NUMERICAL IMPLEMENTATION OF THE HILBERT TRANSFORM

A Thesis

Submitted to the College of Graduate Studies and Research

in Partial Fulfillment of the Requirements

For the Degree of Master of Science

in the Department of Electrical Engineering

University of Saskatchewan

Saskatoon, Saskatchewan

by

Xiangling Wang

PERMISSION TO USE

I agree that the Library, University of Saskatchewan, may make this thesis freely available for inspection. I further agree that permission for copying of this thesis for scholarly purpose may be granted to the professor or professors who supervised the thesis work recorded herein or, in their absence, by the Head of the Department or the Dean of the College in which the thesis work was done. It is understood that due recognition will be given to me and to the University of Saskatchewan in any use of the material in this thesis. Copying or publication or any other use of this thesis for financial gain without approval by the University of Saskatchewan and my written permission is prohibited.

Request for permission to copy or to make any other use of the material in this thesis in whole or part should be addressed to:

Head of the Department of Electrical Engineering
57 Campus Drive
University of Saskatchewan
Saskatoon, Saskatchewan
Canada S7N 5A9

ACKNOWLEDGEMENTS

I wish to express my gratitude to the following people who not only made this thesis possible but also an enjoyable experience:

Dr. Ronald J. Bolton: my supervisor, for his valuable guidance, criticisms and consistent encouragement throughout the course of this research work.

My husband, Zhanghai Wang: for his love and encouragement.

My parents, my sister Xiangrong Wang and brother-in-law Xianggang Yu: for the support they provided to me.

All my friends in Saskatoon, Sha Li, Song Hu, Jing Yin, Ying Cui, Quan Wan, Yajun Wang, Xin Xu and Yanan Xing: for making me feel welcome. They will always be special friends in my life.

The Department of Electrical Engineering: for supplying the opportunity to study in Canada and the necessary facilities with which to work.

ABSTRACT

Many people have abnormal heartbeats from time to time. A Holter monitor is a device used to record the electrical impulses of the heart when people do ordinary activities. Holter monitoring systems that can record heart rate and rhythm when you feel chest pain or symptoms of an irregular heartbeat (called an arrhythmia) and automatically perform electrocardiogram (ECG) signal analysis are desirable.

The use of the Hilbert transform (HT) in the area of electrocardiogram analysis is investigated. A property of the Hilbert transform, i.e., to form the analytic signal, was used in this thesis. Subsequently pattern recognition can be used to analyse the ECG data and lossless compression techniques can be used to reduce the ECG data for storage.

The thesis discusses one part of the Holter Monitoring System, Input processing.

Four different approaches, including the Time-Domain approach, the Frequency-Domain approach, the Boche approach and the Remez filter approach for calculating the Hilbert transform of an ECG wave are discussed in this thesis. By comparing them from the running time and the ease of software and hardware implementations, an efficient approach (the Remez approach) for use in calculating the Hilbert transform to build a Holter Monitoring System is proposed.

Using the Parks-McClellan algorithm, the Remez approach was present, and a digital filter was developed to filter the data sequence.

Accurate determination of the QRS complex, in particular, accurate detection of the R wave peak, is important in ECG analysis and is another task in this thesis. A program was developed to detect the R wave peak in an ECG wave.

The whole algorithm is implemented using Altera's Nios SOPC (system on a program chip) Builder system development tool. The performance of the algorithm was tested using the standard ECG waveform records from the MIT-BIH Arrhythmia database. The results will be used in pattern recognition to judge whether the ECG wave is normal or abnormal.

Table of Contents

List of Tables

List of Figures

List of Abbreviations

APBs	atrial premature beats
BPM	beats per minute
DHCP	dynamic host configuration protocol
ECG	electrocardiogram
FIR	finite impulse response
FT	Fourier transform
FFT	fast Fourier transform
FPGA	field programmable gate array
HW	hardware
HT	Hilbert transform
IFFT	inverse fast Fourier transform
ISA	instruction set architecture
LED	light emitting diode
MIT-BIH	Massachusetts Institute of Technology- Beth Israel Hospital
PCG	polarcardiogram
PVC	premature ventricular contraction
PHY/MAC	physical layer/media access control
RISC	reduced instruction set computer
RMS	root mean square
SDK	software development kit
SOPC	system on a program chip

SW	software
VCG	vectorcardiogram
VPB	ventricular premature beat

Chapter 1 Introduction

Since the development of medical science, many instruments for improving people's health have been developed. The electrocardiogram (ECG) monitoring system is one of them. The most common type of ECG monitoring is called Holter monitoring. Holt monitoring is a portable recording tool and can help doctors make a precise diagnosis.

1.1 Research Motivation

Many people have irregular heartbeats from time to time. Some heart problems occur during certain activities, such as eating, exercise or even sleeping. Sometimes the irregular heartbeats don't influence life style and are usually harmless in normal hearts. But it is also possible that these irregular heartbeats with pre-existing illness can cause heart attacks that lead to death. A device that can record the activities of the heart is very useful in preventing heart attacks. The Holter monitoring system was developed for this objective in an ambulatory situation.

A Holter monitoring system is a small recording instrument that is used to capture ECG data of the heart's electrical activities over a period of time. The patient can carry it in a pocket or in a small pouch. The monitor is battery operated. The electrocardiogram is saved in a memory card. The electrocardiographer can analyse the recordings visually by means of a computer.

1.1.1 Electrocadiogram

The standard ECG is a representation of the heart's electrical activities recorded from electrodes that are placed on different parts of patient's body.

The electrocardiogram is composed of complexes and waves. In normal sinus rhythm, waves and complexes are the P wave, QRS complex, ST Segment, T wave and U wave. Measurements are PR interval, QRS duration, QT interval, RR interval and PP interval. Figure 1.1 illustrates a typical waveform of normal heartbeats and intervals as well as standard time and voltage measures.

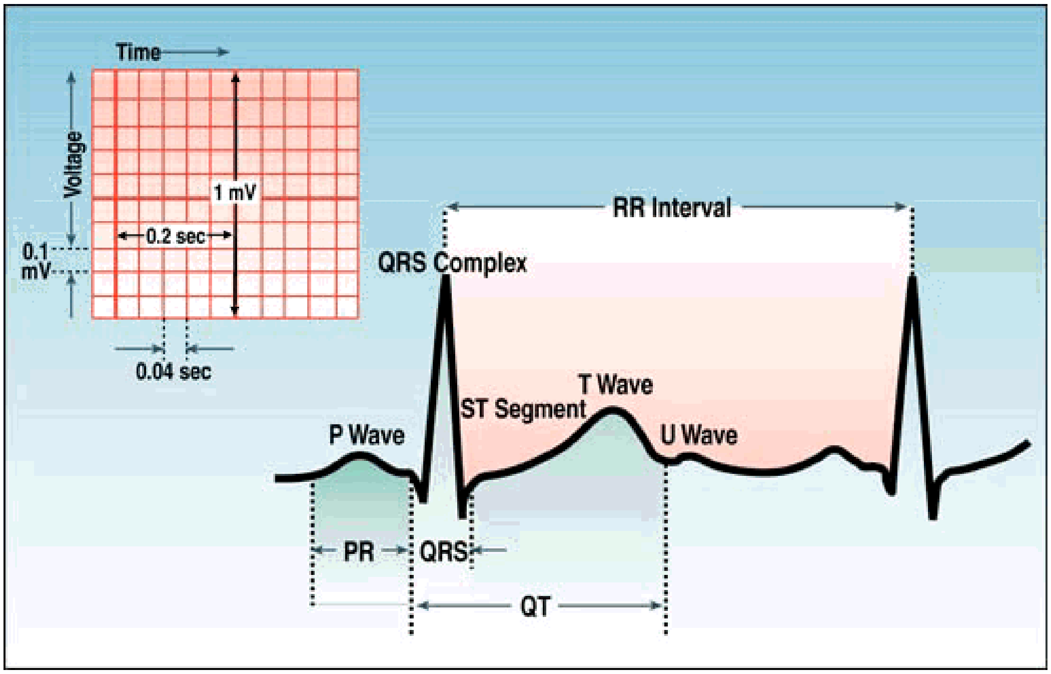


Figure 1.1 ECG signal [1]

Different parts of the ECG waves are caused differently. Detailed information will be given below to explain each part of the ECG waveform.

- The **P wave** is due to the electrical activation (depolarization) of the heart (atria). It is usually positive, low amplitude and smooth. In normal situation, the time of the P wave should be smaller than 0.12 seconds.

- The **QRS complex** represents right and left ventricular depolarization. It is high amplitude in normal situations. The shape of the QRS complex will be changed if the electrodes are placed on different parts of the body. It also changes when abnormal heartbeats occur. A QRS complex can have positive (upwards) or negative (downwards) deflections. The figure below summarizes the nomenclature used to define the different components of the QRS complex.

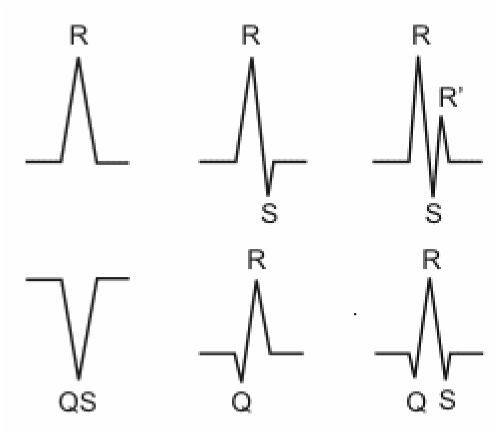


Figure 1.2 The different components of the QRS complex [2].

- The **ST segment** represents the time following the QRS it takes for depolarization of the ventricles before repolarization. Repolarization of the atria is a low amplitude signal that occurs during the time of the high amplitude QRS and consequently it can’t be seen on a standard ECG.

- The **T wave** is caused by the repolarization of the ventricles. Usually it is positive and rounded.

- The reason that causes the **U wave** is not that clear, “afterdepolarizations” in the ventricles maybe is the answer.

- The **PR interval** is the time interval from the beginning of the P wave to the beginning of the QRS complex. In normal situation the PR interval should be 0.12-0.2 s, Short PR < 0.12 s, Prolonged PR >0.2 s.

- The **QRS duration** is the time of ventricular depolarization. Normal: 0.06 s-0.1 s, Prolonged QRS duration: >0.1s.

- The **QT interval** represents the duration of ventricular depolarization and repolarization. It is between the onset of the QRS complex and the end of the T wave. It normally depends on heart rate.

- The **RR interval** is the duration of ventricular cardiac cycle. The value of the RR interval indicates the ventricular rate.

- The **PP interval** is the duration of atrial cycle. It indicates the atrial rate.

The normal adult heart beats regularly between 60 to 100 beats per minute. Bradycardia occurs once the heart rate is slower than 60 beats per minute. The waveform is similar to the normal ECG wave, but the RR interval is longer. A rate of above 100 beats per minute is called tachycardia, in this case the RR interval is shorter and the waveform is also similar to the regular sinus rhythm. Each P wave is following by a QRS complex. A waveform of a regular ECG wave is shown in

Figure 1.3. The wave of the bradycardia is shown in Figure 1.4. Figure 1.5 illustrates the tachycardia. Figure 1.7 shows some abnormal ECG waves.

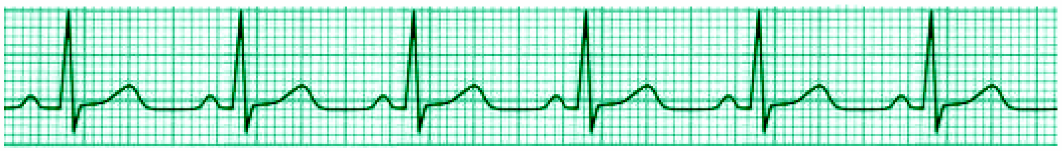


Figure 1.3 A normal ECG wave [3].

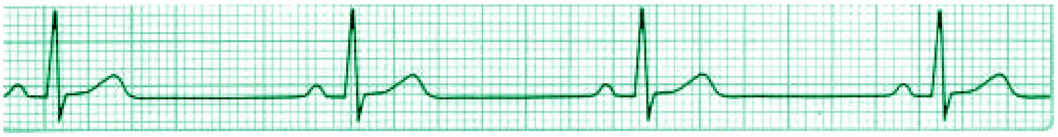


Figure 1.4 Bradycardia [3].

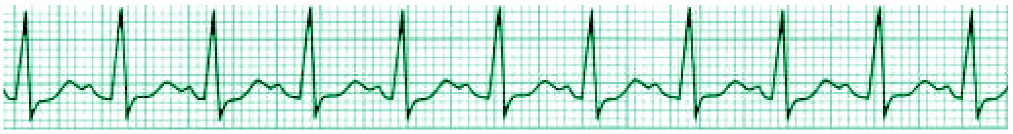


Figure 1.5 Tachycardia [3].

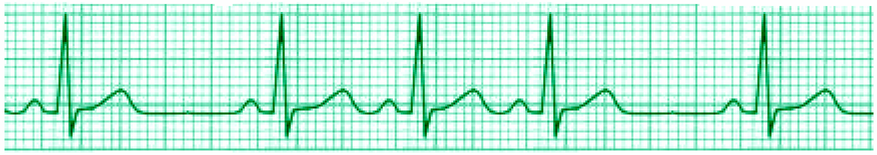


Figure 1.6 An irregular heartbeat wave [3].

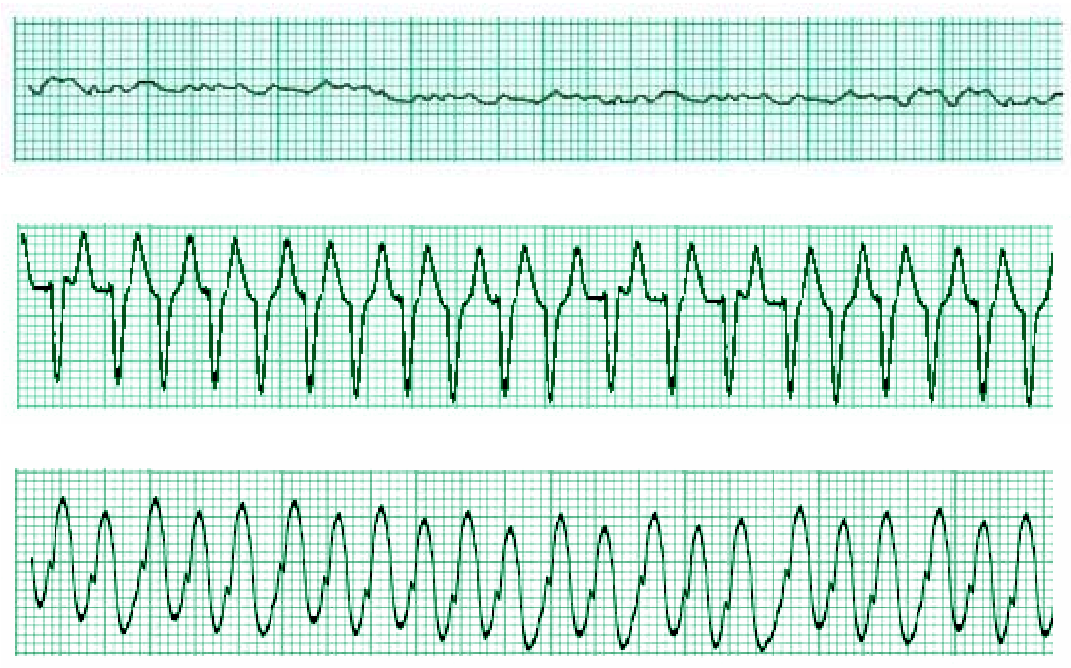


Figure 1.7 Three abnormal ECG waveforms [3].

The importance of irregular heartbeats depends on the type of pattern they produce, how often they occur, how long they last, and whether they occur at the same time the patient had symptoms.

1.1.2 Advantanges of Holter Monitoring System

As discussed previously, the Holter monitoring system records the electrical activity of heart during usual daily activities. A recording is much more likely to detect any abnormal heartbeats that occur during these activities.

During the late 1960s, computerized ECG's came into use in many of the larger hospitals. General Holter monitoring system records continuous electrocardiographic measurements of the heart’s rhythm. Usually the recording time is around 24 to 48 hours. That means even when the heartbeat is normal, the Holter monitor also works as well.

The system discussed in this thesis automatically records the ECG wave when the user is not feeling good or the heartbeats are not regular. The recording algorithm is not continuous any more. It also can record the heartbeats manually; the wearers can record the heartbeat if wanted when the heart rhythms are ordinary. The differences between the general Holter monitor and the system developed in this thesis are shown in the Table 1.1.

Table 1.1 Comparison general Holter Monitor with new system in this thesis

	General Holter Monitor	**New system in this thesis**
Continuity	Continuous	Intermittent
Saving time	24-48 hours	More than 48 hours
Operation	Manually operated	Automatic/Manually operated

The advantages in the system discussed in this thesis are:

- Record the ECG wave automatically when wearer does not feel good.
- Save memory space and extend the recording time.
- Record activities manually when the wearer wants.
- Normally record for a few hours or for a few seconds.

1.1.3 Structure of the Holter Monitoring System

The structure of the Holter monitoring system discussed in this thesis is shown in Figure 1.8.

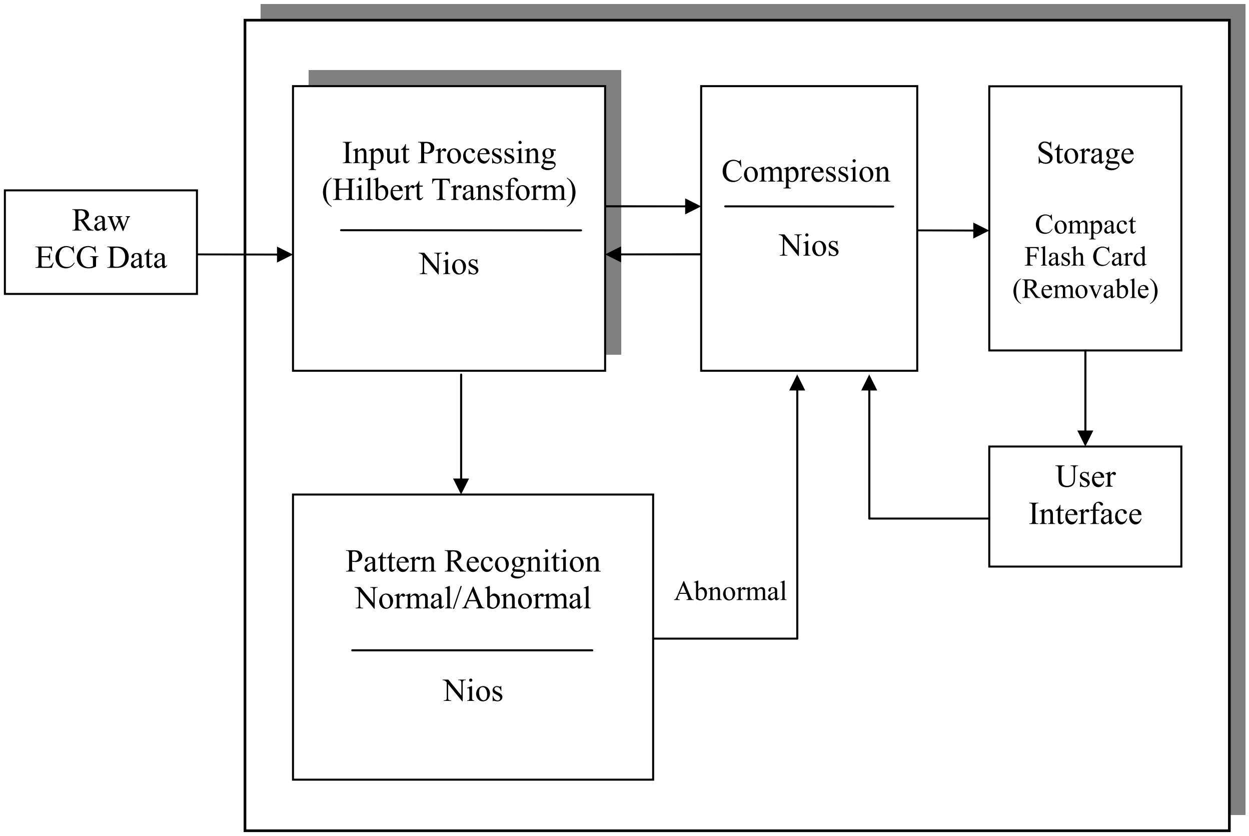


Figure 1.8 The structure of the Holter Monitoring System.

From Figure 1.8 it can be seen that the system includes four sub-systems: the Input processing sub-system, the Pattern Recognition sub-system, the Compression sub-system and the Storage sub-system.

The input data is the raw ECG data. These data record the activities of the heart. Every heartbeat is caused by a section of the heart generating an electrical signal that then conducts through specialized pathways to all parts of the heart.

These electrical signals also get transmitted through the chest to the skin where they can be recorded [4].

1.1.3.1 Input Processing

The main objective of this sub-system is to implement the Hilbert transform of the input ECG data on the Nios embedded processor. Then the zero crossing points corresponding to the input R wave peaks are found and the results information is sent to the Pattern Recognition sub-system.

1.1.3.2 Pattern Recognition

Obtain the results of the input processing, and by using properties of the Hilbert transform, create the analytic signal and use it to assist in doing pattern recognition to determine if the ECG wave is normal or abnormal.

1.1.3.3 Compression

Compress all the abnormal data obtained from the “Input processing” sub-system.

1.1.3.4 Storage

Save all the compressed data in compact flash card. This card is removable and can be inserted into the computer to read the information recorded in this card. Electrocardiographer can analyze the data or by means of some software to draw the waveform to see what happened visually.

Holter monitoring gives doctors the record of patient's heart rate and rhythm over a period of time. The Holter monitor can record heart rate and rhythm when the patient feels chest pain or symptoms of an irregular heartbeat (an arrhythmia). The doctor can then look at the time when the patient noticed their symptoms. Reading this printout will give the doctor an idea about the nature of the heart problem. A Holter monitor provides the physician a better opportunity to capture any abnormal heartbeats or rhythms that may be causing the patient's symptoms. It's necessary to develop an efficient (i.e., long time durations and automatic pattern recognition analysis) system to record these irregular heartbeats so that the doctor can know what had happened when patient had those symptoms. This system is a very good tool to prevent people from fatal heart problems.

Briefly, the reasons for using a Holter monitor may include: to detect problems missed in a regular ECG; to check activity after an arrhythmia; to see if a new pacemaker works and to see if drug therapy is working.

The work of this thesis is to implement the input processing sub-system. That is to develop an efficient algorithm to compute the Hilbert transform of the input ECG waves and determine R-R intervals.

1.2 Objectives

The objectives of this thesis are to address the aforementioned problems and to propose solutions to the following problems.

- To use different methods to compute the Hilbert transform of an input signal.

- To compare these algorithms with each other from running time and hardware/software implementation.
- To develop an efficient algorithm for calculating the Hilbert transform to build a Holter ECG Monitoring System.
- To develop a detector to find all of the zero crossing points that correspond to the R wave peaks in the output wave, i.e., the Hilbert transform sequence of an ECG wave.

1.3 Outline of Thesis

Chapter 2 gives the background materials of the Holter monitor and the Hilbert transform including the past history, notation and definition.

Chapter 3 is devoted to four different methods used to compute the Hilbert transform of an input signal. The four approaches include the Time-Domain approach, the Frequency-Domain approach, the Boche approach and the Remez filter approach. In this chapter, the examples and the results are given for using different approaches to calculate the Hilbert transform. By comparing four methods with each other in running time and the ease of software and hardware implementations, an efficient algorithm for the Hilbert transform to build a Holter ECG Monitoring System will be present.

Chapter 4 describes how to design and implement the Remez filter approach that was mentioned in Chapter 3, and also describes how to design and implement a filter with an optimal fit. The information about Alter’s Nios SOPC Builder system development tool is introduced as well.

Chapter 5 concentrates on analyzing the results and compares them to the MIT-BIH Arrhythmia database to make sure the results are correct.

The thesis is summarized in Chapter 6. At the end of the thesis, an appendix is given.

Chapter 2 Background

In this chapter, a brief review of Holter Monitoring is given. The definition, the properties and the applications of the Hilbert transform are also contained in this chapter.

2.1 Holter Monitoring Review

As discussed in the previous chapter, an electrocardiogram (ECG) is a record produced by an electrocardiography, which indicates the electrical voltage in the heart. An ECG provides information on the condition and performance of the heart. It is one of the simplest and fastest procedures used to evaluate the heart. Because an arrhythmia can occur irregularly, it will be difficult to record when the patient is in the doctor's office.

In 1949, American physician Norman Jeff Holter (1914-1983) developed a 75 pound backpack that can record the ECG of the wearer. This portable monitoring device is called the Holter monitor, named after its inventor.

The Holter monitor is battery-powered and can continuously record the electrical activities of the heart over a specified period of time, normally 24 to 48 hours. Usually the patient will undergo Holter monitoring as an outpatient, meaning that the monitor will be placed on the body of the patient by a technician in a cardiologist’s office. Then the patient will go home and do normal activities. With the development of technology, the Holter monitor is greatly reduced in size. It is now very compact and combined with digital recording and used to record ECGs. The Holter monitor can be easily carried without interfering with the patient’s activities. At the end of the recording period, the patient will go back to the doctor’s office to remove the Holter monitor. The data saved in the Holter monitor will be analyzed by an electrocardiographer and a computer. The analysis results will provide the information about the patient’s heart rhythm, the frequency of the beats and the irregularities. This portable monitor can be an effective and powerful diagnostic tool that can directly determine how the physician treats the patient’s condition.

As Figure 2.1 shows, the Holter monitor is a small-size recording device. The monitor has wires called leads. The leads attach to metal disks called electrodes, which the user wears on his chest. These electrodes are very sensitive, and they can pick up the electrical impulses of the heart. The impulses are recorded by the Holter monitor record the heart’s electrical activity.

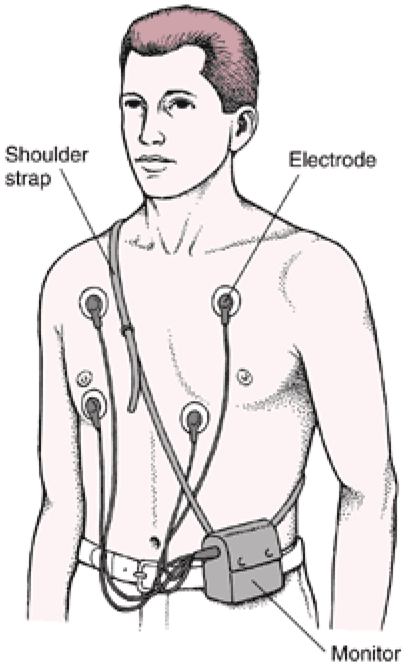


Figure 2.1 A man with the Holter monitor [5].

If necessary, the ECG can be transmitted by telephone to a computer at the hospital or doctor's office for an immediate reading as soon as symptoms occur. The use of the effective home care monitor of the heart patients will decrease the incidence of the readmissions and lower the costs of the hearth care.

Advanced Holter monitors have been developed that employ digital electrocardiographic recordings, extended memory for more than 24 hours recording, pacemaker pulse detection and analysis, software for analysis of digital ECG recordings that are downloaded and stored on a computer, and capability of transmission of results over the internet [6].

The system discussed in this thesis is to record the rhythm of the heartbeat automatically when the symptoms occur. The monitor does not continuously record but records the heart rate and rhythm when the patients feel symptoms of an irregular heartbeat or when abnormal heart beats or rhythms occur. That means the device automatically capture the arrhythmias when they occur. It also can be activated manually by the patients when chest pain is felt during a symptomatic event. So it can record for a long time. Up to now, the Holter monitor in the market can record over 24-48 hours; the longest one is not much more than 72 hours. The system discussed in this thesis will much improve the recording time. The advantages of the system have been discussed in Chapter 1.

2.2 Hilbert Transform Review

In 1893, the physicist Arthur E. Kennelly (1861-1939) and the scientist Charles P. Steinmetz (1865-1923) first used the Euler formula

$$e^{jz} = \cos(z) + j\sin(z) \tag{2.1}$$

which was derived by a famous Swiss mathematician Lenonard Euler (1707-1783) to introduce the complex notation of harmonic wave forms in electrical engineering, that is:

$$e^{jwt} = \cos(\omega t) + j\sin(\omega t), \tag{2.2}$$

where j is the imaginary unit.

In the beginning of the 20th century, the German scientist David Hilbert (1862-1943) proved that the Hilbert transform of the function $\cos(\omega t)$ is $\sin(\omega t)$. This is the one of properties of the Hilbert transform, i.e., basic $\pm\frac{\pi}{2}$ phase-shift.

2.2.1 Definition

In mathematics and in signal processing, the Hilbert transform $\hat{x}(t)$ of a real time function $x(t)$ is defined as:

$$\hat{x}(t) = H[x(t)] = \frac{1}{\pi}\int_{-\infty}^{\infty}\frac{x(\tau)}{t-\tau}d\tau \tag{2.3}$$

when the integral exists.

It can be seen from the Equation (2.3) that the independent variable is not changed as result of this transformation, so the output $\hat{x}(t)$ is also a time dependent function.

It is normally not possible to calculate the Hilbert transform as an ordinary improper integral because of the possible singularity at $\tau = t$. The integral is to be considered as a Cauchy principal value.

In mathematics, the Cauchy principal value of certain is defined as

$$\lim_{\varepsilon\to 0+}\left[\int_{a}^{b-\varepsilon}x(t)dt + \int_{b+\varepsilon}^{c}x(t)dt\right], \tag{2.4}$$

where b is a point at which the behaviour of the function $x(t)$ is such that

$$\int_{a}^{b}x(t)dt = \pm\infty \quad \text{for any } a < b$$

and $$\int_{b}^{c}x(t)dt = \mp\infty \quad \text{for any } c > b\text{. [7]}$$

So when the Hilbert transform exists, it is written as presented at Equation (2.3). Other forms for $H(x(t))$ can be obtained by change of variable, that is

$$H[x(t)] = \frac{1}{\pi}\int_{-\infty}^{\infty}\frac{x(t-\tau)}{\tau}d\tau \tag{2.5}$$

$$H[x(t)] = \frac{1}{\pi}\int_{-\infty}^{\infty}\frac{x(t+\tau)}{\tau}d\tau \ . \tag{2.6}$$

A Hilbert transform of a square wave is shown below:

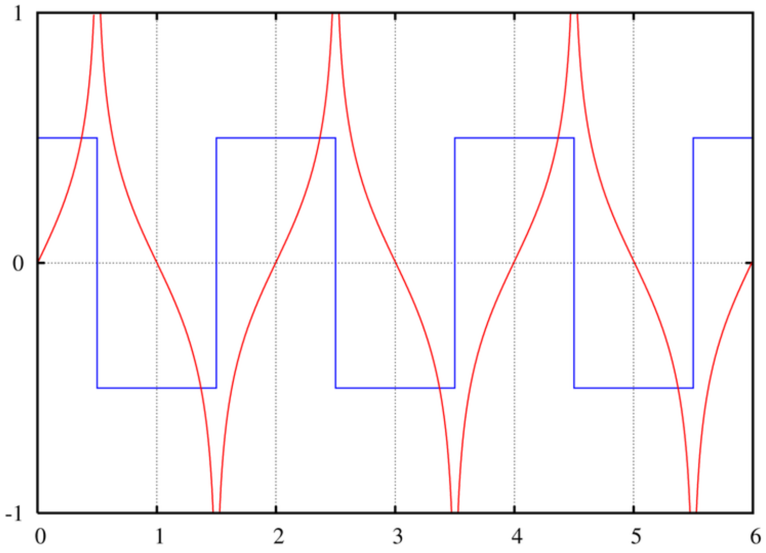


Figure 2.2 The Hilbert transform of a square wave.

2.2.2 Frequency Response of the Hilbert Transform

From the Equation (2.3), (2.4) and (2.5), it can be seen that Hilbert transform is a convolution:

$$H[x(t)] = \hat{x}(t) = \frac{1}{\pi t} * x(t) \tag{2.7}$$

Equation (2.7) shows that $\hat{x}(t)$ is a linear function of $x(t)$. It is obtained from $x(t)$ applying convolution with $1/(\pi t)$.

According to the convolution theorem (the Fourier transform of a convolution of two functions is the product of their Fourier Transforms.), it can be seen that the spectrum of $H[x(t)]$ is related to that of $x(t)$.

Applied the Fourier transform to the Equation (2.7), that is

$$F\{\hat{x}(t)\} = \frac{1}{\pi} F\left\{\frac{1}{t}\right\} F\{x(t)\} \tag{2.8}$$

where F is the Fourier transform.

Since

$$F\left\{\frac{1}{t}\right\} = \int_{-\infty}^{\infty} \frac{1}{x} e^{-j2\pi^{*} fx} dx = -j\pi \operatorname{sgn}(f) \tag{2.9}$$

where

$\operatorname{sgn} f$ is +1 for $f > 0$, 0 for $f = 0$ and -1 for $f < 0$.

Rewriting Equation (2.9), that is:

$$F\left(\frac{1}{\pi t}\right) = -j\operatorname{sgn}(f) = \begin{cases} -j & for\ f > 0 \\ +j & for\ f < 0 \end{cases}.$$

Therefore, the Fourier transform of $\frac{1}{\pi t}$ is $-j\operatorname{sgn}(f)$, which is equal to $-j$ for positive frequency and $+j$ for negative frequency. Hence the Hilbert transform is equivalent to a kind of filter, in which the amplitudes of the spectral components are left unchanged, but their phases are altered by $\frac{\pi}{2}$, positively or negatively according to the sign of frequency [8].

Therefore, the Fourier transform of the Hilbert transform of $f(t)$ is given by

$$F\{(\hat{x})\} = -j\operatorname{sgn} fF\{x(t)\}\,. \tag{2.10}$$

The time domain result can be obtained performing an inverse Fourier transform.

2.3 Hilbert Transform Properties

In this part, some properties of the Hilbert transform will be discussed.

(1) The Hilbert transform of a real function is linear.

As discussed in the section 1.2, the Hilbert transform of a function $f(t)$ is defined as

$$H[f(t)] = \frac{1}{\pi}\int_{-\infty}^{\infty} \frac{f(\tau)}{t-\tau} d\tau \, .$$

Because of the possible singularity at $t = \tau$, the integral is to be considered as a Cauchy principal value. It is expressed on the form

$$H[f(t)] = \lim_{\varepsilon \to 0} \frac{1}{\pi}\int_{|x-t|>\varepsilon} \frac{f(\tau)}{t-\tau} d\tau. \tag{2.11}$$

Suppose $f(t) = c_1 f_1(t) + c_2 f_2(t)$, then the Hilbert transform of $f(t)$ should be

$$\begin{aligned} H[f(t)] &= H\left[c_1 f_1(t) + c_2 f_2(t)\right] \\ &= \lim_{\varepsilon \to 0} \frac{1}{\pi}\int_{|x-t|>\varepsilon} \frac{c_1 f_1(\tau) + c_2 f_2(\tau)}{t-\tau} d\tau \\ &= c_1 \lim_{\varepsilon \to 0} \frac{1}{\pi}\int_{|x-t|>\varepsilon} \frac{f_1(\tau)}{t-\tau} d\tau + c_2 \lim_{\varepsilon \to 0} \frac{1}{\pi}\int_{|x-t|>\varepsilon} \frac{f_2(\tau)}{t-\tau} d\tau \\ &= c_1 H[f_1(t)] + c_2 H[f_2(t)]. \end{aligned} \tag{2.12}$$

Equation (2.12) shows the linearity of the Hilbert transform.

(2) The Hilbert transform of a Hilbert transform is the negative of the original function.

$$H[\hat{f}(t)] = \hat{\hat{f}}(t) = F^{-1}\left[-j\,\mathrm{sgn}(\omega)F[\hat{f}(t)]\right]$$

$$= F^{-1}\left[-j\,\mathrm{sgn}(\omega)[-j\,\mathrm{sgn}(\omega)F(j\omega)]\right]$$

$$= F^{-1}\left[-F(j\omega)\right]$$

$$= -f(t) \qquad (2.13)$$

(3) The Hilbert transform of the derivative of a function is equivalent to the derivative of the Hilbert transform of a function. [9]

$$H[f(t)] = \frac{1}{\pi}\int_{-\infty}^{\infty}\frac{f(\tau)}{t-\tau}d\tau = \frac{1}{\pi}\int_{-\infty}^{\infty}\frac{f(t-s)}{s}ds$$

So $$\frac{d}{dt}\{H[f(t)]\} = \frac{d}{dt}\left\{\frac{1}{\pi}\int_{-\infty}^{\infty}\frac{f(t-s)}{s}ds\right\}$$

$$= \frac{1}{\pi}\int_{-\infty}^{\infty}\frac{f'(t-s)}{s}ds$$

$$= \frac{1}{\pi}\int_{-\infty}^{\infty}\frac{f'(\tau)}{t-\tau}d\tau$$

$$= H[f'(t)]\,. \qquad (2.14)$$

(4) A function and its Hilbert transform are orthogonal

$$\int_{-\infty}^{\infty} f(t)\hat{f}(t)dt = \int_{-\infty}^{\infty} f(t)\left[\frac{1}{2\pi}\int_{-\infty}^{\infty} -j\,\mathrm{sgn}(\omega)F(j\omega)e^{j\omega t}d\omega\right]dt$$

$$= \frac{1}{2\pi}\int_{-\infty}^{\infty} - j\,\mathrm{sgn}(\omega)F(j\omega)\left[\int_{-\infty}^{\infty} f(t)e^{j\omega t}dt\right]d\omega$$

$$= \frac{-j}{2\pi}\int_{-\infty}^{\infty} \mathrm{sgn}(\omega)\left|F(j\omega)\right|^{2} d\omega$$

$$=0. \qquad (2.15)$$

Since integrand $\mathrm{sgn}(\omega)\left|F(j\omega)\right|^{2}$ is an odd function which is integrated over symmetric limits, the result is 0.

Equation (2.15) proves that a real function and its Hilbert transform are orthogonal.

This property can be used in energy and power signals.

(5) The energy in a real function and its Hilbert transform are equal.

The signal and its Hilbert transform have identical energy because a phase shift does not change the energy of the signal only amplitude changes can do that.

The energy of $f(t)$ and $F(\omega)$ is defined as

$$E_f = \int_{-\infty}^{\infty}\left|f(t)\right|^{2} dt = \frac{1}{2\pi}\int_{-\infty}^{\infty}\left|F(\omega)\right|^{2} d\omega\,. \qquad (2.16)$$

So the energy of the Hilbert transform of $f(t)$ can be computed as

$$E_{\hat{f}} = \int_{-\infty}^{\infty}\left|\hat{f}(t)\right|^{2} dt$$

$$= \frac{1}{2\pi}\int_{-\infty}^{\infty}\left|- j\,\mathrm{sgn}(\omega)F(\omega)\right|^{2} d\omega$$

$$= \frac{1}{2\pi}\int_{-\infty}^{\infty}\left|F(\omega)\right|^{2} d\omega\,. \qquad (2.17)$$

From Equation (2.16) and Equation (2.17), it shows that $E_{\hat{f}} = E_f$.

