395 Wellington Street
Ottawa, Ontario
K1A 0N4

395, rue Wellington
Ottawa (Ontario)
K1A 0N4

Your file Votre référence

Our file Notre référence

NOTICE

The quality of this microform is heavily dependent upon the quality of the original thesis submitted for microfilming. Every effort has been made to ensure the highest quality of reproduction possible.

If pages are missing, contact the university which granted the degree.

Some pages may have indistinct print especially if the original pages were typed with a poor typewriter ribbon or if the university sent us an inferior photocopy.

Reproduction in full or in part of this microform is governed by the Canadian Copyright Act, R.S.C. 1970, c. C-30, and subsequent amendments.

AVIS

La qualité de cette microforme dépend grandement de la qualité de la thèse soumise au microfilmage. Nous avons tout fait pour assurer une qualité supérieure de reproduction.

S'il manque des pages, veuillez communiquer avec l'université qui a conféré le grade.

La qualité d'impression de certaines pages peut laisser à désirer, surtout si les pages originales ont été dactylographiées à l'aide d'un ruban usé ou si l'université nous a fait parvenir une photocopie de qualité inférieure.

La reproduction, même partielle, de cette microforme est soumise à la Loi canadienne sur le droit d'auteur, SRC 1970, c. C-30, et ses amendements subséquents.

Canada

UNIVERSITY OF ALBERTA

Reduction of Pathological Tremor by Functional Electrical Stimulation Using Digital Feedback Control

By

JAMNAN LAW ©

A thesis submitted to the Faculty of Graduate Studies and Research in partial fulfillment of the requirements for the degree of MASTER OF SCIENCE

DEPARTMENT OF ELECTRICAL ENGINEERING

Edmonton, Alberta

Spring, 1995

Acquisitions and
Bibliographic Services Branch

395 Wellington Street
Ottawa, Ontario
K1A 0N4

Direction des acquisitions et
des services bibliographiques

395, rue Wellington
Ottawa (Ontario)
K1A 0N4

Your file *Votre référence*

Our file *Notre référence*

THE AUTHOR HAS GRANTED AN IRREVOCABLE NON-EXCLUSIVE LICENCE ALLOWING THE NATIONAL LIBRARY OF CANADA TO REPRODUCE, LOAN, DISTRIBUTE OR SELL COPIES OF HIS/HER THESIS BY ANY MEANS AND IN ANY FORM OR FORMAT, MAKING THIS THESIS AVAILABLE TO INTERESTED PERSONS.

L'AUTEUR A ACCORDE UNE LICENCE IRREVOCABLE ET NON EXCLUSIVE PERMETTANT A LA BIBLIOTHEQUE NATIONALE DU CANADA DE REPRODUIRE, PRETER, DISTRIBUER OU VENDRE DES COPIES DE SA THESE DE QUELQUE MANIERE ET SOUS QUELQUE FORME QUE CE SOIT POUR METTRE DES EXEMPLAIRES DE CETTE THESE A LA DISPOSITION DES PERSONNE INTERESSEES.

THE AUTHOR RETAINS OWNERSHIP OF THE COPYRIGHT IN HIS/HER THESIS. NEITHER THE THESIS NOR SUBSTANTIAL EXTRACTS FROM IT MAY BE PRINTED OR OTHERWISE REPRODUCED WITHOUT HIS/HER PERMISSION.

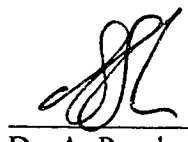
L'AUTEUR CONSERVE LA PROPRIETE DU DROIT D'AUTEUR QUI PROTEGE SA THESE. NI LA THESE NI DES EXTRAITS SUBSTANTIELS DE CELLE-CI NE DOIVENT ETRE IMPRIMES OU AUTREMENT REPRODUITS SANS SON AUTORISATION.

ISBN 0-612-01619-6

Canada

January 17, 1995

This is to indicate that the author of the thesis "Attenuation of Pathological Tremor by Functional Electrical Stimulation using Digital Feedback Control" has my permission to include the diagram titled "wrist flexion-extension movements" from the publication "Attenuation of Pathological Tremors By Functional Electrical Stimulation I. Method", Ann. of Biomed. Eng., vol 20, pp. 205-224, 1992, authored by J. Elek, M. Javidan and myself.



Dr. A. Prochazka

University of Alberta

Library Release Form

Name of Author: Jamnan Jarm-yan Law

Title of Thesis:

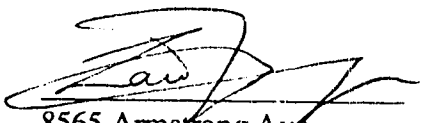
Attenuation of Pathological Tremor by Functional Electrical Stimulation using Digital Feedback Control.

Degree: Master of Science

Year this Degree Granted: 1995

Permission is hereby granted to the University of Alberta Library to reproduce single copies of this thesis and to lend or sell such copies for private, scholarly, or scientific research purposes only.

The author reserves all other publication and other rights in association with the copyright in the thesis, and except as hereinbefore provided, neither the thesis nor any substantial portion thereof may be printed or otherwise reproduced in any material form whatever without the author's prior written permission.



8565 Armstrong Ave,
Burnaby, B.C. Canada
V3N 2H4

1995 January 25

University of Alberta

Faculty of Graduate Studies and Research

The undersigned certify that they have read, and recommend to the Faculty of Graduate Studies and Research for acceptance, a thesis entitled **Attenuation of Pathological Tremor by Functional Electrical Stimulation using Digital Feedback Control** submitted by Jamnan Jarm-yan Law in partial fulfillment of the requirements for the degree of Master of Science in Electrical Engineering

Dr. R. Rink (Supervisor)

Dr. A. Prochazka

Dr. K. Stromsmoe

Dr. R. P. W. Lawson (Chair)

Date: 27 Jan 1995

Abstract

Pathological tremor is a common motor disorder which can prevent a patient from feeding himself or doing other daily tasks. Only about half of the patients respond to medication. Mechanical means of treatment like attaching an inertial or viscous load to the affected limb have been experimented by different people with unsatisfactory results. This type of treatment usually lacks selectivity which means it attenuates both tremor and useful movements. Patients tend to reject this type of treatment because it induces fatigue. In 1989, Prochazka et al. at the University of Alberta suggested the use of functional electrical stimulation (FES) to suppress tremor. The idea is to use a closed-loop feedback system to activate the tremorogenic muscles out-of-phase. They built an analog system and experimented on several patients with satisfactory results.

This thesis presents development of the FES suppression method using digital signal processing. A digital implementation allows easy modification to filter parameters and more complex algorithms. The availability of portable stimulators which use microcontrollers permits filter calculations to be done within the stimulator, achieving a truly small and portable implementation. The bionic glove developed at the University of Alberta is chosen to be the platform for implementing the digital tremor suppression system. Experiments have been done with this system on normal subjects mimicking tremor movements. The results were encouraging. Further research to test the system on patients now awaits.

Acknowledgments

This work and the thesis could never have been performed without the consistent and dedicated support of a group of very valuable and committed individuals.

I wish to thank my supervisor Dr. Arthur Prochazka for his valuable guidance and contribution during my research and especially for him volunteering to be the subject of most of my experiments. I would also like to thank my co-supervisor Dr. Ray Rink for his advice on digital control theories.

Special thank goes to Dr. Brian Andrew who provided a special mechanical brake for evaluation. I would also like to thank Kelly on helping me to setup the brake for experiments.

Practical implementation of my design would not be possible without the bionic glove developed at the University of Alberta. I would like to express my gratitude to my colleagues Michel Gauthier and Zoltan Kenwell for their efforts in perfecting the bionic glove and making special modifications to suit my application.

Table of Contents

1. INTRODUCTION 1

2. TREMOR 5

2.1 Definition of tremor 5

2.2 Types of tremors 6
2.2.1 Physiological tremor 6
2.2.2 Pathological tremor 7
2.2.3 Tremor frequency 8

3. GENERATION OF MOVEMENTS 12

3.1 Movement Generation by the Central Nervous System 12
3.1.1 Background 12
3.1.2 Propagation of Excitation Signal 12
3.1.3 Muscle Contraction 13

3.2 Functional Electrical Stimulation 14
3.2.1 Background 14
3.2.2 The parameters of electrical stimulation 16
3.2.3 Generation of bi-directional movements 21

3.3 Properties of load-moving muscle 21
3.3.1 low-pass frequency response and phase lag 22
3.3.2 non-linearity 22
3.3.3 time-variant 23

4. TREMOR SUPPRESSION USING A MECHANICAL BRAKE 26

4.1 The Structure of the Spring Brake 26

4.2 The control mechanism 27

4.3 The mechanical brake idea 30

5. FILTER DESIGN 31

5.1 Problem definition 31

5.2 Block-diagram analysis 32

5.3 Load-moving muscle model 35

5.4 Linear Time Invariant filter 38
5.4.1 Design Criteria 38

5.4.2 The original linear filter: 39
5.4.3 The improved digital filter 42
5.4.4 Performance analysis of the system 45

5.5 Conditional filter for safety purpose 47
5.5.1 Estimation method 49

6. IMPLEMENTATION 51

6.1 Overview: the filter implementation 51
6.1.1 Fixed-point mathematics 51
6.1.2 Structure of Infinite Impulse Response Filter (IIR filter) 52
6.1.3 Constructing the filter 52

6.2 Implementation using a digital signal processor 55
6.2.1 Implementation on the 68HC16 56
6.2.2 Implementation on the 68HC11 60

6.3 Length transducer as an angle sensor 63
6.3.1 Experimental configuration 63
6.3.2 Proposed portable configuration 66

7. TRIALS IN NORMAL SUBJECTS 71

7.1 Experimental setup 71
7.1.1 Electrode placement 71
7.1.2 Threshold and Gain Setup 72

7.2 Trials with the 68HC16 system 73

7.3 Trials with the bionic glove system (68HC11 system) 75

7.4 The final implementation 80

8. DISCUSSION 81

8.1 Problem analysis 81
8.1.1 Length transducer construction 81
8.1.2 Signal-to-noise ratio 91
8.1.3 Cross-over frequency 92
8.1.4 Possible solutions 95

