Linear Predictive Coding is an analysis/synthesis technique to lossy speech compression that attempts to model the human production of sound instead of transmitting an estimate of the sound wave. The tradeoff for LPC's low bit rate is that it does have some difficulty with certain sounds and it produces speech that sound synthetic.

Linear predictive coding encoders break up a sound signal into different segments and then send information on each segment to the decoder. The encoder send information on whether the segment is voiced or unvoiced and the pitch period for voiced segment which is used to create an excitement signal in the decoder. The encoder also sends information about the vocal tract which is used to build a filter on the decoder side which when given the excitement signal as input can reproduce the original speech. Hence the reflection coefficients form a base for future speech codec model.

REFERENCES

1. **Wai C. Chu**. Speech Coding Algorithms: Foundation and Evolution of Standardized Coders.

2. **Avi Peretz** and **Cagdas Gumusoglu,** Wideband Speech Codec Implementation on Bf-533 DSP.

3. **Marwan Al-Akaidi,** Introduction to speech processing: Cambridge University Press 0521814588 - Fractal Speech Processing.

4. **Nimrod Peleg Update March (2009),** Linear Prediction Coding.

5. **Jan Cernocky, Valentina Hubeika, FIT BUT Brno,** Fundamental Frequency, Encoding and Decoding of Speech.

6. **L. R. Rabiner, M. J. Cheng, A. E. Rosenberg and C. A. McGonegal,** "A Comparative Performance Study of Several Pitch Detection Algorithms," IEEE Trans on Acoustics, Speech, and Signal Processing, vol. 24, no. 5, pp. 399418, 1976.

7. **J. A. Moorer**, "The Optimum Comb Method of Pitch Period Analysis of Continuous Digitized Speech," IEEE Trans. on Acoustics, Speech, and Signal Processing, vol. 22, no. 5, pp. 330-338, 1974.

8. **Randy Goldberg** and **Lance Riek**. A Practical Handbook of Speech Coders (1999) Chapter 2:1-28, Chapter 4: 1-14, Chapter 9: 1-9, Chapter 10:1-18.