2.4 Hilbert Transform Applications

The Hilbert transform is a very useful tool for the analysis of problems in various research areas. The Hilbert transform has a variety of applications, such as in the field of radio and signal processing, communication and power area.

2.4.1 Analytic Signal

In digital signal processing, it is often needed to look at the relationships between the real part and imaginary part of a complex signal. The relationships are usually described by Hilbert transforms. Hilbert transform is also used to create special signals called analytic signals which are especially important in simulation.

An analytic signal is a complex function created by taking a signal and then adding in quadrature its Hilbert transform [10]. An analytic signal is defined as

$$z(t) = f(t) + j\hat{f}(t) = A(t)e^{j\theta(t)}, \tag{2.18}$$

where

$f(t)$ is the input signal.

$\hat{f}(t)$ is the Hilbert transform of the input signal.

$z(t)$ is the analytic signal constructed from $f(t)$ and its Hilbert transform. It is called the pre-envelope of $f(t)$.

The real and imaginary parts can be expressed in polar coordinates as:

$$z(t) = A(t)e^{j\theta(t)}. \tag{2.19}$$

where

$A(t)$ is the “envelope” or amplitude of the analytic signal given as

$$A(t) = \sqrt{f^2(t) + \hat{f}^2(t)}\,. \tag{2.20}$$

$\theta(t)$ is the phase of the analytic signal given as

$$\theta(t) = \arctan\left(\frac{\hat{f}(t)}{f(t)}\right). \tag{2.21}$$

$A(t)$ and $f(t)$ have common tangents and the same values at the points where $\hat{f}(t) = 0$, i.e., the envelop determined using Equation (2.20) will have the same slope and magnitude of the original signal $f(t)$ at its maxima.

2.4.2 Analytic Signal Applied in Pattern Recognition [11]

In this thesis, given a real function $f(t)$, such as an ECG wave, it is possible to compute the Hilbert transform, $\hat{f}(t)$. This allows the calculation of the envelope of $f(t)$ and also the phase of the pre-envelope of $f(t)$ and $\hat{f}(t)$. If the two functions are then plotted in polar form (polar plot), the result is a waveform display very similar to a Vectorcardiogram (VCG) or a Polar-cardiogram (PCG).

Thus the resulting magnitude versus angle plot is used for further analysis. The temporal dependence of the ECG data is removed. In effect the data has been shifted from a magnitude versus time system into a magnitude versus angle system. At the same time that because using a sampled-data waveform sampled at a fixed frequency (usually 360 Hz), the time information is still implicitly available to the user. A major disadvantage of the time normalisation is that it implicitly assumes linear distortion of the ECG waveform over the length of the normalised segment.

As mentioned previously, while the Hilbert transform display (polar plot) is reasonably familiar to a vectorcardiographic display, the data display is subject to different interpretation. Depending on the abnormality occurring in the ECG data, different displays are presented to the user. The main use of this display format is to monitor the data being placed in the pattern recognition so that different waveform segments (P,Q,R,S,T) and different time locations (before QRS ,after QRS) could be found. According to these information, the input ECG wave is normal or abnormal can be judged

Chapter 3 Computation of the Hilbert Transform

In this chapter, four methods of implementing the Hilbert transform are given. It includes the Time-domain approach, the Frequency-Domain approach, the Boche approach and Remez filter approach.

3.1 Time-Domain Approach

The Hilbert transform of a signal $y(t)$ at time t is given by

$$\hat{y}(t) = \frac{1}{\pi}\int_{-\infty}^{+\infty} \frac{y(\tau)}{t-\tau} d\tau \,. \tag{3.1}$$

Assume that the signal $y(t)$ has been sampled every Δt second to give the sequence $y_k = y(k\Delta t), \quad k = 1,2,3,\ldots,N$ and that the sampled Hilbert transform signal $\hat{y}(k)$ is to be computed. If the signal $y(t)$ is assumed to vary linearly during the sampling interval [12], for time from Δt to $N\Delta t$, the Hilbert transform at time Δt is

$$\hat{y}_k = \hat{y}(k\Delta t) = \frac{1}{\pi}\int_{\Delta t}^{N\Delta t} \frac{y(\tau)}{k\Delta t - \tau} d\tau$$

$$= \frac{1}{\pi}(\int_{\Delta t}^{2\Delta t} \frac{y(\tau)}{k\Delta t - \tau} d\tau + \ldots + \int_{(k-2)\Delta t}^{(k-1)\Delta t} \frac{y(\tau)}{k\Delta t - \tau} d\tau + \int_{(k-1)\Delta t}^{k\Delta t} \frac{y(\tau)}{k\Delta t - \tau} d\tau$$

$$+\int_{k\Delta t}^{(k+1)\Delta t}\frac{y(\tau)}{k\Delta t-\tau}+\int_{(k+1)\Delta t}^{(k+2)\Delta t}\frac{y(\tau)}{k\Delta t-\tau}d\tau+\ldots+\int_{(N-1)\Delta t}^{N\Delta t}\frac{y(\tau)}{k\Delta t-\tau}d\tau)$$

$$=\frac{1}{\pi}(\sum_{i=1}^{k-2}I_i^{(f)}+I_k^{(c)}+\sum_{i=k+2}^{N}I_i^{(r)}), \tag{3.2}$$

where

$$I_i^{(f)}\equiv\int_{i\Delta t}^{(i+1)\Delta t}\frac{y(\tau)}{k\Delta t-\tau}d\tau$$

$$I_k^{(c)}\equiv\int_{(k-1)\Delta t}^{k\Delta t}\frac{y(\tau)}{k\Delta t-\tau}d\tau+\int_{k\Delta t}^{(k+1)\Delta t}\frac{y(\tau)}{k\Delta t-\tau}d\tau$$

$$I_i^{(r)}\equiv\int_{(i-1)\Delta t}^{i\Delta t}\frac{y(\tau)}{k\Delta t-\tau}d\tau\ .$$

When $y(t)$ is linear during the sampling period,

$$y(t)=y_i+(y_{i+1}-y_i)(t-i\Delta t)/\Delta t \quad \text{for} \quad i\Delta t\le t\le(i+1)\Delta t$$

$$y(t)=y_i+(y_{i-1}-y_i)(t-i\Delta t)/(-\Delta t) \quad \text{for} \ (i-1)\Delta t\le t\le i\Delta t\,.$$

So

$$I_i^{(f)}\equiv\int_{i\Delta t}^{(i+1)\Delta t}\frac{y(\tau)}{k\Delta t-\tau}d\tau$$

$$=\int_{i\Delta t}^{(i+1)\Delta t}\frac{y_i+(y_{i+1}-y_i)(\tau-i\Delta t)/\Delta t}{k\Delta t-\tau}d\tau$$

$$=\int_0^{\Delta t}\frac{y_i+(y_{i+1}-y_i)x/\Delta t}{k\Delta t-i\Delta t-x}dx \quad (x\equiv\tau-i\Delta t)$$

$$=\int_0^{\Delta t}\frac{y_i+(y_{i+1}-y_i)\tau/\Delta t}{(k-i)\Delta t-\tau}d\tau \tag{3.3}$$

and

$$I_i^{(r)}\equiv\int_{(i-1)\Delta t}^{i\Delta t}\frac{y(\tau)}{k\Delta t-\tau}d\tau$$

$$= \int_{(i-1)\Delta t}^{i\Delta t} \frac{y_i + (y_{i-1} - y_i)(\tau - i\Delta t)/(-\Delta t)}{k\Delta t - \tau} d\tau$$

$$= -\int_{\Delta t}^{0} \frac{y_i + (y_{i-1} - y_i)x/\Delta t}{k\Delta t - i\Delta t + x} dx \quad (x \equiv -(\tau - i\Delta t))$$

$$= -\int_{0}^{\Delta t} \frac{y_i + (y_{i-1} - y_i)\tau/\Delta t}{(i-k)\Delta t - \tau} d\tau \tag{3.4}$$

and

$$I_k^{(c)} \equiv \int_{(k-1)\Delta t}^{k\Delta t} \frac{y(\tau)}{k\Delta t - \tau} d\tau + \int_{k\Delta t}^{(k+1)\Delta t} \frac{y(\tau)}{k\Delta t - \tau} d\tau$$

$$= \int_{(k-1)\Delta t}^{k\Delta t} \frac{y_k + (y_{k-1} - y_k)(\tau - k\Delta t)/(-\Delta t)}{k\Delta t - \tau} d\tau + \int_{k\Delta t}^{(k+1)\Delta t} \frac{y_k + (y_{k+1} - y_k)(\tau - k\Delta t)/\Delta t}{k\Delta t - \tau} d\tau$$

$$= \int_{-\Delta t}^{0} \frac{y_k + (y_k - y_{k-1})x/\Delta t}{-x} dx + \int_{0}^{\Delta t} \frac{y_k + (y_{k+1} - y_k)\omega/\Delta t}{-w} dw$$

$$(x \equiv \tau - k\Delta t, \quad w \equiv \tau - k\Delta t)$$

$$= \int_{-\Delta t}^{0} \frac{y_k + (y_k - y_{k-1})\tau/\Delta t}{-\tau} d\tau + \int_{0}^{\Delta t} \frac{y_k + (y_{k+1} - y_k)\tau/\Delta t}{-\tau} d\tau\,. \tag{3.5}$$

Further calculation gives

$$I_i^{(f)} = \int_{0}^{\Delta t} \frac{y_i + (y_{i+1} - y_i)\tau/\Delta t}{(k-i)\Delta t - \tau} d\tau$$

$$= -\int_{0}^{\Delta t} \frac{y_i + (y_{i+1} - y_i)(\tau - (k-i)\Delta t)/\Delta t + (y_{i+1} - y_i)(k-i)}{\tau - (k-i)\Delta t} d\tau$$

$$= -\int_{0}^{\Delta t} (y_{i+1} - y_i)/\Delta t d\tau - \int_{0}^{\Delta t} \frac{y_i + (y_{i+1} - y_i)(k-i)}{\tau - (k-i)\Delta t} d\tau$$

$$= -(y_{i+1} - y_i) - (y_i + (y_{i+1} - y_i)(k-i)) \ln\left|\tau - (k-i)\Delta t\right|_0^{\Delta t}$$

$$= y_i \ln\frac{k-i}{k-i-1} + (y_{i+1} - y_i)(-1 + (k-i)\ln\frac{k-i}{k-i-1}) \tag{3.6}$$

and

$$I_i^{(r)} = -\int_0^{\Delta t} \frac{y_i + (y_{i-1} - y_i)\tau / \Delta t}{(i-k)\Delta t - \tau} d\tau$$

$$= \int_0^{\Delta t} \frac{y_i + (y_{i-1} - y_i)(\tau - (i-k)\Delta t) / \Delta t + (y_{i-1} - y_i)(i-k)}{\tau - (i-k)\Delta t} d\tau$$

$$= \int_0^{\Delta t} (y_{i-1} - y_i) / \Delta t d\tau + \int_0^{\Delta t} \frac{y_i + (y_{i-1} - y_i)(i-k)}{\tau - (i-k)\Delta t} d\tau$$

$$= (y_{i-1} - y_i) + (y_i + (y_{i-1} - y_i)(i-k)) \ln\left|\tau - (i-k)\Delta t\right|_0^{\Delta t}$$

$$= y_i \ln \frac{i-k-1}{i-k} + (y_{i-1} - y_i)(1 + (i-k)\ln \frac{i-k-1}{i-k}) \qquad (3.7)$$

and

$$I_k^{(c)} = \int_{-\Delta t}^{0} \frac{y_k + (y_k - y_{k-1})\tau / \Delta t}{-\tau} d\tau + \int_0^{\Delta t} \frac{y_k + (y_{k+1} - y_k)\tau / \Delta t}{-\tau} d\tau$$

$$= -\int_{\Delta t}^{0} \frac{y_k + (y_k - y_{k-1})(-x) / \Delta t}{x} dx \ - \int_0^{\Delta t} \frac{y_k + (y_{k+1} - y_k)\tau / \Delta t}{\tau} d\tau$$

$$= \int_0^{\Delta t} \frac{y_k + (y_k - y_{k-1})(-\tau) / \Delta t}{\tau} d\tau - \int_0^{\Delta t} \frac{y_k + (y_{k+1} - y_k)\tau / \Delta t}{\tau} d\tau$$

$$= \int_0^{\Delta t} (y_{k-1} - y_{k+1}) / \Delta t d\tau \ .$$

$$= y_{k-1} - y_{k+1} \ . \qquad (3.8)$$

So the Hilbert transform of y_k is

$$\hat{y}_k = \frac{1}{\pi}(\sum_{i=1}^{k-2} I_i^{(f)} + I_k^{(c)} + \sum_{i=k+2}^{N} I_i^{(r)})$$

$$= \frac{1}{\pi}(\sum_{i-1}^{k-2} (y_i \ln \frac{k-i}{k-i-1} + (y_{i+1} - y_i)(-1 + (k-i)\ln \frac{k-i}{k-i-1}))$$

$$+ (y_{k-1} - y_{k+1})$$

$$+\sum_{i=k+2}^{N}(y_i \ln\frac{i-k-1}{i-k}+(y_{i-1}-y_i)(1+(i-k)\ln\frac{i-k-1}{i-k}))) . \tag{3.9}$$

The results given on the reference [5] is

$$I_i^{(f)} = y_i \ln\frac{k-i}{k-i-1}+(y_{i+1}-y_i)(-1+(k-i)\ln\frac{k-i}{k-i-1})$$

$$I_k^{(c)} = y_{k-1}-y_{k+1}$$

$$I_i^{(r)} = y_i \ln\frac{i-k}{i-k-1}+(y_{i-1}-y_i)(-1+(i-k)\ln\frac{i-k}{i-k-1}) .$$

Noticed that $I_i^{(f)}$ and Equation (3.6), $I_k^{(c)}$ and Equation (3.8) are the exactly same. But $I_i^{(r)}$ is different from Equation (3.7). The equation does influence the result of the Hilbert transform of a signal. So it is important to prove which one (Equation (3.10) or the one given on the reference [12]) is correct, This can be proved from the example below.

Here an example is given. Consider the signal $f(x)=\sin(x)$. From Equation (3.1), its Hilbert transform is

$$H[\sin(x)] = \frac{1}{\pi}\int_{-\infty}^{\infty}\frac{\sin(\tau)}{x-\tau}d\tau . \tag{3.10}$$

Letting $s = x-\tau$, get

$$H(\sin x) = \frac{1}{\pi}\int_{-\infty}^{\infty}\frac{\sin(x-s)}{s}(-ds)$$

$$= -\frac{1}{\pi}\int_{-\infty}^{\infty}\frac{\sin x\cos s-\cos x\sin s}{s}ds$$

$$= -\frac{1}{\pi}\left[\int_{-\infty}^{\infty}\frac{\sin x\cos s}{s}ds-\int_{-\infty}^{\infty}\frac{\cos x\sin s}{s}ds\right]$$

$$= -\frac{1}{\pi}\left[\sin x\int_{-\infty}^{\infty}\frac{\cos s}{s}ds - \cos x\int_{-\infty}^{\infty}\frac{\sin s}{s}ds\right]$$

It is well known that $\frac{1}{\pi}\int_{-\infty}^{\infty}\frac{\cos s}{s}ds = 0$ and $\frac{1}{\pi}\int_{-\infty}^{\infty}\frac{\sin s}{s}ds = 1$.

So $$H(\sin x) = \cos x\frac{1}{\pi}\int_{-\infty}^{\infty}\frac{\sin s}{s}ds$$

$$= \cos x \ . \tag{3.11}$$

In order to test that the Equation (3.9) is right, here an example is given. Assuming the input function is $y = \sin(2\pi * f * N)$, letting $f = 0.02$ Hz, $N = 500$. From the Equation (3.11), it can be seen that the Hilbert transform of $y = \sin(2\pi * 0.02 * 500)$ should be $\cos(2\pi * 0.02 * 500)$.

Writing a MATLAB program for this algorithm, Figure 3.1 and Figure 3.2 were obtained. Figure 3.1 shows the input waveform and Figure 3.2 (a) illustrates the Hilbert transform waveform of this input wave using Equation (3.9). Figure 3.2 (b) shows the results using the equation from reference [12].

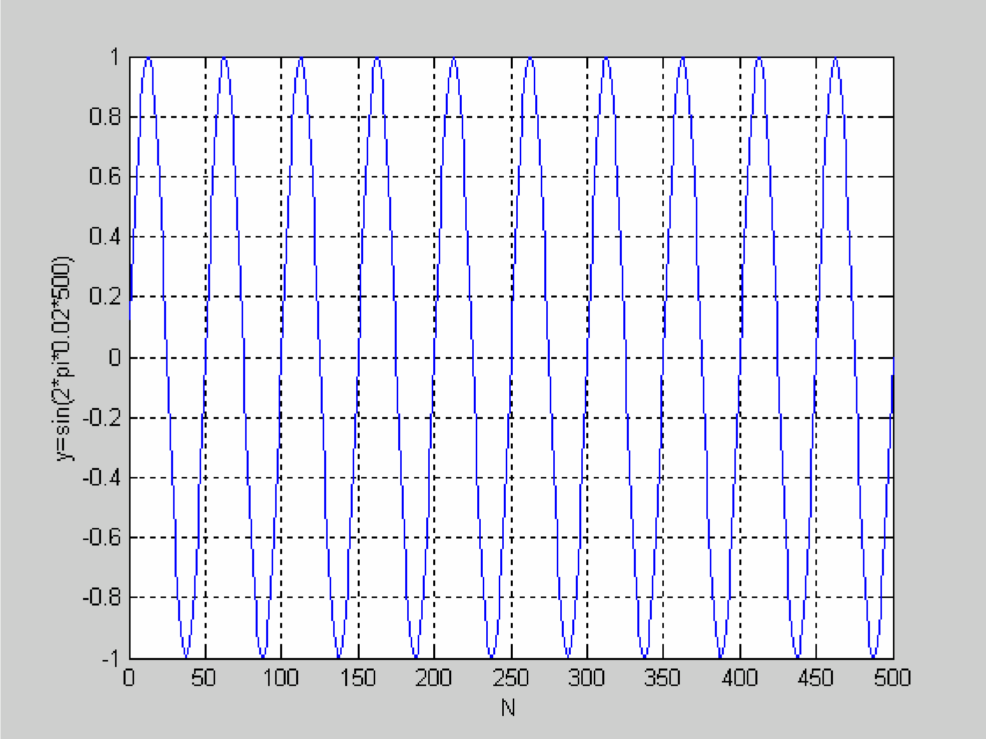


Figure 3.1 Input waveform: $\sin(2\pi * 0.02 * 500)$.

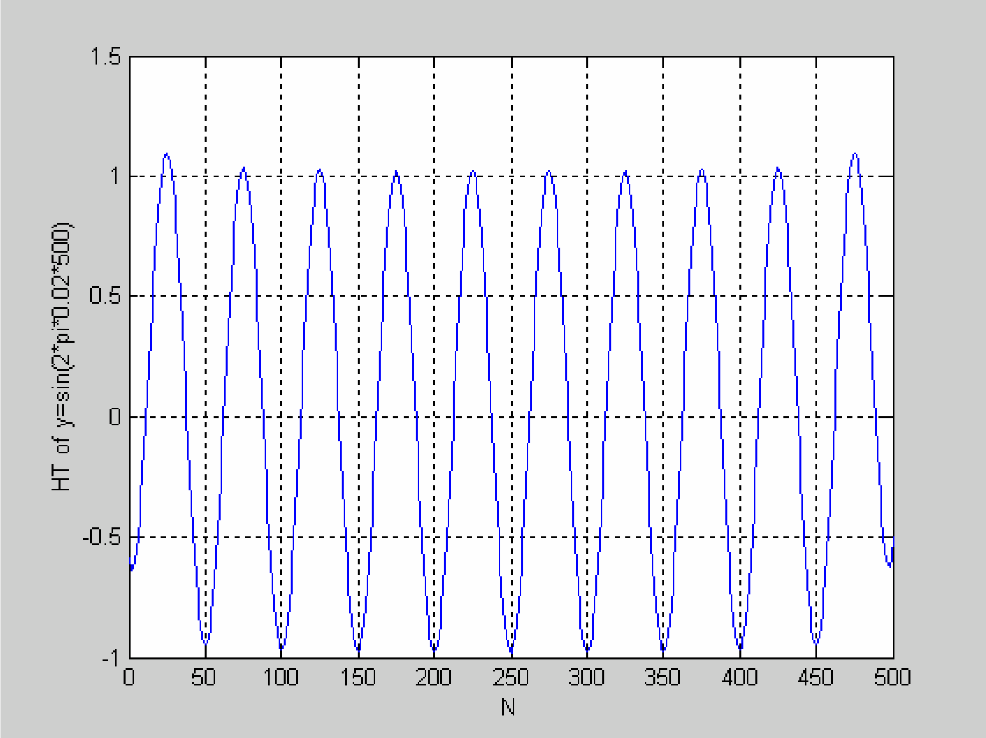


Figure 3.2 (a) Output waveform: The Hilbert transform of $\sin(2\pi * 0.02 * 500)$.

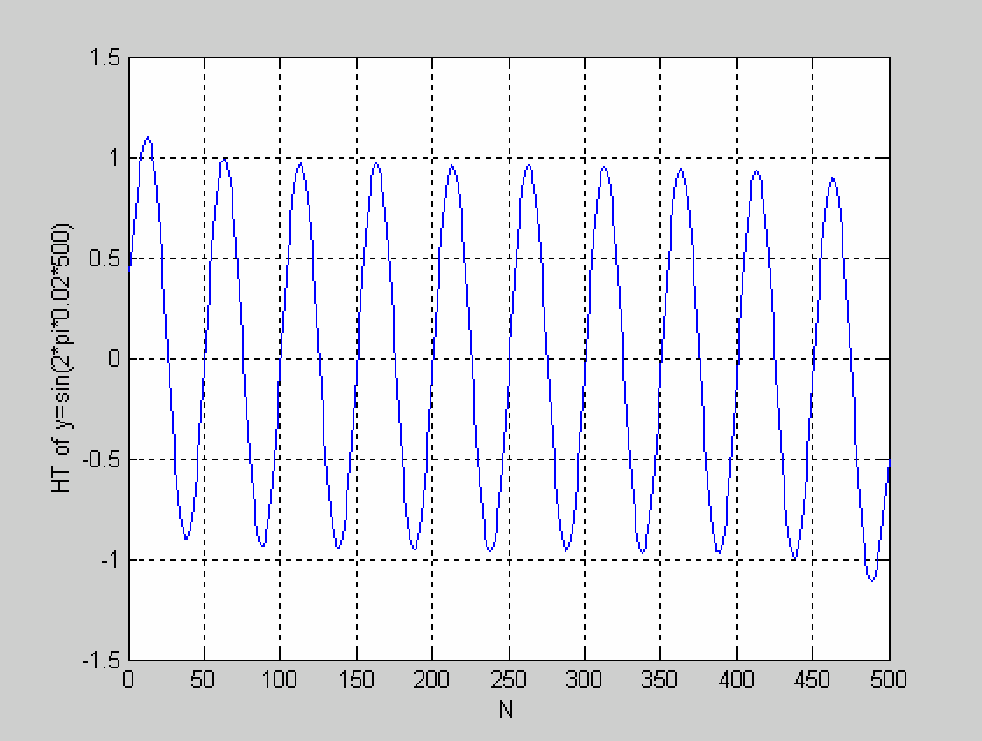


Figure 3.2 (b) Output waveform using the equation from [12].

As shown in Figure 3.2 (a), the output waveform is the Hilbert transform of the input sine wave. It is a cosine wave. It also illustrates that the Hilbert transform of a real function does not change the amplitude of the signal but only changes its phase by $\frac{\pi}{2}$ rad/s.

The waveform shown in Figure 3.2 (b) is not a cosine wave. From the waveform it also can be seen that $I_i^{(r)}$ from [12] is not correct and Equation (3. 9) is correct.

As discussed in previous chapters, the input wave in the Holter monitoring system is an ECG wave. Figure 3.3 is the wave obtained from the MIT-BIH

(Massachusetts Institute of Technology-Beth Israel Hospital) arrhythmia database [see Appendix A] MIT213. According to the Time-Domain Approach, using Equation (3.9), the Hilbert transform of this ECG wave can be computed.

The input ECG waveform is shown in Figure 3.3. According to the explanation about the ECG waves in Chapter 1, from the value of the P wave, the QRS complex, PR interval, QRS duration, RR interval and PP interval, it can be seen that Figure 3.3 is a normal ECG wave. Because each R wave stands for a beat, Figure 3.3 describes 4 beats of a heart.

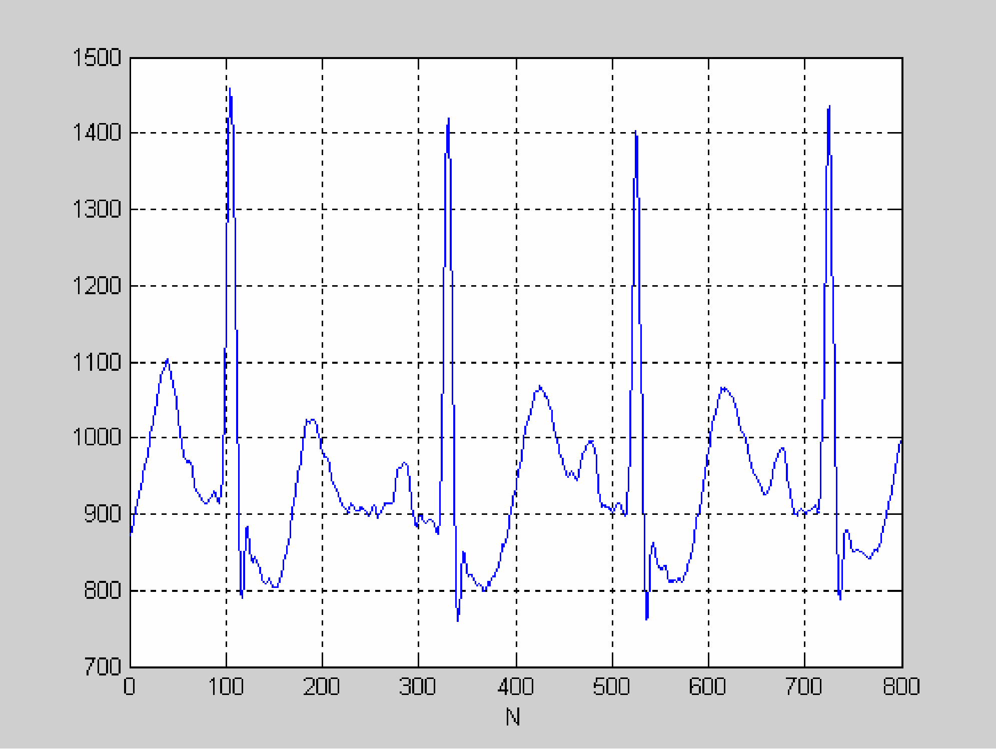


Figure 3.3 A normal ECG wave.

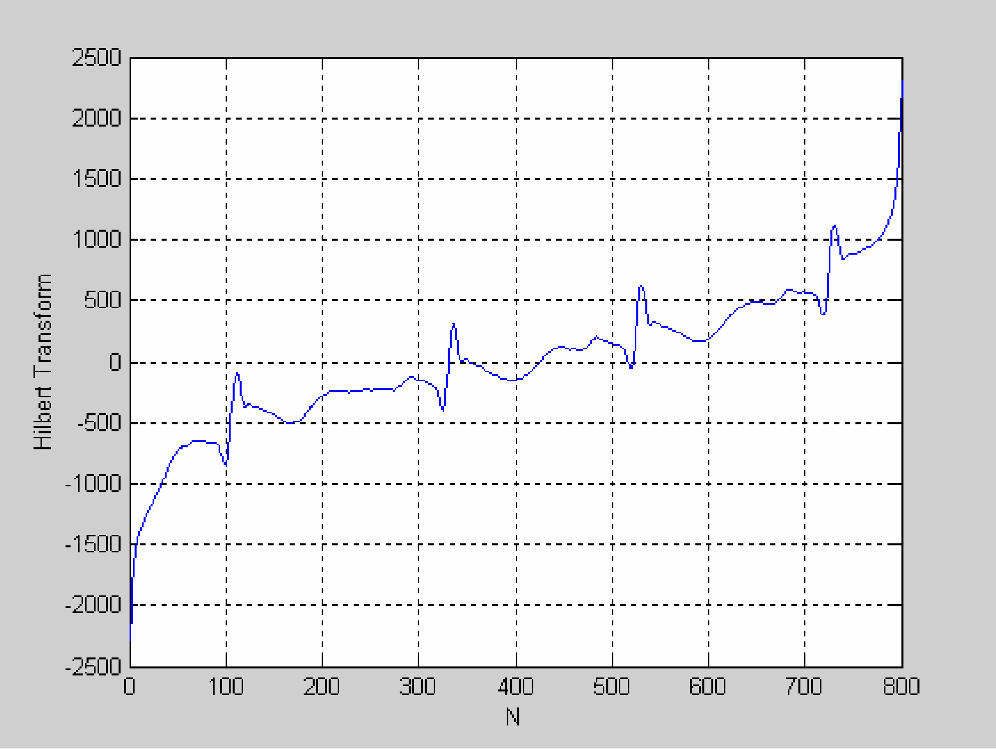


Figure 3.4 The Hilbert transform of the ECG wave.

Figure 3.4 shows the results of the Hilbert transform of this ECG wave. The Hilbert transform waveform of the ECG wave should oscillate from negative to positive or from positive to negative around the X-axis. The points corresponding to peak values of R wave should be zero in the output waveform. But from Figure 3.4, it can be seen that the output waveform is distorted. It’s not the correct waveform, so this method of computing the Hilbert transform may not be suitable for the ECG wave.

3.2 Frequency-Domain Approach

To further investigate the Hilbert transform, the frequency domain analysis is very useful. The second method to compute the Hilbert transform of a function is Frequency-Domain approach.

As shown before, the Hilbert transform of the function $y(t)$ is:

$$\widehat{y}(t) = \frac{1}{\pi}\int_{-\infty}^{\infty}\frac{y(\tau)}{t-\tau}d\tau\,.$$

Because the usual time domain definition based on the Cauchy principal value of an integral is usually not easy to calculate, the Hilbert transform in the frequency domain is defined. Suppose $Y(f)$ and $\widehat{Y}(f)$ are the Fourier transform of $y(t)$ and $\hat{y}(t)$. $Y(f)$ and $\widehat{Y}(f)$ are defined as

$$Y(f) = \int_{-\infty}^{\infty} y(t)e^{-j2\pi ft}dt$$

$$\widehat{Y}(f) = -j\,\mathrm{sgn}(f)Y(f)\,. \tag{3.12}$$

Applying the Fourier transform to the convolution defined in the equation above, can obtain

$$\widehat{Y}(j\varpi) = FT[\widehat{y}(t)] = FT\left[\frac{1}{\pi t}\right]Y(j\varpi) = -j\,\mathrm{sgn}(\varpi)Y(j\varpi)\,[5], \tag{3.13}$$

where

$FT[]$ represents Fourier transform.

This equation indicates that the Hilbert transform can be interpreted in the frequency domain.

So given a sampled signal y_k, the sequence $\hat{y}_k$ can then be computed using fast Fourier transform (FFT) techniques as

$$\hat{y}_k = IFFT[-j\,\mathrm{sgn}(\varpi_n)FFT[y_k]], \tag{3.14}$$

where

$FFT[]$ represents the fast Fourier Transform operation.

$IFFT[]$ represents inverse fast Fourier transform operations.

ϖ_n represents the nth frequency of the discrete Fourier transform.

sgn is the sign function.

This formula can be used to calculate the Hilbert transform, by first taking the Fourier transform of y_k, multiplying it by $-j\,\mathrm{sgn}(\varpi_n)$, then taking the inverse Fourier transform, thus obtaining $\hat{y}_k$. Thus the Hilbert transform is a $-\frac{\pi}{2}$ phase shifter when viewed as a linear system whose input is y_k and output is $\hat{y}_k$.

Here an example is also given. The input is the sine wave, $\sin(2\pi * 0.02 * 500)$, used previously, and the output should be the Hilbert transform of the input, i.e., a cosine wave. Figure 3.5 shows the input waveform $\sin(2\pi * 0.02 * 500)$ and Figure 3.6 shows the output results using Frequency-Domain approach to compute the Hilbert transform of a function.

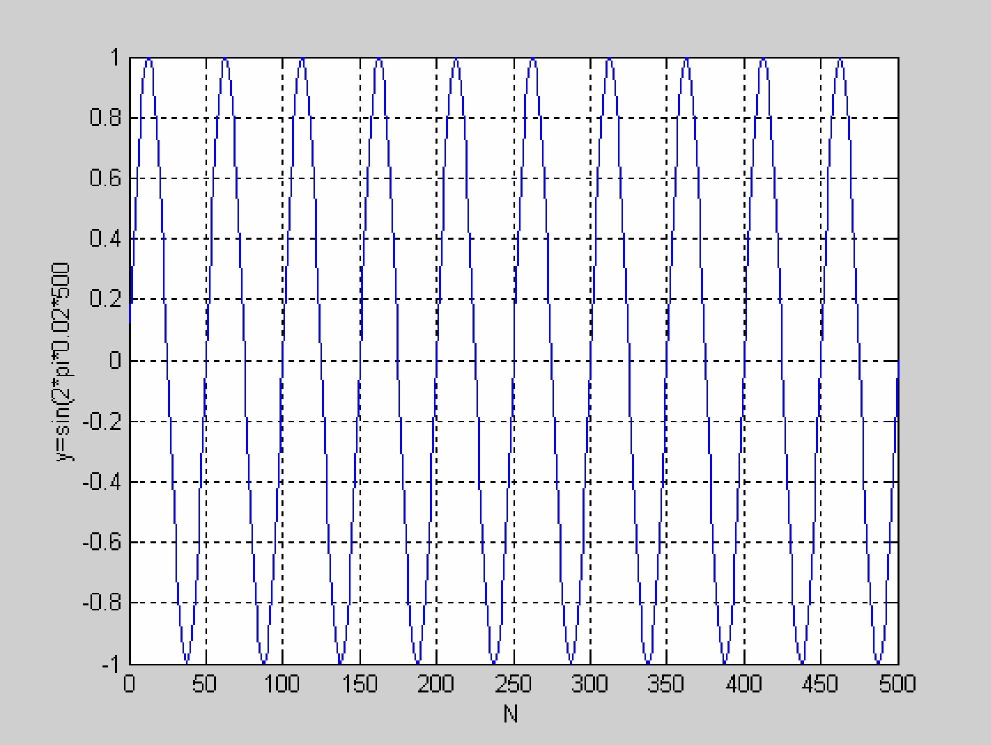


Figure 3.5 Input wave: $\sin(2\pi * 0.02 * 500)$.

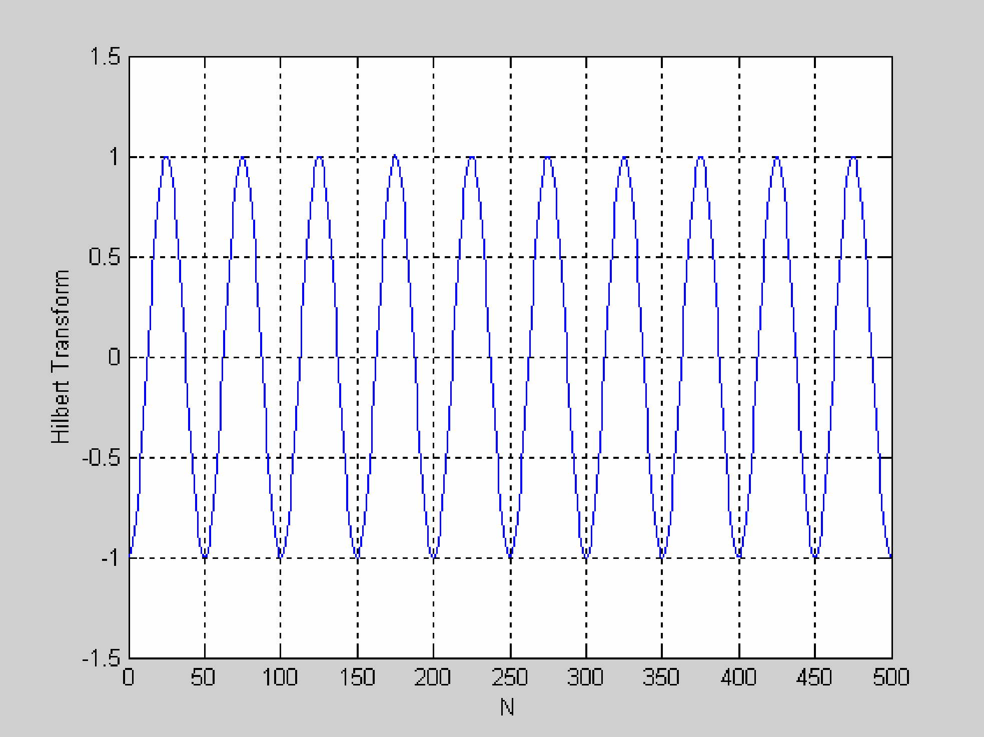


Figure 3.6 The Hilbert transform of $\sin(2\pi * 0.02 * 500)$ **.**

Figure 3.7 shows the same section of ECG wave taken from the MIT-BIH database, MIT213, used previously (Figure 3.3). Using the Frequency-Domain approach to obtain the Hilbert transform, the output waveform is shown in Figure 3.8.