8.2 The evolution of the original analog device 95

8.3 Useful tools 97

8.4 Conclusion 97

List of Figures

Figure 3-1: Arrangement of thin and thick filaments in skeletal muscle 13
Figure 3-2: The parameters for a typical train of stimulation pulses 18
Figure 4-3 Exploded three-dimensional view of the brake drums 27
Figure 4-4 Top and side views of the brake 28
Figure 4-5 Control of the spring rod by a servo motor 28
Figure 5-1: Block diagram for tremor suppression feedback system. *The transfer function $G_1(s)$ converts the CNS command signals to actual movements.* 32
Figure 5-2: Digital equivalent block diagram for tremor suppression feedback control system 35
Figure 5-3 Actual data from normal subjects 36
Figure 5-4: Frequency response of the load-moving wrist muscle model. *1) continuous frequency response; 2) digital zero-order hold equivalent of the continuous frequency response, with T 0.01s* 37
Figure 5-5: Frequency response of the original tremor suppression filter. *1) continuous version; 2) bilinear transformation of the continuous filter with T = 0.01s* 41
Figure 5-6: The open loop frequency response of the digitized wrist transfer function $D(z)$ with the digitized original high-pass filter 42
Figure 5-7: Frequency response of the improved digital filter (solid line) and the digitized original filter (dotted line). *Notice the extended range of low gain at frequencies 0-1Hz and a greater phase advance overall.* 43
Figure 5-8: Pole-zero map of the improved digital filter 44
Figure 5-9: The open loop frequency response of the digitized wrist transfer function $D(z)$ with the improved digital filter 45
Figure 5-10: Overall closed loop frequency response of marginally stable tremor suppression systems with 1) the digitized original filter; 2) the improved digital filter. 46
Figure 5-11: Left - the gain from frequencies 0 to 13Hz. 8Hz is the cut-off frequency for the non-linear filter. Right - the gain for frequencies up to the nyquist frequency. All gains above unity will saturate the filter. 48
Figure 6-1: Transposed Direct Form II 55
Figure 6-2: Frequency response plots at different stages of the cascaded system. *Graphs (a) to (e) are the frequency response plots of the individual sections. Graphs (f) to (i) are the frequency response plots at the end of the sections including all previous sections.* 59
Figure 6-3: Effect of coefficient quantization on frequency response 60
Figure 6-4: Difference from original frequency response after coefficient quantization 61
Figure 6-5: Length transducer experimental setup 63
Figure 6-6: A simplified model of the experimental length transducer configuration 64
Figure 6-7: Length to wrist angle relationship with the experimental setup. *Notice that the relationship is quite non-linear close to 180 degrees. The useful range of this transducer is very limited.* 65
Figure 6-8: A portable length sensor placement on the arm 66
Figure 6-9: A simplified model for the portable length transducer configuration. 66
Figure 6-10: Relationship between the angle and length with $y/x = 1$ 67
Figure 6-11: Relationship between the angle and length with $y/x = 3$ 67
Figure 6-12 Structure of the length transducer using an elastic string 68
Figure 6-13: Structure of LVDT 68
Figure 7-1: Surface electrode and stimulator setup 71
Figure 7-2: The 68HC16 system experimental setup 73
Figure 7-3: 68HC16 tremor suppression system operation on a normal subject performing a tremor-like 4Hz movement. This signal was recorded from the length transducer directly. 74
Figure 7-4: Fourier transform plots from the length signal before the after the 68HC16 tremor suppression system was turned on. *As seen from the plot, an attenuation of around 10dB is achieved. This corresponds to a 1:3.1 attenuation.* 74
Figure 7-5: 68HC11 tremor suppression system experimental setup 75

Figure 7-6: 68HC11 tremor suppression system working on a normal subject performing a tremor-like 4Hz movement. This signal was recorded directly from the length transducer. 76
Figure 7-7: Fourier transform plots from the length signal before and after the 68HC11 tremor suppression system was turned on for a movement similar to one shown in Figure 7-6. 77
Figure 7-8: 68HC11 tremor suppression system on a normal subject performing a slow movement. The signal was recorded from the length transducer directly. 78
Figure 7-9: 68HC11 tremor suppression system on a normal subject performing a tremor-like 4Hz movement mixed with a slow intended movement. The signal was recorded directly from the length transducer. 78
Figure 7-10: Fourier transform plots from the length signal before and after the 68HC11 tremor suppression system was turned on for a movement similar to the one shown in Figure 7-9. *Notice the intentional low frequency component of the signal, which was not attenuated.* 79
Figure 8-1: Weaknesses of the portable length transducer. *The construction of this portable length transducer has some weaknesses. However, until a better length/angle sensor can be found, these problems must be dealt with carefully.* 82
Figure 8-2: A plot showing a hysteresis relationship between the length and the wrist angle. 83
Figure 8-3: Actual recorded length to wrist angle relationship 84
Figure 8-4: A best fit cubic curve for the recorded data using least square algorithm. *The best fit cubic curve is shown as the solid line. The relationship is nearly but not quite linear. Notice that this curve obtained from actual data agrees with the theoretical relationship proposed in chapter 6 for x/y = 1 (shown as dash line in the plot)* 85
Figure 8-5: Relationship between length and wrist angle with static friction. *With large static friction between the metal rod and the guiding tube, the length to angle relationship became very non-linear and chaotic.* 87
Figure 8-6: Fourier spectrum plots showing the effect of large static friction on the length transducer. *The top Fourier spectrum is obtained from the signal of an angle sensor at the wrist. A normal subject was mimicking a 4Hz to 5Hz tremor movement, which is shown as a peak around the frequency range. The Fourier spectrum at the bottom is obtained from the length transducer signal collected simultaneously from the same tremor movement. Notice that high amplitude harmonics exist in the length transducer signal. These harmonics are nearly non-existing when the length transducer is properly lubricated (Figure 8-7).* 88
Figure 8-7: Fourier spectrum plots obtained from a properly constructed and lubricated length transducer. *A properly constructed and lubricated length transducer gives a faithful translation of the wrist angle. The top plot shows the Fourier transform plot from the angle sensor signal and the bottom is for the length sensor. These signals were recorded from a person mimicking a 4-Hz tremor. The two plots are nearly identical.* 89
Figure 8-8: The Fourier spectrum plots of the signals the 68HC11 internal A-to-D converter saw. *1) solid line is the spectrum for the input signal from the portable length sensor. 2) dotted line is the spectrum from a signal generator input at 4Hz. The quantization noise for both inputs are comparable. This proves that the length transducer does not introduce any more high frequency noise. The unwanted harmonics can be explained by the static friction discussed above.* 90
Figure 8-9: Fourier spectrum plots for both the input and output of the 68HC11 digital filter in an open loop system. *These are numbers directly obtained from the 68HC11 program. Notice that the signal-to-noise ratio decreases significantly because the high frequency components were never attenuated.* 92
Figure 8-10: Overall closed-loop gain of the system and sample input and output. *The system is just below the instability margin. Notice that the second harmonic is amplified dramatically.* 94

List of Tables

Table 2-1 Frequencies of common types of tremors 9
Table 6-1: Poles and zeros pairing for the improved digital filter 57

List of Symbols, Nomenclature or Abbreviations

A/D	analog to digital converter
Ca^{2+}	Calcium ion
CNS	Central nervous system
DASA2	Data Acquisition, Stimulation and Analysis, a program developed by Jamnan Law
DC	direct current
DSP	Digital signal processor, e.g. 68HC16
EEPROM	Electrically erasable programmable read-only memory
FES	Functional electrical stimulation
FIR	Finite impulse response
IIR	Infinite impulse response
LED	Light-emitted diode
LTI	Linear time-invariant
LVDT	Linear voltage differential transducer
MATLAB	A program especially designed for signal-processing theory calculations
RAM	Random-access memory
S/N	Signal to noise ratio
ZOH	Zero-order hold

1. INTRODUCTION

Tremor is the most common of all motor disorders. It can be described as an involuntary rhythmical oscillation about a fixed point or trajectory. There are two main classes of tremor: physiological and pathological. Physiological tremor usually results from muscle fatigue or emotional stress, and thus is not considered an illness. It is characterized by a small amplitude movement with a frequency of 8 to 12 Hz (compared to 0 to 1 Hz bandwidth of most intentional movements). Pathological tremor, on the other hand, results from damage to the nervous system. This type of tremor is of lower frequency but larger amplitude, and can manifest itself either while the patient is at rest or is trying to execute a motor task. Severe pathological tremor can prevent a patient from feeding himself or doing other tasks. This is the type of tremor dealt with in this thesis. Unless otherwise specified, the word 'tremor' refers to pathological tremor for the rest of this thesis.

Of the three main types of pathological tremor: parkinsonian, essential and cerebellar, only 50% respond to medication *[Weiner et al., 1989]*. Cerebellar tremor, often seen in multiple sclerosis patients, is particularly intractable. Different non-medication treatments have been suggested and put into practice. An inertial load can be attached to the affected limb to act like a second order mechanical filter *[Hewer et al., 1972]*. This is a simple and easy treatment, but obviously not selective enough and it attenuates both the tremor movement and useful movement. The patient tends to reject this treatment mainly because it induces fatigue. Viscous loading is another alternative suggested *[Morrice et al., 1987]*, the idea being to attach a device to the affected limb, which would offer an impedance to

movement proportional to limb velocity. This method has the same drawback as a inertial load, so it is not used practically either.

In 1989, Prochazka et al. at the University of Alberta discovered that feedback-controlled functional electrical stimulation (FES) could attenuate tremor effectively and selectively *[Elek and Prochazka, 1989] [Javidan et al., 1990]*. FES is used to stimulate a muscle to contract involuntarily by passing a train of small and short electric current pulses through the muscle. The amplitude of the contraction, thus the force generated by the muscle, can be adjusted by modulating the duration or amplitude of the electric pulses. Using this technique, tremor can be counteracted by using closed-loop functional electrical stimulation to activate the tremorogenic muscles out-of-phase. With a good knowledge of the transfer characteristics of human muscles, and an appropriate feedback controller, the fast tremorous components of movement can be suppressed, leaving slower intentional movements almost unaffected. A continuous feedback controller built from operational amplifiers was implemented and experimented on several patients *[Prochazka et al., 1992]*, and the results were quite satisfactory.

One of the main problems in making the system described above practical was its size. The filter, built from operational amplifiers (op-amps) and resistors, and the stimulator, was far from portable. Although it is very easy to reduce the size of the filter significantly by using a printed circuit board design, the inflexibility in the ability to change the filter parameters made the analogue device unfeasible.

To solve many of the problems of the present device, a digital controller was suggested *[Prochazka, personal communication]* as a replacement for the analogue controller in the first prototype. First of all, a digital system would allow quick changes to the parameters of the controller, not to mention the relative ease in changing the order of the overall transfer function. Adaptive (self-tuning) and nonlinear feedback control that may otherwise be very difficult, if not impossible to incorporate into an analogue system, is feasible in a digital system. Digital signal processors now available commonly allow adaptive control and parameter identification techniques to be applied readily.

A more important reason for using a digital controller is the ease in combining the stimulator with the filter. Most new FES stimulators already use microcontrollers. The idea is to include the filter calculation algorithm in the software that also controls the stimulator. This enables the system to be reduced dramatically in size. A portable, self-contained device is the ultimate goal of this thesis research.

In this thesis, the development of such a system is described. The first design uses the old experimental setup (the stimulator with the continuous filter) and replaces the filter with a digital version. After some successful trials on normal subjects, a portable system was then implemented with the same algorithm in a portable stimulator/controller. Although some successful trials have been done with this portable system on normal subjects, several problems still have to be overcome before a robust system is ready for clinical trials on patients.

References

Elek, J. and A. Prochazka, "Attenuation of wrist tremor with closed-loop electrical stimulation of muscles", J. Physiol. 414:17P, 1989

Hewer, L. R., R. Cooper, M. H. Morgan, "An investigation into the value of treating intention tremor by weighting the affected limb", Brain 95, pp.579-590, 1972

Javidan, M., J. Elek, A. Prochazka, "Tremor reduction by functional electrical stimulation.", Neurol. 40:369, 1990

Morrice, B. L., R. G. Lee, W. J. Becker, D. G. White, J. A. Hoffer, "Viscous loading improves manual tracking accuracy in humans with cerebellar incoordination", Soc. Neurosci. Abstracts, 13:1700, 1987

Prochazka, A., J. Elek, M. Javidan, "Attenuation of pathological tremors by functional electrical stimulation I: Method", Ann. of Biomed. Eng., vol 20, pp. 205-224, 1992a

Prochazka, A., J. Elek, M. Javidan, "Attenuation of pathological tremors by functional electrical stimulation II: Clinical evaluation", Ann. of Biomed. Eng., vol 20, pp. 225-236, 1992b

Weiner, W. J., A. E. Lang, "Tremor in: movement disorders: a comprehensive survey", Futura Publishing, pp. 221-256, 1989

2. TREMOR

In this chapter, a brief review on the types of tremors and their characteristics is given. Most of the details, e.g. the neurophysiology of tremor generation or medical treatment methods, are deliberately omitted because the primary focus of this thesis is on the engineering aspects of functional electrical stimulation..

2.1 Definition of tremor

In the literature to date, there has not been a commonly accepted precise definition for tremor. In 1914, Dejerine characterized tremor as

> "involuntary rhythmic oscillations about a position of equilibrium which involve either the whole body or part of it". *[Dejerine, 1914]*

R.B. Stein and R. G. Lee in the 'Handbook of Physiology' say

> "... tremor may be defined operationally as an involuntary movement characterized by a rhythmical oscillation about a fixed point or trajectory." *[Stein and Lee, 1977]*

Although there have been many definitions for tremor, most have a few keywords in common: involuntary, rhythmical and oscillation. Tremor results from the alternating or synchronous contractions of reciprocally innervated antagonistic muscles (flexor and extensor). It has a relatively fixed periodicity and has an amplitude and waveform which are to some extent invariant over a period of time. These characteristics enable clinicians

to distinguish tremor from other movements with irregular appearance such as myoclonus, chorea or clonus.

2.2 Types of tremors

From a clinical point of view, tremor occurring in human subjects can be classified into two sub-divisions: physiological and pathological.

2.2.1 Physiological tremor

Physiological tremor is characterized as a low-amplitude oscillation found in almost all normal subjects. The rate of oscillation depends on the age of the subject or the part of the body being examined and is usually around 8 to 12 Hz. Because it is small in amplitude, physiological tremor is unnoticeable most of the time. The tremor becomes more apparent when the body is affected by fatigue, anxiety or certain pharmacological agents like alcohol.

The origin of physiological tremor remains uncertain, despite the fact that it has been the subject of numerous investigations. One explanation, which may be of interest to control engineers, is that the tremor is a result of instability of the servomechanism associated with the spinal stretch reflex *[Halliday and Redferarn, 1956] [Henatsch, 1967] [Joyce and Rack, 1974] [Lippold, 1970]*. The spinal stretch reflex is a feedback control system within the body to control the position or force of the muscle. When a person is nervous, especially when a dangerous situation arises, the gain of this feedback system increases, perhaps in preparation for faster movements (i.e. short rise time). Similarly, when a person is tired, the delay around the loop becomes longer (larger phase lag). In both cases, the feedback

system will move to a less stable state. When the system becomes only marginally stable, oscillations (tremors) will occur.

Since physiological tremor is not considered an illness, a treatment is not necessary. The system developed in this thesis is intended to suppress another kind of tremor: pathological tremor.

2.2.2 Pathological tremor

This kind of tremor is a result of nervous system disorder. There has not yet been universal agreement as to how to classify distinct types of pathological tremors. Nevertheless, the most widely accepted clinical classification is based on the posture or type of volitional activity during which the tremor is likely to occur *[Marshall, 1970]*. Using these criteria, pathological tremors can then be divided into three major categories:

1. resting tremor
2. postural tremor
3. intention tremor

Postural and intention are also collectively known as action tremor.

2.2.2.1 Resting tremor

This type of tremor only occurs when the limb affected is at rest and disappears with the initiation of voluntary movement. The classical example of this type of tremor is the Parkinsonian tremor, which occurs in Parkinson's patients. This was originally described by Parkinson *[Parkinson, 1817]*. It is characterized by a lower-frequency oscillation than

physiological tremor (around 4 to 7Hz) with a larger amplitude. Notice that although Parkinsonian tremor is most commonly a resting tremor, in some patients it also manifests as an action tremor.

2.2.2.2 Postural tremor

This type of tremor is most apparent when the arms are held outstretched to maintain a fixed posture. The most common example of this type of tremor is the "essential" tremor. This is characterized by a frequency of around 5 to 7Hz. As the term essential implies, the cause of this form of tremor is not well understood.

2.2.2.3 Intention tremor

This tremor occurs during the execution of an intended voluntary movement. However, there is little tremor at rest or during maintenance of a fixed position. Usually the frequencies of oscillations lie between 2 and 4Hz, and the amplitude becomes progressively larger as the final position is approached. This type of tremor is most commonly associated with cerebellar disorders. *[Marshall, 1968] [Fahn, 1972] [Jankovic and Fahn, 1980] [Marsden, 1984] [Weiner and Lang, 1989]*

2.2.3 Tremor frequency

Mathematical analysis techniques such as Fourier transformation and power spectral analysis allow a separation of complex tremors that is often difficult by clinical examination alone. Spectral analysis of tremor was first introduced by Halliday and Redfearn *[Halliday and Redfearn, 1956]*. This and many of the research that followed *[Merton et al, 1967] [Sutton and Sykes, 1967a, 1967b] [Stephens and Taylor, 1974]* reported that a

consistent peak frequency was observed for individual types of pathological tremors. This consistency is the basic principle underlying the research reported in this thesis. With the knowledge of the dominant frequency of tremor movements for a patient, an appropriately tuned filter can be designed for the tremor suppression system. Table 2-1 shows typical frequencies of common types of tremors.

Frequency (Hz)	Type of tremor
2.5 to 4	Cerebellar tremor
4 to 5	Parkinson's disease (rest tremor)
5.5 to 7	Essential tremor Parkinson's disease (postural tremor)
7 to 12	Physiological

Table 2-1 Frequencies of common types of tremors

References

Dejerine, J., "Semiologie des affections du systeme nerveus", Masson, Paris, 1914

Fahn, S., "Differential diagnosis of tremors", Med. Clin. N. Am 56:1363-75, 1972

Halliday, A. M., and J. W. T. Redferarn, "An analysis of the frequency of finger tremor in healthy subjects", J. Physiol. London 134, pp. 600-611, 1956

Henatsch, H. D., "Instability of the proprioceptive length servo; Its possible role in tremor phenomena." in Neurophysiological Basis of Normal and Abnormal Motor Activities, New York: Raven, pp. 75-89, 1967

Jankovic, J. and S. Fahn, "Physiologic and pathologic tremors. Diagnosis, mechanism, and management", Ann. Intern. Med. 93:460-465, 1980

Joyce, G. C., and P. M. H. Rack. "The effects of load and force on tremor at the normal human elbow joint", J. Physiol. London 240: 375-396, 1974

Lippold, O. C., "Oscillation in the stretch reflex arc and the origin of the rhythmical 8-12 c/s component of physiological tremor", J. Physiol. London 206: 359-382, 1970

Marsden, C. D., "Origins of normal and pathological tremor", in Movement Disorders: Tremor, London: Macmillan, 1948

Marshall, J., "Clinical symptomatology of disease of the basal ganglia.", in Handbook of Clinical Neurology, vol 6, Diseases of Basal Ganglia. Amsterdam: North-Holland, pp. 809-825, 1968

Marshall, J., "Tremor in: Handbook of Clinical Neurology", Amsterdam: North Holland, vol 6, pp. 809-825, 1970.

Merton, P. A., H. B. Morton and C. Rashbass, "Visual feedback in hand tremor", Nature 216, pp. 583-584, 1967

Parkinson, J., "An essay on the shaking palsy", London: Wittingham & Newland, 1817

Stein, R. B., and R. G. Lee, "Tremor and clonus" in Handbook of Physiology, pp. 325-343, 1977

Stephens, J. A., and Taylor, A., "The effect of visual feedback on physiological muscle tremor", Electroenceph Clin Neurophysiol 36, pp. 457-464, 1974

Sutton, G. C. and K. Sykes, "The effect of withdrawal of visual presentation of errors upon the frequency spectrum of tremor in a manual task.", J. Physiol 190, pp. 281-293, 1967a

Sutton, G. C. and K. Sykes, "The variation in hand tremor with force in healthy subjects", J. Physiol 191, pp. 699-711, 1967b

Weiner, W. J., and A. E. Lang, "Tremor in movement disorders: a comprehensive survey", Futura Publishing Co. Inc., 1989

3. GENERATION OF MOVEMENTS

3.1 Movement Generation by the Central Nervous System

3.1.1 Background

The Central Nervous System (CNS) controls all voluntary movements in humans. A single movement includes two events:

1. the propagation of an excitation signal, and
2. the conversion of this signal into mechanical force, which moves the limb.

3.1.2 Propagation of Excitation Signal

Excitation signal is usually transmitted by using electrical pulses. However, this propagation method is very different from the conduction of electricity on copper wires. The media in which the electricity is conducted are called nerves. These are very long and narrow extensions (axons) of cells called neurons. Basically, chemical changes at the nerve ending near the CNS initiate active ions flux across the local cell boundaries. This creates an imbalance of electrical charge across the cell wall. When voltage difference reaches a threshold, this induces further chemical changes in adjacent part of the cell axon, causing a similar voltage difference. A series of such events will cause an electrical pulse (action potential) to move along the axon. This conduction is an active, self-propagating process, and the pulse moves along the axon at a constant amplitude and velocity. The process is often compared to what happens when a match is applied to one end of a train of

gunpowder; by igniting the powder particles immediately in front of it, the flame moves steadily down the train to its end. At the end of the axon is the muscle fibre, which is the smallest distinguishable unit of the muscle.