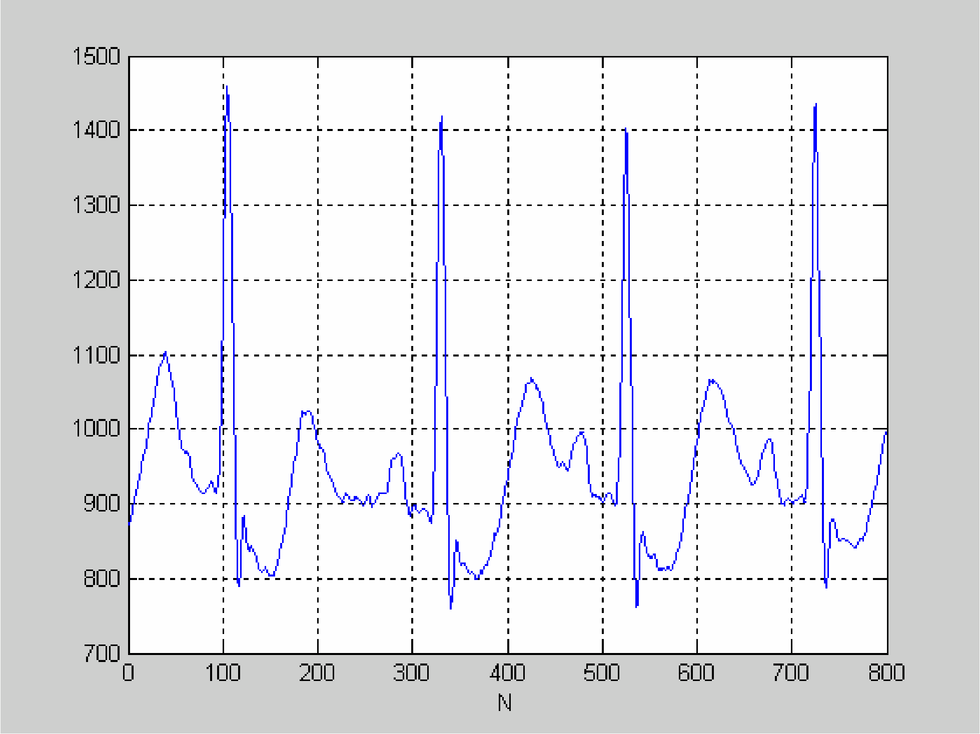


Figure 3.7 Input ECG wave

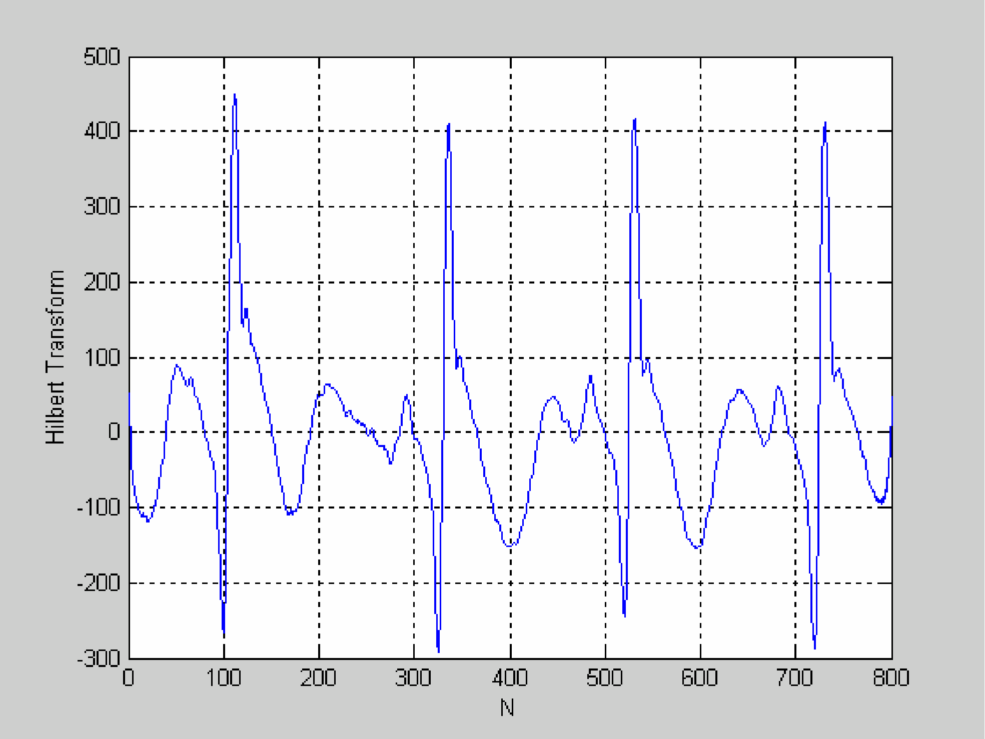


Figure 3.8 Output: the Hilbert transform of the ECG wave.

The Hilbert transform waveform shown in Figure 3.8 is good. The wave oscillates around the X-axis. The zero crossing points corresponding to the R peak wave are right.

From the examples given above, the figures show that the Hilbert transform of the sine wave is cosine wave. The output wave oscillates around zero as it should when the input wave is the ECG wave. Even though this algorithm works, it’s also not suitable for the system developed in this thesis. The reason will be given later.

3.3 Boche Approach

The Boche approach [13] presents a new algorithm by reconstructing a band-limited function from samples to calculate the Hilbert transform.

The algorithm can be described as follows:

A set of discrete instants $\{t_i\}$ are given with the corresponding values $\{y_i\}$ where exists a function $f(t_i) = y_i$. A statement can be made for approximating function as Equation (3.15):

$$f_n(t) = \sum_{k=1}^{n} b_{k,n} \frac{\sin \pi(t - t_k)}{\pi(t - t_k)} \tag{3.15}$$

$$f_n(t_i) = \sum_{k=1}^{n} b_{k,n} \frac{\sin \pi(t_i - t_k)}{\pi(t_i - t_k)}. \tag{3.16}$$

To calculate the coefficients $b_{k,n}$ of the Equation (3.16), a system of n linear equations, Equation (3.17), have to be solved:

$$\begin{bmatrix} f_n(t_1) \\ \vdots \\ f_n(t_n) \end{bmatrix} = \begin{bmatrix} a_{11} & \cdots & a_{1n} \\ \vdots & a_{ik} & \vdots \\ a_{n1} & \cdots & a_{nn} \end{bmatrix} \begin{bmatrix} b_{1n} \\ \vdots \\ b_{nn} \end{bmatrix}, \tag{3.17}$$

where $a_{ik} = \dfrac{\sin \pi(t_i - t_k)}{\pi(t_i - t_k)} \quad t_i \neq t_k$ for $i \neq k$.

Using the iteration method to solve the linear equations, the coefficients $b_{k,n}$ were obtained.

The Hilbert transform of the Equation (3.17) can be derived as follows:

It is well known that

$$\int_{-\infty}^{\infty} \frac{\sin(\pi(t - t_k))}{\pi(t - t_k)} dt = 1$$

and

$$\int_{-\infty}^{\infty} \frac{\cos(\pi(t-t_k))}{\pi(t-t_k)} dt = 0 .$$

According to the definition of the Hilbert transform of a real function, the Hilbert transform $\hat{f}(t)$ of $f(t) = \frac{\sin(\pi(t-t_k))}{\pi(t-t_k)}$ is derived as follows:

$$\hat{f}(t) = \frac{1}{\pi}\int_{-\infty}^{\infty} \frac{\sin(\pi(t-t_k))}{\pi(t-t_k)} \frac{1}{\lambda - t} dt$$

$$= \frac{1}{\pi}\int_{-\infty}^{\infty} \frac{\sin(\pi(t-t_k))}{\pi(\lambda-t_k)} (\frac{1}{\lambda - t} + \frac{1}{t-t_k}) dt$$

$$= \frac{1}{\pi(\lambda-t_k)}\int_{-\infty}^{\infty} (\frac{\sin(\pi(t-t_k))}{\pi(t-t_k)} - \frac{\sin(\pi(t-\lambda+\lambda-t_k))}{\pi(t-\lambda)}) dt$$

$$= \frac{1}{\pi(\lambda-t_k)}\int_{-\infty}^{\infty} (\frac{\sin(\pi(t-t_k))}{\pi(t-t_k)} - \frac{\sin(\pi(t-\lambda))\cos(\pi(\lambda-t_k)) + \cos(\pi(t-\lambda))\sin(\pi(\lambda-t_k))}{\pi(t-\lambda)}) dt$$

$$= \frac{1}{\pi(\lambda-t_k)}(1-\cos(\pi(\lambda-t_k)))$$

$$= \frac{1-\cos(\pi(\lambda-t_k))}{\pi(\lambda-t_k)} . \qquad (3.18)$$

It is shown that the Hilbert transform of the series

$$f_n(t) = \sum_{k=1}^{n} b_{k,n} \frac{\sin \pi(t-t_k)}{\pi(t-t_k)}$$

is

$$H\{f_n(t)\} = \sum_{k=1}^{n} b_{k,n} \frac{1-\cos(\pi(t-t_k))}{\pi(t-t_k)} , \qquad (3.19)$$

where $b_{k,n}$ are the coefficients in Equation (3.17).

Here an example is given to demonstrate the algorithm for the sampled signal as well as its Hilbert transform. Assuming the sampled function g is given by

$$f(t) = \frac{\sin(2t)}{2t} - \frac{3}{4}\frac{\sin(3(t-5))}{3(t-5)} + \frac{1}{2}\frac{\sin(2.5(t+6))}{2.5(t+6)}.$$

The samples were taken in the interval $-10 \le t \le 10$ with a sampling interval of 0.25, thus yielding 81 sampling points. Based on the Equation (3. 19), i.e.,

$$\hat{f}(\lambda) = \frac{1-\cos(\pi(\lambda - t_k))}{\pi(\lambda - t_k)}$$

and the linear property of the Hilbert transform that discussed in Chapter 2, the Hilbert transform, $\hat{f}(t)$, of the function , $f(t)$, can be written as follows :

$$\hat{f}(t) = \frac{1-\cos(2t)}{2t} - \frac{3}{4}\frac{(1-\cos(3(t-5)))}{3(t-5)} + \frac{1}{2}\frac{(1-\cos(2.5(t+6)))}{2.5(t+6)}.$$

Running a MATLAB program, the results are shown in Figure 3.9 and Figure 3.10. Figure 3.9 (a) shows the waveform of original function $f(t)$, Figure3.9 (b) is the 26th approximation $f_{26}(t)$ after 26 iterations. The error function is obtained using $f_{26}(t)$ minus $f(t)$, and the result is shown in Figure 3.9 (c). The error function is smaller than 0.0001 shown in Figure 3.9 (c). The sample sequence t_i is given in Figure 3.9 (d).

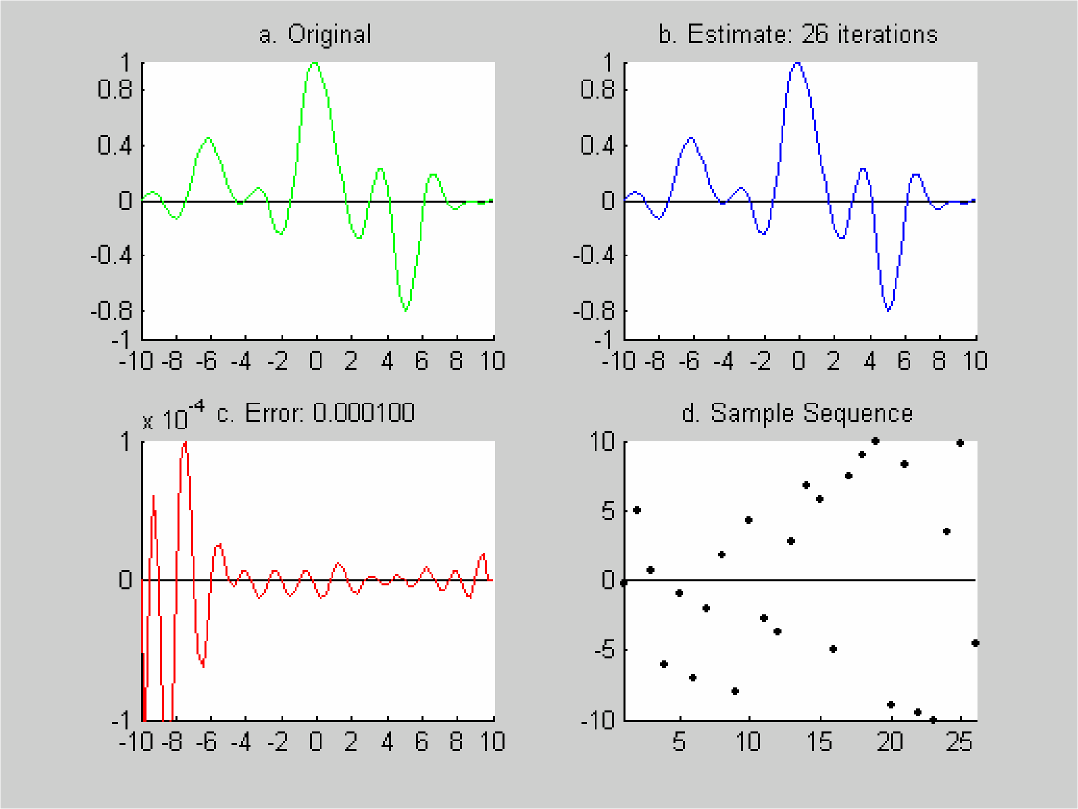


Figure 3.9 (a) $f(t)$ **(b)** f_{26} **(c)** $f_{26}(t)-f(t)$ **(d)** t_i **.**

Figure 3.10 (a) shows the Hilbert transform $\hat{f}(t)$ of the original function $f(t)$, and the 26th approximation $\hat{f}_{26}(t)$ is shown in Figure 3.10 (b), and then the error function of the Hilbert transform is illustrated in Figure 3.10 (c).

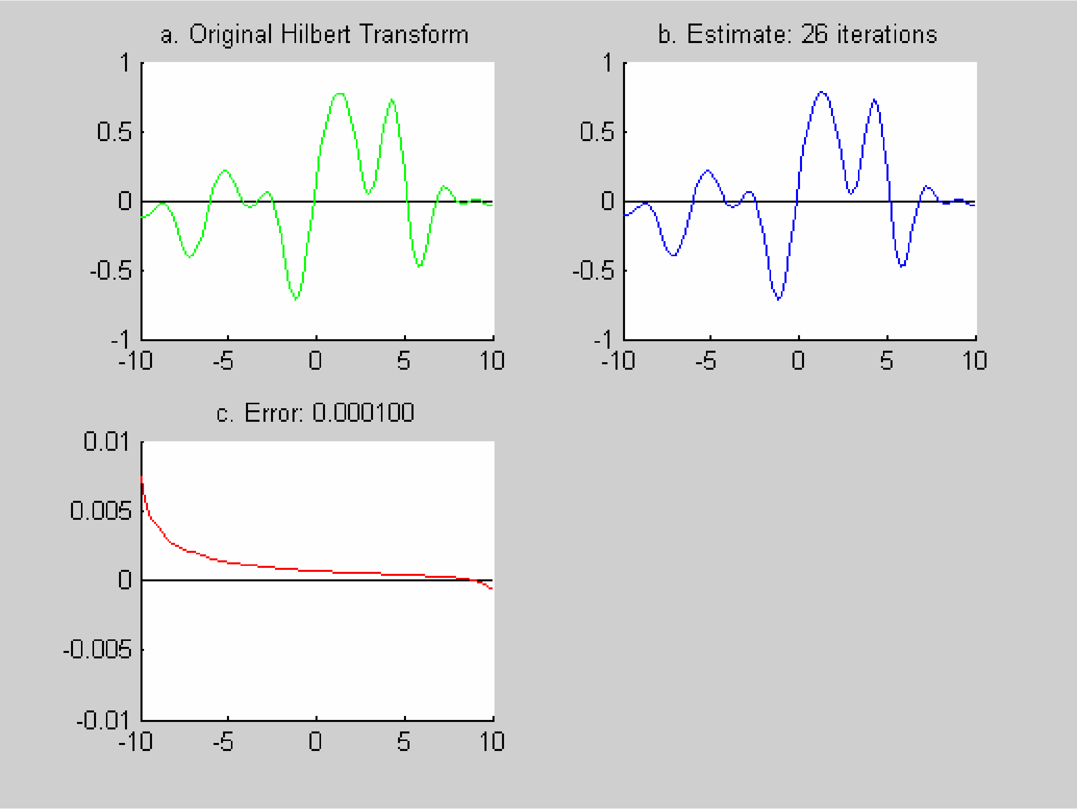


Figure 3.10 (a) $\hat{f}(t)$ **(b)** $\hat{f}_{26}(t)$ **(c)** $\hat{f}_{26}(t)-\hat{f}(t)$

Boche algorithm permits reconstruct the bandlimited function from samples and recovery of the Hilbert transform of this function. Compared with other known solutions for computing Hilbert transform of a function, this algorithm does not need to calculate integrals. However a set of linear equations has to be solved in each iteration step.

Here the algorithm is just given as a reference. In this thesis, the difficulty of computing the solution to a variable set of linear equations is not easier than calculating an integral. So this method is not suitable for the Holter monitoring system in this thesis.

3.4 Remez Approach

This section mainly talks about using “ *remez* ” function in MATLAB to obtain the Hilbert transform of a signal.

The Remez Exchange FIR filter design approach (also called the Parks-McClellan or Optimal method) is a popular technique used to design FIR filters. The well known Parks-McClellan algorithm uses this approach and Chebyshev approximation theory to generate filters with an optimal fit between the desired frequency responses and actual frequency responses. Filters designed in this way illustrate an equiripple wave in the frequency response. By implementing the Parks-McClellan algorithm [see Appendix B], the Remez approach designs a linear-phase FIR filter.

The syntax of the remez function can be written the following way:

$$b = remez(n, f, a)$$

$b = remez(n, f, a)$ returns a row vector b including the $n+1$ coefficients of the order n FIR filter.

where

“ n ” represents the order of the filter.

“ f ” represents a vector of pairs of normalized frequency points. The frequencies are specified in the range between 0 and 1, where 1 corresponds to the Nyquist frequency. The frequencies must be in increasing order.

“ a ” represents a vector containing the desired amplitudes at the points specified in f . f and a are the same length. The length is an even number.

The output coefficients in b satisfied the symmetry relation $b(k) = b(n+2-k),\ k = 1,...,n+1$;

Remez function can specify the different filter type:

$$b = remez(n, f, a, 'ftype').$$

$b = remez(n, f, a, w, 'ftype')$ is used when the special filter type is needed.

where

$'ftype'$ represents the filter type parameter. It includes three types: Multiband, Differentiator and Hilbert transform. The one used in this thesis is 'hilbert', that is the Hilbert transformer, for linear-phase filters with odd symmetry. The output filter coefficients in b satisfies $b(k) = -b(n+2-k)$, $k = 1,2,\ldots,n+1$. The Hilbert transformer has the desired amplitude of 1 across the entire band.

Here an example is given:

$$h = remez\,(100, [0.05\ 0.95], [1\ 1], 'hilbert')$$

designs an approximate FIR Hilbert transformer of length 100. The frequencies are specified from 0.05 to 0.95 and their corresponding amplitudes are 1. The amplitude will be 0 at other frequencies.

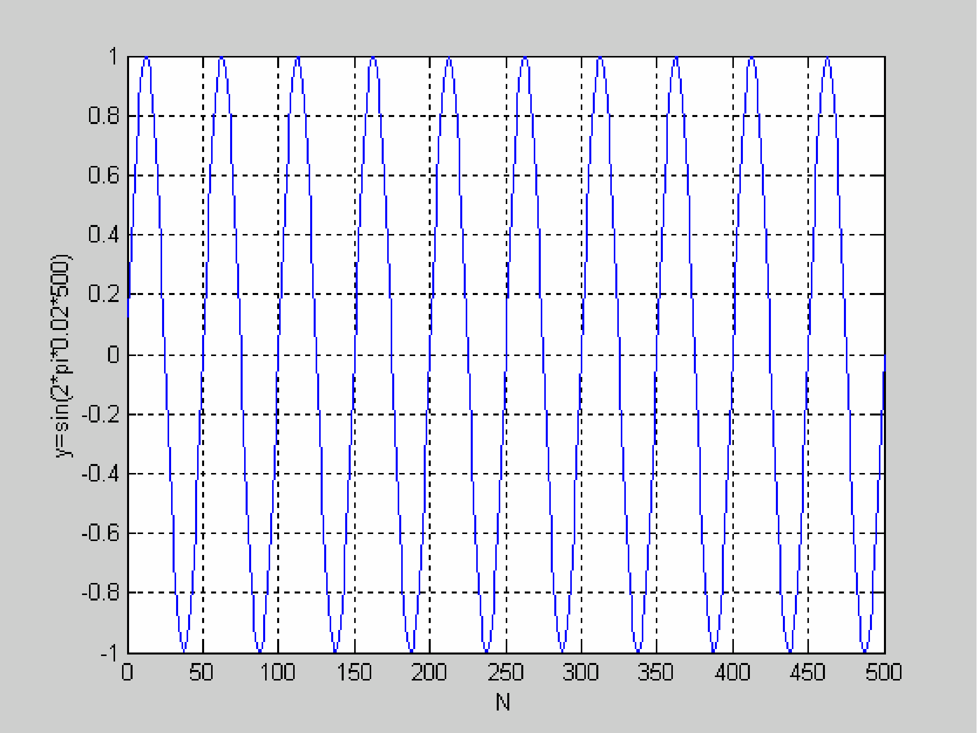


Figure 3.11 The input wave: $y = \sin(2\pi * 0.02 * 500)$.

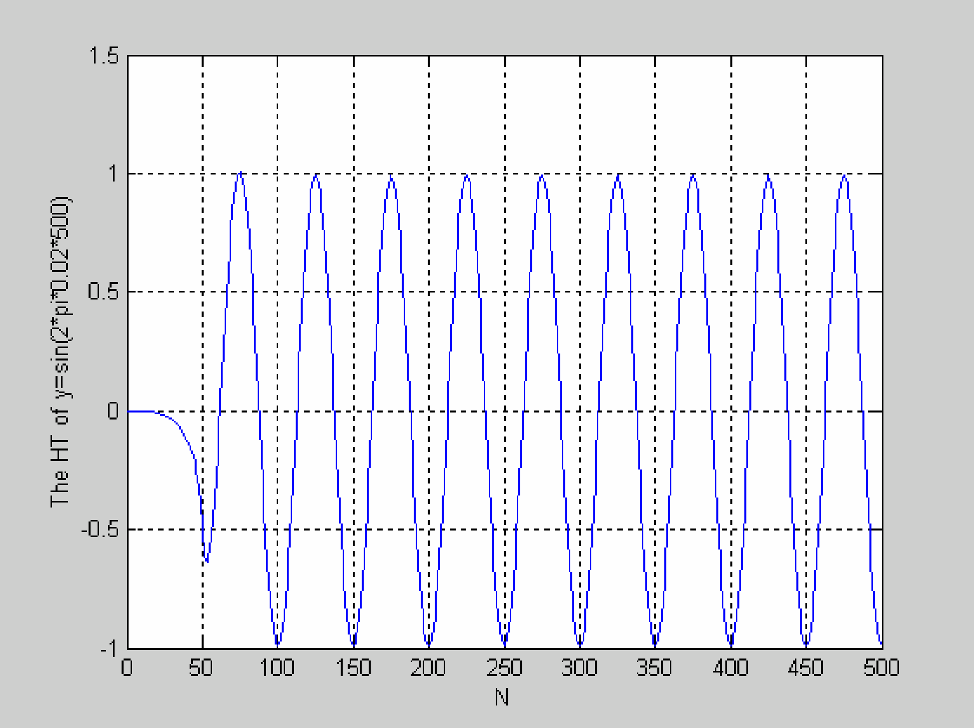


Figure 3.12 The output wave: HT of $y = \sin(2\pi * 0.02 * 500)$.

As before, the input signal is the same sine wave $y = \sin(2\pi * 0.02 * 500)$, used previously. The *remez* function discussed above was used to generate a Hilbert transformer and obtain the Hilbert transform of the input sine signal. The input waveform is shown in Figure 3.11.

Figure 3.12 shows the Hilbert transform of the input signal $y = \sin(2\pi * 0.02 * 500)$. Because the order of the filter used here is $M = 100$, the filter phase delay should be $100/2 = 50$. From the output waveform, it can be seen clearly that the filter has a phase delay for N from 0 to 50. The output results corresponding to the input signal should be calculated from $N = 51$. Figure 3.12 illustrates that the Hilbert transform of a sine function is a cosine function.

Another example is given here. The input wave is still the normal ECG wave from the MIT-BITH Arrhythmia database MIT213 used previously (see Figure 3.3).

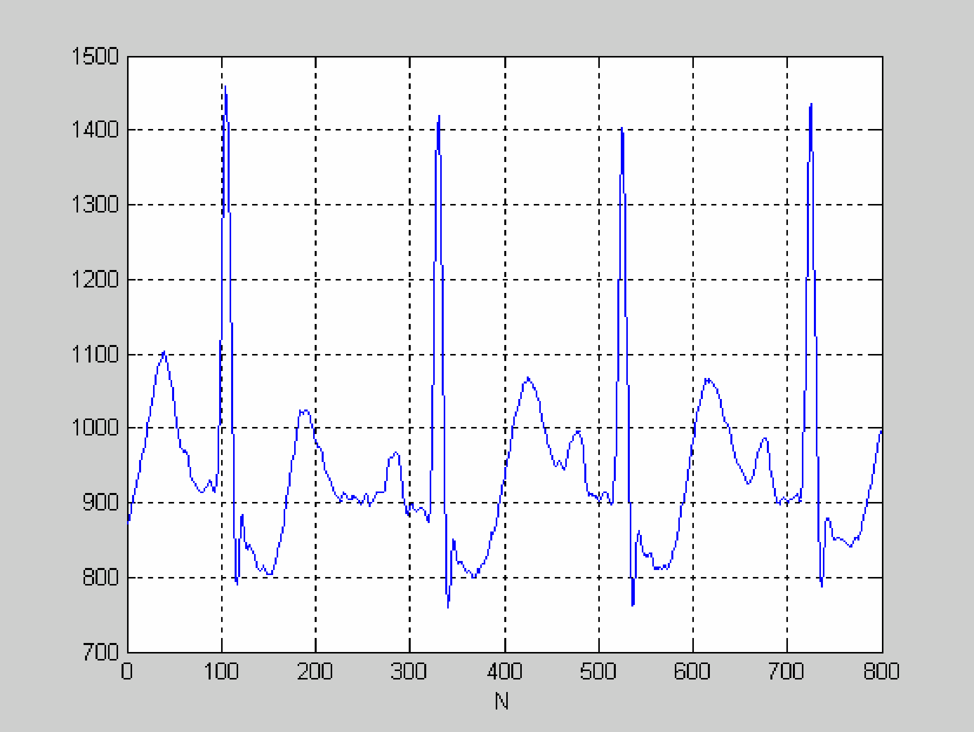


Figure 3.13 An ECG wave.

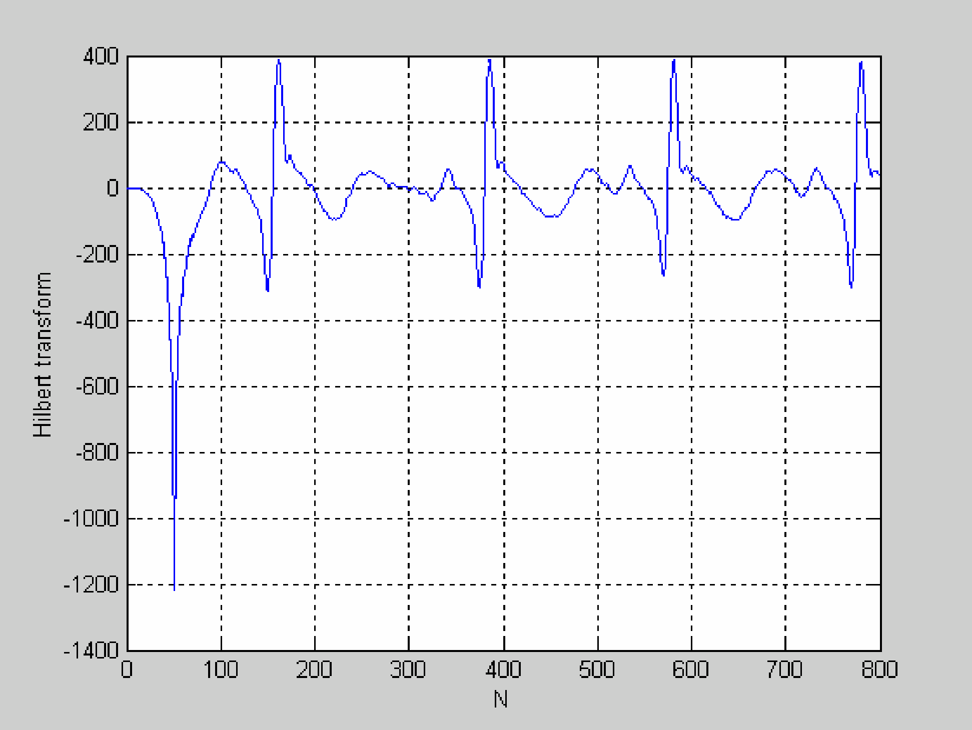


Figure 3.14 The Hilbert transform of an ECG wave.

Figure 3.14 shows the output, i.e., the Hilbert transform of the input ECG signal. For the same reason, the filter phase delay, the output wave lags by 50 samples.

3.5 Comparison

Four methods for computing Hilbert transform are discussed above. From this section, the comparison of these four methods is given. Table 3.1 shows the different running time and the easy level of hardware (HW) and software (SW) implementation. All of programs are MATLAB programs and timing was done using the tic/toc functions.

Table 3.1 The comparison of the four methods for computing Hilbert transform.

	Time Domain Approach		**FrequencyDomain Approach**		**Boche Approach**	**Remez Filter Approach**	
	Sine (N=500)	ECG (N=800)	Sine (N=500)	ECG (N=800)		Sine (N=500)	ECG (N=800)
Running Time	2.7740s	7.1110s	0.010s	0.010s	0.6110	0.0140s	0.030s
HW/SW	HW&SW Problem		HW Problem		HW&SW Problem	OK	

For Time-Domain approach, even though the equations for computing the Hilbert transform have been derived and it is not needed to calculate the integral anymore, it still needs to compute the “ln” function. It is hard to implement in the Nios system that will be used in this thesis because there is no hardware “ln” function in the Nios processor. The running time was measured for sine wave and ECG wave individually. Note that the sample number is 500 for sine wave and 800 for ECG wave. The results are shown in the Table 3.1. Using this method, the output waveform that corresponding to an input ECG signal is distorted to some extent.

For Frequency-Domain approach, the running time is very fast, but it is inconvenient to be implemented on the hardware because it needs to compute the FFT and IFFT for this method. This is not that easy to implement on the hardware.

The Boche approach supplied a simple way to calculate the Hilbert transform of the bandlimited function. Even though it does not need to calculate the integral, a set of linear equations has to be solved. The size of the equations is variable. It's not that easy to implement in the Nios processor used in this thesis.

Compared with the first three methods discussed above, the Remez filter approach is a better choice to compute Hilbert transform of a function in this thesis. Its running time is shorter and can be implemented on both hardware and software.

The computer performance that used in this thesis:

Operation System: Microsoft Window XP

CPU: Intel® Pentium®4 2.66GHz

RAM: 512MB.

Chapter 4 Implementation

In the previous chapters, four methods to compute the Hilbert transform applied to the input data have been discussed. After comparing them in terms of running time and the complexities of the software and hardware implementation, the Remez approach was selected to be the best method for the ECG Holter Monitoring System. The problem now, is how to implement the Hilbert transform algorithm to build the Holter monitoring system on the Nios system. In the following sections, this will be discussed in detail.

4.1 Nios Embedded Processor Overview

The Nios embedded processor is a user-configurable, 16-bit ISA (Instruction Set Architecture), general-purpose RISC (Reduced Instruction Set Computer) embedded processor that was designed to be a very flexible and powerful processor solution [14]. The Nios embedded processor has become a commonly used embedded processor because of its ease-of-use and flexibility. The Nios embedded processor system structure is shown in Figure 4.1.

The Quartus II software, the SOPC (System on a Programmable Chip) Builder system development tool, is used to build and evaluate custom processor-based systems. Designers can use SOPC Builder to integrate one or more configurable Nios CPUs with any number of standard peripherals, gluing the system together [14]. Using SOPC Builder, a user can combine the Nios processor with user logic and program it into a FPGA (Field Programmable Gate Array) easily.

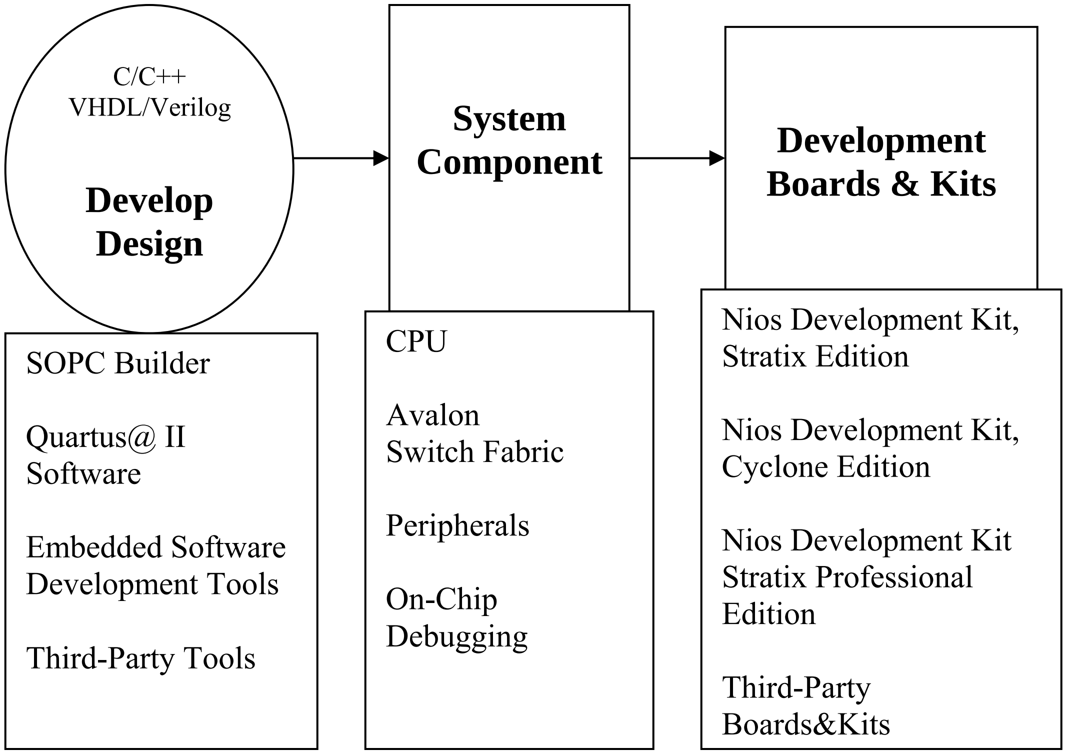

C/C++
VHDL/Verilog
Develop Design
System Component
Development Boards & Kits
SOPC Builder
Quartus@ II Software
Embedded Software Development Tools
Third-Party Tools
CPU
Avalon Switch Fabric
Peripherals
On-Chip Debugging
Nios Development Kit, Stratix Edition
Nios Development Kit, Cyclone Edition
Nios Development Kit Stratix Professional Edition
Third-Party Boards&Kits

Figure 4.1 Nios Embedded Processor System.

In this thesis, the development tool is SOPC Builder, using a Nios development kit, Stratix Professional Edition. It is a complete embedded systems development kit for the Nios embedded processor.

There are a number of necessary steps to create a Nios system on the Nios development board. The procedure is shown in Figure 4.2. The flow includes both the hardware and software design tasks required to create a working system. The right side illustrates the software development flow and the left side illustrates the hardware design flow.

Based on the system requirements, the hardware design begins with the SOPC builder system integration software. At this point, the designer can begin writing device-independent C/C++ software.

After the hardware designer defines the customer Nios processor hardware system using SOPC Builder, SOPC Builder generates a custom software development kit (SDK) that forms the foundation for the software development flow [16]. With the SDK, the designer can begin writing software that interacts at the low level with hardware components.

The Nios SDK Shell provides an UNIX bash shell environment board on a PC platform. It is a very useful utility. Figure 4.3 shows the Nios SDK shell (bash environment).

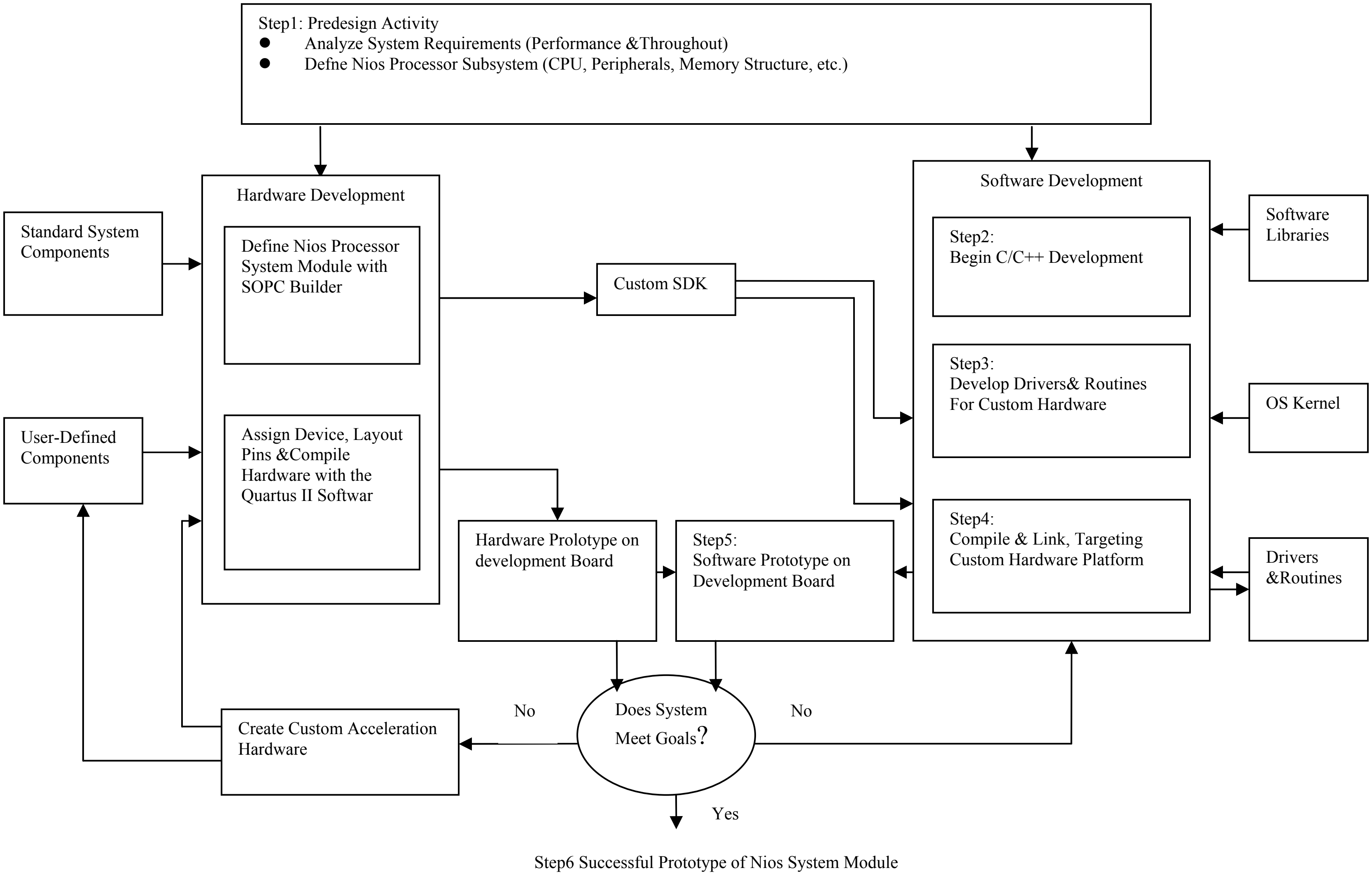
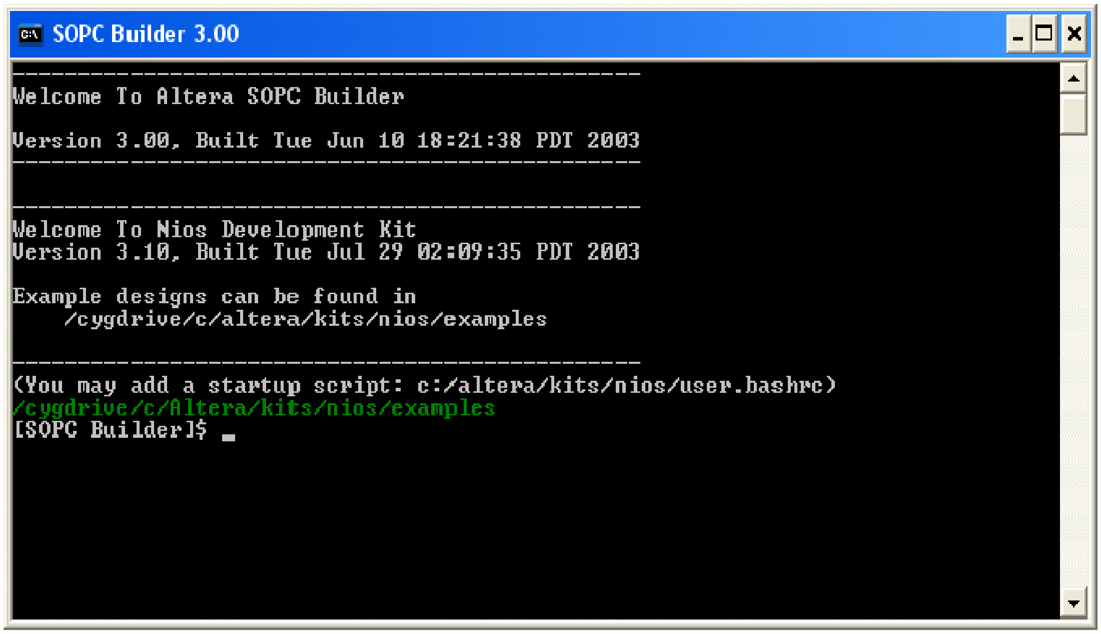

Step1: Predesign Activity
● Analyze System Requirements (Performance &Throughout)
● Defne Nios Processor Subsystem (CPU, Peripherals, Memory Structure, etc.)
Hardware Development
Define Nios Processor System Module with SOPC Builder
Assign Device, Layout Pins &Compile Hardware with the Quartus II Softwar
Standard System Components
User-Defined Components
Custom SDK
Software Development
Step2: Begin C/C++ Development
Step3: Develop Drivers& Routines For Custom Hardware
Step4: Compile & Link, Targeting Custom Hardware Platform
Software Libraries
OS Kernel
Drivers &Routines
Hardware Prolotype on development Board
Step5: Software Prototype on Development Board
Does System Meet Goals?
No
No
Yes
Create Custom Acceleration Hardware
Step6 Successful Prototype of Nios System Module

Figure 4.2 Hardware/Software development flow for a Nios processor system [15]



```
--------------------------------------------------
Welcome To Altera SOPC Builder

Version 3.00, Built Tue Jun 10 18:21:38 PDT 2003
--------------------------------------------------

--------------------------------------------------
Welcome To Nios Development Kit
Version 3.10, Built Tue Jul 29 02:09:35 PDT 2003

Example designs can be found in
    /cygdrive/c/altera/kits/nios/examples

--------------------------------------------------
(You may add a startup script: c:/altera/kits/nios/user.bashrc)
/cygdrive/c/Altera/kits/nios/examples
[SOPC Builder]$ _
```

Figure 4.3 Nios SDK Shell (bash).