3.1.3 Muscle Contraction

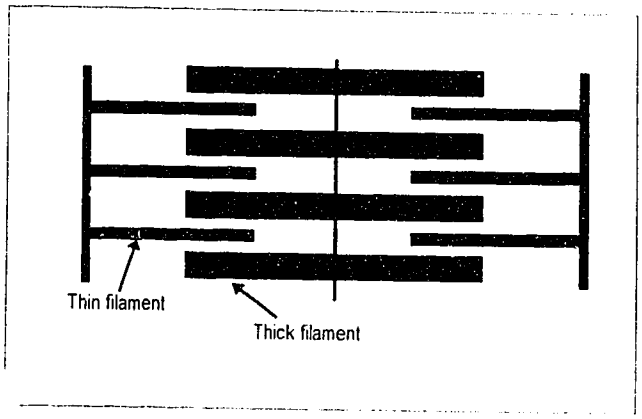


Figure 3-1. Arrangement of thin and thick filaments in skeletal muscle

Muscle is responsible for converting an electrical signal into a mechanical contraction. It is generally divided into 3 types, skeletal, cardiac, and smooth, although smooth muscle is not a homogeneous single category. Most voluntary movements involve only skeletal muscle, which is the type found at the wrist. Skeletal muscle is made up of individual muscle fibres, which are further made up of myofibrils. These myofibrils contain interleaving thin filaments and thick filaments arranged as shown in Figure 3-1. The thick filaments, which are about twice the diameter of the thin filaments, are made up of myosin; the thin filaments are made up mainly of actin.

An action potential arriving at a muscle fibre triggers the release of Ca^{2+}. This will initiate a series of chemical reactions resulting in the formation of cross-linkages between actin and myosin and sliding of thin filaments on thick filaments, shortening the muscle fibre.

muscle twitch. The duration of the twitch varies with the different types of muscle fibres, which are fast fatiguable, fast fatigue resistant and slow fatigue resistant. Fast fatiguable muscle fibres have the shortest twitch duration, whereas slow fatigue resistant muscle fibres have the longest. Most muscles in the body are a mixture of the three.

3.2 Functional Electrical Stimulation

3.2.1 Background

As described above, the excitation signal is propagated as electrical pulses. The possibility of exciting the muscle with electric current appears to be very attractive because electricity is easily generated. As early as the eighteenth century, applications of electricity in medicine were already put into practice *[Reswick, 1973]*. Perhaps the most novel and famous recent application is the artificial cardiac pacemaker which uses electrical stimulation of cardiac muscle to stabilize the rhythmic beating of the heart *[Zoll, 1952]*. At about the same time as artificial cardiac pacemaker became a reality, electrical stimulators used to stimulate skeletal muscles were also in development by several research groups, and with the advance of electronics, these had been implemented and used experimentally on paralyzed patients *[Reswick, 1973]*. The idea of using electricity in muscle stimulation known as functional electrical stimulation (FES) has subsequently became an active research field in rehabilitation *[Liberson et al., 1961] [Vodovnik, 1971]*. In 1967, FES was described as

movement" *[Gracanin et al., 1967]*

In spinal cord injured patients, whose CNS loses part of the nerve connections to the motor system (muscle), FES is a very promising alternative to rehabilitation modalities. Excellent research work has been performed in the use of FES in the upper extremities producing functional hand rehabilitation in quadriplegics. *[Peckham, Mortimer and Marsolais, 1980]*. An even more exciting application is the use of FES in the lower extremities of paraplegic patients, producing the standing and biped gait of completely paralyzed persons. The first application of FES to a paraplegic patient was reported by Kantrowiz in 1963 *[Kantrowiz, 1963]*. Using surface electrodes to stimulate the quadriceps and glutei muscles of a paraplegic patient, the patient was able to achieve a few minutes of erect standing. The next similar trial was made by Wilemon et al. *[Wilemon et al., 1970]* and Reswick *[Reswick, 1972]* at Rancho Los Amigos Hospital in California. However, the above trails were only single trials and no research or clinical activities followed. First continual research efforts were started by Kralj et al. in 1972 *[Kralj, 1973]*. The data were gathered on muscle forces exerted by FES in 50 paraplegic patients with different etiology, time past injury, sex and age. The influence of FES training on muscle force and fatigue resistance was documented. These information formed the basis for more extensive research in the area.

Electrical stimulation exists in many different forms. Basically, it is performed by passing a series of rectangular electrical pulses (either constant current or constant voltage, as shown in Figure 3-2) through an electrode into the axons entering a muscle. The current density (in the range of 10 A/m^2) determines the amplitude of muscle contraction. The electrode, either surface or implanted, is carefully placed near where the motor nerve enters the muscle of interest. This position is termed 'motor point' which is defined as an area on the surface of the skin which, when stimulated, produces the greatest isometric torque about a joint *[Crochetiere et al., 1967].*There are many configurations to consider which best suit the application, and there are different parameters that can be changed during the stimulation to control the force and time course of muscle contraction.

3.2.2.1 Implanted or Surface Electrode

Surface Electrode

Surface Electrodes are basically metallic and electrolyte interfaces placed over the skin. Electric current passes through the skin to the muscle nerve fibres. This provides a convenient means of stimulation for experimental purposes. The electrodes are readily used on any subjects (either normal subjects or patients) and are widely available and relatively inexpensive. However, there are disadvantages when using surface electrodes:

- These electrodes have to be removed and carefully reapplied to the appropriate position every time they are used. Apart from being time consuming, finding the

the stimulation depends heavily on precise electrode position.

- High electrical stimulation amplitudes applied transcutaneously by surface electrodes can result in local pain, strong muscle spasms or even burn injuries to the skin when applied inappropriately.

- It is not always possible to obtain satisfactory stimulation selectivity because of the relatively large surface area of the commonly used electrodes.

Implanted Electrode

Electrode implantation requires a serious commitment of the patient and clinicians, and the cost of implantation surgery is often high. Permanently implanted electrodes are not ideal for experiments because inappropriate command signals into the electrodes can cause electrode damage. However, they have many advantages over surface electrodes:

- The main advantage of implanted electrodes is presented by the low stimulation currents or voltages required for muscle activation. In this way, smaller energy sources are needed, providing more specific excitation possibilities and considerably reduced size of stimulators.

- Since the position of the electrode is fixed after implantation, variations in contraction due to position change become minimal.

- It is possible to stimulate a very specific muscle because of the small electrode size.

It is a usual practice that monophasic pulses are used in surface electrodes and biphasic pulses are used in implanted electrodes. Biphasic pulses minimize electrolytic corrosion of the implanted electrodes *[Kralj and Bajd, 1989]*. Corrosion is not a concern with the large surface area, disposable electrodes used on the skin.

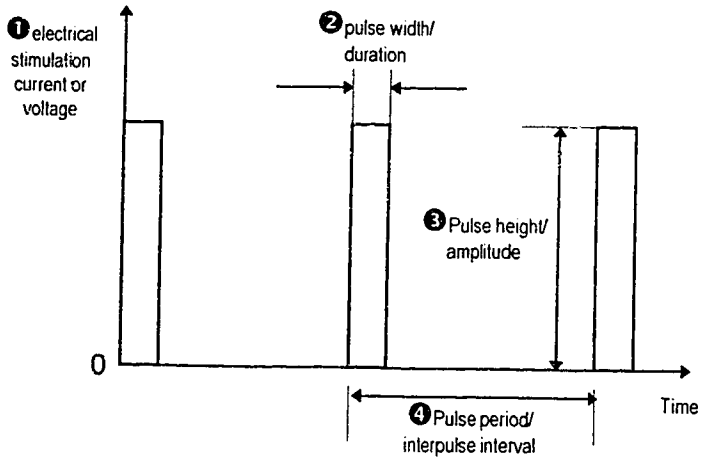


Figure 3-2: The parameters for a typical train of stimulation pulses

3.2.2.3 Voltage or current control

Muscle contraction can be controlled by maintaining either the current output or the voltage output. An important difference between the two types becomes evident in the case of an incomplete contact between a surface electrode and the skin. In the case of a constant-current stimulation, the smaller effective contact surface results in a much greater current density, which can cause skin burns. For a constant-voltage stimulation, the smaller contact surface increases the resistance thus a decrease in the current. This decrease in current will offset the decrease in area, leaving the current density nearly unaffected. Notice that this problem only occurs when a surface electrode is used. For

will be equally effective.

One problem associated with constant-voltage stimulator is that the rectangular voltage pulse will be distorted across the electrodes *[Crochetiere et al., 1967]*. The impedance across two points on the body is capacitive so voltage cannot change instataneously across the two points. Only constant-current rectangular-pulse type of stimulus will retain its shape when impressed across stimulating electrodes.

In all the experiments reported in this paper, surface electrodes and monophasic current-controlled stimulators were used. The electrical stimulator has a limit on how much current it can output. Therefore the possibility of injuries caused by a decrease in electrode contact area is small.

3.2.2.4 Stimulation frequency

A single stimulation pulse produces only a short-lived muscle twitch which generally lasts for no more than 0.2s and the duration being highly dependent on the type of muscle fibre (fast fatiguable, slow fatigue resistant or fast fatigue resistant). If the stimulation period is close to or longer than this twitch period, the muscle force produced will be seen as individual twitches. If the frequency of stimulation pulses increases to the point that, between two twitches, there is no time left for muscle relaxation, a summation of muscle force will occur. The twitching is considerably reduced at stimulation frequencies above 15Hz. At higher frequencies, the response of the muscle becomes smooth, a condition known as tetanic contraction. The frequency at which tetanic contraction occurs is called

fusion frequency. For the human wrist muscle, the fusion frequency is around 20Hz, which poses a lower bound for the stimulation frequency.

Carroll et al. conducted a set of experiments to examine the effects of different stimulation frequencies on muscle contraction *[Carrol et al., 1989]*. In particular, the effects on the rise time (delay in muscle response) and muscle fatigue were studied. In general, within the range of stimulation periods of interest (7ms to 62ms), the rise times decreased as the stimulation periods were decreased. It was also found that a short stimulation period of around 12ms induced a phenomenon known as "high-frequency fatigue". This is a rapid torque decline during stimulation at high frequencies. To avoid this phenomenon, the stimulation frequency should be chosen to be as low as the application permits. Compromises must be made to have a fast enough response time without inducing rapid fatigue in the muscle. In the experiments reported in this thesis, a stimulation frequency of 25-35Hz was used.