This "bash" environment can be used for all related development work for the Nios system and communicate with the Nios development board. The Nios development board is shown in Figure 4.4. The Nios development kit includes many Nios-specific utilities that can run in the Nios SDK Shell to generate and debug software. The Nios SDK Shell also can be used to run test programs on the Nios development board.

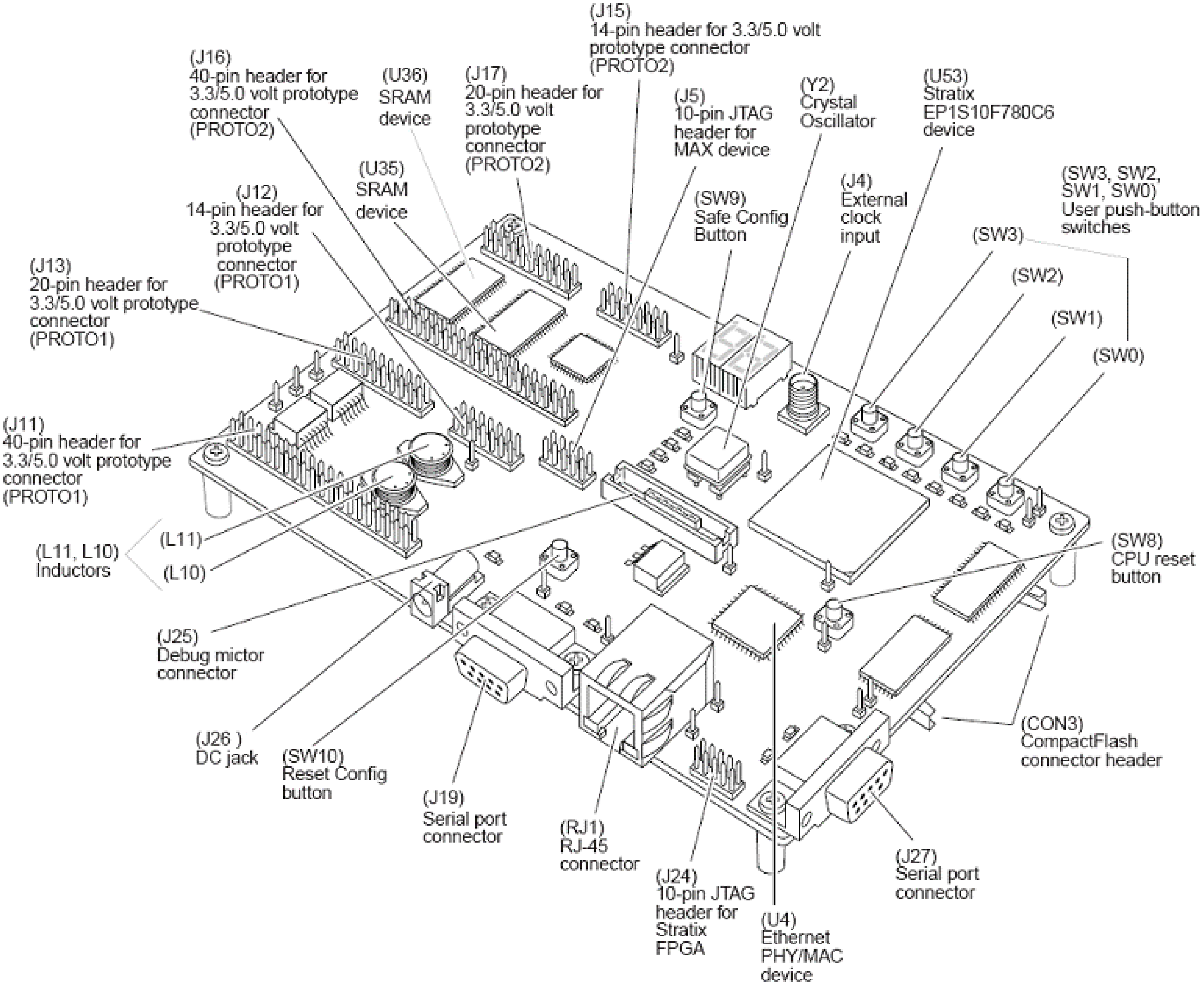

(J15)
14-pin header for 3.3/5.0 volt prototype connector (PROTO2)
(J16)
40-pin header for 3.3/5.0 volt prototype connector (PROTO2)
(U36)
SRAM device
(J17)
20-pin header for 3.3/5.0 volt prototype connector (PROTO2)
(J5)
10-pin JTAG header for MAX device
(Y2)
Crystal Oscillator
(U53)
Stratix EP1S10F780C6 device
(U35)
SRAM device
(J12)
14-pin header for 3.3/5.0 volt prototype connector (PROTO1)
(SW9)
Safe Config Button
(J4)
External clock input
(SW3, SW2, SW1, SW0)
User push-button switches
(J13)
20-pin header for 3.3/5.0 volt prototype connector (PROTO1)
(SW3)
(SW2)
(SW1)
(SW0)
(J11)
40-pin header for 3.3/5.0 volt prototype connector (PROTO1)
(L11)
(L11, L10)
Inductors
(L10)
(SW8)
CPU reset button
(J25)
Debug mictor connector
(J26)
DC jack
(SW10)
Reset Config button
(CON3)
CompactFlash connector header
(J19)
Serial port connector
(RJ1)
RJ-45 connector
(J24)
10-pin JTAG header for Stratix FPGA
(U4)
Ethernet PHY/MAC device
(J27)
Serial port connector

Figure 4.4 Nios Development Board Components [17]

Figure 4.4 shows the Nios development board components. It includes [18]

- Stratix EP1S40F780 device
- MAX EPM7128AE CPLD configuration control logic
- SRAM (1 Mbyte in two banks of 512 Kbytes, 16-bit wide)
- SDR SDRAM (16 Mbytes, 32-bit wide)
- Flash (8 Mbytes)
- CompactFlash connector header for Type 1 CompactFlash cards
- 10/100 Ethernet physical layer/media access control (PHY/MAC)
- Ethernet connector (RJ-45)
- Two serial connectors (RS-232 DB9 port)
- Two 5-V-tolerant expansion/prototype headers
- Two JTAG connectors
- 50-MHz crystal (socket), external clock input
- Mictor connector for debugging
- Four user-defined push-button switches
- Eight user-defined LEDs
- Dual 7-segment LED display
- Power-on reset circuitry.

Hardware designers can use the Nios development board as a platform to prototype complex embedded systems. Software developers can use the Nios reference design pre-programmed on the development board to begin prototyping software immediately.

4.2 Digital Filter

In digital signal processing, an important function of a filter is to remove unwanted parts of the signal, such as random noise, or to extract useful parts of the signal, such as the components lying within a certain frequency range [19]. Figure 4.5 illustrates the basic concept.

raw (unfiltered) signal
FILTER
filtered signal

Figure 4.5 Filter.

The filter function is implemented as a direct form II transposed structure as shown in Figure 4.6.

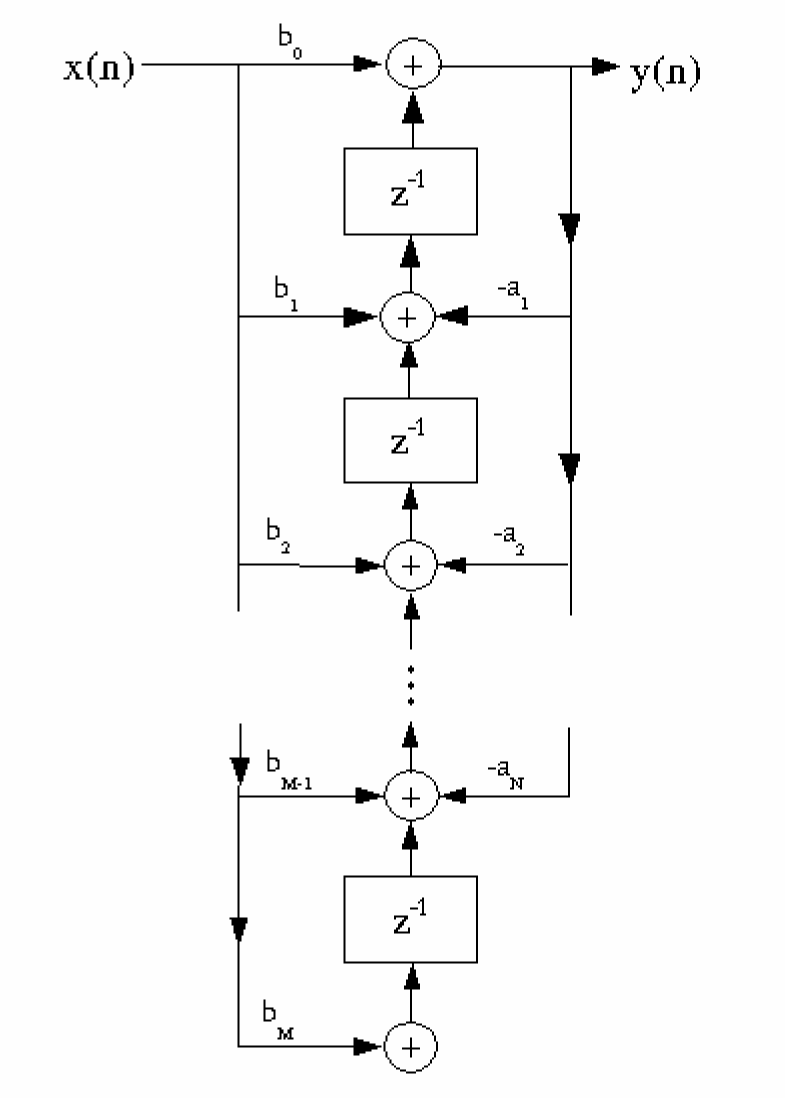


Figure 4.6 Digital filter transposed structure.

For a linear time-invariant system (Figure 4.6), its input and output satisfy the following equation

$$y[n]-\sum_{k=1}^{N}a_k y[n-k]=\sum_{k=0}^{M}b_k x[n-k] \tag{4.1}$$

with the corresponding rational system function

$$H(z)=\frac{\sum_{k=0}^{M}b_k z^{-k}}{1-\sum_{k=1}^{N}a_k z^{-k}} \text{ [20]}, \tag{4.2}$$

where:

$x[n-k]$ is the previous input.

$y[n]$ is the output.

$y[n-k]$ is the previous output.

a_k and b_k are the filter coefficients.

$H(z)$ is the filter's Z transform.

For causal FIR (Finite Impulse Response) system, the system function has only zeros (except for poles at z=0), FIR filter does not depend on the past values of the output. FIR filters are therefore non-recursive. Since the coefficients a_k are all zero, the Equation (4.1) reduces to

$$y[n]=\sum_{k=0}^{M}b_k x[n-k]. \tag{4.3}$$

When the filter sequence (impulse response) of FIR filter is either symmetric or anti-symmetric, the filter is of linear phase. Such filters do not distort the phase of the input signal. It is well known that FIR filters can always be designed such that they exhibit the desirable characteristic.

If the impulse response of the FIR filter satisfies the condition

$$h[M-n] = h[n] \quad \text{for } n = 0,1,\ldots,M,$$

it is called symmetric.

If the impulse response of the FIR filter satisfies the condition

$$h[M-n] = -h[n] \quad \text{for } n = 0,1,\ldots,M,$$

it is called anti-symmetric. Table 4.1 categorizes linear phase filters according to their symmetry and length.

Table 4.1 Four types of the linear phase FIR filter

Type	Impulse Response	
1	symmetric	Length (M+1) is odd
2	symmetric	Length (M+1) is even
3	anti-symmetric	Length (M+1) is odd
4	anti-symmetric	Length (M+1) is even

Examples of the four types of impulse response sequences are shown in Figure 4.7.

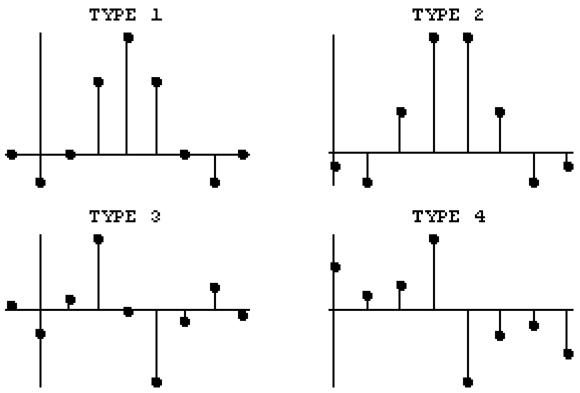


Figure 4.7 Illustration of four types of impulse response symmetry.

A Type 1 filter may be used to implement any desired bandpass frequency response. A Type 2 filter may not be used to define a highpass filter since the symmetry condition requires $H(\pi)=0$. It can be used instead of Type 1 in cases where an even length filter is preferable [21]. Antisymmetric filters can be used to design FIR differentiators and Hilbert transformers. Differentiators are anti-symmetric FIR filters with approximately linear magnitude responses. Hilbert transformers are anti-symmetric FIR filters with approximately constant magnitude.

4.3 Implementation

In this section, the implementation for performing the Hilbert transform of an input signal will be discussed in detail.

4.3.1 Filter Order

As discussed in previous chapter, the order of the filter should be determined to meet certain filter specifications including the passband ripple, stopband attenuation and the transition bandwidth. MATLAB is a perfect tool for this purpose. A sine wave signal was used as the test signal. Let M represent the filter order.

The Hilbert transform experiments were conducted with the MATLAB program and 501 samples of the test signal $\sin(2*\pi*0.02*n)$ with $n=0,\ldots,500$. The input signal is shown in Figure 4.8. The Hilbert transform outputs are shown in Figures 4.9-4.13 for the filter order $M =$ 51,71,91,101 and 201.

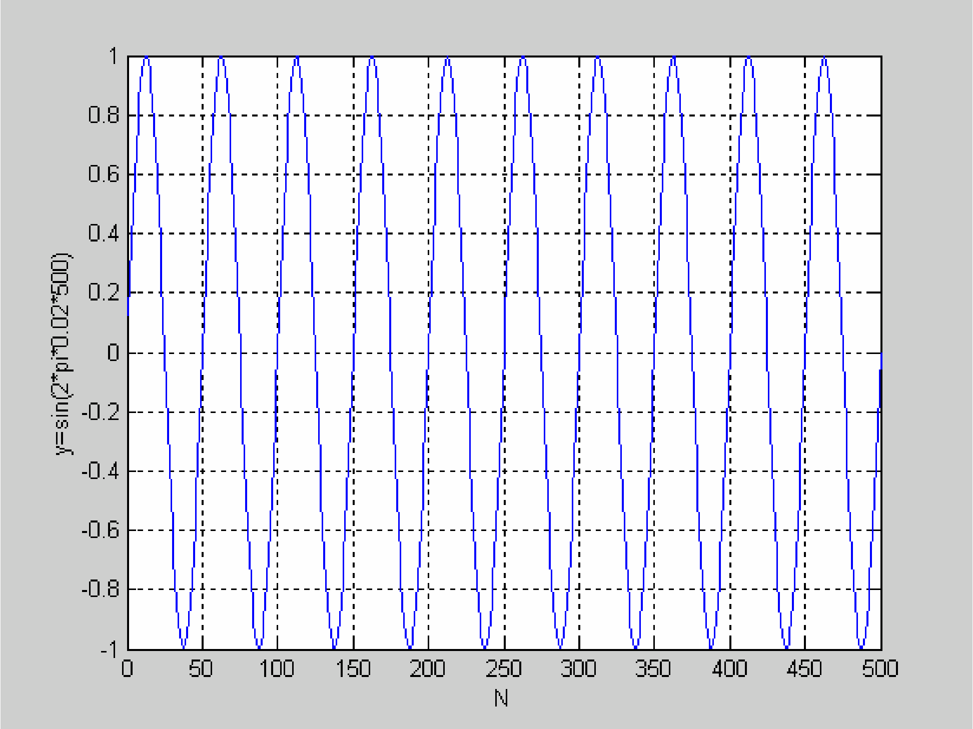


Figure 4.8 Input waveform: $\sin(2 * \pi * 0.02 * 500)$.

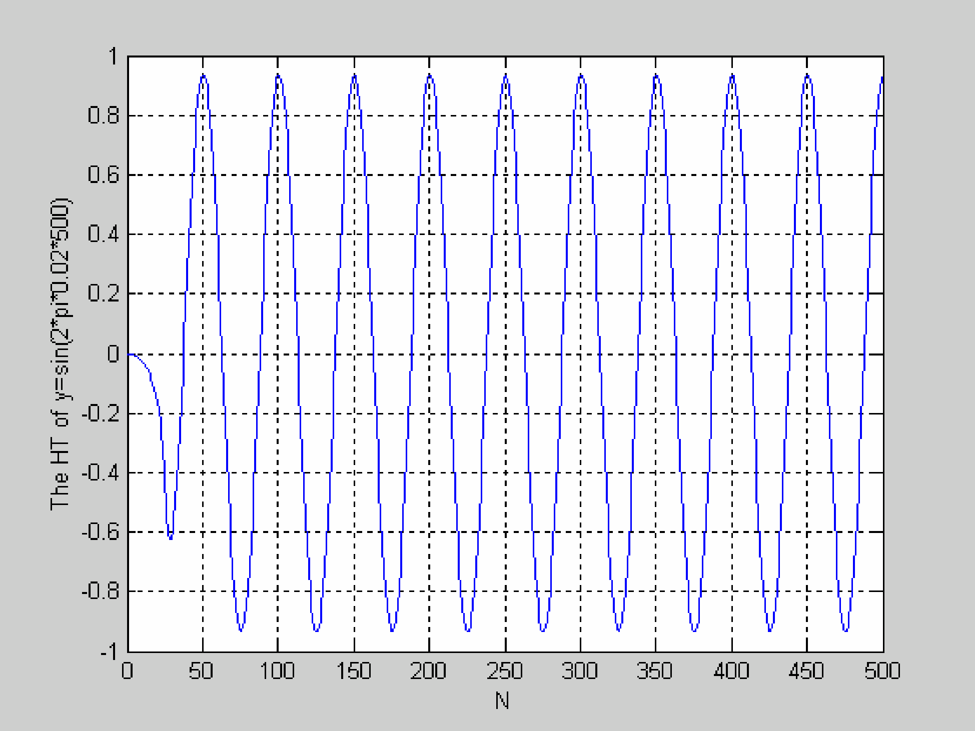


Figure 4.9 Output waveform, $M = 51$.

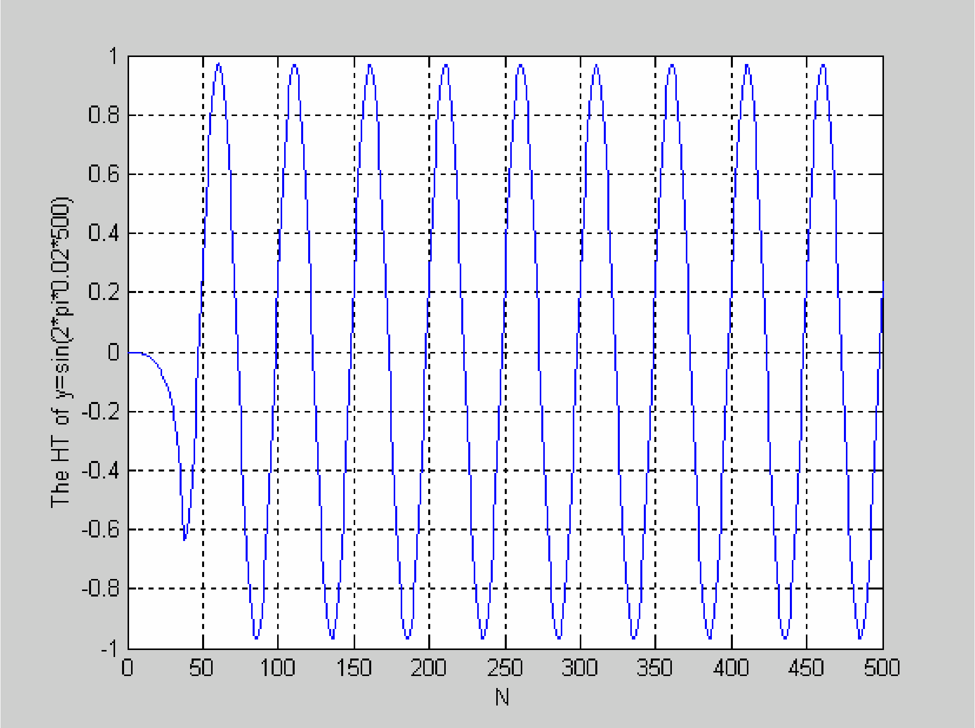


Figure 4.10 Output waveform, $M = 71$.

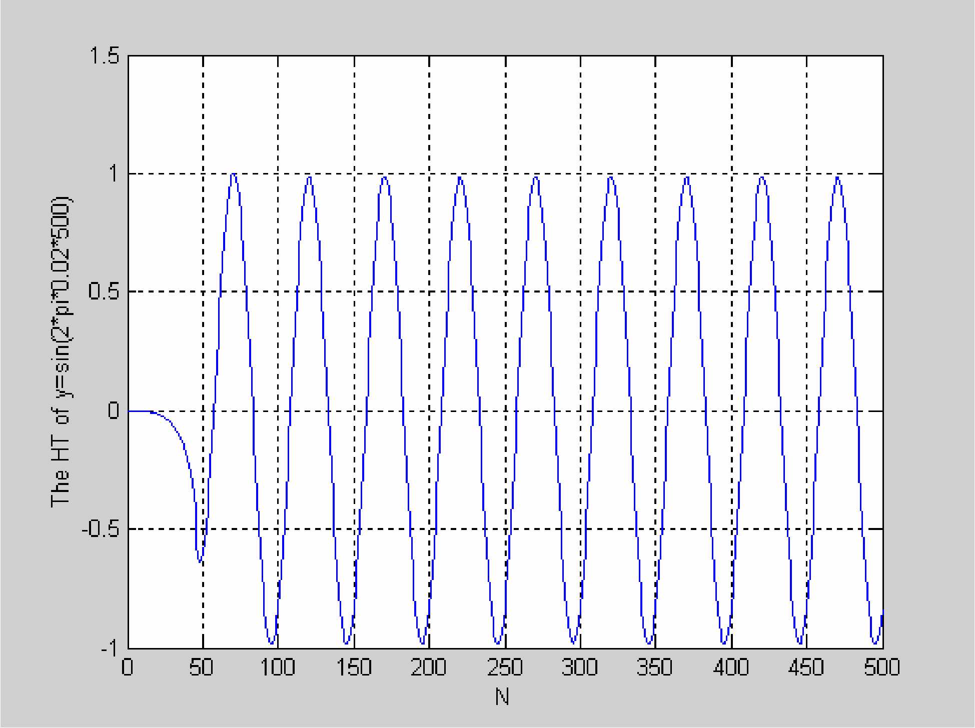


Figure 4.11 Output waveform, $M = 91$.

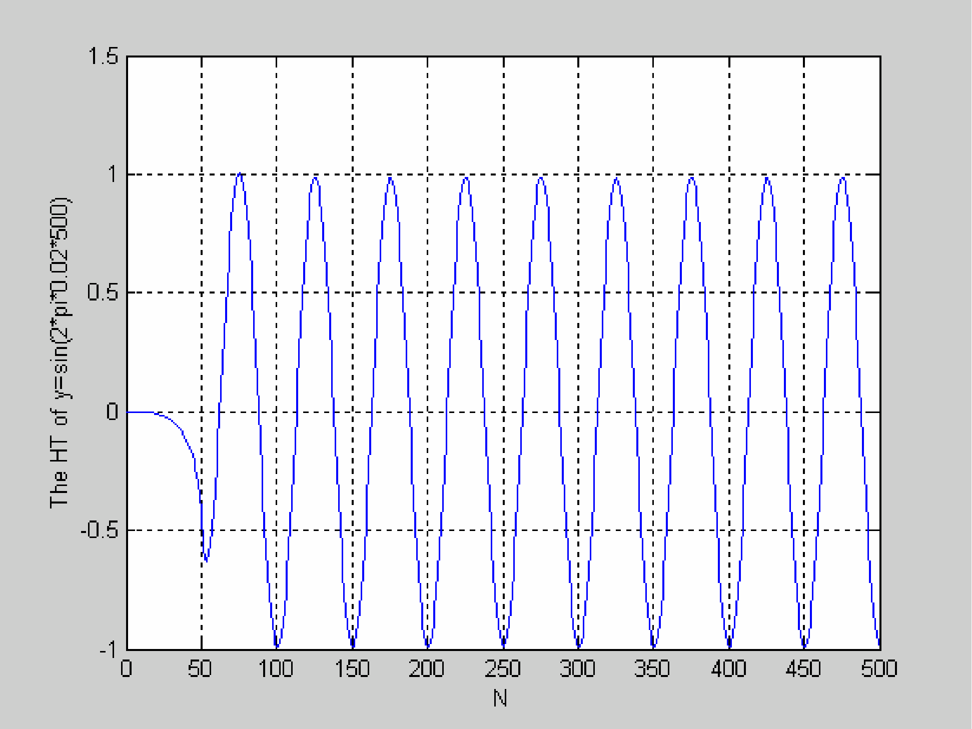


Figure 4.12 Output waveform, $M = 101$.

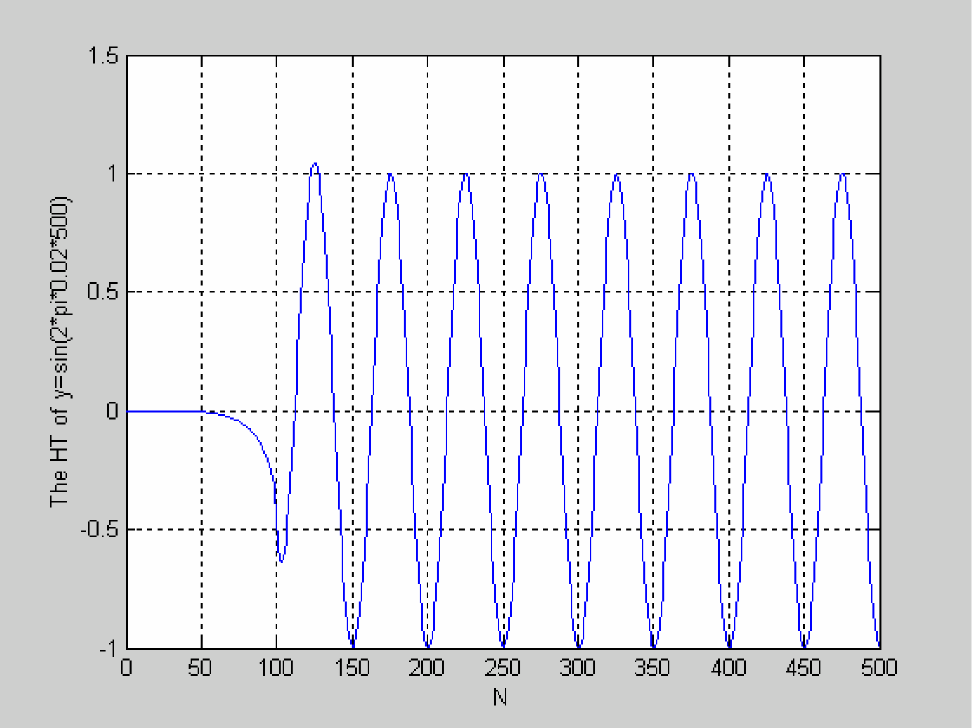


Figure 4.13 Output waveform, $M = 201$.

The group delay caused by the FIR filter with the anti-symmetric coefficients and order M is $\frac{M}{2}$. As result, the phase delay when the input signal $\sin(2*\pi*0.02*n)$ passes through the filter is $\frac{M}{2}*0.02*2\pi+\frac{\pi}{2}=(0.02*M+0.5)\pi$.

The output maximum and the minimum values were checked. The amplitude error and the phase delay for different order filters were also illustrated in Table 4.2. From the Table 4.2, it can be seen that the higher the order is, the less the error is when M is changed from 51 to 101. In order to not take long time for compute the Hilbert transform of the ECG wave, the filter order should be chosen properly.

Table 4.2 Filter order comparison

M	Maximum	Minimum	Error	Phase Delay
51	0.934020	-0.933397	<6.63%	1.52π
71	0.975858	-0.967608	<2.83%	1.92π
91	1.000212	-0.983314	<0.8%	2.32π
101	1.008626	-0.987704	<0.2%	2.52π
201	1.046485	-0.997695	<2.44%	4.52π

The frequency responses are shown in Figure 4.14 and Figure 4.15 when the order of the filter is 100 or 101.

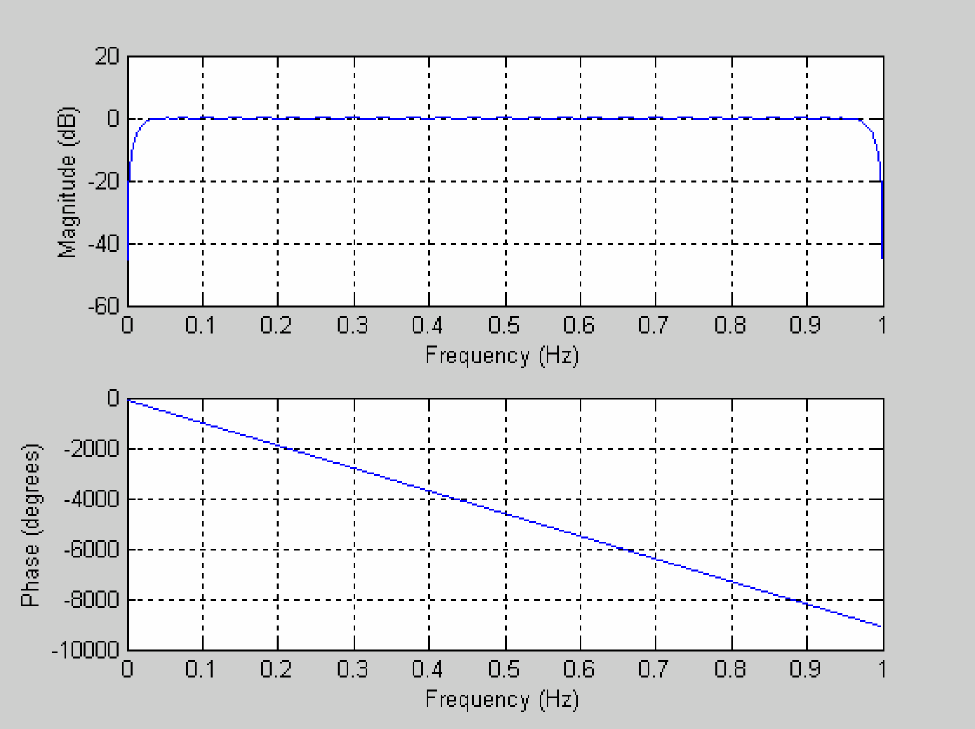


Figure 4.14 Frequency response, $M = 100$.

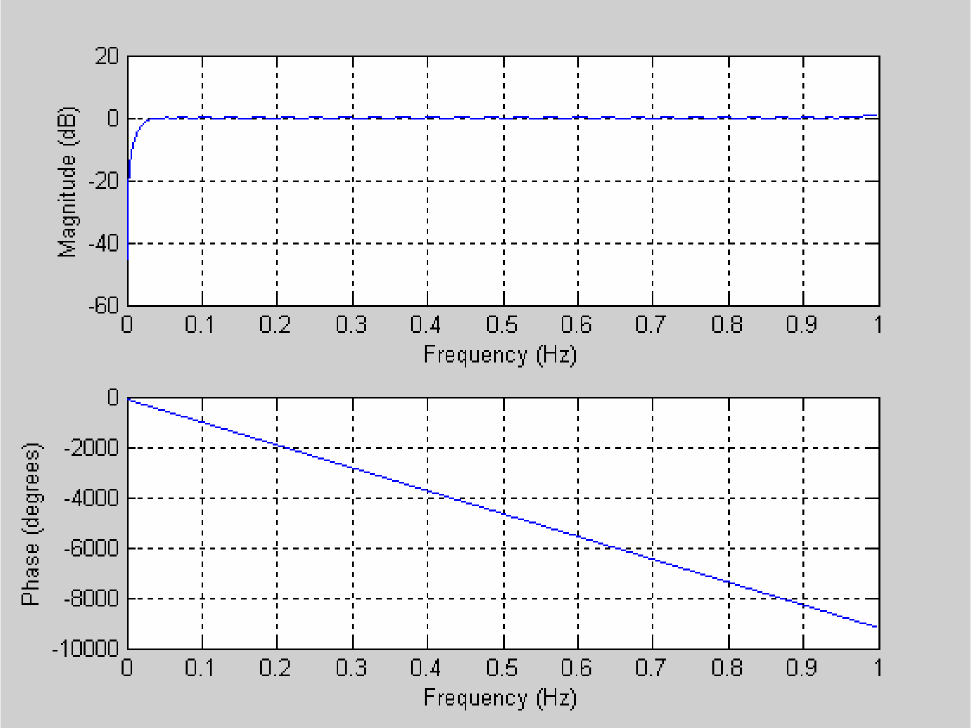


Figure 4.15 Frequency response, $M = 101$.

Based on the delay analysis above, the phase response is $-360*\frac{M}{2}*\frac{f}{2}-90$ degree for the normalized sampling frequency of 2. The slopes of the phase responses in Figure 4.14 and 4.15 are -9090 degree/Hz and -9180 degree/Hz, respectively.

From the Figure 4.9, Figure 4.10, Figure 4.11 and Figure 4.12 and Figure 4.13 it can be seen that the order of the filter must be bigger than 91 so that the amplitude of the output wave is ≈ 1. The phase shift depends on the filter order as shown above. In addition, in this thesis, the error requirement is to be smaller than 0.5%. From the Figure 4.14 and Figure 4.15, when the order is odd, the frequency response is not symmetrical (see the right hand side of Figure 4.14 and Figure 4.15). So the order of the filter in this thesis is determined as 100.

4.3.2 Filter Coefficients

1. When using MATLAB, the remez function:

$$b = remez(n, f, a, 'filtertype')$$

can be used to get the coefficients of the filter directly. That is

$$b = remez(100,[0.05\ 0.95],[1\ \ 1],'h'),$$

where

$n = 100$ is the filter order.

f $= [0.05\ 0.95]$ is a vector of pairs of normalized frequency point, specified in the range 0 to 1, where 1 corresponds to the Nyquist frequency. This frequency range is determined according to a few tests.

$a = [1\ 1]$ is a vector containing the desired amplitudes 1 at the points specified in $f = [0.05\ 0.95]$. It has the same length as f.

$'filtertype' = 'h'$ specifies that the filter is Hilbert transformer. This parameter allows specifying one of the following filters: Multiband, Differentiator, and Hilbert transformer.

Running a MATLAB program, Figure 4.16 shows that the calculated remez filter is an equi-ripple bandpass filter with a symmetrical magnitude response around $f = 0.5$. The coefficient b was obtained as shown in Figure 4.17.

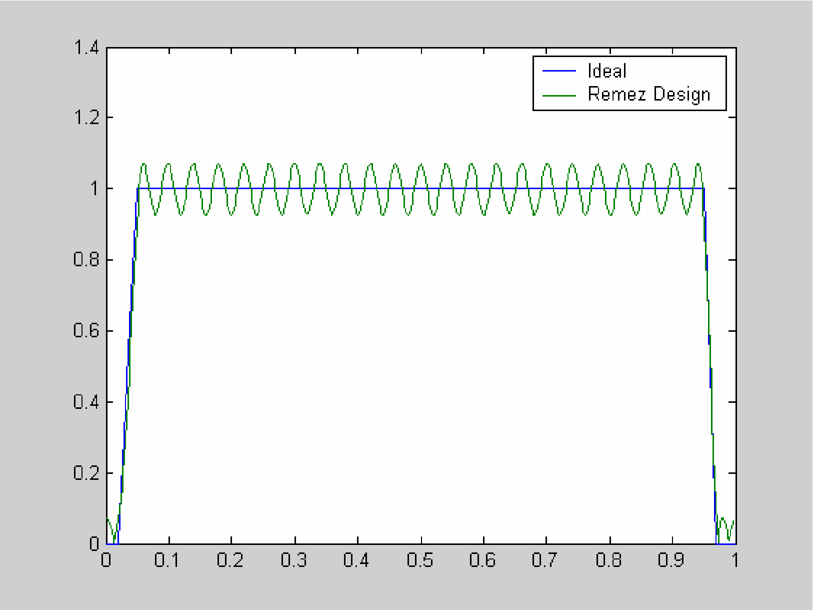
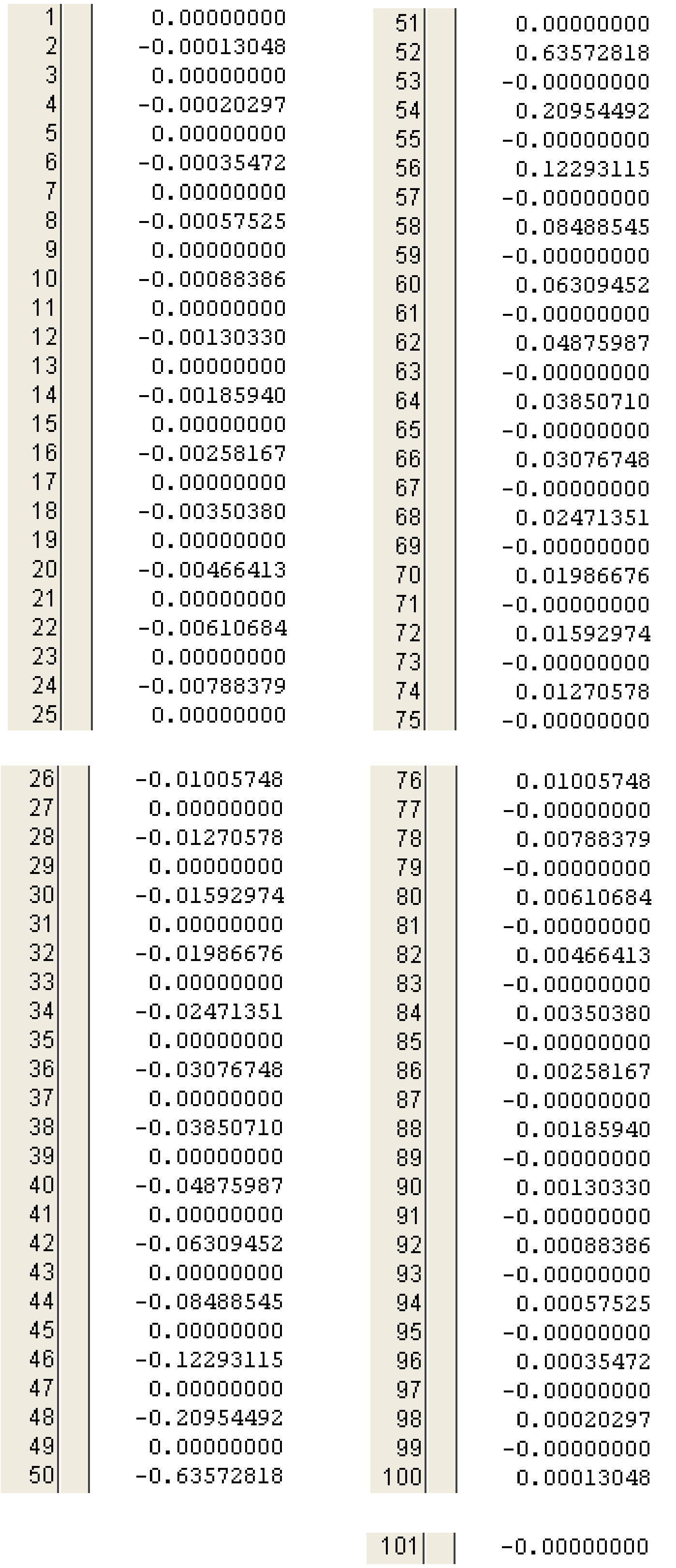


Figure 4.16 Frequency response of the ideal and Remez design filter.



1	0.00000000
2	-0.00013048
3	0.00000000
4	-0.00020297
5	0.00000000
6	-0.00035472
7	0.00000000
8	-0.00057525
9	0.00000000
10	-0.00088386
11	0.00000000
12	-0.00130330
13	0.00000000
14	-0.00185940
15	0.00000000
16	-0.00258167
17	0.00000000
18	-0.00350380
19	0.00000000
20	-0.00466413
21	0.00000000
22	-0.00610684
23	0.00000000
24	-0.00788379
25	0.00000000
26	-0.01005748
27	0.00000000
28	-0.01270578
29	0.00000000
30	-0.01592974
31	0.00000000
32	-0.01986676
33	0.00000000
34	-0.02471351
35	0.00000000
36	-0.03076748
37	0.00000000
38	-0.03850710
39	0.00000000
40	-0.04875987
41	0.00000000
42	-0.06309452
43	0.00000000
44	-0.08488545
45	0.00000000
46	-0.12293115
47	0.00000000
48	-0.20954492
49	0.00000000
50	-0.63572818
51	0.00000000
52	0.63572818
53	-0.00000000
54	0.20954492
55	-0.00000000
56	0.12293115
57	-0.00000000
58	0.08488545
59	-0.00000000
60	0.06309452
61	-0.00000000
62	0.04875987
63	-0.00000000
64	0.03850710
65	-0.00000000
66	0.03076748
67	-0.00000000
68	0.02471351
69	-0.00000000
70	0.01986676
71	-0.00000000
72	0.01592974
73	-0.00000000
74	0.01270578
75	-0.00000000
76	0.01005748
77	-0.00000000
78	0.00788379
79	-0.00000000
80	0.00610684
81	-0.00000000
82	0.00466413
83	-0.00000000
84	0.00350380
85	-0.00000000
86	0.00258167
87	-0.00000000
88	0.00185940
89	-0.00000000
90	0.00130330
91	-0.00000000
92	0.00088386
93	-0.00000000
94	0.00057525
95	-0.00000000
96	0.00035472
97	-0.00000000
98	0.00020297
99	-0.00000000
100	0.00013048
101	-0.00000000

Figure 4.17 Coefficients of the filter when $M = 100$.

From the output coefficients shown in the Figure 4.17, it can be seen that the Hilbert transformer has negative symmetry.

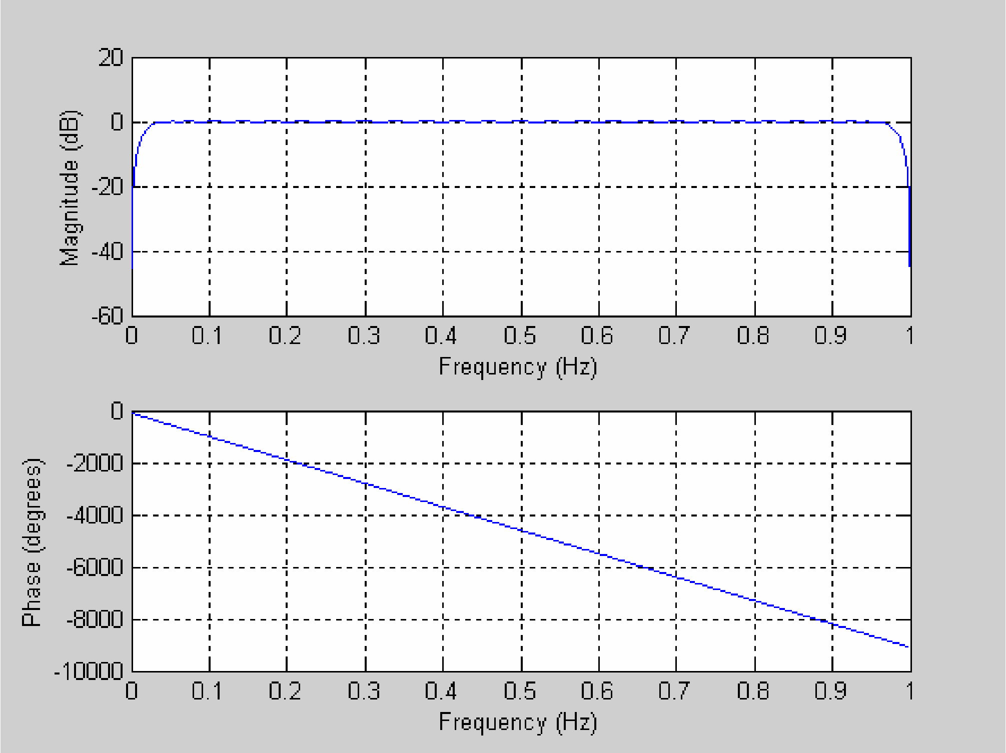


Figure 4.18 Frequency response of Remez filter.

Figure 4.18 demonstrates that the calculated filter has a symmetrical magnitude response around $f = 0.5$ and a linear phase response with the slope of -9090 degree/Hz.

2. Using C program *remez.c* to calculate the coefficients of the filter

This program uses the Remez exchange algorithm to design linear phase FIR digital filters with minimum weighted Chebyshev error in approximating a desired ideal frequency response [22]. The program has a special built-in section for specifying the more common ideal filter types such as multi-band, bandpass filters, Hilbert transform filters, and differentiators [23].

remez.c calculates the optimal FIR filter impulse response for a set of given band edges, the desired response and the weight on those bands. It includes a main program that handles the input, sets up the appropriate approximation problem and handles the output of the optimal filter coefficients.

Function

$$remez(h[], numtaps, numband, bands, desired[], weight[], type)$$

has input values $numtaps$, $numband$, $bands$, $desired[]$, $weight[]$, $type$ and output value $h[]$,

where

$h[]$ is the impulse response of the filter, i.e. the coefficients of the filter.

$numtaps$ is an integer. Specifying number of the filter coefficients. It should be $M+1$, M is the order of the filter.

$numband$ is an integer, specifying number of bands in filter specification.

$bands$ is a double variable, specifying user_specified band edges, using upper and lower cutoff frequencies. The bands array specifys the set F to be the form $F = UB_i$ where each frequency band B_i is a closed subinterval of the frequency axis [0, 1/2]. The number of $bands$ should be $2*numband$.

$desired[]$ is an array, which is the user_specified band responses, the desired frequency response in each band. The number of $desired[]$ should equal to the number of $bands$.

weight[] is an array, which is the user_specified error weights, a positive weight function in each band. The number of *weight*[] equals to the number of bands. The array *desired*[] and *weight*[] specify the ideal response and weight function in each band.

type is the type of the filter. It includes:

(a) Multi-band filter;

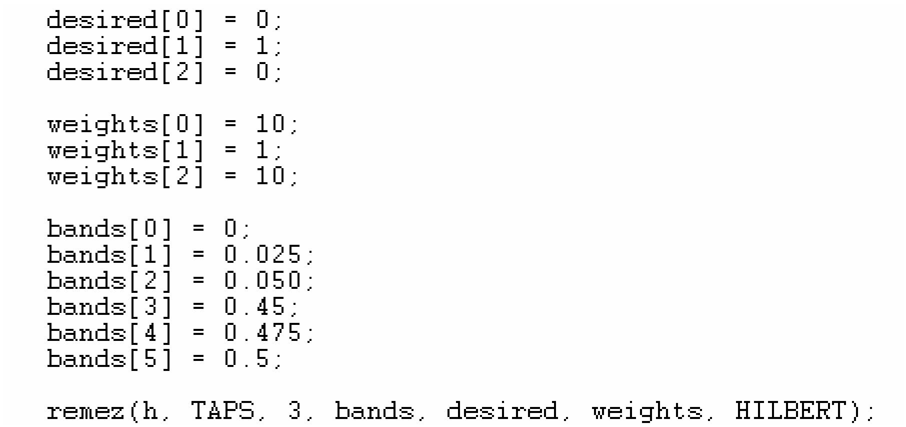
(b) Bandpass filters;

(c) Hilbert transform filters;

(d) Differentiators.

In this thesis, it has been explained that the order of the filter is $M = 100$, so the *numtaps* should be $M + 1 = 101$. The *numband* is 3. The *bands*, the *desired*[] and the *weight*[] is defined as shown in Figure 4.19. The *type* parameter is '*HILBERT*'. The output result *h*[] is saved in a text file “coefile.txt”.

To test program for the *remez*() function, the appropriate arguments to *remez*() was used to generate a filter. The initial parameters are shown in Figure 4.19. The resulting coefficients are shown as the Figure 4.20.

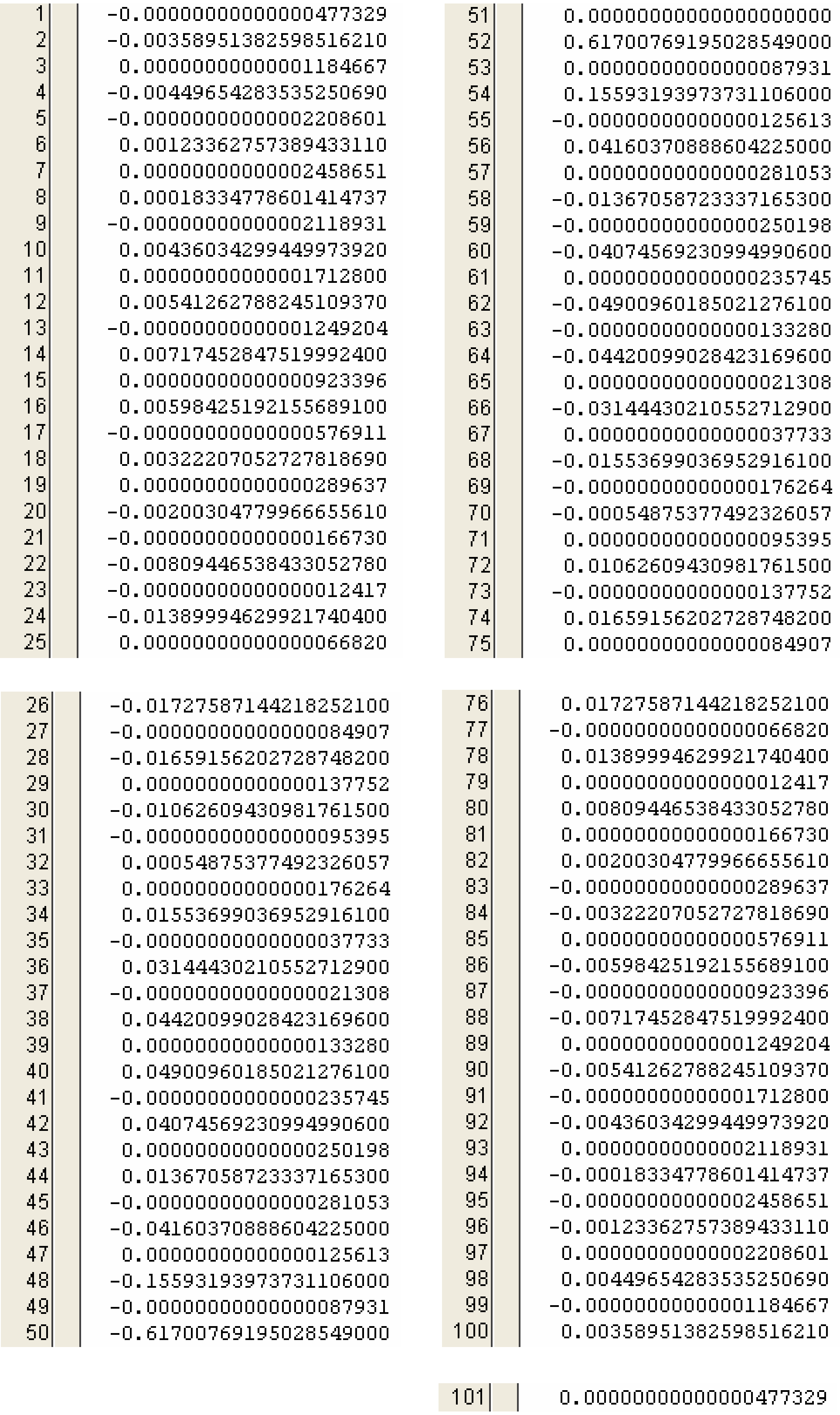


```
desired[0] = 0;
desired[1] = 1;
desired[2] = 0;

weights[0] = 10;
weights[1] = 1;
weights[2] = 10;

bands[0] = 0;
bands[1] = 0.025;
bands[2] = 0.050;
bands[3] = 0.45;
bands[4] = 0.475;
bands[5] = 0.5;

remez(h, TAPS, 3, bands, desired, weights, HILBERT);
```

Figure 4.19 Initial parameter of *remez.c* **.**

A C program was run in Visual C++ and the filter coefficients are obtained. They are shown in Figure 4.20. The frequency response of the filter is illustrated in Figure 4.21.

It can be noticed that the coefficients obtained from the C code are not exactly the same as those from MATLAB. Since the source code in MATLAB is not available, it is not possible to check the differences in the specific calculations. The frequency responses in Figure 4.18 and Figure 4.21 also show the differences. It looks like the response from C code is closer to the ideal filter response (Figure 4.16).



```
  1   -0.00000000000000477329         51    0.00000000000000000000
  2   -0.00358951382598516210         52    0.61700769195028549000
  3    0.00000000000001184667         53    0.00000000000000087931
  4   -0.00449654283535250690         54    0.15593193973731106000
  5   -0.00000000000002208601         55   -0.00000000000000125613
  6    0.00123362757389433110         56    0.04160370888604225000
  7    0.00000000000002458651         57    0.00000000000000281053
  8    0.00018334778601414737         58   -0.01367058723337165300
  9   -0.00000000000002118931         59   -0.00000000000000250198
 10    0.00436034299449973920         60   -0.04074569230994990600
 11    0.00000000000001712800         61    0.00000000000000235745
 12    0.00541262788245109370         62   -0.04900960185021276100
 13   -0.00000000000001249204         63   -0.00000000000000133280
 14    0.00717452847519992400         64   -0.04420099028423169600
 15    0.00000000000000923396         65    0.00000000000000021308
 16    0.00598425192155689100         66   -0.03144430210552712900
 17   -0.00000000000000576911         67    0.00000000000000037733
 18    0.00322207052727818690         68   -0.01553699036952916100
 19    0.00000000000000289637         69   -0.00000000000000176264
 20   -0.00200304779966655610         70   -0.00054875377492326057
 21   -0.00000000000000166730         71    0.00000000000000095395
 22   -0.00809446538433052780         72    0.01062609430981761500
 23   -0.00000000000000012417         73   -0.00000000000000137752
 24   -0.01389994629921740400         74    0.01659156202728748200
 25    0.00000000000000066820         75    0.00000000000000084907

 26   -0.01727587144218252100         76    0.01727587144218252100
 27   -0.00000000000000084907         77   -0.00000000000000066820
 28   -0.01659156202728748200         78    0.01389994629921740400
 29    0.00000000000000137752         79    0.00000000000000012417
 30   -0.01062609430981761500         80    0.00809446538433052780
 31   -0.00000000000000095395         81    0.00000000000000166730
 32    0.00054875377492326057         82    0.00200304779966655610
 33    0.00000000000000176264         83   -0.00000000000000289637
 34    0.01553699036952916100         84   -0.00322207052727818690
 35   -0.00000000000000037733         85    0.00000000000000576911
 36    0.03144430210552712900         86   -0.00598425192155689100
 37   -0.00000000000000021308         87   -0.00000000000000923396
 38    0.04420099028423169600         88   -0.00717452847519992400
 39    0.00000000000000133280         89    0.00000000000001249204
 40    0.04900960185021276100         90   -0.00541262788245109370
 41   -0.00000000000000235745         91   -0.00000000000001712800
 42    0.04074569230994990600         92   -0.00436034299449973920
 43    0.00000000000000250198         93    0.00000000000002118931
 44    0.01367058723337165300         94   -0.00018334778601414737
 45   -0.00000000000000281053         95   -0.00000000000002458651
 46   -0.04160370888604225000         96   -0.00123362757389433110
 47    0.00000000000000125613         97    0.00000000000002208601
 48   -0.15593193973731106000         98    0.00449654283535250690
 49   -0.00000000000000087931         99   -0.00000000000001184667
 50   -0.61700769195028549000        100    0.00358951382598516210

                                     101    0.00000000000000477329
```

Figure 4.20 Coefficients from the C program.

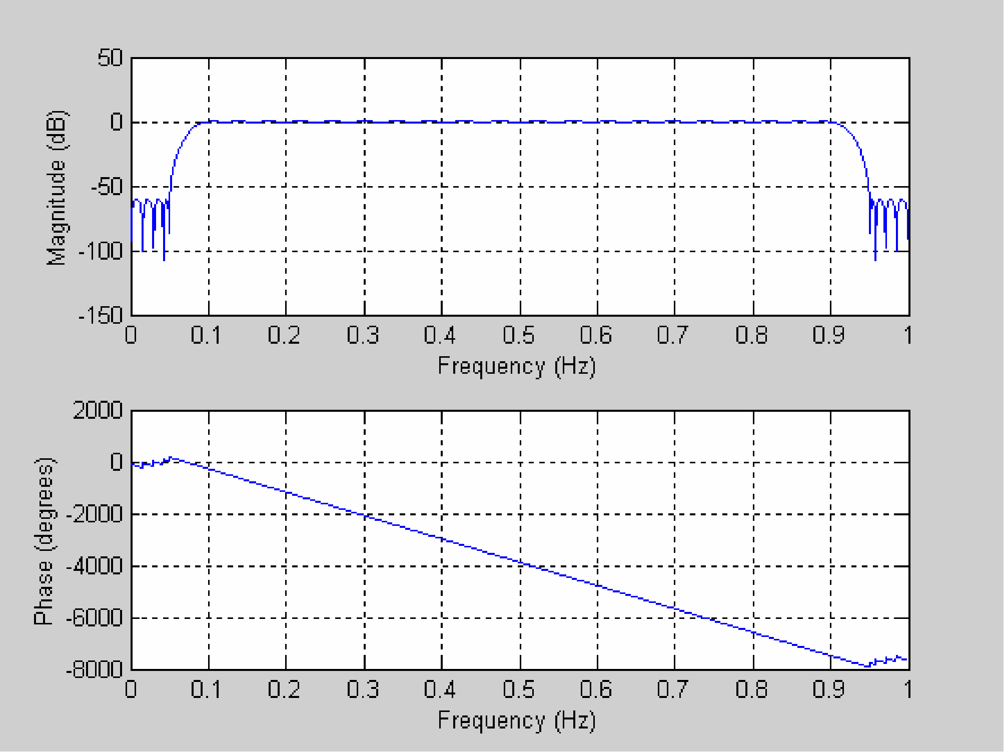


Figure 4.21 Frequency response for a $M = 100$ Hilbert transform filter.

This filter should also have a linear phase response as the one in Figure 4.18. However the phase response of the plot does not have the same slope in the stopband as one in the passband. It is caused by the calculation error because the transfer function values in the stopband are very small.

4.3.3 Digital Filter

According to the FIR filter definition, the function $fir\_filter(M, Coef, Xinput)$ was written in C program that is used to apply filtering operations on the data sequence in vector $Xinput$,

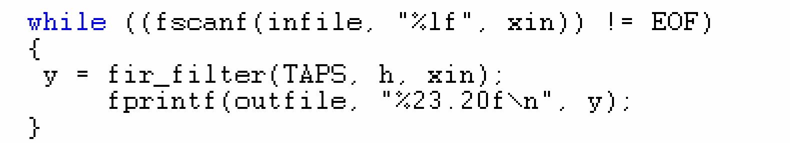
where

M is an integer. It is the number of coefficients of the filter.

Coef is the coefficients of the filter.

Xinput is the input data.

The following codes illustrate how to use this function.



```
while ((fscanf(infile, "%lf", xin)) != EOF)
{
 y = fir_filter(TAPS, h, xin);
     fprintf(outfile, "%23.20f\n", y);
}
```

where

"xin" contains the input data, it could be the ECG data.

"infile" is the file name which saved the input data. It is named by users.

"h" is the coefficients obtained from the *remez.c* .

"outfile" is the file name which saved the results filtered from the filter. It is named by user. In this case, the *outfile* saved the Hilbert transform of the input ECG data.

4.3.4 Detector for R Wave Peak

Accurate determination of the QRS complex, in particular accurate detection of the R wave peak, is essential in computer-based ECG analysis [24]. As described in the previous chapter, one of the properties of the Hilbert transform is that it is an odd function. That is to say that it will cross zero on the x-axis every time that there is an inflection point in the original waveform [24]. Similarly a crossing of the zero between consecutive positive and negative inflection points in the Hilbert transformed conjugate will be represented as a peak in its original waveform. Using this characteristic, a detector for determining the R wave peak in the input ECG waveform was developed.

The Hilbert transform of the ECG wave was obtained in section 4.3.3. The peaks in the Hilbert transform sequence $h(n)$ represent regions of high probability of finding R wave peaks. An adaptive threshold is used to locate the peaks in the $h(n)$ sequence. For finding the R wave peak accurately, a moving 1000 points window is used to subdivide the Hilbert transform $h(n)$ sequence. The RMS (Root Mean Square) value and the maximum amplitude in the present window are then calculated. The threshold in this window was determined according to the criteria below:

1. If (the RMS value) $\geq$ (18%*maximum amplitude) in the Hilbert transform sequence, the threshold is set up at (39%*maximum amplitude).
2. If (the present maximum amplitude) $\geq$ (2* previous maximum amplitude), the threshold will be (39%*previous maximum amplitude).
3. If (the RMS value) < (18%* maximum amplitude) in the Hilbert transform sequence, the threshold will be (1.6*RMS value).
4. If the two peaks in the $h(n)$ sequence are two close together, only one of them is the real R peak.

4.3.5 R Wave Peak Detection Test

In this section, two ECG waveforms taken from the MIT-BIH Arrhythmia database are used as the test signals.

The MIT-BIH Arrhythmia database consists of 48 records, each containing 30 minutes of two-channel ECG with heartbeat and rhythm annotations. The recordings were digitized at 360 samples per second per channel with 11-bit resolution over a 10 mV range. All samples are represented as positive numbers. The entire 30-minute record is annotated.

XW_1 and XW_2 are the normal ECG waves. They are the small sections of MIT213 from the MIT-BIH database. Figures 4.22 and 4.23 illustrate the input data XW_1 and its filtered version, i.e., the Hilbert transform of the input signal.

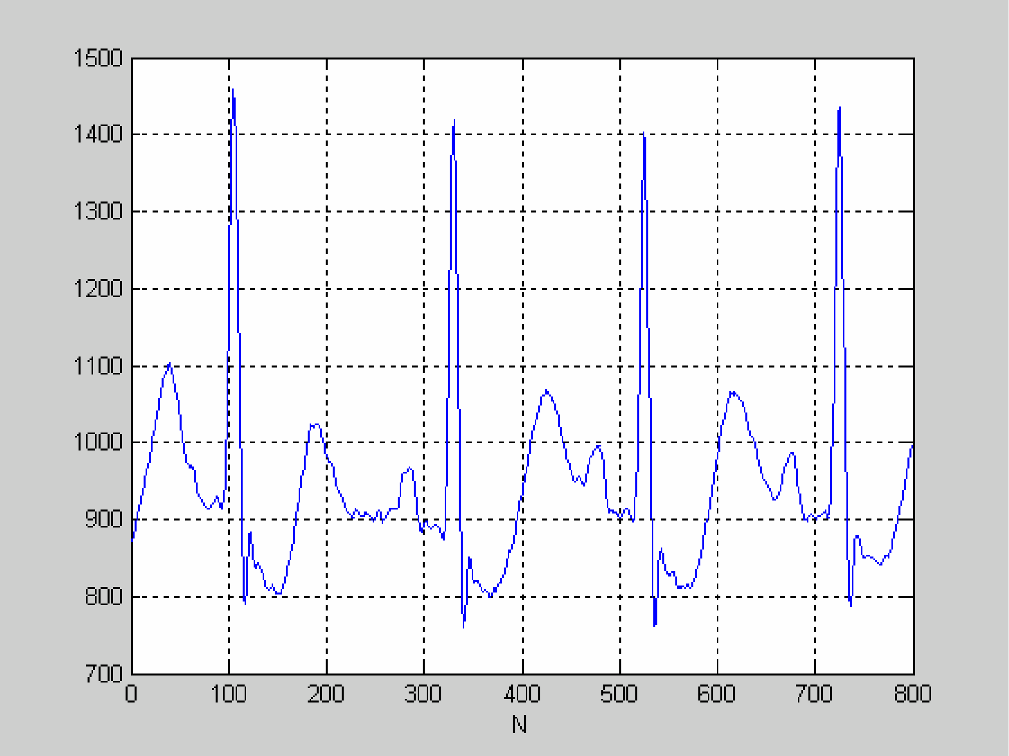


Figure 4.22 Input wavefom: XW_1.

As discussed in the previous chapter, the standard ECG is a representation of the heart electrical activity recorded from electrodes on the body face. Figure 4.22

shows a small section beats of a normal heart. There are 4 heartbeats since each R wave represents a heartbeat.

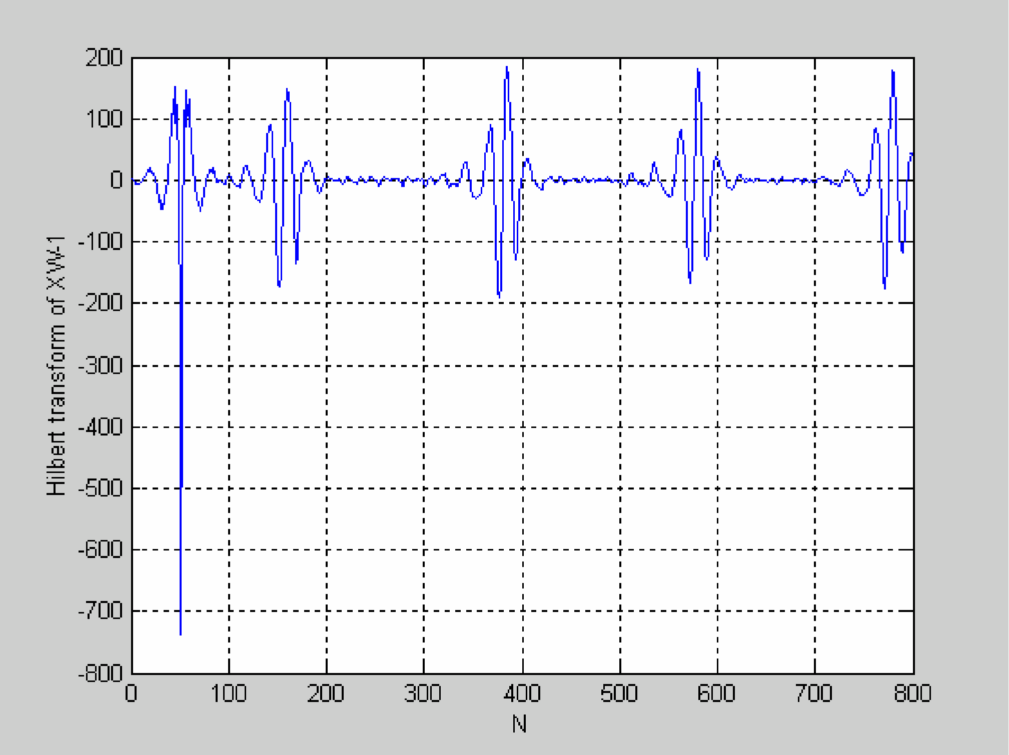


Figure 4.23 Output waveform for XW_1.

Table 4.3 XW_1 extreme points and values

Input (Figure 4.22)		Output (Figure 4.23)		Error
N	Extreme Value	N	Value	$(N_{output} - N_{Input} - 50)$
104	1459	155	≈ 0	1
329	1419	380	≈ 0	1
524	1404	575	≈ 0	1
724	1435	774	≈ 0	0

Figure 4.23 shows the Hilbert transform of the input wave, i.e. XW_1. For those points in Figure 4.22 where the slope changed from positive to negative, their outputs should be the zero crossing points with the extreme values changing from negative to positive. Because of the filter delay, the output results lag the input by 50 $(M/2)$. Table 4.3 shows N and extreme values corresponding to every heartbeat, i.e., the R wave peak. Compared the results with the MIT-BIH database annotation file, it can be seen that the output values are close.

Figure 4.24 and Figure 4.25 are for data XW_2.

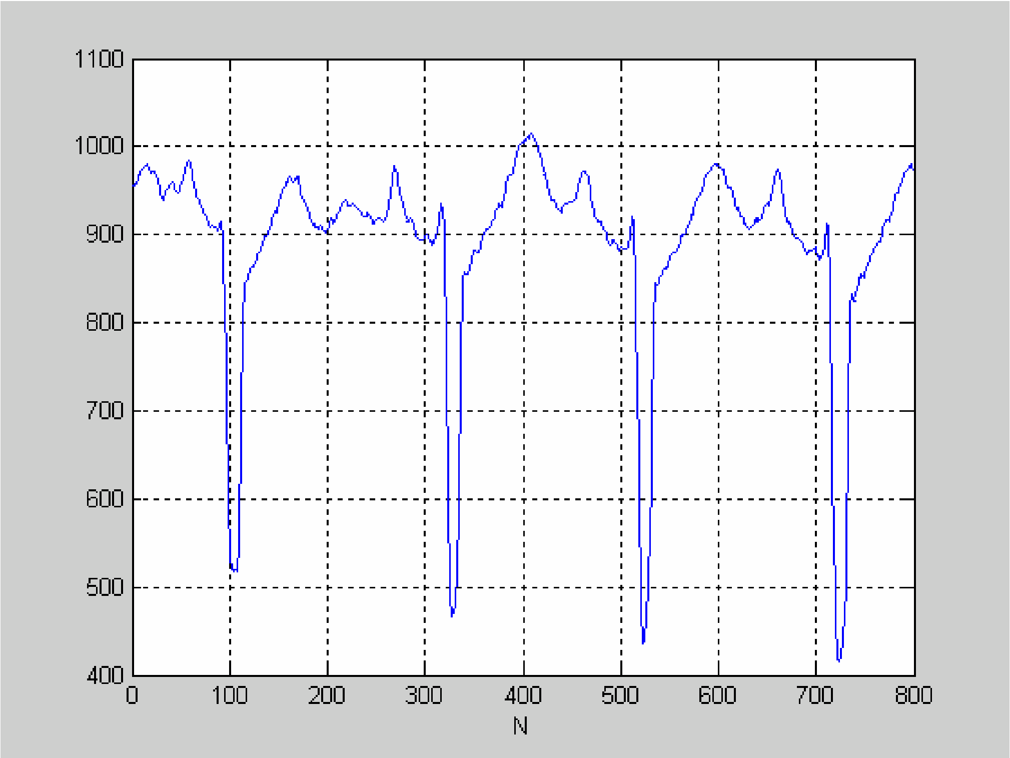


Figure 4.24 Input waveform: XW_2.

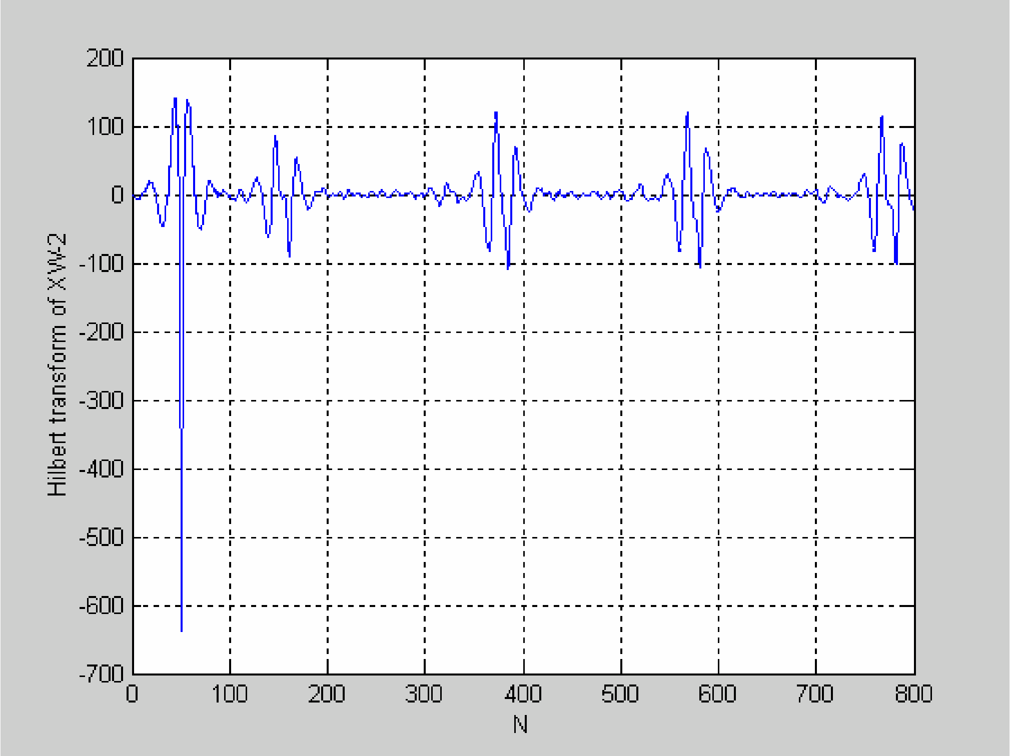


Figure 4.25 Output waveform for XW_2.

Figure 4.24 shows another channel waveform for the same person. The reason why Figure 4.24 is different from the Figure 4.22 is the electrode is placed at a different location. XW_1 is obtained when the electrode is placed at the front of the body while XW_2 is obtained when the electrode is placed on the back of the body. For this case, for those points which slope changed from negative to positive, the output point should be the zero crossing point with the extreme values changing from positive to negative.