3.2.2.5 Pulse height/width

Varying either the amplitude (height) or the duration (width) of stimulus pulses are methods of recruitment modulation of contraction. Recruitment modulation involves varying the number of muscle fibres activated, thus controlling the force of contraction. In a paper by Crago et al, both of these recruitment methods were studied. The conclusion was:

> "Recruitment has similar input-output relationships regardless of whether pulse amplitude or pulsewidth is modulated. There is however an important reason for modulating pulsewidth at the highest stimulus amplitude obtainable... Pulsewidth modulation requires less charge transfer per stimulus pulse at any given force level when compared to pulse amplitude modulation, and it is important to minimize charge transfer to prevent damage to tissue." *[Crago et al., 1990]*

It was made clear that a pulse-width modulation is preferred. For the experiments reported in this paper, whenever possible, pulse-width modulation was used.

3.2.3 Generation of bi-directional movements

Remember that muscle can only pull, but never push. To achieve movements in both directions, there are a pair of muscles for each joint: the extensor and the flexor. They work to generate force in opposite directions. When one is contracting, the other will be relaxing, and vice versa. When performing functional electrical stimulation, two separate cathodes are needed for the flexor and the extensor, although a common anode can be used. The two cathodes transmit control signals that are out of phase with each other, making sure that the two muscles will not contract at the same time (a condition known as co-contraction).

3.3 Properties of load-moving muscle

Since it is the movement of the limbs that is of interest, the dynamic properties of the limb must be considered in addition to the force generation by the muscle alone. In this thesis,

the muscle and the limb (the load) it is attached to are collectively called a load-moving muscle. It is assumed that there is no extra load is on the limb.

3.3.1 low-pass frequency response and phase lag

The muscle fibre does not take the electrical pulse and transform it into a mechanical force in a simple one-to-one relationship. The fibre is unable to transform high frequencies in the modulation of electrical pulses into corresponding fluctuation in force hence acts as a low-pass filter in terms of frequency response. This important property of muscle derives from the fact that the time course of a muscle twitch produced by a single pulse is longer (by several hundred milliseconds) than that of the pulse itself. Before the muscle has time to relax, a second pulse could come and cause another twitch. Because the relaxation is incomplete with high frequency pulses, the muscle will generate smaller movements.

Most of the phase lag comes from the dynamic properties of the limb. The inertia of the limb gives rise to a phase lag in the frequency response. If there is extra load on the limb, for example, if the patient is holding a cup, the phase lag will be increased.

3.3.2 non-linearity

- The amount of contractile force that a muscle can produce depends markedly on its initial length. There are two factors in this non-linearity: 1) the force generating mechanism, and 2) the stiffness of muscle. As described above, the muscle fibre consists basically of thin and thick filaments. There are cross-bridges between the two filaments which are the main force generators. The more cross-bridges

formed, the stronger the contraction is. At different lengths, different numbers of cross-bridges can be formed, giving rise to different contraction strengths.

- Like a spring, muscles generate a restoring force when they are stretched beyond their resting length. This passive force (stiffness) will be added on top of the active force generated by the muscle fibres. *[Matthews, 1959] [Rack and Westbury, 1969]*

- In response to external stimulation (e.g. functional electrical stimulation), the muscle exhibits threshold and saturation characteristics. The stimulation must be greater than the threshold level before any contraction can result. Beyond the threshold, the muscle reaction becomes fairly linear (see the previous two points). However, after a maximum contraction is obtained, any greater stimulation will not increase the force further: the response becomes saturated.

3.3.3 time-variant

One obvious property of muscle is that the force generated changes over time. The reason could be that the muscle becomes fatigued after prolonged stimulation (short term effect), or becomes stronger after appropriate training (long term effect). No matter the cause, a given stimulus will have different results at different time; that is, the system is time-variant.

References

Carroll, S. G., R. J. Triolo, H. J. Chizeck, R. Kobetic, and B. Marsolais, "Tetanic responses of electrically stimulated paralyzed muscle at varying interpulse intervals," IEEE Trans. Biomed. Eng., vol. BME-30, pp. 644-653, July 1989.

Crago, P. E., P. H. Peckham and G. B. Thrope, "Modulation of muscle force by recruitment during intramuscular stimulation", IEEE Trans. Biomed. Eng., vol BME-27, no. 12, pp. 644-651, Dec. 1990

Crochetiere, W. J., Vodovnik., L., and Reswick., J. B., "Electrical Stimulation of Skeletal Muscle - A study of muscle as an actuator", Med. & biol. Engng., Vol5. pp. 111-125, 1967

Gracanin, F., T. Preves, and J. Trontelj, "Evaluation of use of functional electronic perennial brace in hemiparetic patients", in Proc. Int. Symp. External Control Human Extremities, Dubrovnik, Yugoslavia, August 29 to September 2, 1967, 198.

Kantrowitz, A., "Electronic physiologic aids", Report of the Maimonides Hospital, Brooklyn, NY, 1963

Kralj, A. R. and T. Bajd, "Functional electrical stimulation: standing and walking after spinal cord injury", CRC Press Inc, 1989

Kralj, A., Grobelnik, S., and Vodovnik, L., "Electrical stimulation of paraplegic patients - feasibility study", in Proc. Int. Symp. External Control Human Extremities, Dubrovnik, Yugoslavia, August 28 to September 2, pp. 561, 1973

Liberson, W. T., H. J. Holmquest, D. Scott and A. Dow, "Functional electrotherapy: stimulation of the peroneal nerve synchronized with the swing phase of the gait in hemiplegic patients", Arch. Phys. Med. Rehabil., 42, 101, 1961

Matthews, P. B. C., "The dependence of tension upon extension in the stretch reflex of the decerebrate cat", J. Physiol., vol. 147, pp. 521-546, 1959

Peckham, P. H., Mortimer, J. T., and Marsolais, E. B., "Controlled prehension and release in the C5 quadroplegic elicited by functional electrical stimulation of the paralyzed forearm musculature", Ann. Biomed. Eng., vol. 8, pp. 369, 1980.

Rack P. M. H., and D. R. Westbury, "The effects of length and stimulus rate on tension in the isometric cat soleus muscle", J. Physiol., vol. 204, pp. 443-460, 1969

Reswick, J. B., "A brief history of functional electrical stimulation", in Neural Organization and Its Relevance to Prosthetics, Med Book Corp, 1973

Reswick, J. B., "Development of feedback control prosthetic and orthotic devices", Advances in Biomedical Engineering, vol. 2, pp. 139, 1972

Vodovnik, L., "Functional electrical stimulation of extremities", in Advances in Electronics and Electron Physics, vol. 30, Academic Press, 1971

Wilemon, W. K., Mooney, V., McNeal, D. R., and Reswick, J. B., "Surgically implanted peripheral neuroelectric stimulation", Internal report of Rancho Los Amigos Hospital, Downey, CA, 1970

Zoll, P. M., "Resuscitation of the heart in ventricular standstill by external electric stimulation", N. Engl. J. Med., 247, 768, 1952

4. TREMOR SUPPRESSION USING A MECHANICAL BRAKE

An alternative to the functional electrical stimulation approach of suppressing tremor was the use of the mechanical brake to stop fast limb movements while minimally affecting intended slow movements. Dr. Brian Andrews from Glenrose Rehabilitation Hospital has been using a small and portable mechanical brake in a hip orthosis. This brake provides a large rotational impedance and requires a small lever movement to reduce the impedance to nearly zero, in contrast to other brakes previously studied that are usually large and insensitive. An interesting property of the brake is that it is unidirectional. The large impedance only acts on movements in one direction, whereas the movements in the other direction experience a much smaller resistance. To control resistance in both directions, a pair of brakes are needed. Two brakes are mounted on a common axis, with one braking clockwise movements and the other brakiing anti-clockwise movements.

4.1 The Structure of the Spring Brake

The structure of the brake is quite simple. It consists mainly of two cylindrical drums (see Figure 4-3) arranged in a coaxial manner, wrapped around by a strong spring. One end of the spring is attached to one of the drum. The other end of the spring is extended outwards to form a small rod which is used to change the spring's diameter which in turn controls the frictional force between the spring and the second drum (see Figure 4-4). When there is no force on the rod, the spring will wrap itself tightly on the drum when the drum is turned against the spring wraping pattern, generating a large rotational resistance. If, on the other hand, the drum is turned in the opposite direction, the fictional force will help to unwarp the

spring, resulting in only a small resistance. When the rod is pushed such that the diameter of the spring is increased, the spring to drum contact is reduced and the brake's resistance is reduced to nearly zero in both directions.

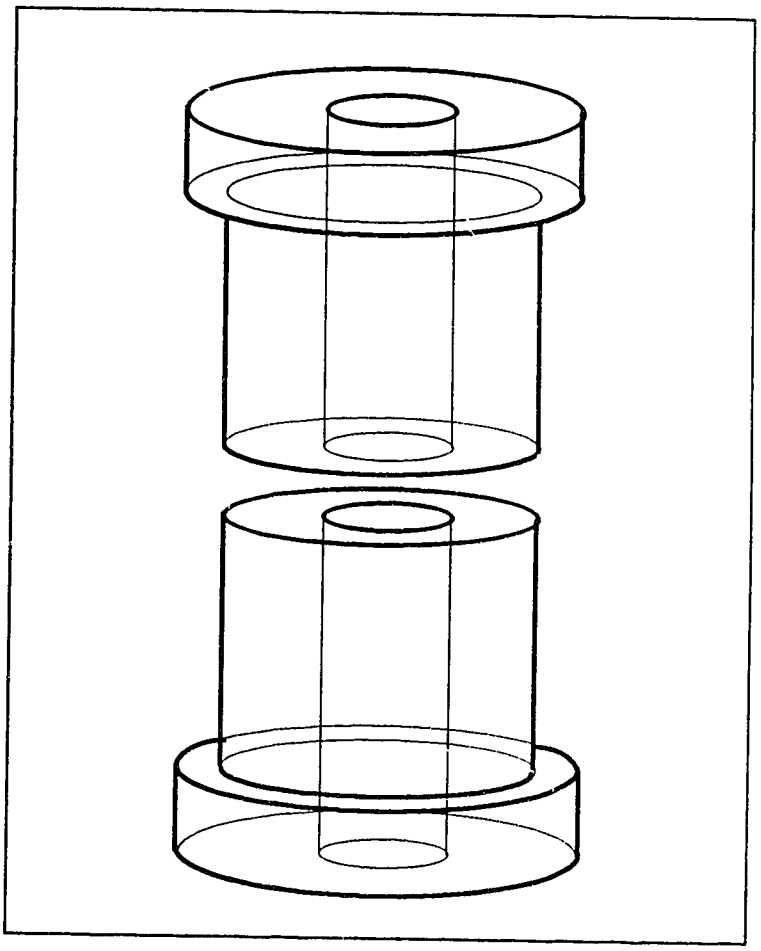


Figure 4-3 Exploded three-dimensional view of the brake drums

4.2 The control mechanism

Two angle servo motors are used to control the position of the rods. A round shape plastic piece is fixed on the shaft of each servo motor, as shown in Figure 4-5. The angular position of the round piece is used to control the position of the spring end. The angular position can be controlled by feeding an appropriate pulse width modulated signal to the angle servo motor.