Table 4.4 XW_2 extreme points and values

Input (Figure 4.24)		Output (Figure 4.25)		Error
N	Extreme Value	N	Value	$(N_{output} - N_{Input} - 50)$
103	518	152	≈ 0	-1
327	466	377	≈ 0	0
522	436	572	≈ 0	0
722	416	771	≈ 0	-1

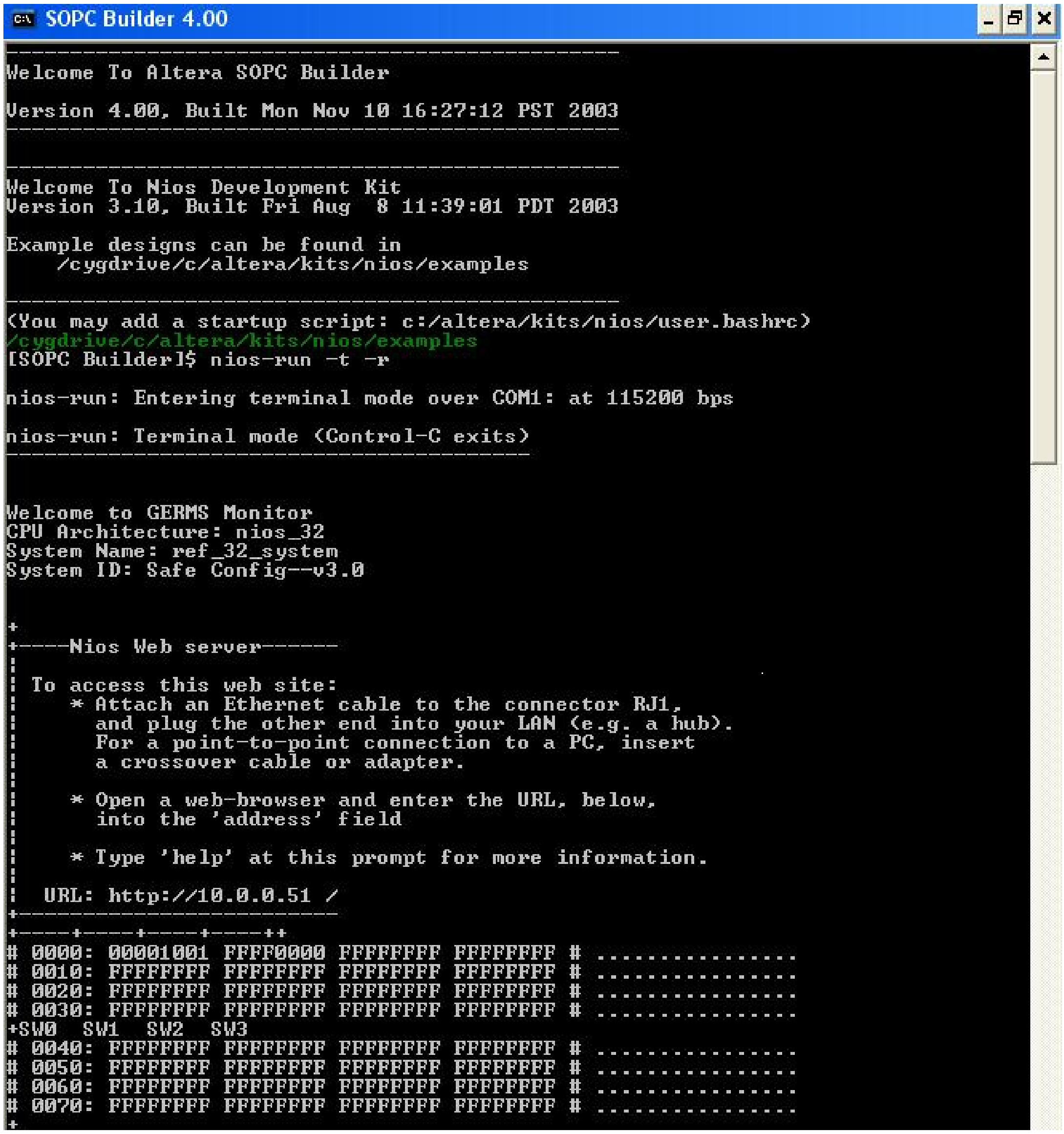
Table 4.4 shows the results, for the same reason with the previous wave, filter phase delay, the output wave lag the input by 50 ($M/2$). Comparing the output results with the MIT-BIH database annotation file, the error between the output results and the annotation results is acceptable.

4.4 Nios Implementation

In order to use the Nios development board [Figure 4.4], connect the host PC and open a Nios SDK shell and type

$$\$ \ nios - run \ -t \ -r$$

This establishes a simple terminal connection with the development board. Press the *SafeConfig* button on the Nios development board to reset the Nios development board and reconfigure the Stratix FPGA. The reference design emits a text message to the serial port when the Nios processor boots. After the LEDs begin to blink and the LED displays network-initialization status messages, press SW3 to abort DHCP network configuration. Text will display in the Nios SDK Shell window. Press the *Enter* key on the PC several times to provide stimulus to the reference design. The interface is shown in Figure 4.26.



```
------------------------------------------------
Welcome To Altera SOPC Builder

Version 4.00, Built Mon Nov 10 16:27:12 PST 2003
------------------------------------------------

------------------------------------------------
Welcome To Nios Development Kit
Version 3.10, Built Fri Aug  8 11:39:01 PDT 2003

Example designs can be found in
    /cygdrive/c/altera/kits/nios/examples

------------------------------------------------
(You may add a startup script: c:/altera/kits/nios/user.bashrc)
/cygdrive/c/altera/kits/nios/examples
[SOPC Builder]$ nios-run -t -r

nios-run: Entering terminal mode over COM1: at 115200 bps

nios-run: Terminal mode (Control-C exits)
-----------------------------------------


Welcome to GERMS Monitor
CPU Architecture: nios_32
System Name: ref_32_system
System ID: Safe Config--v3.0


+
+----Nios Web server------
|
| To access this web site:
|    * Attach an Ethernet cable to the connector RJ1,
|      and plug the other end into your LAN (e.g. a hub).
|      For a point-to-point connection to a PC, insert
|      a crossover cable or adapter.
|
|    * Open a web-browser and enter the URL, below,
|      into the 'address' field
|
|    * Type 'help' at this prompt for more information.
|
|  URL: http://10.0.0.51 /
+--------------------------
+-----+-----+-----+-----++
# 0000: 00001001 FFFF0000 FFFFFFFF FFFFFFFF # ................
# 0010: FFFFFFFF FFFFFFFF FFFFFFFF FFFFFFFF # ................
# 0020: FFFFFFFF FFFFFFFF FFFFFFFF FFFFFFFF # ................
# 0030: FFFFFFFF FFFFFFFF FFFFFFFF FFFFFFFF # ................
+SW0  SW1  SW2  SW3
# 0040: FFFFFFFF FFFFFFFF FFFFFFFF FFFFFFFF # ................
# 0050: FFFFFFFF FFFFFFFF FFFFFFFF FFFFFFFF # ................
# 0060: FFFFFFFF FFFFFFFF FFFFFFFF FFFFFFFF # ................
# 0070: FFFFFFFF FFFFFFFF FFFFFFFF FFFFFFFF # ................
+
```

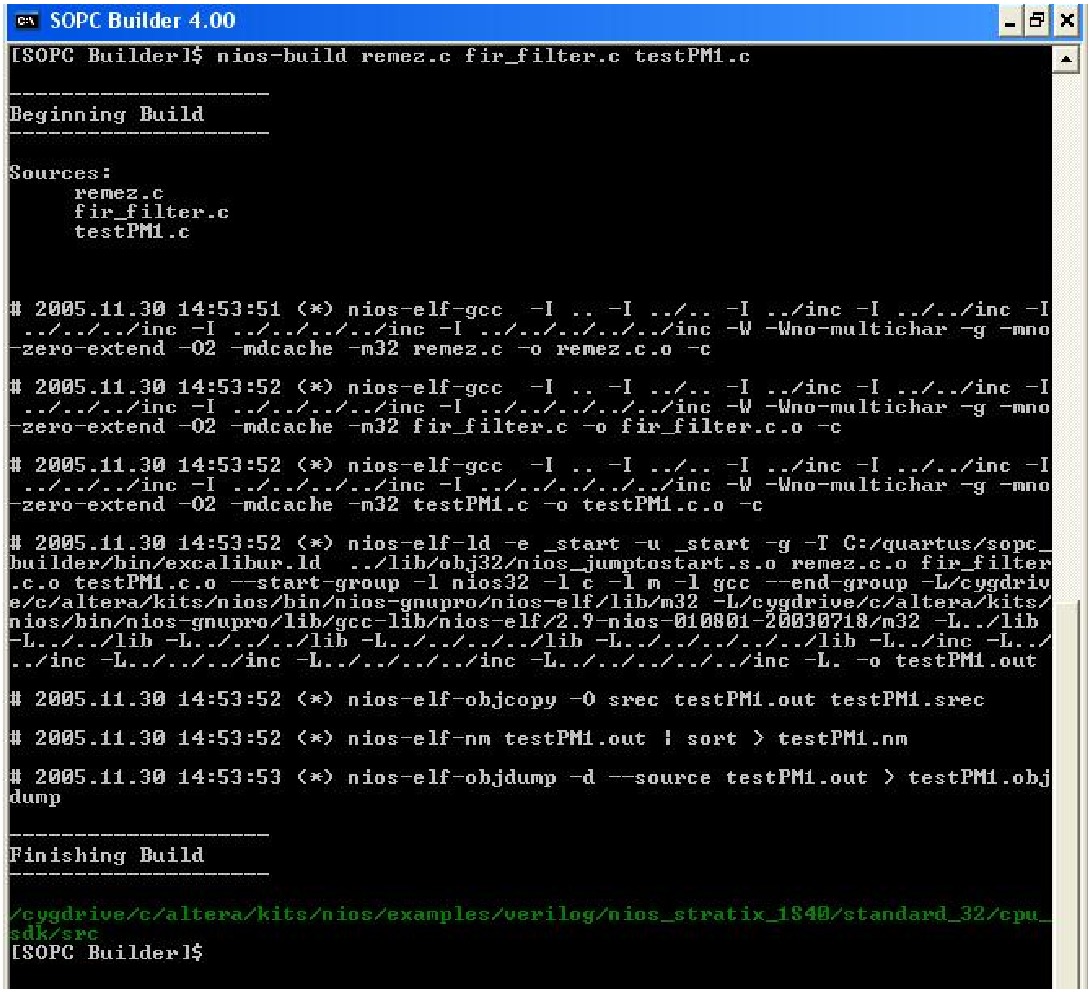
Figure 4.26 Nios SDK Shell Prompt.

If the activity in the Nios SDK shell looks like in Figure 4.26, then the PC is communicating correctly with the Nios development board. Press $Ctrl + C$ to exit the terminal program and return to the bash shell.

To compile all programs at the at the bash prompt, use command

$$\$ nios - build$$

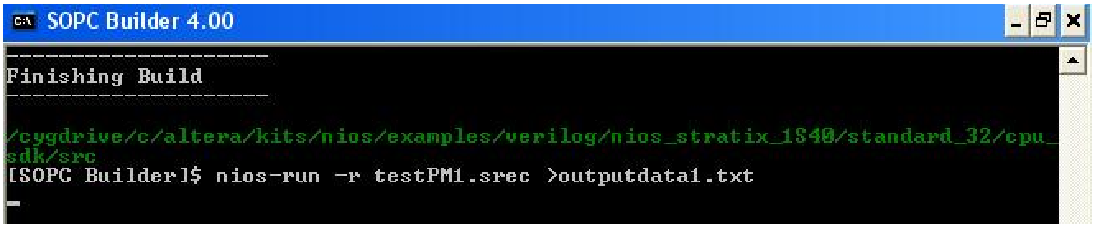
The GNU C/C++ compiler and linker will be invoked. Several intermediate files and an executable ($.srec$) file will be produced. The messages are shown in Figure 4.27.



```
[SOPC Builder]$ nios-build remez.c fir_filter.c testPM1.c

--------------------
Beginning Build
--------------------

Sources:
      remez.c
      fir_filter.c
      testPM1.c


# 2005.11.30 14:53:51 (*) nios-elf-gcc  -I .. -I ../.. -I ../inc -I ../../inc -I
 ../../../inc -I ../../../../inc -I ../../../../../inc -W -Wno-multichar -g -mno
-zero-extend -O2 -mdcache -m32 remez.c -o remez.c.o -c

# 2005.11.30 14:53:52 (*) nios-elf-gcc  -I .. -I ../.. -I ../inc -I ../../inc -I
 ../../../inc -I ../../../../inc -I ../../../../../inc -W -Wno-multichar -g -mno
-zero-extend -O2 -mdcache -m32 fir_filter.c -o fir_filter.c.o -c

# 2005.11.30 14:53:52 (*) nios-elf-gcc  -I .. -I ../.. -I ../inc -I ../../inc -I
 ../../../inc -I ../../../../inc -I ../../../../../inc -W -Wno-multichar -g -mno
-zero-extend -O2 -mdcache -m32 testPM1.c -o testPM1.c.o -c

# 2005.11.30 14:53:52 (*) nios-elf-ld -e _start -u _start -g -T C:/quartus/sopc_
builder/bin/excalibur.ld  ../lib/obj32/nios_jumptostart.s.o remez.c.o fir_filter
.c.o testPM1.c.o --start-group -l nios32 -l c -l m -l gcc --end-group -L/cygdriv
e/c/altera/kits/nios/bin/nios-gnupro/nios-elf/lib/m32 -L/cygdrive/c/altera/kits/
nios/bin/nios-gnupro/lib/gcc-lib/nios-elf/2.9-nios-010801-20030718/m32 -L../lib
-L../../lib -L../../../lib -L../../../../lib -L../../../../../lib -L../inc -L../
../inc -L../../../inc -L../../../../inc -L../../../../../inc -L. -o testPM1.out

# 2005.11.30 14:53:52 (*) nios-elf-objcopy -O srec testPM1.out testPM1.srec

# 2005.11.30 14:53:52 (*) nios-elf-nm testPM1.out | sort > testPM1.nm

# 2005.11.30 14:53:53 (*) nios-elf-objdump -d --source testPM1.out > testPM1.obj
dump

--------------------
Finishing Build
--------------------

/cygdrive/c/altera/kits/nios/examples/verilog/nios_stratix_1S40/standard_32/cpu_
sdk/src
[SOPC Builder]$
```

Figure 4.27 Nios-Build messages.

To download and run the code compiled in Figure 4.27, download the *srec* file to the Nios development board. Redirect stdout to a data file *outputdata*1*.txt* . The message is shown in Figure 4.28.



```
---------------------
Finishing Build
---------------------

/cygdrive/c/altera/kits/nios/examples/verilog/nios_stratix_1S40/standard_32/cpu_
sdk/src
[SOPC Builder]$ nios-run -r testPM1.srec >outputdata1.txt
_
```

Figure 4.28 Nios SDK shell prompt.

The input waveforms XW_1 and XW_2 are a section wave of MIT 213 taken from the MIT_BIH Arrhythmia database (Figure 4.29 and Figure 4.31). The output file record the data of the Hilbert transform of the ECG wave. The output waveforms are shown in the Figure 4.30 and Figure 4.32.

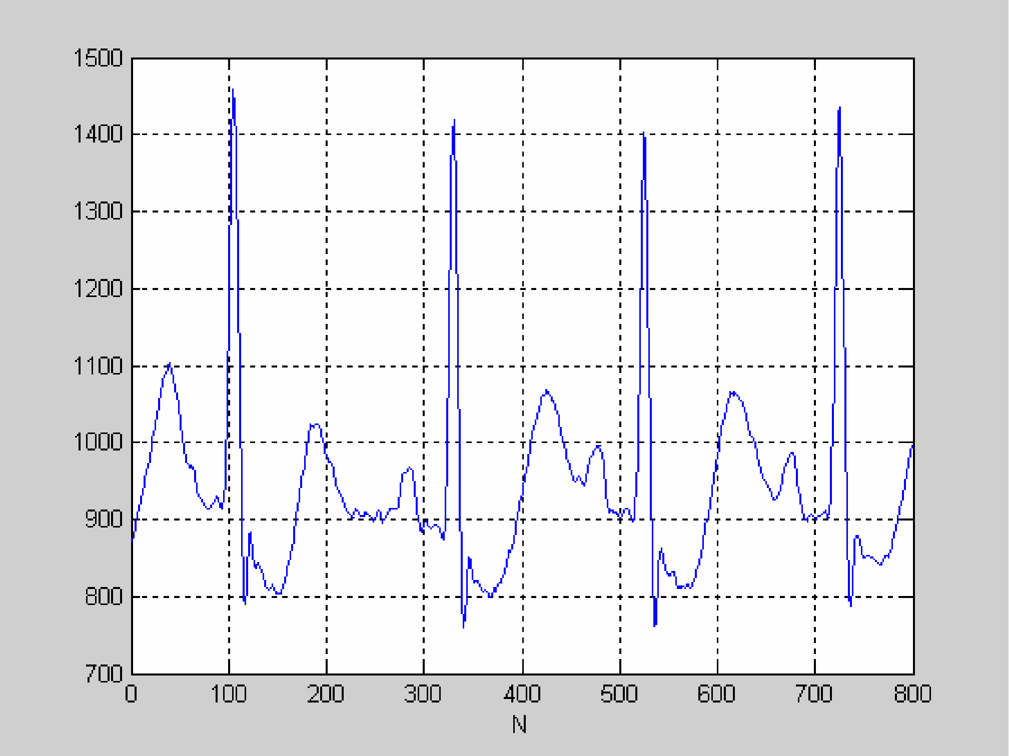


Figure 4.29 Input ECG waveform: XW_1.

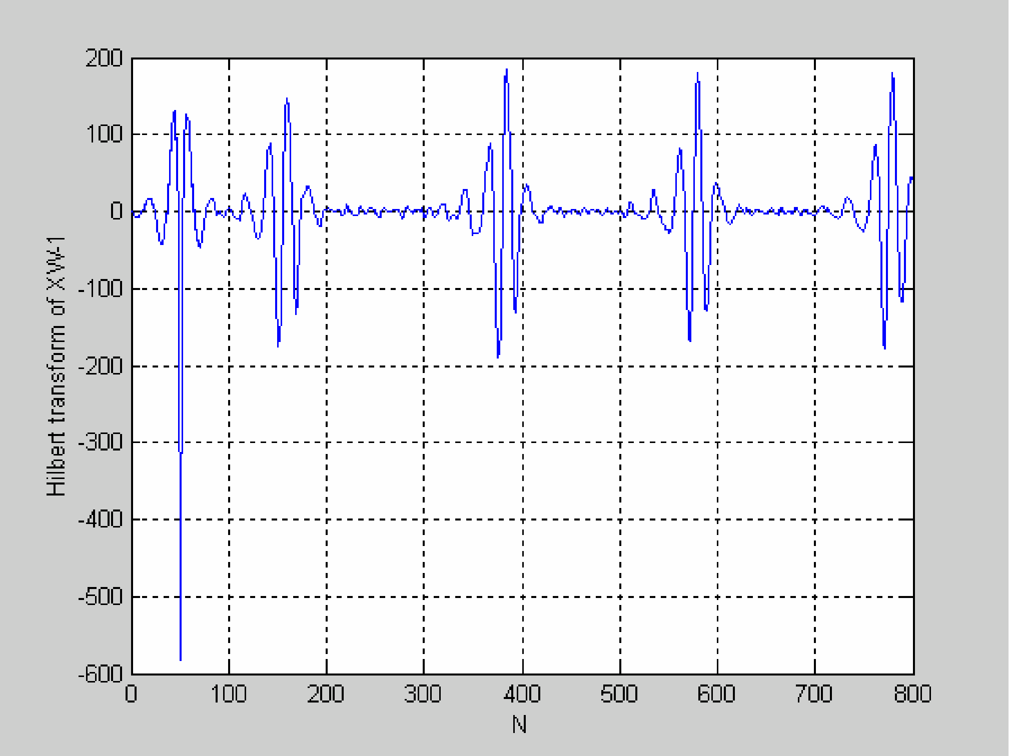


Figure 4.30 The Hilbert transform of XW_1.

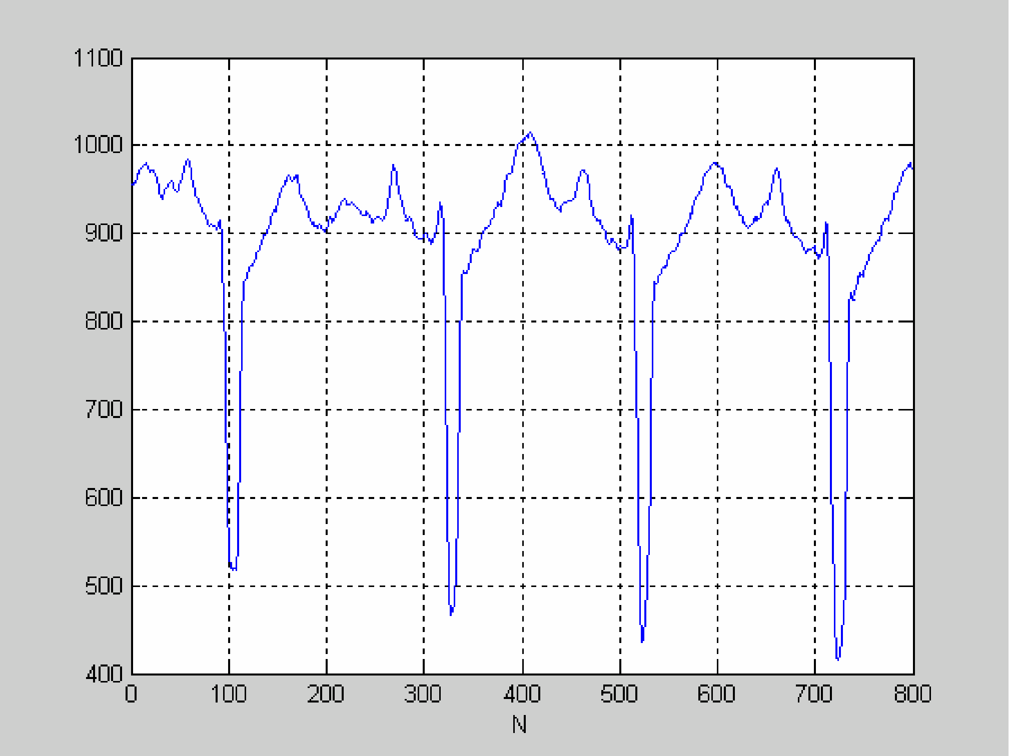


Figure 4.31 Input ECG waveform: XW_2.

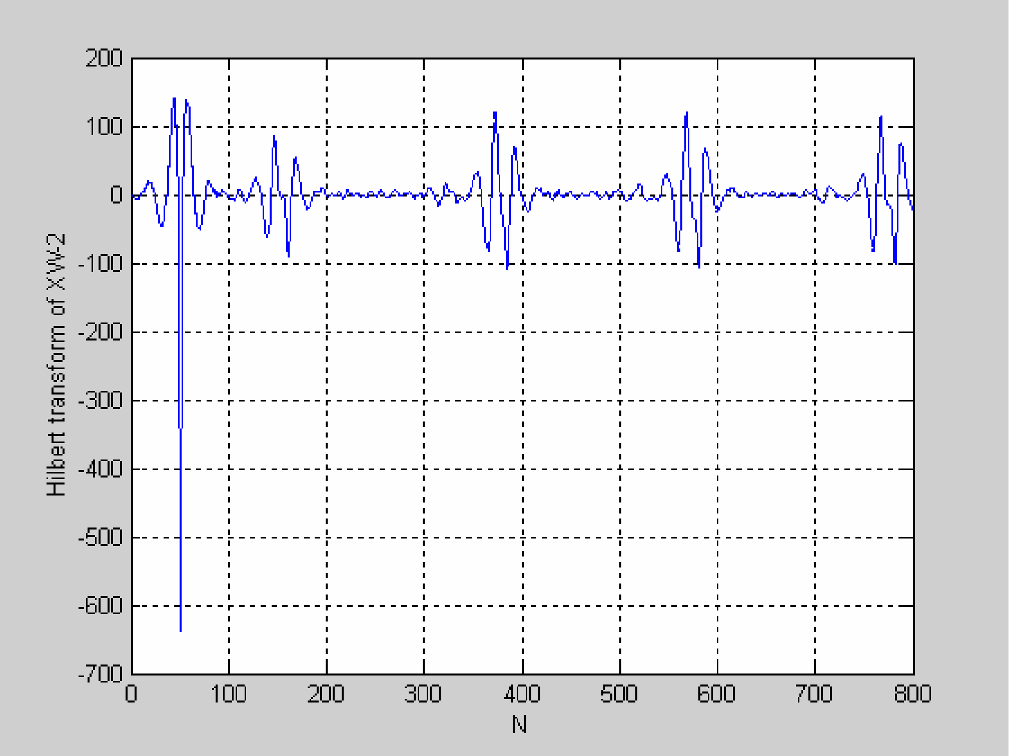


Figure 4.32 The Hilbert transform of XW_2.

From the output waveforms in Figure 4.30 and Figure 4.32, it can be seen that the waveforms are almost the same as the results obtained from Visual C++.

Table 4.5 XW_1 R wave points and their Hilbert transform points

Input (Figure 4.29)		Output (Figure 4.30)		Error
N	Extreme Value	N	Value	$(N_{output} - N_{Input} - 50)$
104	1459	155	≈ 0	1
329	1419	380	≈ 0	1
524	1404	575	≈ 0	1
724	1435	774	≈ 0	0

Table 4.6 XW_2 R wave points and their Hilbert transform points

Input (Figure 4.31)		Output (Figure 4.32)		Error
N	Extreme Value	N	Value	$(N_{output} - N_{Input} - 50)$
103	518	152	≈ 0	-1
327	466	377	≈ 0	0
522	436	572	≈ 0	0
722	416	771	≈ 0	-1

Table 4.5 and Table 4.6 show the N values of the R wave points and their Hilbert transform points. It’s the same as the one discussed in the section 4.2. The results illustrates that the programs can work correctly on the Nios embedded processor.

Chapter 5 Results

In the previous chapter, implementation of the algorithm for the Hilbert transform of an ECG wave has been discussed. In order to prove the program is working well in the Nios processor, some ECG waves taking from the MIT-BIH Arrhythmia database were used as verifications. The results will be compared with the annotation files recorded in the MIT-BIH database.

5.1 Experimental Results

Further examples of some ECG waves and their Hilbert transformed output waveforms are shown in this part.

The file $mit212from100.txt$ is a small section from MIT212 recorded in the MIT-BIH arrhythmia database. It includes 1000 data samples from 100 to1099 in MIT212. Figure 5.1 shows the original ECG waveform. The output, i.e., the Hilbert transform of $mit212from100.txt$ is shown in Figure 5.2. The file $mit212\_1from47200.txt$ is another section from the MIT212. It also includes 1000 data samples. The original waveform and the Hilbert transform of this ECG wave are shown in Figure 5.3 and Figure 5.4.

As discussed preciously, accurate detection of the R wave peaks is important in ECG analysis. The zero crossing points corresponding to the true R wave peak in the Hilbert transformed data of the original ECG waveform are shown in Table 5.1 and Table 5.2. The actual position of R wave points in the MIT-BIH arrhythmia database is also given. The error between the located R and actual R is calculated using Equation 5.1:

$$Error = N_{output} - N_{MIT-BIH} - 50\,, \tag{5.1}$$

where

N_{ouput} is the number of the zero crossing point in the Hilbert transform sequence of the original ECG data located by detector.

$N_{MIT-BIH}$ is the actual R wave peak location recorded in the MIT-BIH annotation file.

Since the order of the filter used in this thesis is 100, the output waveform lagged the input waveform 50.

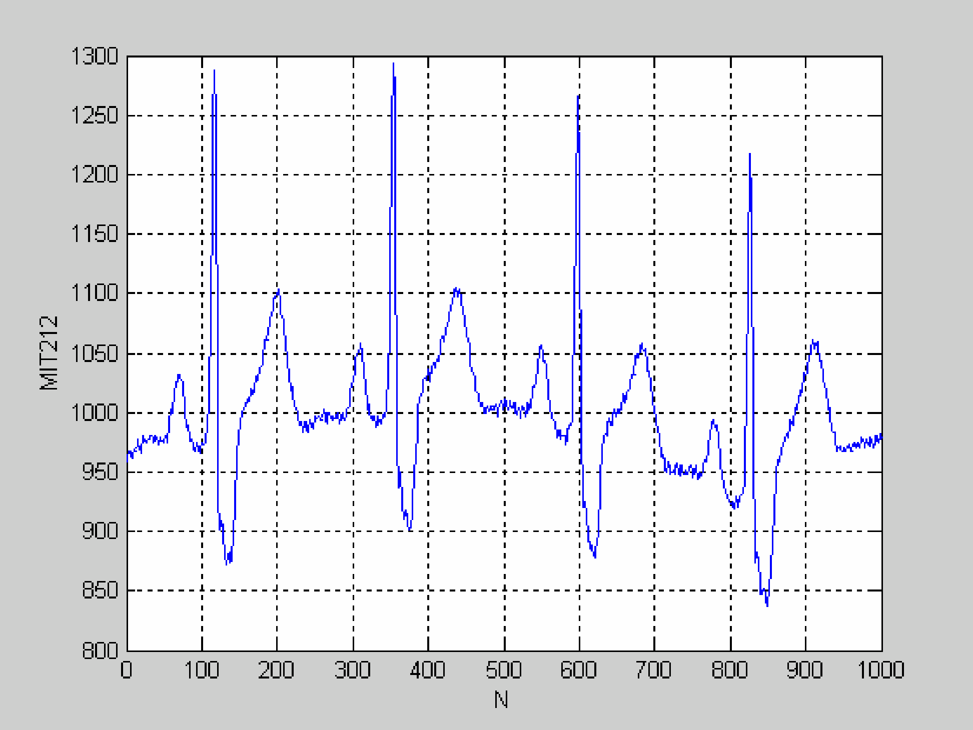


Figure 5.1 Input waveforms: mit212from100.txt.

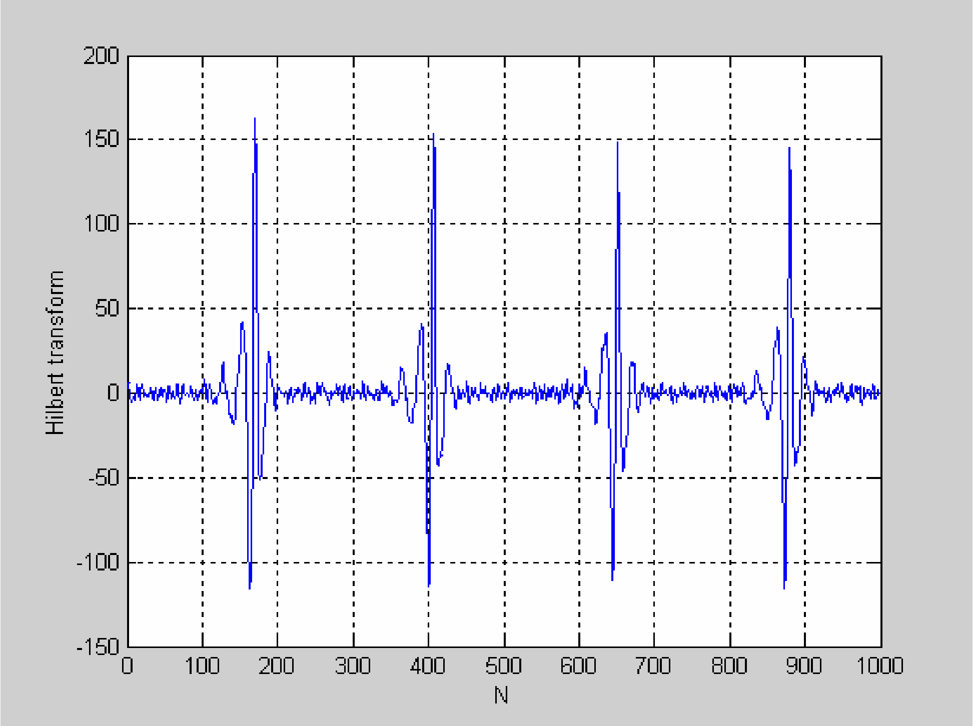


Figure 5.2 Output waveform: outMIT212from100.txt.

Table 5.1 mit212from100 extreme points and values

Input		Output		MIT-BIH Value	Error
N	**Extreme Value**	**N**	**Value**		
116	1287	165	≈ 0	114N	1
353	1294	402	≈ 0	351N	1
597	1265	647	≈ 0	596N	1
825	1218	875	≈ 0	824N	1

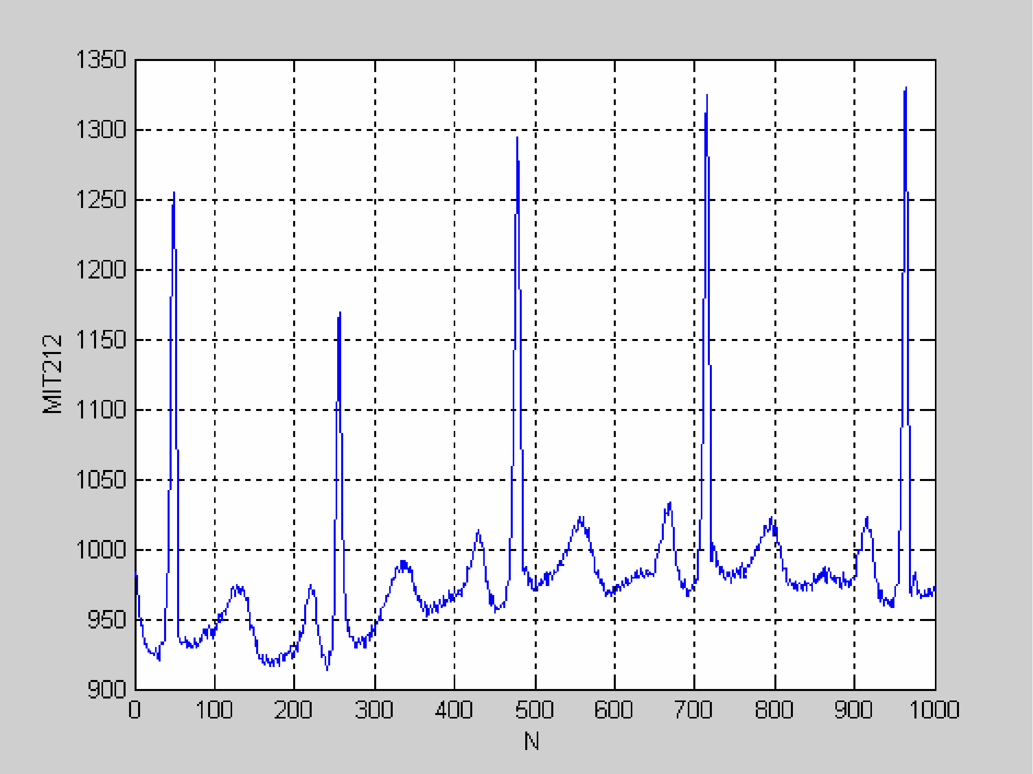


Figure 5.3 Input waveform: mit212_1from47200.txt.

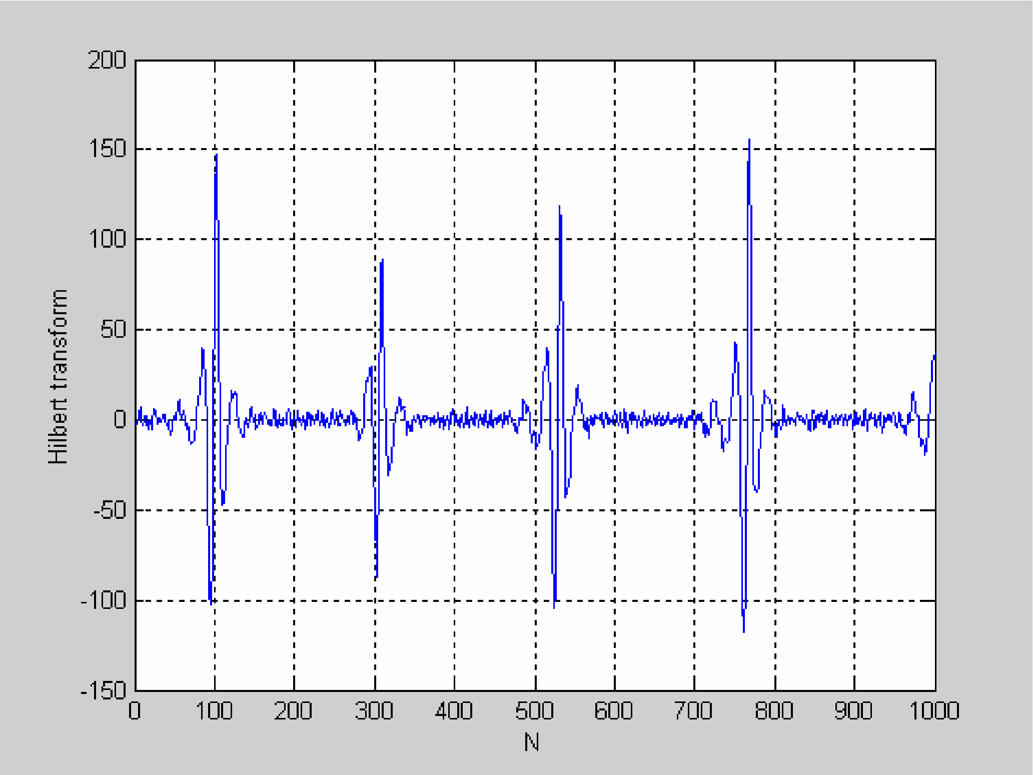


Figure 5.4 Output waveform: outMIT212from47200.txt.

Table 5.2 mit212_1from47200 extreme points and values

Input		**Output**		**MIT-BIH Value**	**Error**
N	**Extreme Value**	**N**	**Value**		
48	1255	97	≈ 0	46N	1
255	1169	304	≈ 0	254N	0
477	1294	527	≈ 0	476N	1
714	1324	763	≈ 0	713N	0
963	1330				

The ECG waveforms (Figure 5.1 and Figure 5.3) are normal because the P waves, QRS complexes, ST segments, T waves and U waves are normal. Each R wave peak presents a heartbeat.

Table 5.1 and Table 5.2 show the positions of the zero crossing points in the output waveform which corresponding to the R peaks in the input ECG wave. Because of the filter delay, the output results lag the input by 50 $(M/2)$. From Table 5.1 and Table 5.2, it can be seen that the N in the output waveform is lagging about 50. The error between the zero crossing points obtained in the output waves with the value recorded in the MIT-BIH annotation file is very small. It proves that the algorithm for calculating the Hilbert transform of the original ECG wave is correct. Note in the Table 5.1 and Table 5.2, the meaning of the letter in the "MIT-BIH value" is the type of the beat recorded in the MIT-BIH annotation file. N means normal QRS, V means premature ventricular contraction (PVC). F means fusion PVC. The details about the MIT-BIH annotation code are shown in the Appendix A.

The files $mit213\_1from67000.txt$ and $mit213\_1from1450.txt$ are small sections from another record, MIT213, in the MIT-BIH arrhythmia database. Figure 5.5 and Figure 5.7 show the ECG waveform. The outputs, i.e., the Hilbert transform of the input wave, were illustrated in Figure 5.6 and Figure 5.8. It can be seen there is an irregular heartbeat between $N = 600$ and 700 in Figure 5.5. This kind of beat is called a PVC (Premature Ventricular Contraction). PVCs are premature heartbeats originating from the ventricals of heart. PVCs are early or extra heartbeats that commonly occur and are usually harmless in normal hearts, but can cause problems in hearts with pre-exiting disease. A person with PVCs may or may not feel the irregular heartbeat, usually as a skip heartbeat [25]. The characteristic of PVCs is that there is no P wave and PR interval and the QRS complex is greater than 0.12s.

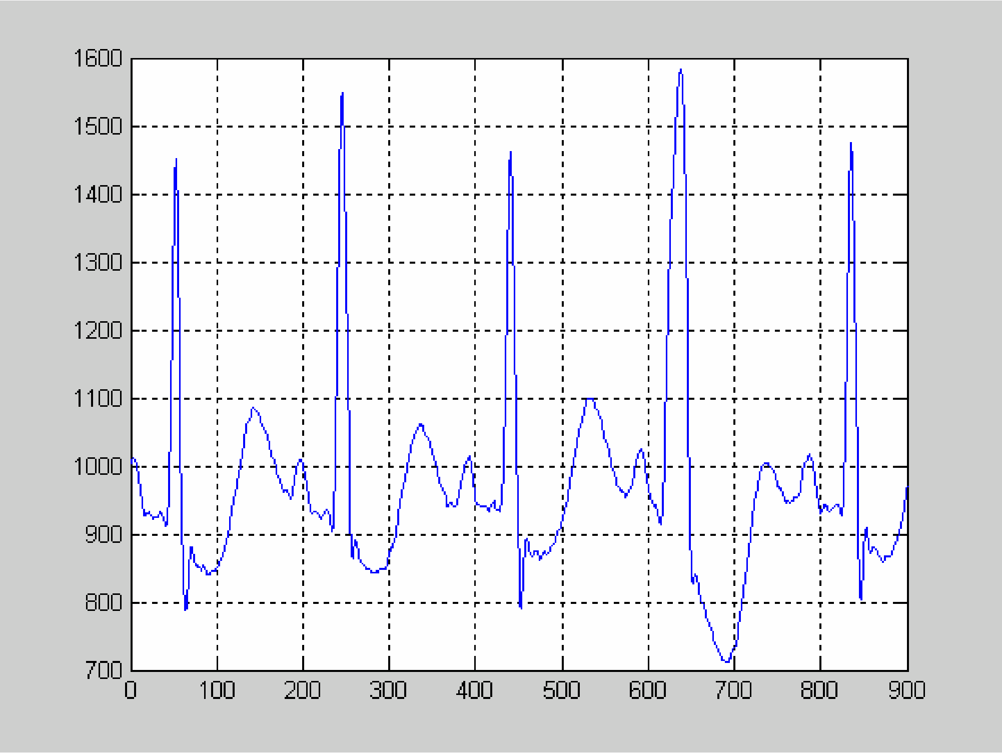


Figure 5.5 Input waveform: mit213_1from67000.txt

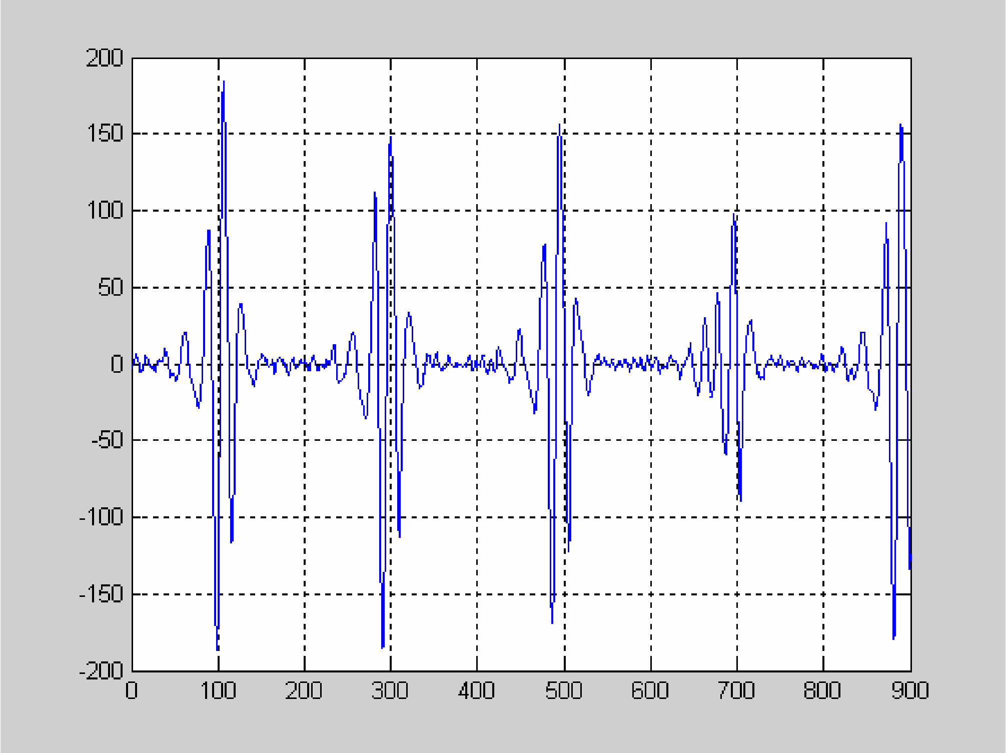


Figure 5.6 Output waveform: outMIT213_1from67000.txt.

Table 5.3 mit213_1from67000 extreme points and values

Input		**Output**		**Value**	**Error**
N	**ExtremeValue**	**N**	**Value**	**(MIT-BIH)**	
51	1451	101	≈ 0	50 N	1
244	1549	293	≈ 0	243 F	0
439	1463	489	≈ 0	438 N	1
637	1582	689	≈ 0	635 V	4
834	1476	884	≈ 0	833 N	1

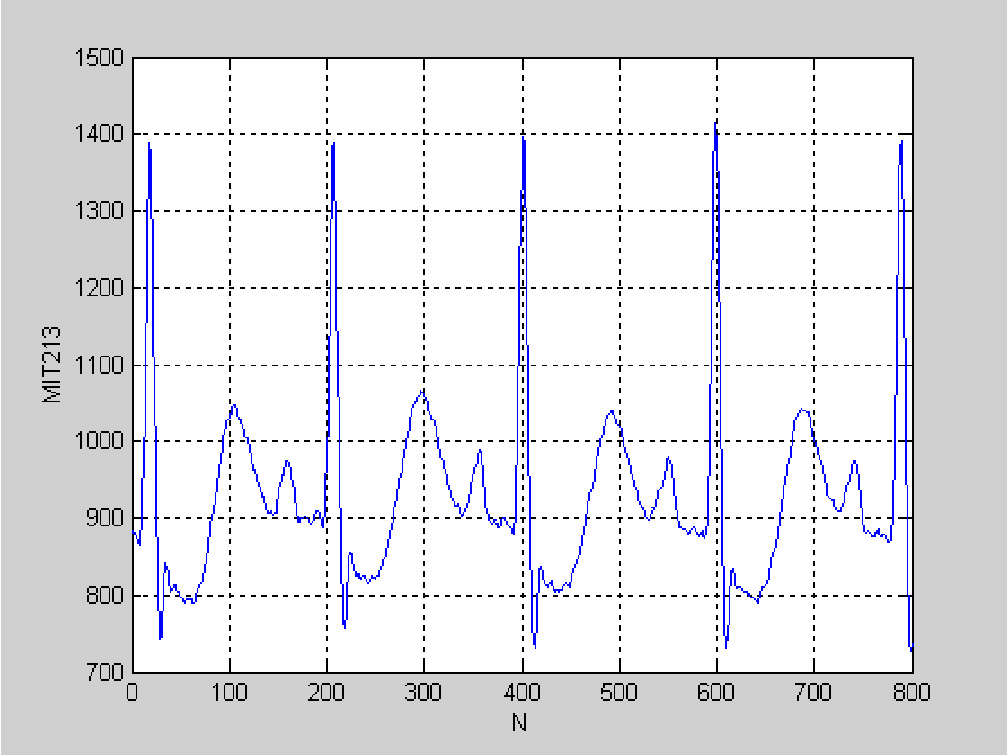


Figure 5.7 Input waveform: mit213_1from1450.txt.

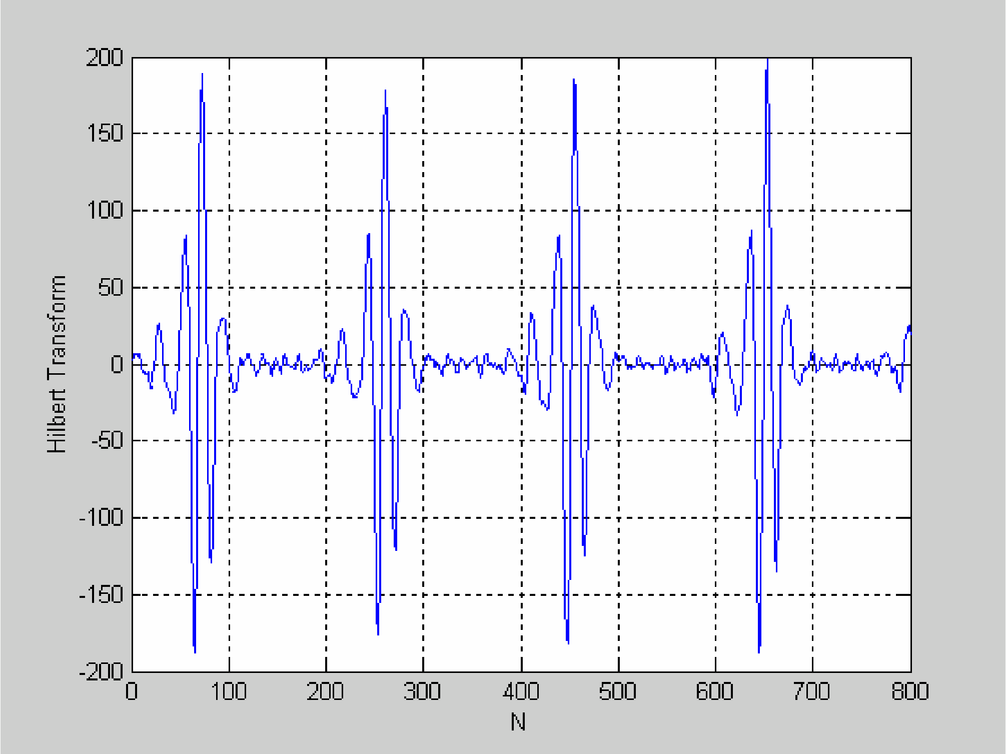


Figure 5.8 Output waveform: outMIT213_1from1450.txt.

Table 5.4 mit213_1from1450 extreme points and values

Input		**Output**		**Value (MIT-BIH)**	**Error**
N	**ExtremeValue**	**N**	**Value**		
17	1388	67	≈ 0	16 N	1
206	1388	255	≈ 0	205 N	0
400	1396	450	≈ 0	399 N	1
598	1415	647	≈ 0	597 N	0

From the results obtained in Table 5.3 and Table 5.4, it can be seen that the error for normal heartbeat and for the PVC beat are small and are acceptable.

Here other examples are given, the files $mit223\_1from47700.txt$ and $mit223\_1from7900.txt$ are sections from the MIT 223 in the MIT-BIH database. The input waves are shown in Figure 5.9 and Figure 5.11, the Hilbert transforms of the input ECG waves are shown in Figure 5.10 and Figure 5.12. In this case, there are a few irregular heartbeats, such as PVCs. In Figure 5.9, the PVCs happened between $N = 0$ to 100 and $N = 800$ to 900. Table 5.5 and Table 5.6 show the compared results.

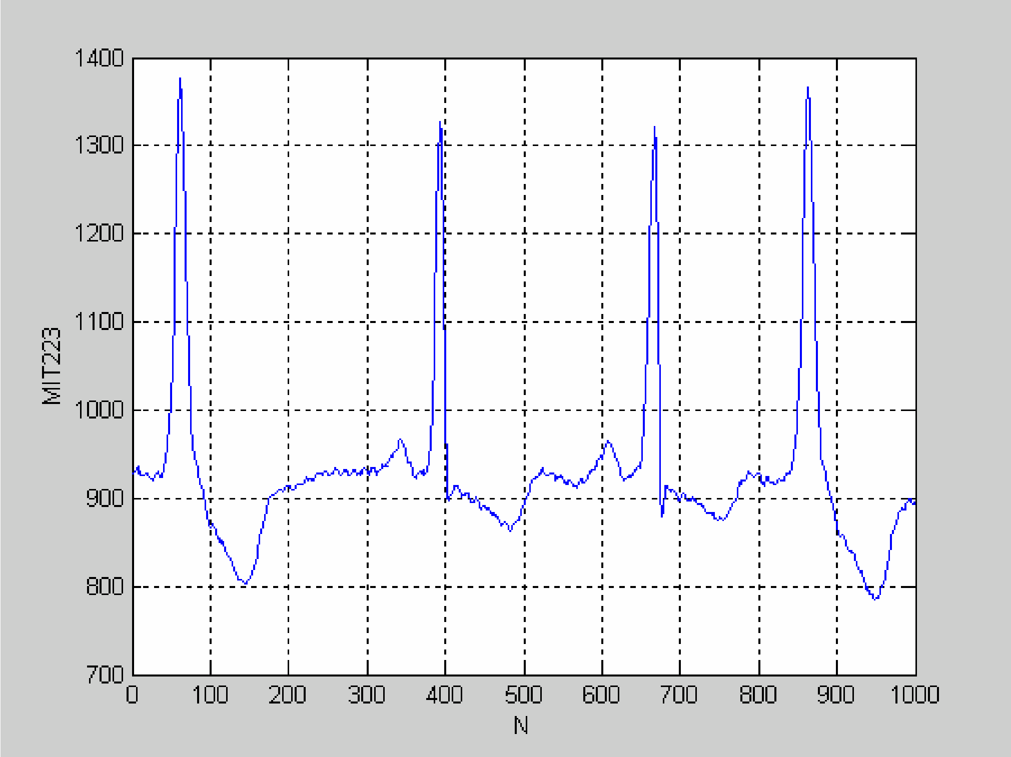


Figure 5.9 Input waveform: mit223_1from47700.txt.

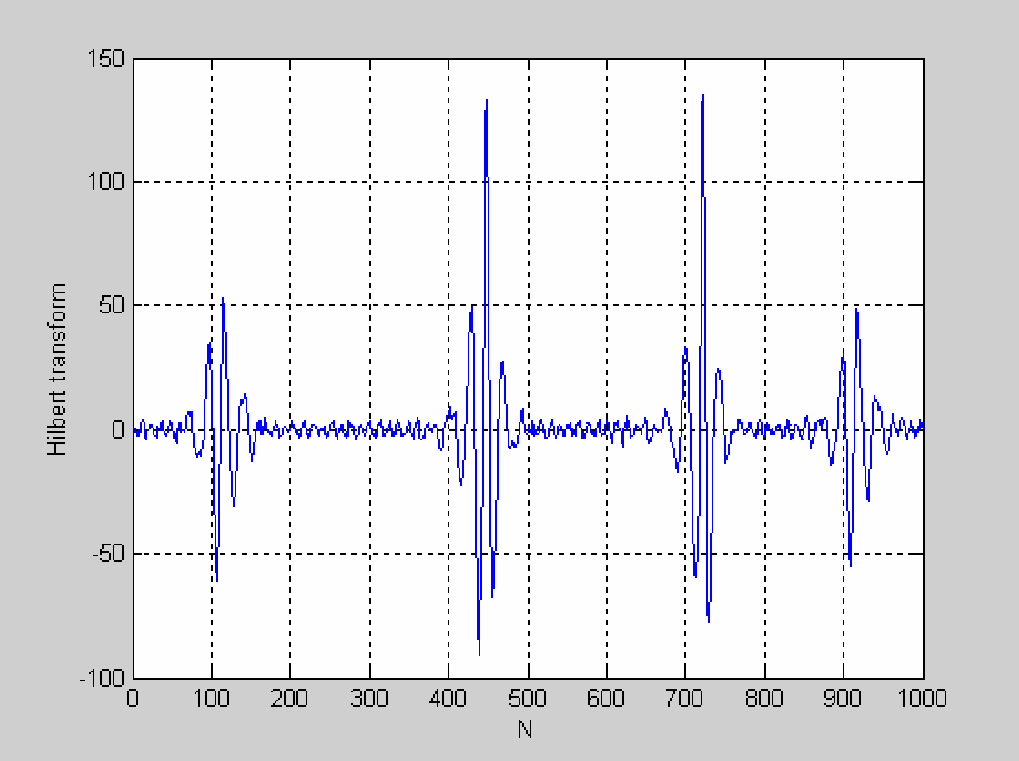


Figure 5.10 Output waveform: outMIT223_1_from47700.txt.

Table 5.5 mit223_1from47700 extreme points and values

Input		Output		Value (MIT-BIH)	Error
N	**Extreme Value**	**N**	**Value**		
60	1376	110	≈ 0	60 V	0
393	1327	442	≈ 0	391 N	1
667	1321	716	≈ 0	664 N	2
862	1367	912	≈ 0	861 V	1

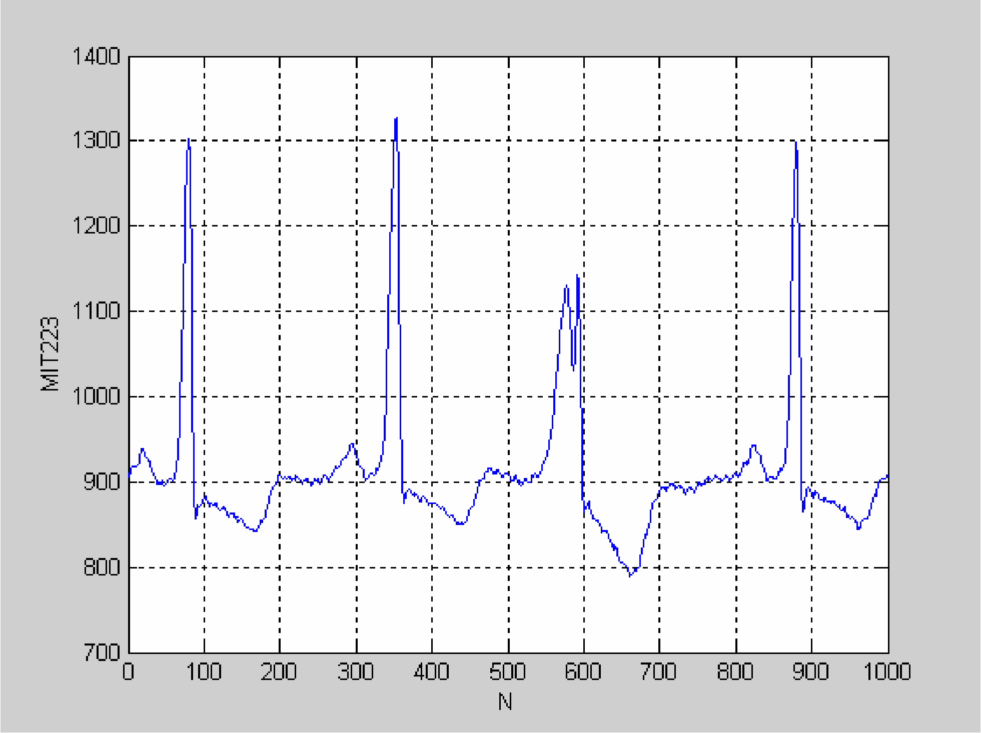


Figure 5.11 Input waveform: mit223_1from7900.txt.

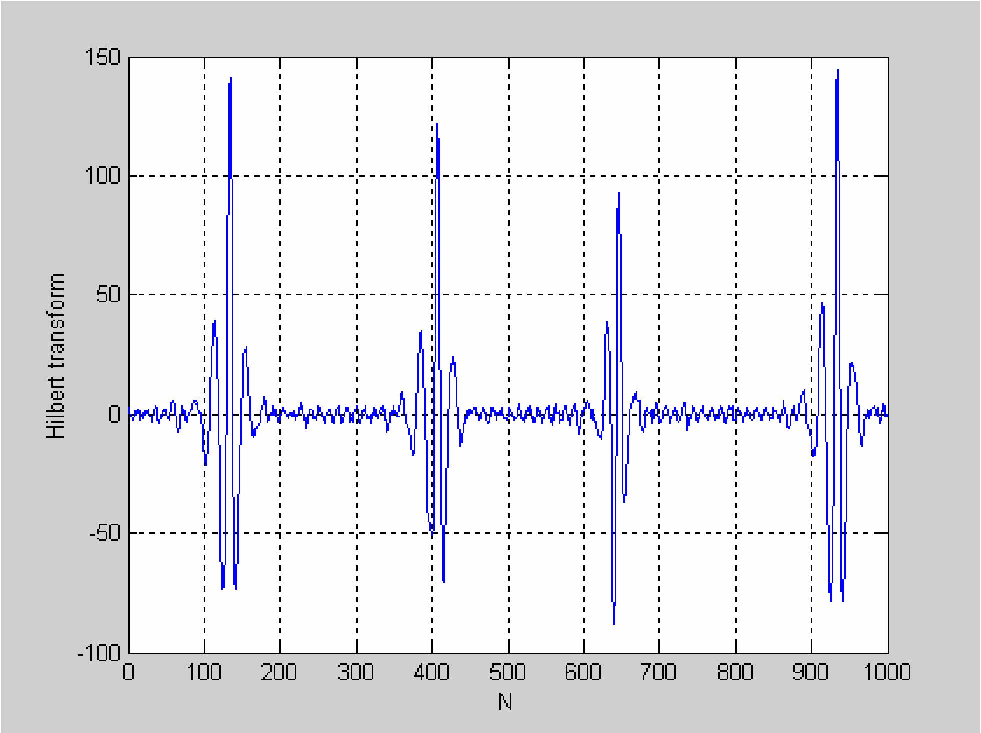


Figure 5.12 Output waveform: outMIT223_1_from7900.txt.

Table 5.6 mit223_1from7900 extreme points and values

Input		**Output**		**Value**	**Error**
N	**ExtremeValue**	**N**	**Value**	**(MIT-BIH)**	
79	1303	128	≈ 0	76 N	2
352	1328	401	≈ 0	349 N	2
591	1143	641	≈ 0	590 V	1
879	1298	928	≈ 0	876 N	2

In Figure 5.11, the PVC happened between $N = 500$ to 600. The R wave peak is not that clear, there are two peaks and the extreme values are close together. Table 5.5 and Table 5.6 record the R wave peaks in the input and output waveform. From Table 5.5 and Table 5.6, it can be seen that the output results lag the input data by 50 because the filter delay is 50 $(M/2)$.

As discussed in the Chapter 4, the points which are of interest are those zero crossing points with the extreme value changing from negative to positive or from positive to negative. Comparing the results shown in Table 5.1-Table 5.6 with the data recorded in the MIT-BIH arrhythmia database, the error is small and is acceptable. It illustrated that the algorithm and the program are correct and working properly.

5.2 Complete ECG Testing

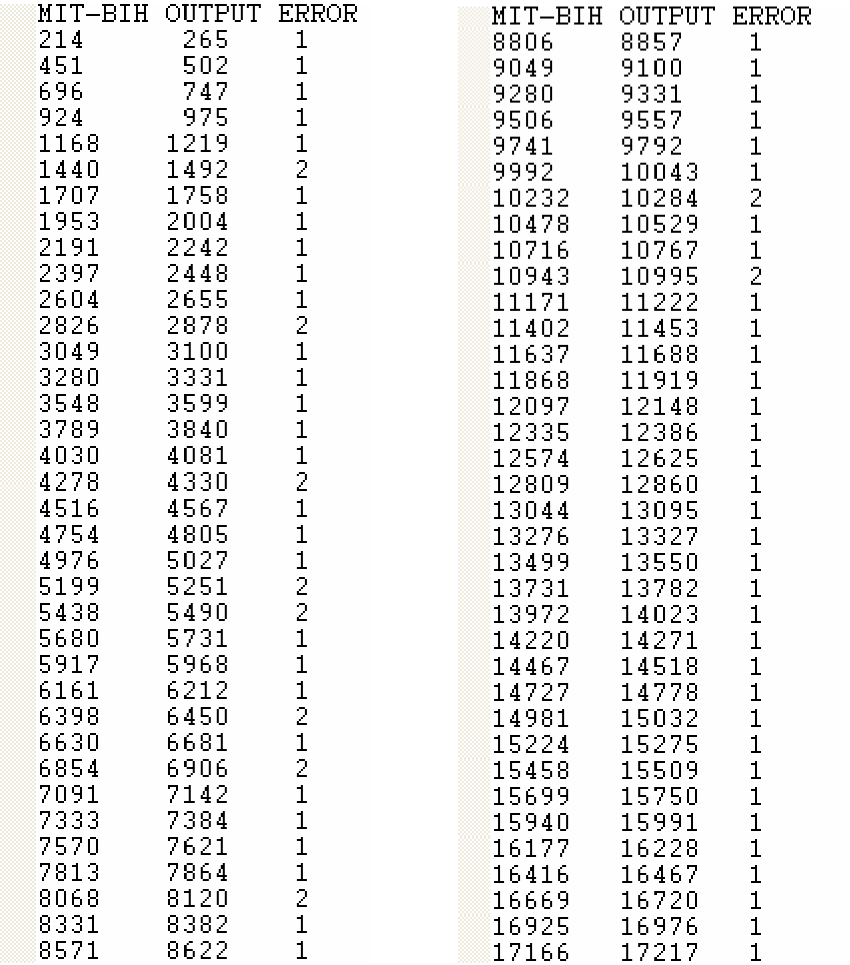
In the previous section, just small sections from the record of the MIT-BIH Arrhythmia database are tested. As mentioned previously, each MIT-BIH excerpt contains 30 minutes of ECG with heartbeat recording. In this section, the whole excerpt will be used to test. Every excerpt includes 649999 data samples.

MIT212 is the heart recording of an adult (31 years old). The predominant rhythm is normal sinus at rates of 75-90 BPM.

MIT213 is the record of a 61year_old person’s heart; heart rate is 100-110 BPM. Predominant is normal but there are occasional APBs (Atrial Premature Beats).

MIT223 is the record of an 84 year_old person’s heart. APBs are present throughout. There is high-grade ventricular ectopic activity with frequent multifocal VPBs, couplets, and runs of Vtach.

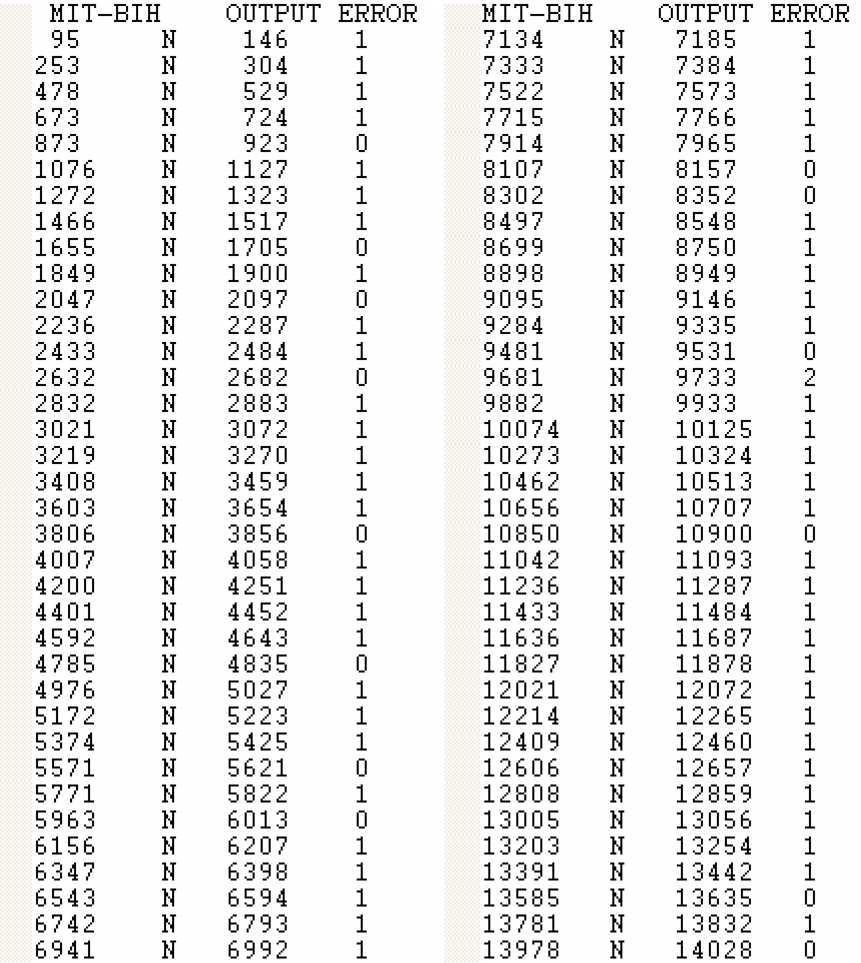
The entire excerpt from the MIT-BIH Arrhythmia database was tested by the system discussed in this thesis. The Hilbert transform of this excerpt is obtained and the R wave peaks are detected. Since the data is too long, just portion of the results of MIT212, MIT213 and MIT223 were shown in Figure 5.13, Figure 14-15 and Figure 5.16-17. Beat by beat comparison was performed according to the annotation file. The results are shown in Table 5.7.



```
MIT-BIH OUTPUT ERROR
214       265    1
451       502    1
696       747    1
924       975    1
1168     1219    1
1440     1492    2
1707     1758    1
1953     2004    1
2191     2242    1
2397     2448    1
2604     2655    1
2826     2878    2
3049     3100    1
3280     3331    1
3548     3599    1
3789     3840    1
4030     4081    1
4278     4330    2
4516     4567    1
4754     4805    1
4976     5027    1
5199     5251    2
5438     5490    2
5680     5731    1
5917     5968    1
6161     6212    1
6398     6450    2
6630     6681    1
6854     6906    2
7091     7142    1
7333     7384    1
7570     7621    1
7813     7864    1
8068     8120    2
8331     8382    1
8571     8622    1
```

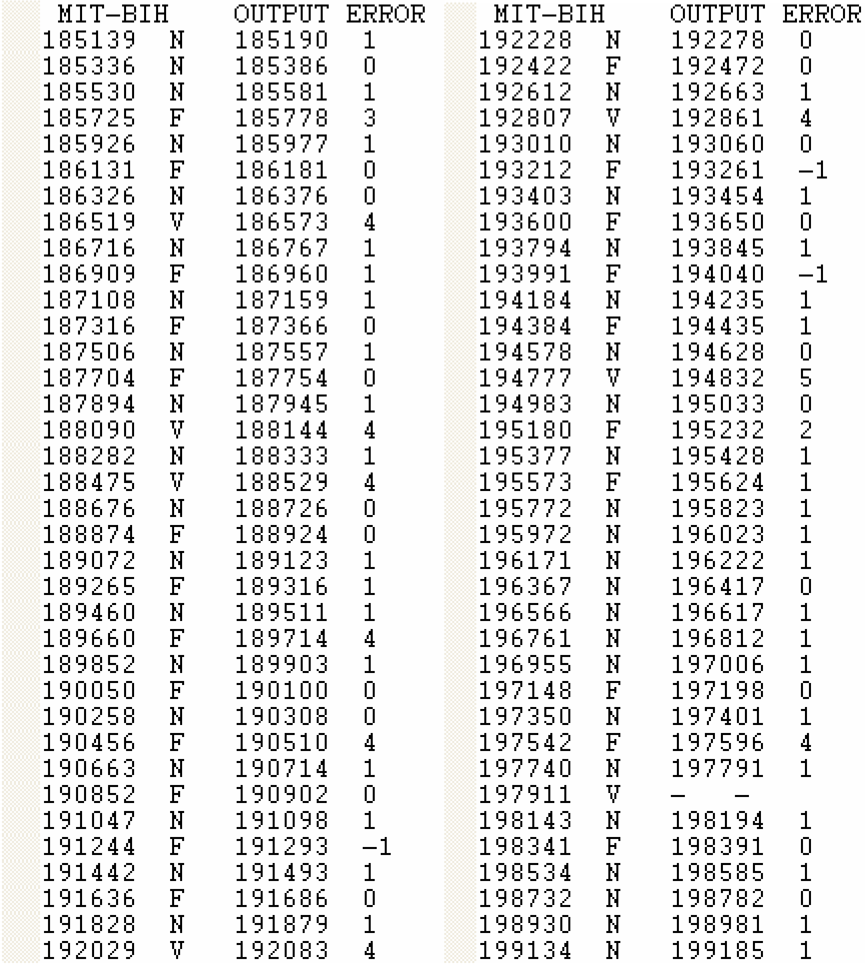
```
MIT-BIH OUTPUT ERROR
8806     8857    1
9049     9100    1
9280     9331    1
9506     9557    1
9741     9792    1
9992     10043   1
10232    10284   2
10478    10529   1
10716    10767   1
10943    10995   2
11171    11222   1
11402    11453   1
11637    11688   1
11868    11919   1
12097    12148   1
12335    12386   1
12574    12625   1
12809    12860   1
13044    13095   1
13276    13327   1
13499    13550   1
13731    13782   1
13972    14023   1
14220    14271   1
14467    14518   1
14727    14778   1
14981    15032   1
15224    15275   1
15458    15509   1
15699    15750   1
15940    15991   1
16177    16228   1
16416    16467   1
16669    16720   1
16925    16976   1
17166    17217   1
```

Figure 5.13 The portion of results of MIT212.



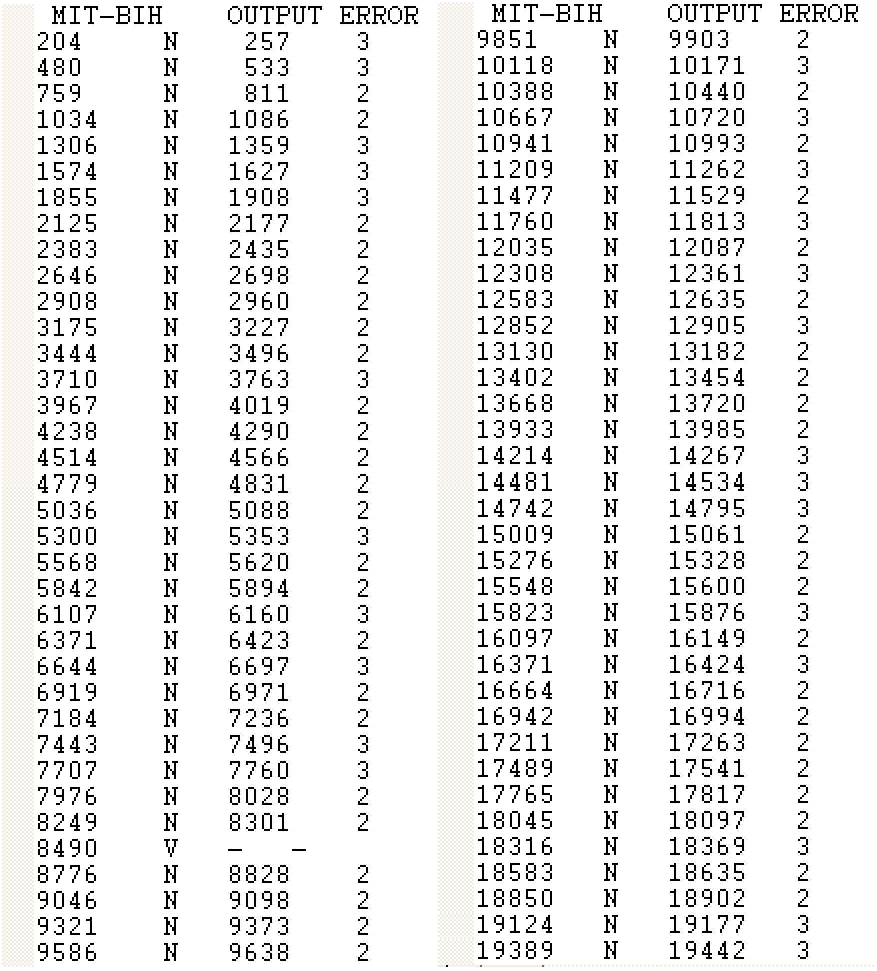
```
 MIT-BIH      OUTPUT ERROR     MIT-BIH       OUTPUT ERROR
 95      N     146    1        7134     N    7185     1
253      N     304    1        7333     N    7384     1
478      N     529    1        7522     N    7573     1
673      N     724    1        7715     N    7766     1
873      N     923    0        7914     N    7965     1
1076     N    1127    1        8107     N    8157     0
1272     N    1323    1        8302     N    8352     0
1466     N    1517    1        8497     N    8548     1
1655     N    1705    0        8699     N    8750     1
1849     N    1900    1        8898     N    8949     1
2047     N    2097    0        9095     N    9146     1
2236     N    2287    1        9284     N    9335     1
2433     N    2484    1        9481     N    9531     0
2632     N    2682    0        9681     N    9733     2
2832     N    2883    1        9882     N    9933     1
3021     N    3072    1        10074    N    10125    1
3219     N    3270    1        10273    N    10324    1
3408     N    3459    1        10462    N    10513    1
3603     N    3654    1        10656    N    10707    1
3806     N    3856    0        10850    N    10900    0
4007     N    4058    1        11042    N    11093    1
4200     N    4251    1        11236    N    11287    1
4401     N    4452    1        11433    N    11484    1
4592     N    4643    1        11636    N    11687    1
4785     N    4835    0        11827    N    11878    1
4976     N    5027    1        12021    N    12072    1
5172     N    5223    1        12214    N    12265    1
5374     N    5425    1        12409    N    12460    1
5571     N    5621    0        12606    N    12657    1
5771     N    5822    1        12808    N    12859    1
5963     N    6013    0        13005    N    13056    1
6156     N    6207    1        13203    N    13254    1
6347     N    6398    1        13391    N    13442    1
6543     N    6594    1        13585    N    13635    0
6742     N    6793    1        13781    N    13832    1
6941     N    6992    1        13978    N    14028    0
```

Figure 5.14 The portion of results of MIT213.



```
 MIT-BIH     OUTPUT ERROR     MIT-BIH     OUTPUT ERROR
185139  N    185190  1       192228  N    192278  0
185336  N    185386  0       192422  F    192472  0
185530  N    185581  1       192612  N    192663  1
185725  F    185778  3       192807  V    192861  4
185926  N    185977  1       193010  N    193060  0
186131  F    186181  0       193212  F    193261  -1
186326  N    186376  0       193403  N    193454  1
186519  V    186573  4       193600  F    193650  0
186716  N    186767  1       193794  N    193845  1
186909  F    186960  1       193991  F    194040  -1
187108  N    187159  1       194184  N    194235  1
187316  F    187366  0       194384  F    194435  1
187506  N    187557  1       194578  N    194628  0
187704  F    187754  0       194777  V    194832  5
187894  N    187945  1       194983  N    195033  0
188090  V    188144  4       195180  F    195232  2
188282  N    188333  1       195377  N    195428  1
188475  V    188529  4       195573  F    195624  1
188676  N    188726  0       195772  N    195823  1
188874  F    188924  0       195972  N    196023  1
189072  N    189123  1       196171  N    196222  1
189265  F    189316  1       196367  N    196417  0
189460  N    189511  1       196566  N    196617  1
189660  F    189714  4       196761  N    196812  1
189852  N    189903  1       196955  N    197006  1
190050  F    190100  0       197148  F    197198  0
190258  N    190308  0       197350  N    197401  1
190456  F    190510  4       197542  F    197596  4
190663  N    190714  1       197740  N    197791  1
190852  F    190902  0       197911  V    -    -
191047  N    191098  1       198143  N    198194  1
191244  F    191293  -1      198341  F    198391  0
191442  N    191493  1       198534  N    198585  1
191636  F    191686  0       198732  N    198782  0
191828  N    191879  1       198930  N    198981  1
192029  V    192083  4       199134  N    199185  1
```