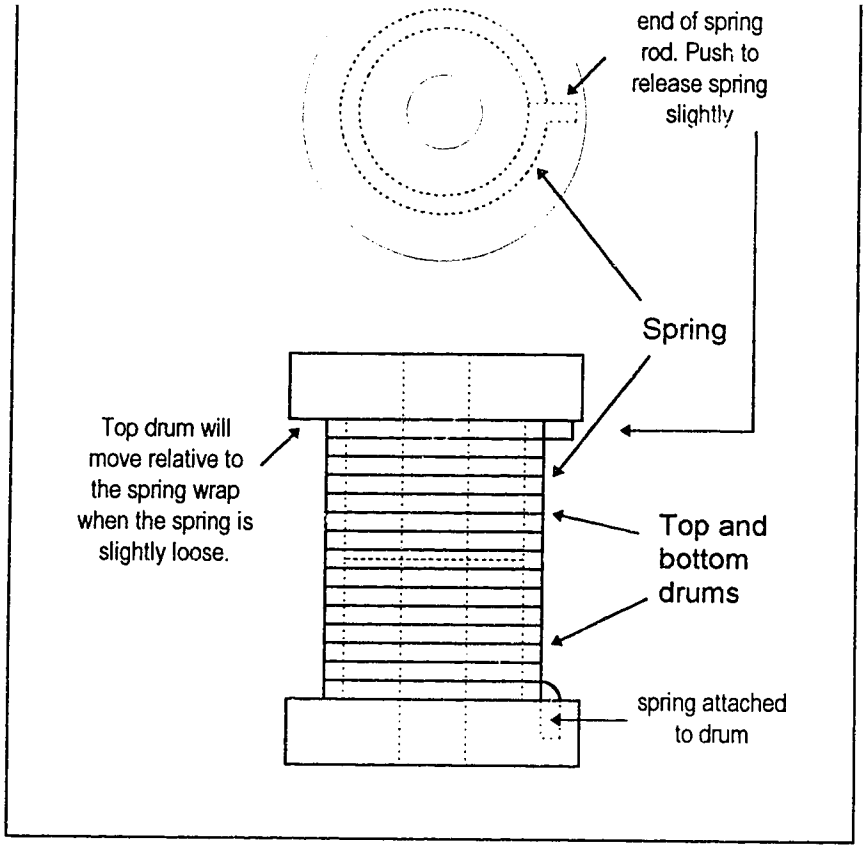


Figure 4-4 Top and side views of the brake

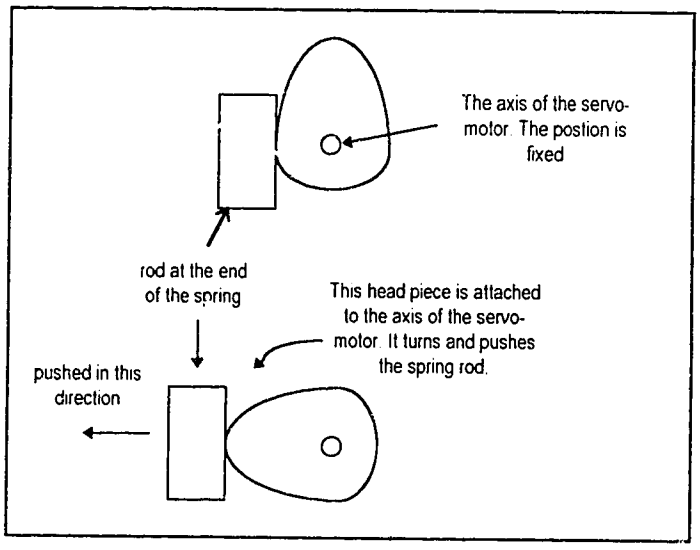


Figure 4-5 Control of the spring rod by a servo motor

controllable in a fairly linear manner. However, further testing revealed several severe problems:

- There was a marked hysteresis in the relationship between the control signals and braking force. To reach a particular resistance force from a larger resistance, the servo motor had to travel a different distance than if it started at a lower resistance. The problem was due to the nature of the contact friction between the rods and discs on the servo motor.

- Even when the spring was fully released (maximum resistance), the drum travelled a short distance (typically 1 to 2 degrees) before it locked up. This characteristic created a dead zone where the behaviour of the brake was uncontrollable.

- The rod travelled only a very small distance (in the order of 1 millimeter) between a locked brake and a open brake. The resolution of the angle servo-motor therefore had to be very high. The servo motors we employed, or other servo motors we considered did not meet the resolution requirements.

To conclude, the problem of attaining adequate servo resolution, the dead zone, and the large errors associated with physical connections and contacts discouraged further attempts to develop the brake for this application.

4.3 The mechanical brake idea

Although this particular spring brake did not suit our applications, the idea of using a mechanical brake instead of FES should not be abandoned. It has several advantages over the FES approach:

- This system does not need any electrical stimulation of the human body which is sometimes difficult and risky.

- There is no need for a mathematical model of the human muscle. The muscle is no longer the actuator. A single filter can be used for all kinds of patients regardless of muscle properties, limb size, etc.

- The risk of injury is minimal with the mechanical system. The FES system will initiate contraction of the muscle. If this is not carefully done, strong contractions can occur and these can cause damage to the muscle or even the joint. Even a short but strong stimulation can cause pain to the patient. Since the mechanical system will only generate a passive resistance, neither of these situations will occur.

5. FILTER DESIGN

5.1 Problem definition

Tremor movement differs from most intended movements in its frequency content (please refer to Chapter 2). This allows an appropriately tuned filter to separate the two kinds of movements. Of course, it is physically impossible with today's technology to insert a device which filters and alters the signals from the central nervous system (CNS) to the actuator, the muscle[1]. A more practical approach to the problem is to cancel the tremor component of movements by adding an opposite stimulus to that causing tremor movements. In practice, this is done by using functional electrical stimulation (FES) within a feedback control system to stimulate the muscle with a signal out-of-phase with the actual tremor movement. This is the approach suggested and experimentally demonstrated by Prochazka et al. in 1992 *[Prochazka et al., 1992a, 1992b]*. In their paper, they gave the design of an analog filter which provided satisfactory clinical results on both normal subjects and patients. In this chapter, a digital version of the filter they used in the feedback system is discussed. An improved digital filter based on it was used in most of the experiments reported in this thesis.

[1] However, it may be possible to filter the output mechanically, achieving the same result. See Chapter 4.

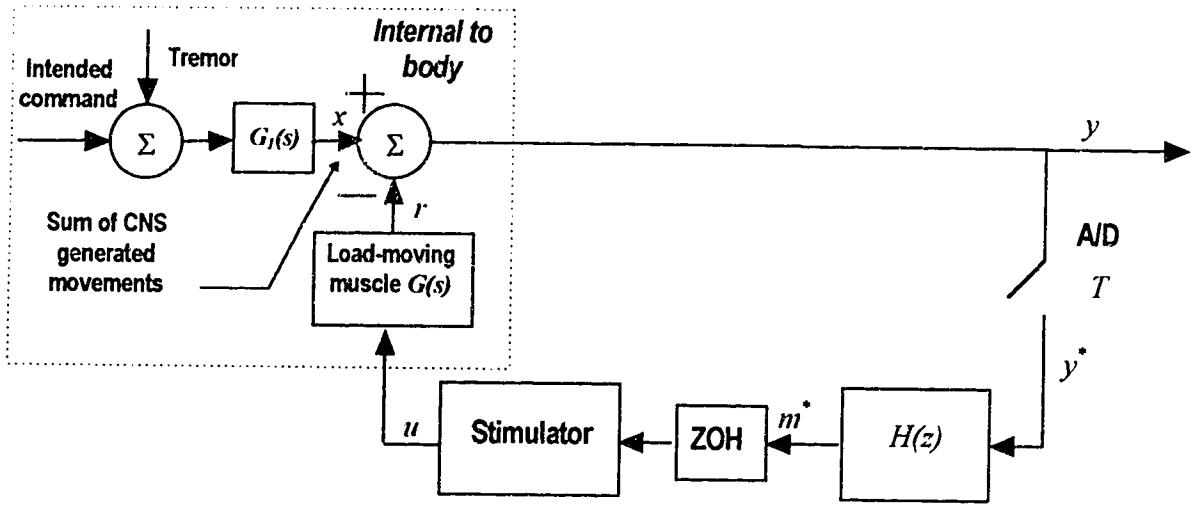


Figure 5-1: Block diagram for tremor suppression feedback system. *The transfer function $G_1(s)$ converts the CNS command signals to actual movements.*

5.2 Block-diagram analysis

Figure 5-1 shows the block diagram for the tremor suppression feedback control system. The input signal x is the movements of the limb which the CNS controls. The signal x is internal to the body and cannot be measured externally with the feedback system in operation. The signal y, on the other hand, is the actual movements of the limb the outside world perceives, and will equal x only if the stimulator is turned off. The ZOH (zero-order hold) block is an essential element which converts digital signals into continuous signals. Th stimulator block converts the filter output into electrical pulses which stimulate the muscle accordingly. The load-moving muscle function $G(s)$ is a linear time-invariant function that models the property of the muscle together with the dynamics of the limb the muscle controls.

Please note that in the block diagram, the input signal x is isolated from the rest of the system. However, in reality, this is not true. The stimulation to the muscle may affect the central nervous system in some unknown ways, which may then change the amplitude or even the frequency of tremor movements. This potentially degrades the performance of the system in ways which cannot be accounted for by any linear system analysis methods.

In Figure 5-1, all small letters denote time-domain variables. T denotes the sampling period. The symbol * means that the variables are sampled in time, i.e. discrete variables. To analyze such a sampled-data system[2], we can start by relating the Laplace transforms of the signals and obtaining a relationship between the sampled output and sampled input. Let us denote the Laplace transforms of signals with capital letters. Notice that the ZOH block can be modeled as the Laplace transfer function: $\frac{1-e^{-sT}}{s}$ *[Franklin et al., 1990, chapter 3]*. Once a sampled-signal transform such as $U^*(s)$ is found, we can then find the corresponding z-transform simply by letting $e^{sT} = z$ or

$$U(z) = U^*(s)\big|_{e^{sT}=z}$$

Now, from Figure 5-1 the relationships are

$$\begin{aligned} Y &= X - R \\ M^* &= Y^* H^* \\ U &= M^* \left(\frac{1-e^{-sT}}{s} \right) \\ R &= GU \end{aligned}$$

[2] Sampled-data system is a hybrid where discrete signals appear in some places and continuous signals occur in other parts.

Suppose we then sample these equations, which gives:

$$Y^* = X^* - R^*$$
$$M^* = Y^* H^*$$
$$R^* = (GU)^* = \left(\frac{G}{s}\right)^* M^* \left(1 - e^{-sT}\right)^*$$

Notice that $[GU]^* \neq G^* U^*$. Combining these equations gives:

$$Y^* = X^* - (GU)^*$$
$$Y^* = X^* - \left(\frac{G}{s}\right)^* M^* \left(1 - e^{-sT}\right)$$
$$Y^* = X^* - \left(\frac{G}{s}\right)^* Y^* H^* \left(1 - e^{-sT}\right)$$

Note that $\left(1 - e^{-sT}\right)^* = \left(1 - e^{-sT}\right)$ since the expression is periodic with time-constant T.

Finally, we have a relationship between Y^* and X^*:

$$Y^* = \frac{1}{1 + H^* D^*} X^* \quad \text{or} \quad Y(z) = \frac{1}{1 + H(z)D(z)} X(z)$$

where

$$D^* = \left(1 - e^{-sT}\right)\left(\frac{G}{s}\right)^* \quad \text{or} \quad D(z) = (1 - z^{-1}) Z\left\{\frac{G(s)}{s}\right\}$$

$D(z)$ is known as the zero-order hold equivalent of the load-moving muscle model $G(s)$.

Thus the equivalent block diagram for Figure 5-1 which is suitable for discrete system analysis is shown in Figure 5-2.

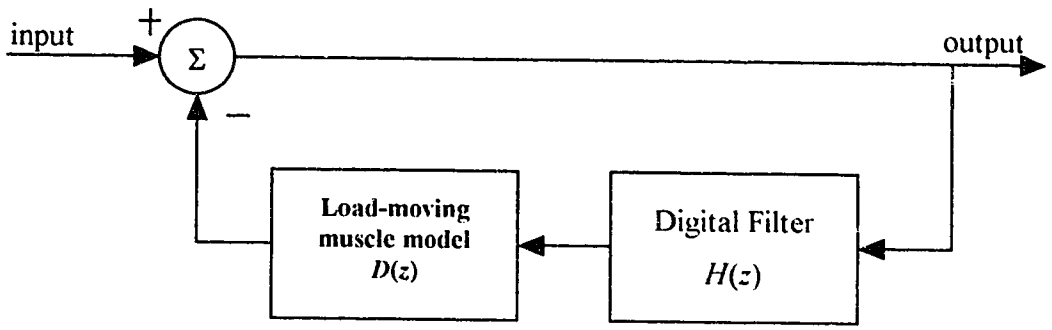


Figure 5-2: Digital equivalent block diagram for tremor suppression feedback control system

5.3 Load-moving muscle model

As discussed above, the human muscle has a low-pass frequency response. Prochazka et al. have carried out clinical experiments to find an average frequency response of the wrist muscle *[Prochazka et al., 1992a]*. They found a transfer function that well fits the experimental frequency response shown in Figure 5-3:

$$G(s) = \frac{K}{(s+8.5)(s+55)^5}$$

where s is the Laplace operator and K is a constant. $D(z)$ can be readily obtained by finding the zero-order hold equivalent of $G(s)$:

$$D(z) = (1-z^{-1})Z\left\{\frac{G(s)}{s}\right\}$$

It must be realized that $D(z)$ is **not** purely the digital equivalent of $G(s)$. $D(z)$ includes a model of a digital-to-analog converter which is needed as a connection between a continuous model and a digital filter. The frequency response plots of the $G(s)$ and $D(z)$ are shown in Figure 5-4, with a sampling period T = 0.01s assumed.

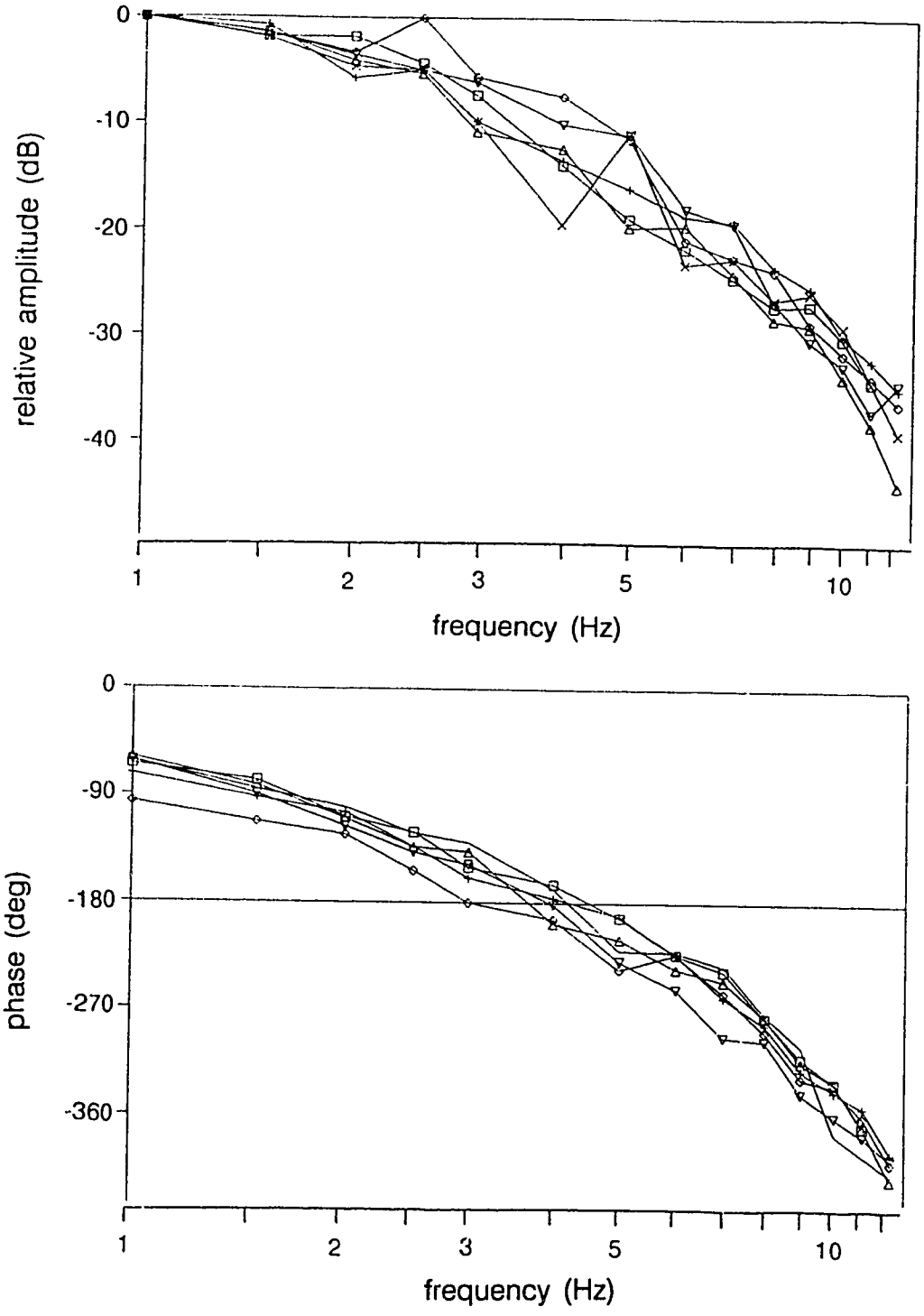


Figure 5-3 Actual data from normal subjects

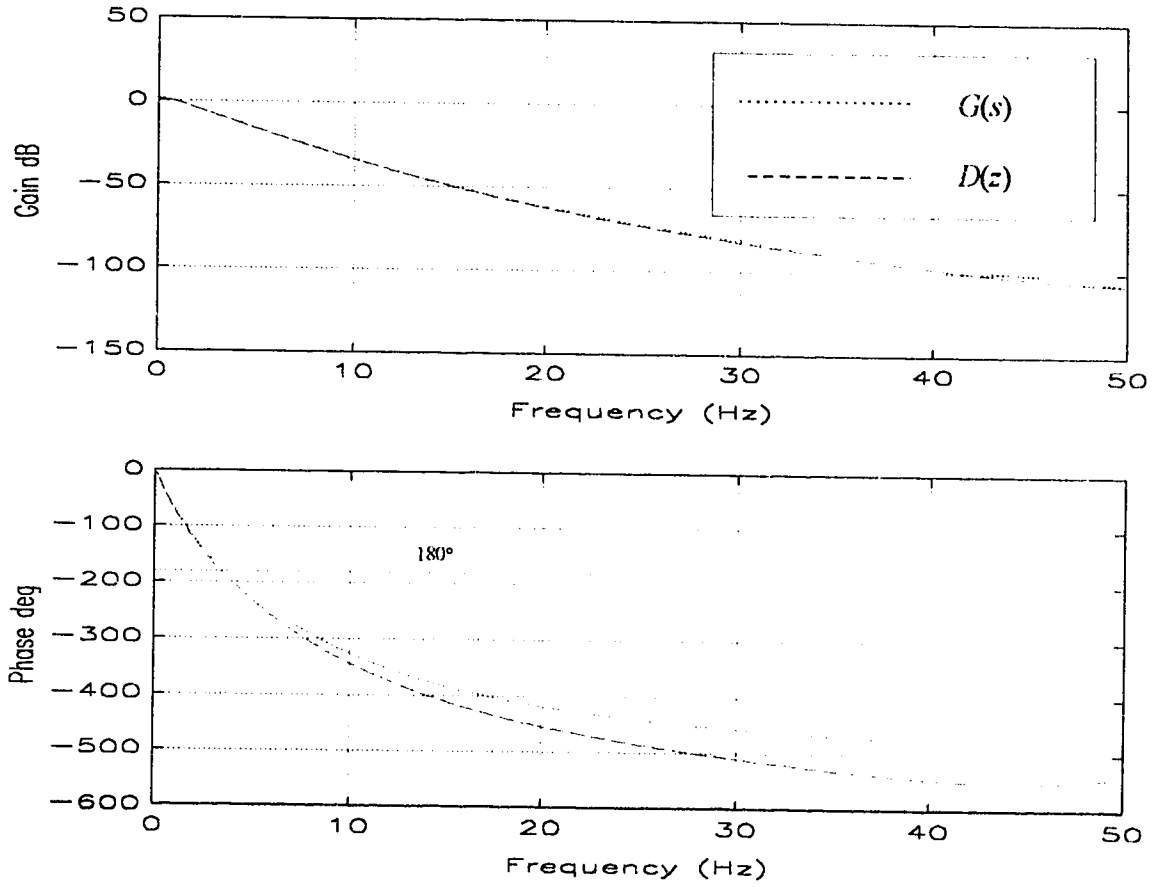


Figure 5-4: Frequency response of the load-moving wrist muscle model. *1) continuous frequency response; 2) digital zero-order hold equivalent of the continuous frequency response, with T = 0.01s*

5.4 Linear Time Invariant filter

5.4.1 Design Criteria

There are several criteria during the design of filter *H(z)*. Many of these criteria cannot be achieved without degrading others. Compromises must be made to get satisfactory experimental results.