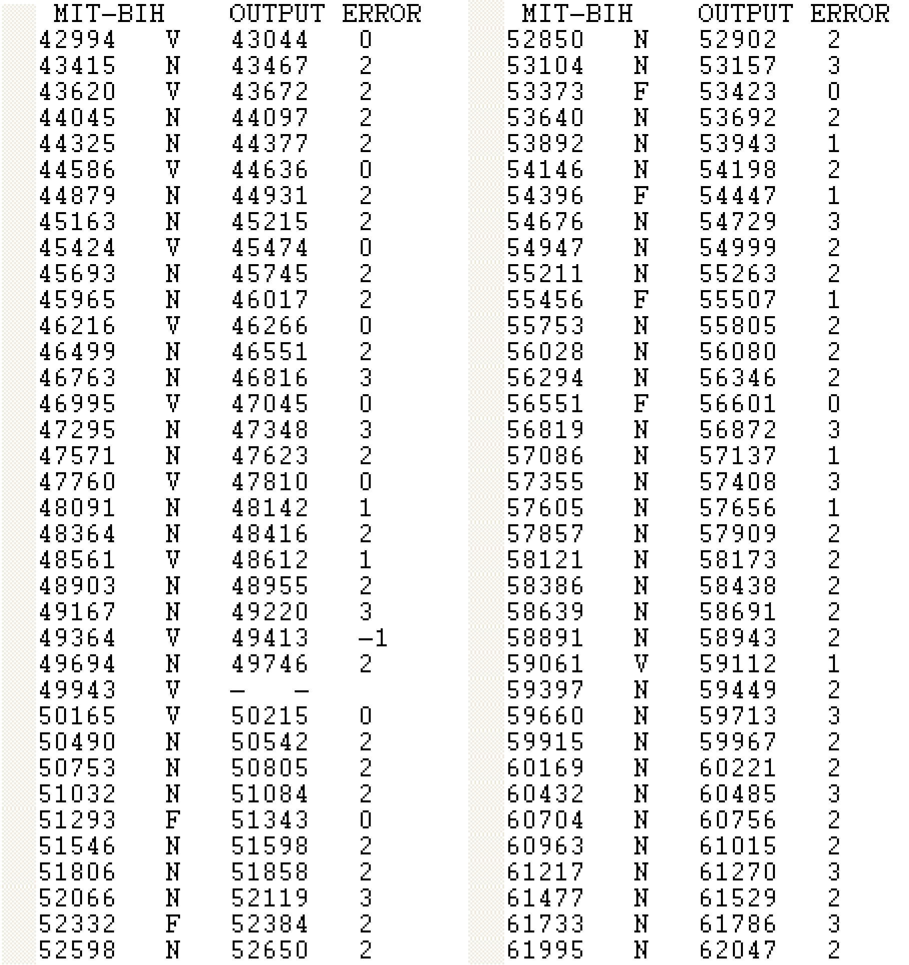
Figure 5.15 The portion of results of MIT213.



```
  MIT-BIH      OUTPUT ERROR
204      N      257    3
480      N      533    3
759      N      811    2
1034     N     1086    2
1306     N     1359    3
1574     N     1627    3
1855     N     1908    3
2125     N     2177    2
2383     N     2435    2
2646     N     2698    2
2908     N     2960    2
3175     N     3227    2
3444     N     3496    2
3710     N     3763    3
3967     N     4019    2
4238     N     4290    2
4514     N     4566    2
4779     N     4831    2
5036     N     5088    2
5300     N     5353    3
5568     N     5620    2
5842     N     5894    2
6107     N     6160    3
6371     N     6423    2
6644     N     6697    3
6919     N     6971    2
7184     N     7236    2
7443     N     7496    3
7707     N     7760    3
7976     N     8028    2
8249     N     8301    2
8490     V     -    -
8776     N     8828    2
9046     N     9098    2
9321     N     9373    2
9586     N     9638    2
```

```
  MIT-BIH      OUTPUT ERROR
9851     N     9903     2
10118    N     10171    3
10388    N     10440    2
10667    N     10720    3
10941    N     10993    2
11209    N     11262    3
11477    N     11529    2
11760    N     11813    3
12035    N     12087    2
12308    N     12361    3
12583    N     12635    2
12852    N     12905    3
13130    N     13182    2
13402    N     13454    2
13668    N     13720    2
13933    N     13985    2
14214    N     14267    3
14481    N     14534    3
14742    N     14795    3
15009    N     15061    2
15276    N     15328    2
15548    N     15600    2
15823    N     15876    3
16097    N     16149    2
16371    N     16424    3
16664    N     16716    2
16942    N     16994    2
17211    N     17263    2
17489    N     17541    2
17765    N     17817    2
18045    N     18097    2
18316    N     18369    3
18583    N     18635    2
18850    N     18902    2
19124    N     19177    3
19389    N     19442    3
```

Figure 5.16 The portion of results of MIT223.



```
 MIT-BIH      OUTPUT  ERROR     MIT-BIH      OUTPUT  ERROR
42994   V    43044    0        52850   N    52902    2
43415   N    43467    2        53104   N    53157    3
43620   V    43672    2        53373   F    53423    0
44045   N    44097    2        53640   N    53692    2
44325   N    44377    2        53892   N    53943    1
44586   V    44636    0        54146   N    54198    2
44879   N    44931    2        54396   F    54447    1
45163   N    45215    2        54676   N    54729    3
45424   V    45474    0        54947   N    54999    2
45693   N    45745    2        55211   N    55263    2
45965   N    46017    2        55456   F    55507    1
46216   V    46266    0        55753   N    55805    2
46499   N    46551    2        56028   N    56080    2
46763   N    46816    3        56294   N    56346    2
46995   V    47045    0        56551   F    56601    0
47295   N    47348    3        56819   N    56872    3
47571   N    47623    2        57086   N    57137    1
47760   V    47810    0        57355   N    57408    3
48091   N    48142    1        57605   N    57656    1
48364   N    48416    2        57857   N    57909    2
48561   V    48612    1        58121   N    58173    2
48903   N    48955    2        58386   N    58438    2
49167   N    49220    3        58639   N    58691    2
49364   V    49413   -1        58891   N    58943    2
49694   N    49746    2        59061   V    59112    1
49943   V    -   -             59397   N    59449    2
50165   V    50215    0        59660   N    59713    3
50490   N    50542    2        59915   N    59967    2
50753   N    50805    2        60169   N    60221    2
51032   N    51084    2        60432   N    60485    3
51293   F    51343    0        60704   N    60756    2
51546   N    51598    2        60963   N    61015    2
51806   N    51858    2        61217   N    61270    3
52066   N    52119    3        61477   N    61529    2
52332   F    52384    2        61733   N    61786    3
52598   N    52650    2        61995   N    62047    2
```

Figure 5.17 The portion of results of MIT223.

Table 5.7 R wave detection performance

MIT-BIH record	Actual number of beats in record	Failed detection number	Detection error rate	Average error
MIT212	2748	10	0.00364	1.11
MIT213	3251	9	0.00277	1.26
MIT223	2605	10	0.00384	2.43

The detection error rate and average error were calculated using the following equations respectively:

$$DetectionErrorRate = \frac{FailedDetectionNumber}{AcutralNumberOfBeats} \tag{5.2}$$

$$AverageError = \frac{\left(\sum_{i=1}^{k} \left|locatedR - actualR\right|\right)}{K}, \tag{5.3}$$

where

K is the total number of R correctly located by the detector.

From the detection error rate and the average error, it can be seen that the results are closed to the data recorded in the MIT-BIH annotation file. The average error in MIT 223 is bigger since MIT 223 records the heart beats of an 83 year old person and there are a lot of PVCs, the beat by beat error is larger. That is why the average error is larger than others. But the error is still acceptable (< 3 samples). The results of the test illustrate the algorithm for calculating the Hilbert transform and detecting the R wave peaks is effective.

Chapter 6 Summary and Conclusion

In the previous chapters, the information about the Holter monitoring system and the process for calculating the Hilbert transform of an ECG signal are discussed in detail. This final chapter summarises the results of the research and the contents of the thesis.

6.1 Summary

The Holter ECG Monitoring System mainly consists of four parts: Input Processing, Pattern Recognition, Compression and Storage. The main objective of the input processing is to calculate the Hilbert transform of the input ECG data. Pattern recognition uses vectorcardiograph and polarcardiograph representations and the concepts of pre-envelope and envelope of a real waveform given by the Hilbert transform to judge whether the ECG wave is normal or abnormal. Lastly data is compressed from the abnormal ECG wave and saved in the flash card. In this thesis, only the first part of the Holter ECG monitoring system, i.e., Input Processing is discussed.

The background of the Holter ECG Monitoring System was presented. The Hilbert transform applications and the basic mathematics and properties of a Hilbert transformer are also presented.

The Hilbert transform is a mathematical method for analysing signal waveforms, and has been widely used in the areas of communication systems analysis. The resulting display of the Hilbert transformed data is similar to that obtained from conventional vectorcardiographic systems. It allows easy visual indication of the different classes of normal and abnormal morphologies. It allows quick, precise segmentation of the incoming ECG into individual heartbeats and also allows the detection of Q-, R-, S-, and T-wave complexes in the data.

In this thesis, four approaches to compute the Hilbert transform including the Time-Domain approach, the Frequency-Domain approach, the Boche approach and the Remez Filter approach have been discussed in detail. The algorithms are deduced and examples are also given for every approach. After comparing them in running time and the ease of implementation, the Remez Filter approach which implements the Parks-McClellan algorithm to design and apply a linear-phase filter was determined to be the best and is used in computing the Hilbert transform of the ECG wave in this thesis. The results for every approach were shown in Chapter 3 and the comparisons were also given.

In the software implementation part, as a first step, the filter order and the frequency range are determined by analyzing a set of test results using MATLAB. The *remez* function was used in MATLAB to test the Hilbert transform of the ECG data.

In additional, a *C* program (*remez.c*) was developed to implement the algorithm of the Remez filter approach. *remez.c* uses the Parks-McClellan algorithm, i.e., uses the Remez exchange algorithm and Chebyshev approximation theory to design a filter with the optimal fit between the desired frequency response and actual frequency response. A main program that handles the input, sets up the appropriate approximation problem and handles the output of the optimal filter coefficients was included. Specifically *remez.c* has a build-in section for the more common ideal filter types such as multi-band, bandpass filters, differentiators and the Hilbert transform filters. All results including the order, frequency range and the coefficients of the filter were shown in Chapter 4.

A digital filter was developed to apply filtering operations on the data sequence. Using the coefficients achieved from *remez.c* and the input data, i.e., the ECG data sequence, the Hilbert transform of the input ECG data was obtained. All results and waveforms are also presented in Chapter 4.

Accurate determination of the QRS complex, in particular, accurate detection of the *R* wave peak, is essential in ECG analysis and is another task in this thesis. In the system discussed in this thesis, the method of ECG waveform analysis uses vectorcardiograph and polarcardiograph representations and examines the concepts of pre-envelope and envelope of a real waveform given by the Hilbert transform. A prototype two stage QRS detector was used based on the determination of a zero crossing in the Hilbert transformed data of the original ECG waveform. The positions of the zero crossing points that correspond to the R wave peaks are useful in judging whether the heartbeat is normal or abnormal.

A *C* program was developed to detect the QRS complex, in particular, to detect the R wave peak used in ECG analysis. All the results were shown in Chapter 4 and Chapter 5. The simulation results are also presented in Chapter 5.

The Nios embedded processor was introduced. SOPC Builder was used to be development tool. Using SOPC builder, users can combine the Nios processor with user logic and program it into an FPGA easily. All programs were run on the Nios embedded processor. The results, i.e., the Hilbert transform of the ECG data sequence, are almost the same as the results obtained in the C program except the first 50 data samples. The first 50 data samples in the output sequence can be ignored since the phase delay of the filter is 50. The results prove that the program can work properly in the Nios embedded processor.

6.2 Conclusion

The data used in test were from standard ECG waveform records in the MIT-BIH arrhythmia database. The performance of the chosen algorithm was tested. The test results were compared with the annotation files recorded in the MIT-BIH arrhythmia database. The detection error rate is smaller than 0.005. The average error is smaller than 3 samples. The results were given in Chapter 5. The error is acceptable. The results illustrated that the algorithm performed effectively with accurate R wave detection.

References

[1] Alan E. Lindsay, "ECG Learning Center in Cyberspace", Available at: http://medlib.med.utah.edu/kw/ecg/ecg_outline/Lesson1/index.html Accessed February 2006.

[2] Richard E. Klabunde, "Cardiovascular Physiology Concepts". Available at: http://www.cvphysiology.com/Arrhythmias/A009.htm Accessed March 2006.

[3] Molson Medical Informatics Student Project, "Cardiac Arrythmias", Available at: http://sprojects.mmi.mcgill.ca/cardiophysio/sinustachycardia.htm Accessed March 2006.

[4] "Electrocardiograms", Available at: http://www.healthyhearts.com/ecg.htm Accessed March 2006.

[5] Merck& Co., Inc., "The Merck Manual of Health & Aging", Available at: http://www.merck.com/pubs/mmanual_ha/figures/fg47_1.html Accessed March 2006.

[6] Aetna, Inc. Available at: http://www.aetna.com/cpb/data/CPBA0019.html Accessed March 2006.

[7] Wikipedia, the free encyclopedia, "Cauchy principal value", Available at: http://en.wikipedia.org/wiki/Cauchy_principal_value Accessed April 2006.

[8] Kunio Takaya, “Digital Signal Processing Electrical Engineering EE-880”, University of Saskatchewan, 2004.

[9] Mathias Johansson, M.Sc. thesis, “The Hilbert transform”, VaxJo University, 1999.

[10] Easy Fourier Analysis, “Signal processing & simulation newsletter”, Available at: http://www.complextoreal.com/tcomplex.htm Accessed April 2006.

[11] Ronald J. Bolton, Ph.D. thesis, “Hilbert transform processing of Electrocardiograms”, University of Queensland, 1983.

[12] Jose A. Inaudi and James M. Kelly, “Linear hysteretic damping and the Hilbert transform”, Journal of Engineering Mechanics, Vol.121, Issue 5, pp.626-632, May 1995.

[13] Holger Boche and Marcus Protzmann, “A new algorithm for the reconstruction of bandlimited functions and their Hilbert transform”, IEEE Transactions on Instrumentation and Mesurement, Vol. 46. No.2, pp. 442-444, April 1997.

[14] Altera website, http://www.altera.com/products/ip/processors/nios2/ni2-index.html Accessed April 2006.

[15] Altera Corporation, “Nios Hardware Development Tutorial”, TU-NIOSHWDV-1.2, January 2004.

[16] Altera Corporation, “Nios Software Development Tutorial”, TU-NIOSSFTWR-1.3, July 2003.

[17] Altera Corporation, “Nios II Development Board Reference Manual, Stratix Professional Edition”, MNL-N2DEVLSTXPRO-1.1, September 2004.

[18] Altera Corporation, “Nios II Development Kit Getting Started User Guide”, UG-NIOSIIDEVKIT-2.2, May 2006.

[19] A. Robin, “Digital Filters: An Introduction”, TechOnLine, Available at: http://www.techonline.com/community/ed_resource/feature_article/20365 Accessed May 2006.

[20] Alan V. Oppenheim and Ronald W. Schafer, “Discrete-Time Signal Processing”, Prentice Hall, Englewood Cliffs, NJ, 1989.

[21] Wolfram Research, “Digital Image Processing Documentation”, Available at: http://documents.wolfram.com/applications/digitalimage/UsersGuide/FilterDesign/ImageProcessing9.2.html , Accessed May 2006.

[22] “Parks-McClellan algorithm for FIR filter design”, Available at: http://search.cpan.org/src/MLEHMANN/PDL-Audio-1.1/remez.c Accessed May 2005.

[23] The digital signal processing committee, “Programs for Digital Signal Processing”, IEEE Acoustics, Speech, and Signal Processing Society, The Institute of Electrical and Electronics Engineers INC., pp5.1-1 to pp.5.1-13, 1979.

[24] D. Benitez, P.A. Gaydecki, A. Zaidi, A.P. Fitzpatrick “The use of the Hilbert transform in ECG signal analysis”, Computers in Biology and Medicine , Vol.31, pp.399-406, 2001.

[25] Encyclopedia, “Premature Ventricular Contractions”, Komo 1000 news, Available at: http://ww3.komotv.com/Global/story.asp?S=1230267 Accessed July 2006.

Appendix A

The MIT-BIH Arrhythmia Database

A.1 Introduction

The database consists of 48 records, each containing 30 minutes of two-channel ECG with beat and rhythm annotations. Each digital record has been copied from an analog recording made with an Avionics 445 two-channel recorder. Annotations have been made by two independent cardiologists with consultation to resolve disagreements.

The data base is recorded on twelve 2400-foot (730m) ANSI standard 9-track tapes at 800 bpi, odd parity, with NRZI recording. A detailed catalog of the contents of each tape, with illustrations, is included with the database.

The format is that which will be used for the American Heart Association (AHA) Database for Ventricular Arrhythmia Detectors, with these differences:

Each 9-track tape contains four 30-minute records.

Sampling frequency is 360 Hz per channel.

Sampling precision is 11 bits, and all samples are represented as positive numbers.

The entire 30-minute record is annotated.

Annotations are referenced to samples (rather than milliseconds).

An additional “0” annotation has been added to the AHA set to specify non-beat annotations (e. g., rhythms, artifact).The “R” (R-on-T PVC) annotation is not used.

Space allocated but unused in the AHA format annotation blocks is used to specify rhythms and beat types more precisely than is allowed using the AHA annotation codes alone. Atrial ectopic beats and conduction defects are among the items specified in this way.

A.2 File Structure

Each tape contains 16 files separated by ANSI standard end-of-file (EOF) marks. The last file is terminated with two EOFs to indicate the end of the tape.

Each record, corresponding to 30 minutes of ECG and annotations, is comprised of four files: an ID block file, a sample data file, an annotation file, and a second ID block file. The order of files is:

(Record 1) ID Block
(EOF)
Sample Data Blocks
(EOF)
Annotation Blocks
(EOF)
ID Block
(EOF)
(Record 2) ID Block
(EOF)
Sample Data Blocks
(EOF)
Annotation Blocks
(EOF)
ID Block
(EOF)

(Record3) Sample Data Blocks
(EOF)
Annotation Blocks
(EOF)
ID Block
(EOF)
(Record4) Sample Data Blocks
(EOF)
Annotation Blocks
(EOF)
ID Block
(EOF)

A.3 Notational and Other Conventions

Multiple-byte Numbers

In this specification, the least significant 8-bit byte of a multiple-byte number is referred to as byte i, the next most significant byte as byte ii, and so on. (The first byte read from the tape in a given block is called byte 1.)

The AHA format specifies that:

16-bit numbers are stored in the order: byte i, byte ii.

32-bit numbers are stored in the order: byte iii, byte iv, byte i, byte ii.

ASCII text

In the ID block and in certain annotation labels (see below) brief comments are present. These are coded as ASCII characters, and should be read from the tape in byte-sequential order.

TOCs

“TOC” means “time of occurrence”, TOCs are always represented as 32-bit numbers. TOCs in the AHA database are given as the number of MILLISECONDS from the beginning of the annotated segment of the record. In the MIT-BIH database, TOCs are given as the number of SAMPLE COUNTS from the beginning of the record. To convert sample counts to milliseconds, multiply sample counts by 1000/360 (=2.777….).

A.4 File Format Specifications

ID block file

The first and fourth files in each record each consist of a single 512 byte ID block, The AHA specification for the ID block is:

Bytes	Use
1-8	record ID (8 ASCII characters)
9-10	number of annotations
11-16	unused
17-20	time of first sample in the annotated segment of the record
21-24	time of last sample in the annotated segment of the record
25-26	number of bytes of sample data, divided by 512 and rounded upward
27-32	unused
33-36	TOC-first annotation, relative to the beginning of the annotated segment
37-40	TOC-last annotation, relative to the beginning of the annotated segment
41-42	numbers of bytes of annotation data, divided by 512 and rounded upward
43-512	unused

In each record in the MIT-BIH database, the entire record has been annotated; thus the time of the first sample is always zero. Each tape has exactly 649999 samples, and an end-of-sample-data mark, per channel (30 minutes and 5.444 seconds), and the annotated segment is considered to end after the end-of-sample-data mark, so that the time of the last sample in the annotated segment is always 650000, and the number of bytes of sample data divided by 512 is always 5079

(649999 samples per channel, times 2 channels, times 2 bytes per sample, divided by 512, rounded up). NOTE THAT THE UNITS OF TIME ARE SAMPLE COUNTS, NOT MILLISECONDS

Sample data file

The second file in each record is the sample data file, which consists of exactly 2540 blocks, each 1024 bytes long.

Each block contains 256 2-byte samples from each channel. Samples are stored alternately in the block:

Byte	Use
1	Channel 1, sample 1, byte i
2	Channel 1, sample 1, byte ii
3	Channel 2, sample 1, byte i
4	Channel 2, sample 1, byte ii
5	Channel 1, sample 2, byte i
⋮	
1021	Channel 1, sample 256, bytei
1022	Channel 1, sample 256, byteii
1023	Channel 2, sample 256, bytei
1024	Channel 2, sample 256, byteii

The AHA database has been recorded using a 12-bit A/D converter with a range of -10V to +10V, and preamp gain adjusted so that a QRS complex is nominally 1V peak-to-peak, or about 200 ADC units. The MIT-BIH database has been recorded using an 11-bit A/D converter with a range of -5mV to +5mV, and the unamplified QRS complexes are nominally 1 mV, or about 200 ADC units; thus the scales are the same though the ranges differ. Both positive and negative (two's complement, with sign extension to 16 bits) samples are recorded in the AHA database; in the MIT-BIH database, all samples are positive (in the range of 0 to 2047).

Block 2540 contains the last fifteen samples for each channel. The end of the sample data is marked in the last sample block by two consecutive sample values of 10000 (base 8) following the last samples. The remainder of the last sample block is padded with zeroes.

Annotation file

The third file in each record is the annotation file, which consists of a variable number (typically 20 to 50) of blocks, each 1024 bytes long.

Each block contains 64 annotation labels, each 16 bytes long. Annotation labels are stored in strict chronologic order.

The AHA format leaves a number of unused bytes in each annotation label, some of which are used in the MIT-BIH database.

The last annotation block is padded with all ones (177 base 8) following the last annotation. If there is no room following the last annotation, an entire block of 177s is written.

A.5 Annotation Specifications

Annotation labels

Byte	AHA format	MIT-BIH format
1	unused	unused(0)
2	AHA annotation code	AHA annotation code
3-6	TOC (milliseconds)	TOC (sample counts)
7-8	Annotation label serial number	Annotation label serial number
9	unused	unused (0)
10	unused	MIT-BIH annotation code
11-16	unused	ASCII text *

*The ASCII text field is filled with zero bytes unless the MIT-BIH annotation code is 22 or 28 (see next page following).

AHA annotation codes

The AHA annotation codes are ASCII characters:

Character	Value (base 8)	Meaning
N	116	supraventricular beat
V	126	premature ventricular contraction (PVC)
E	105	ventricular escape beat
F	106	fusion PVC
R	122	R-on-T PVC
P	120	paced beat
Q	121	beat of indeterminate origin
U	125	data unreadable between preceding and following beat labels
[	133	beginning of ventricular flutter or fibrillation
]	135	end of ventricular flutter/fibrillation

The “R” code does not appear in the MIT-BIH database. An additional code, “O” (117 base 8), has been defined to permit inclusion of rhythm labels, artifact labels, and comments. “O” labels are never QRS labels, and may be ignored for the purpose of counting beats.

MIT-BIH annotation codes

The MIT-BIH annotation codes are not ASCII characters, but numbers between 1 and 37:

Code	AHA equivalent	Meaning
1	N	normal QRS
2	N	left bundle branch block beat
3	N	right bundle branch block beat

4	N	aberrantly conducted beat
5	V	premature ventricular contraction (PVC)
6	F	fusion PVC ***
7	N	nodal premature beat
8	N	atrial premature bat (APB)
9	N	nodal or atrial premature beat
10	E	ventricular escape beat
11	N	nodal escape beat
12	P	paced beat
13	Q	beat of indeterminate origin
14	O,U	beginning of noise *
15	O	end of noise *
16	O	single QRS-like artifact
17-21	O	reserved for future use **
22	O	comment (text) annotation ***
23-24	O	reserved for future use **
25	N	left or right bundle branch block beat
26	O	non-captured pacemaker spike
27	O	axis shift
28	O	rhythm onset (text) annotation ***
29-30	O	reserved for future use **
31	O	ventricular flutter wave
32	[	onset of ventricular flutter or ventricular fibrillation
33	]	end of ventricular flutter/fibrillation atrial ectopic beat
34	N	atrial ectopic beat
35	N	nodal ectopic beat
36	O	missed beat
37	O	blocked APB
38	O	reserved for future use **

* Annotation codes 14 and 15 are used in pairs. If AHA code corresponding to the code 14 “U”, no beats are labeled until the next code 15; otherwise, all beats are labeled.

** Where codes designated “reserved for future use” appear, they should be ignored.

*** “Text” annotations use the last six bytes of the annotation label for an ASCII string (terminated by a zero byte if less than six characters).

**** In the context of paced rhythm (tapes 102,104,107,217) annotation code 6 is used for pacemaker fusion beats.

Rhythm onset annoations

Rhythm onset annotation (MIT-BIH annotation code 28) include an ASCII string which begins with a "(":

String	Meaning
(AB	atrail bigeminy
(AFIB	atrial fibrillation
(AFL	atrial flutter
(B	ventricular bigeminy
(BI	first degree heart block
(BII	second degree heart block
(BIII	third degree heart block
(IVR	idioventricular rhythm
(N	normal sinus rhythm
(NOD	normal (A-V junctional) rhythm
(PAT	paroxsysmal atrial tachycardia
(PREX	pre-excitation (WPW)
(SVTA	supraventricular tachyarrhythmia
(T	ventricular trigeminy
(VFIB	ventricular fibrillation
(VFL	ventricular flutter
(VT	ventricular tachycardia

Comment annotations

Sparse comment annotations exist on a few records. They are:

PSE	pause
TS	tape slippage

Appendix B

B.1 The Parks-McClellan Algorithm

Consider a particularly effective and widely used algorithm procedure for the design of FIR filters with generalized linear phase. Although only type I filters are considered in detail, where appropriate results that apply to types II, III, and IV generalized linear phase filters are indicated.

In designing a causal type I linear phase FIR filter, it is convenient first to consider the design of a zero-phase filter, i.e., one for which

$$h_e[n] = h_e[-n], \tag{1}$$

and then to insert sufficient delay to make it causal. Consequently, consider $h_e[n]$ satisfying the condition of Eq. (1). The corresponding frequency response is given by

$$A_e(e^{j\omega}) = \sum_{n=-L}^{L} h_e[n] e^{-j\omega m} \quad , \tag{2}$$

with $L = M/2$ an integer or, because of Eq. (1),

$$A_e(e^{j\omega}) = h_e[0] + \sum_{n=1}^{L} 2h_e[n]\cos(\omega n). \tag{3}$$

Note that $A_e(e^{j\omega})$ is a real, even, and periodic function of ω. A causal system can be obtained from $h_e[n]$ by delaying it by $L = M/2$ samples. The resulting system has impulse response

$$h[n] = h_e[n - M/2] = h[M - n] \tag{4}$$

and frequency response

$$H(e^{j\omega}) = A_e(e^{j\omega})e^{-j\omega M/2}. \tag{5}$$

The Parks-McClellan algorithm is based on reformulating the filter design problem as a problem in polynomial approximation. Specially, the terms $\cos(\omega n)$ in Eq. (3) can be expressed as a sum of powers of $\cos\omega$ in the form

$$\cos(\omega n) = T_n(\cos\omega), \tag{6}$$

where $T_n(x)$ is an nth-order polynomial. Consequently, Eq. (3) can be rewritten as an Lth-order polynomial in $\cos\omega$. Specially,

$$A_e(e^{j\omega}) = \sum_{k=0}^{L} a_k(\cos\omega)^k, \tag{7}$$

where the a_k's are constants that are related to $h_e[n]$, the values of the impulse response. With the substitution $x = \cos\omega$, Eq. (7) can be expressed as

$$A_e(e^{j\omega}) = P(x)\big|_{x=\cos\omega}, \tag{8}$$

where $P(x)$ is the Lth-order polynomial

$$P(x) = \sum_{k=0}^{L} a_k x^k. \tag{9}$$

It is not necessary to know the relationship between the a_k's and $h_e[n]$; it is enough to know that $A_e(e^{jw})$ can be expressed as the Lth-degree trigonometric polynomial of Eq.(7).

The key to gaining control over ω_p and ω_s is to fix them at their desired values and let δ_1 and δ_2 vary. To formalize the approximation problem, define an approximation error function

$$E(\omega) = W(\omega)[H_d(e^{j\omega}) - A_e(e^{j\omega})], \tag{10}$$

where the weighting function, $W(\omega)$, incorporates the approximation error parameters into the design process. In this design method, the error function $E(\omega)$, the weighting function $W(\omega)$, and the desired frequency response $H_d(e^{j\omega})$ are defined only over closed subintervals of $0 \le \omega \le \pi$.

Parks and McClellan applied the following Alternation Theorem of approximation theory to the filter design problem.

Let F_P denote the closed subset consisting of the disjoint union of closed subsets of the real axis x . $P(x)$ denotes an rth-order polynomial

$$P(x) = \sum_{k=0}^{r} a_k x^k .$$

Also, $D_P(x)$ denotes a given desired function of x that is continuous on F_P, and $E_P(x)$ denotes the weighted error

$$E_P(x) = W_P(x)[D_P(x) - P(x)].$$

The maximum error $\|E\|$ is defined as

$$\|E\| = \max_{x \in F_P} |E_P(x)| .$$

A necessary and sufficient condition that $P(x)$ is the unique rth-order polynomial that minimizes $\|E\|$ is that $E_P(x)$ exhibit at least $(r+2)$ alternations, i.e., there must exist at least $(r+2)$ values x_i in F_P such that $x_1 < x_2 < \cdots < x_{r+2}$ and such that $E_P(x_i) = -E_P(x_{i+1}) = \pm\|E\|$ for $i = 1,2,\ldots,(r+1)$.

The alternation theorem states necessary and sufficient conditions on the error for optimality in the Chebyshev sense. Although the theorem does not state explicitly how to find the optimum filter, the condition is phrased in terms of type I lowpass filters, the algorithm easily generalizes.

From the alternation theorem, the optimum filter $A_e(e^{j\omega})$ will satisfy the following set of equations:

$$W(\omega_i)[H_d(e^{j\omega_i}) - A_e(e^{j\omega_i})] = (-1)^{i+1}\delta, \quad i = 1,2,\ldots,(L+2), \tag{11}$$

where δ is the optimum error and $A_e(e^{j\omega})$ is given by either Eq. (3) or Eq. (7). Using Eq. (.7) for $A_e(e^{j\omega})$, these equations can be written as

$$\begin{bmatrix} 1 & x_1 & x_1^2 & \cdots & x_1^L & \dfrac{1}{W(\omega_1)} \\ 1 & x_2 & x_2^2 & \cdots & x_2^L & \dfrac{-1}{W(\omega_2)} \\ \vdots & \vdots & \vdots & & \vdots & \vdots \\ 1 & x_{L+2} & x_{L+2}^2 & \cdots & x_{L+2}^L & \dfrac{(-1)^{L+2}}{W(\omega_{L+2})} \end{bmatrix} \begin{bmatrix} a_0 \\ a_1 \\ \vdots \\ \delta \end{bmatrix} = \begin{bmatrix} H_d(e^{j\omega_1}) \\ H_d(e^{j\omega_2}) \\ \vdots \\ H_d(e^{j\omega_{L+2}}) \end{bmatrix}, \tag{12}$$

where $x_i = \cos\omega_i$. This set of equation serves as the basis for an iterative algorithm for finding the optimum $A_e(e^{j\omega})$. The procedure begins by guessing a set of alternation frequencies ω_i, $i = 1,2,\ldots,(L+2)$. Note that ω_P and ω_s are fixed and are members of the set of alternation frequencies. Specifically if $\omega_l = \omega_p$, then $\omega_{l+1} = \omega_s$. The set of equations (12) could be solved for the set of coefficients a_k and δ. However, a more efficient alternative is to use polynomial interpolation.

Specifically, Parks and McClellan found that for the given set of the extremal frequencies, δ is given by the formula

$$\delta = \frac{\sum_{k=1}^{L+2} b_k H_d(e^{j\omega_k})}{\sum_{k=1}^{L+2} \frac{b_k (-1)^{k+1}}{W(\omega_k)}}, \tag{13}$$

where

$$b_k = \prod_{\substack{i=1 \\ i \neq k}}^{L+2} \frac{1}{(x_k - x_i)} \tag{14}$$

and, as above, $x_i = \cos\omega_i$. That is, if $A_e(e^{j\omega})$ is determined by the set of coefficients a_k that satisfy Eq. (12), with δ given by Eq. (13), then the error function goes through $\pm\delta$ at the $(L+2)$ frequencies ω_i or, equivalently, $A_e(e^{j\omega})$ has values $1 \pm K\delta$ if $0 \leq \omega_i \leq \omega_p$ and $\pm\delta$ if $\omega_s \leq \omega_i \leq \pi$. Now since $A_e(e^{j\omega})$ is known to be an Lth-order trigonometric polynomial, can interpolate a trigonometric polynomial through $(L+1)$ of the $(L+2)$ known values $E(\omega_i)$ (or equivalently $A_e(e^{j\omega})$). Parks and McClellan used the Lagrange interpolation formula to obtain

$$A_e(e^{j\omega}) = P(\cos\omega) = \frac{\sum_{k=1}^{L+1} [d_k /(x - x_k)] C_k}{\sum_{k=1}^{L+1} [d_k /(x - x_k)]}, \tag{15}$$

where $x = \cos\omega$, $x_i = \cos\omega_i$,

$$C_k = H_d(e^{j\omega_k}) - \frac{(-1)^{k+1}\delta}{W(\omega_k)}, \tag{16}$$

and

$$d_k = \prod_{\substack{i=1 \\ i\neq k}}^{L+1} \frac{1}{(x_k - x_i)} = \frac{b_k}{(x_k - x_{L+2})}. \tag{17}$$

Now, $A_e(e^{j\omega})$ is available at any desired frequency without the need to slove the set of Eq. (12) for the coefficients a_k. The polynomial of Eq. (15) can be used to evaluate $A_e(e^{j\omega})$ and also $E(\omega)$ on a dense set of frequencies in the passband and stopband. If $|E(\omega)| \leq \delta$ for all ω in the passband and stopband, then the optimum approximation has been found. Otherwise a new set of extremal frequencies must be found.

In this algorithm all the impulse response values $h_e[n]$ are implicitly varied on each iteration to obtain the desired optimum approximation, but the values of $h_e[n]$ are never explicitly computed.

Figure B.1 shows the flowchart of Parks-McClellan algorithm.

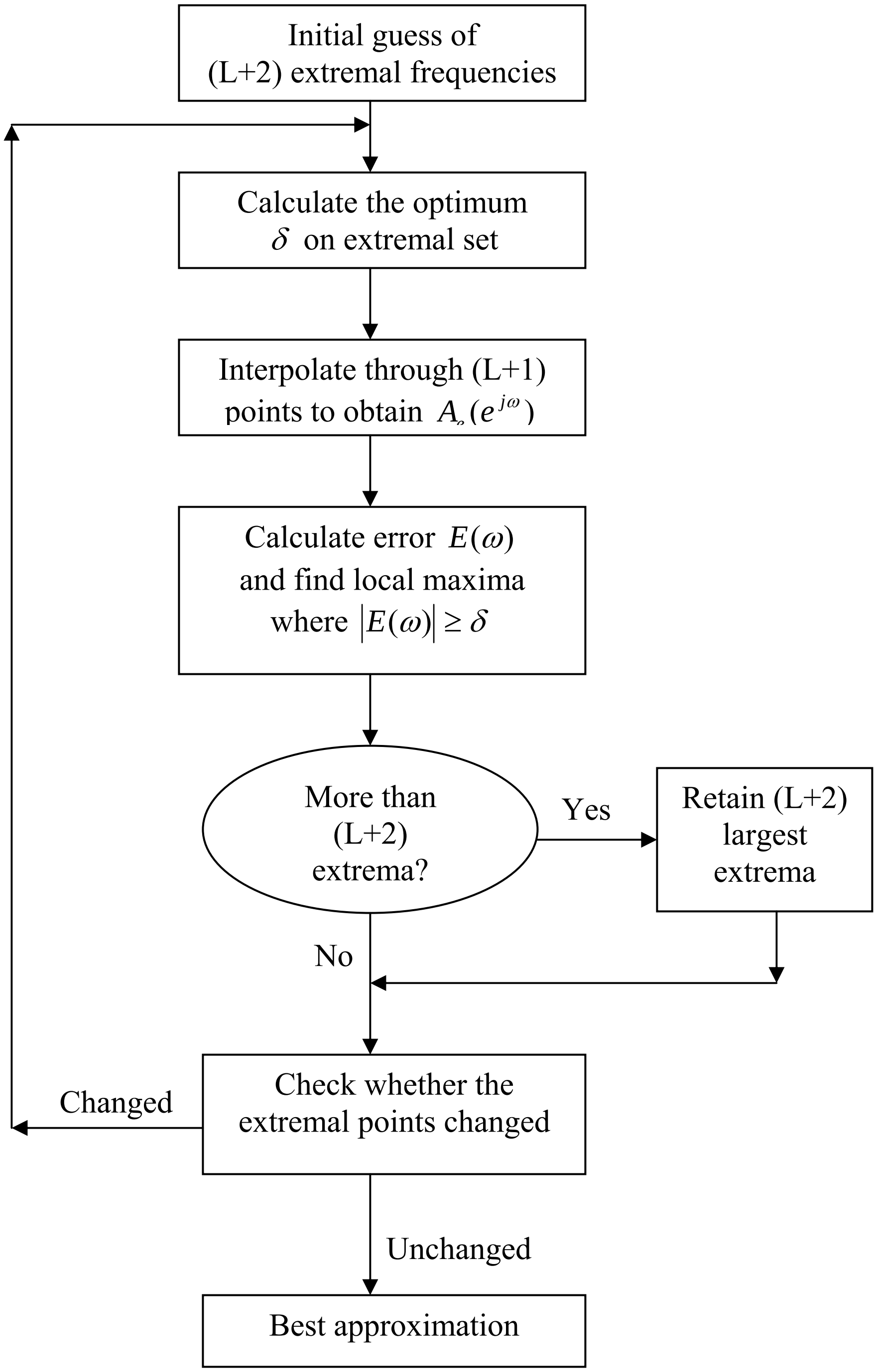

Initial guess of (L+2) extremal frequencies
Calculate the optimum δ on extremal set
Interpolate through (L+1) points to obtain $A_e(e^{j\omega})$
Calculate error $E(\omega)$ and find local maxima where $|E(\omega)| \geq \delta$
More than (L+2) extrema?
Yes
Retain (L+2) largest extrema
No
Check whether the extremal points changed
Changed
Unchanged
Best approximation

Figure B.1 Flowchart of Parks-McClellan algorithm.