5.4.1.1 Minimize gain from 0 to 1Hz and maximize gain from 2 to 6Hz

Most useful voluntary movements are of very low frequency, typically less than 1 Hz. These movements should not be hindered in any way, so the gain of the filter at low frequencies should be close to zero. This is a very subjective measure and must be tested experimentally with patients to get a good compromise.

The frequency range 2 to 6Hz is where most (if not all) of the tremors occur at. A large gain in this range means a better suppression of tremor. A detailed description of the performance of the system can be found in section 5.4.4.

5.4.1.2 Minimize gain above 10Hz

Gain at high frequencies should be small to minimize amplified noise from quantization noise or transducer error. The signal-to-noise will become too small if high frequency noise dominates.

5.4.1.3 Provide a phase lead from 2 to 6Hz

The muscle response is inherently slow, meaning that an electrical stimulus will not produce a movement immediately; that is, there is a lot of phase lag. In order to have good

tremor cancellation, the open loop frequency response of the system (wrist + filter) at around tremor frequency should have approximately zero phase. Therefore, the filter must provide a large phase lead around tremor frequencies 2 to 6Hz.

5.4.1.4 Maximize the gain margin

Because of the high-pass nature of the filter, the system may exhibit instability with a higher gain. The gain margin must be large enough to ensure minimum chance to reach instability.

5.4.1.5 Keep instability frequency (crossover frequency) below 10Hz

If instability should occur, its frequency should be less than 10Hz so that it will not become too uncomfortable for the patient. A frequency too high can even be painful to the patient.

5.4.2 The original linear filter:

Prochazka et al. designed a filter that met the above requirements:

$$H(s) = \frac{K(s+3.3)^6(s+100)^4}{(s+15)^6(s+300)^4}$$

The transfer function has ten poles and ten zeros which are all real. The frequency response of this filter is shown in Figure 5-5 . A bilinear transformation[3] of the filter (also shown in Figure 5-5) is obtained by substituting the Laplace operator s with $\frac{2}{T}\frac{z-1}{z+1}$ where T is the sampling period, and z is the z-transform operator. In this case, a sampling

[3] Also known as the trapezoidal or tustin transformation.

frequency of 100Hz is chosen which corresponds to more than 6 times the highest frequency of interest (around 15Hz). It can be seen from the above figure that the transformed filter follows the continuous one closely below 10Hz. This digital filter yields satisfactory but non-optimal result. The open loop transfer function has a gain margin of only 6dB (see Figure 5-6), degraded from a gain margin of 9dB for the continuous version, and a crossover frequency of around 8Hz.

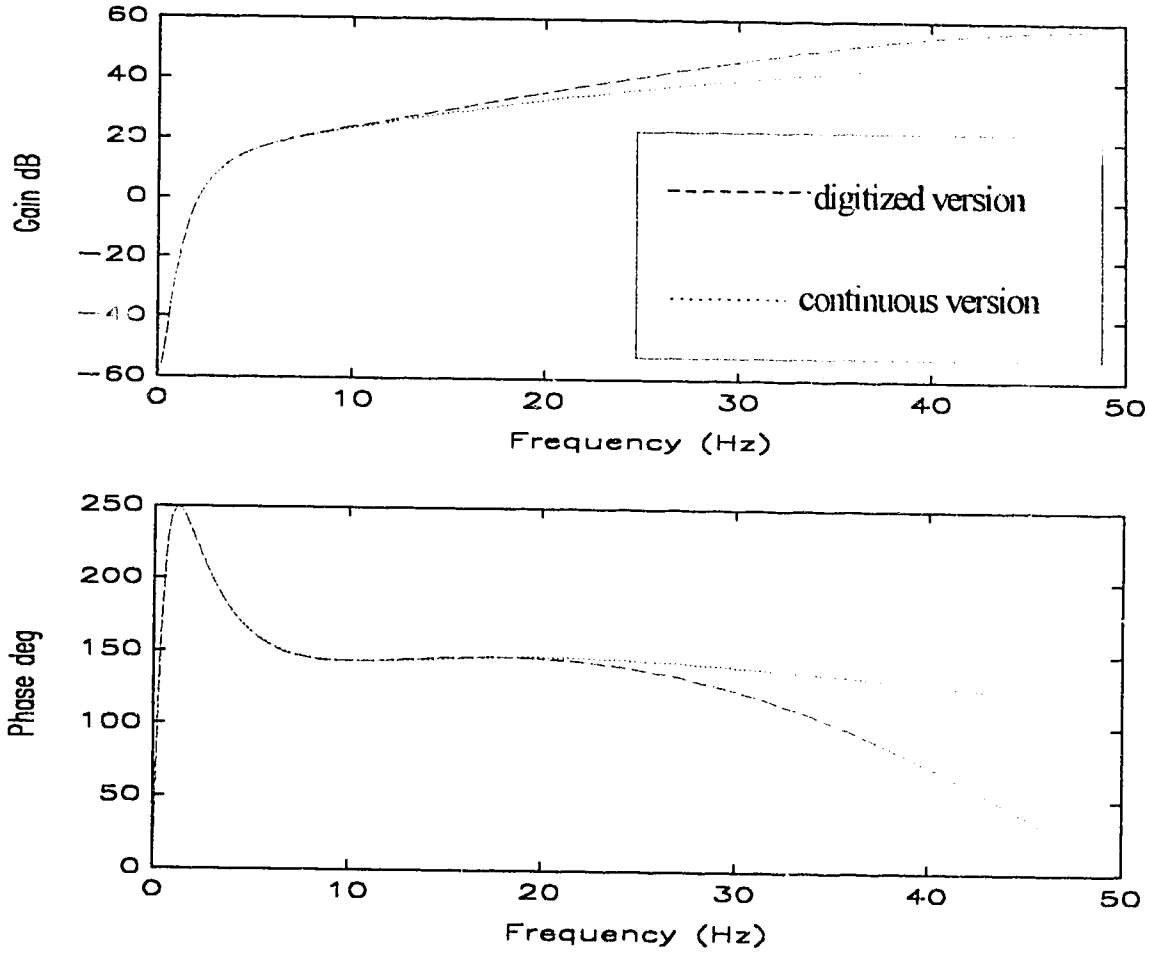


Figure 5-5: Frequency response of the original tremor suppression filter. *1) continuous version; 2) bilinear transformation of the continuous filter with T = 0.01s*

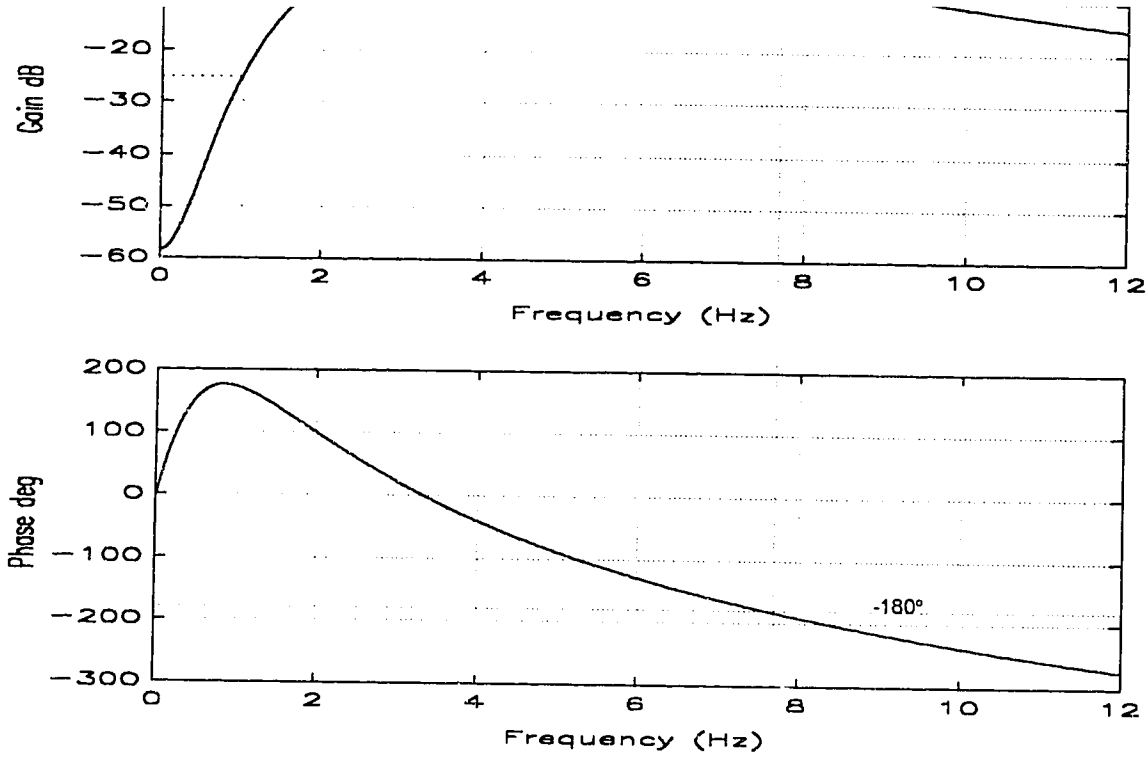


Figure 5-6: The open loop frequency response of the digitized wrist transfer function $D(z)$ with the digitized original high-pass filter

5.4.3 The improved digital filter

The original filter, although having a satisfactory result, can be improved. However, changing the real poles and zeros, or increasing the order of the filter cannot improve any of the above criteria much further without degrading others. Upon reconsidering the problem, there was a unnecessary constraint during the design of the original continuous filter. The s-plane poles and zeros did not have to be real. In another words, the poles and zeros should be placed freely on the s-plane or the z-plane. Using a direct z-plane design, as long as the poles are adequately damped (close to $z = 0$), complex poles and zeros can be used. The use of complex poles and zeros can improve the ability to fine tune the shape

emphasize the gain around 1Hz.

After some trial-and-error iterations, a new transfer function was found. The original general shape of the gain and phase curves is preserved, but some fine tunings are made. The gain around 1 Hz is de-emphasized, so is the gain at around 8 Hz. These changes improve the selectivity of the filter (gain at 1Hz is -35dB compared to -25dB in the original filter), and also increase the gain margin by 3dB (9dB compared with 6dB).

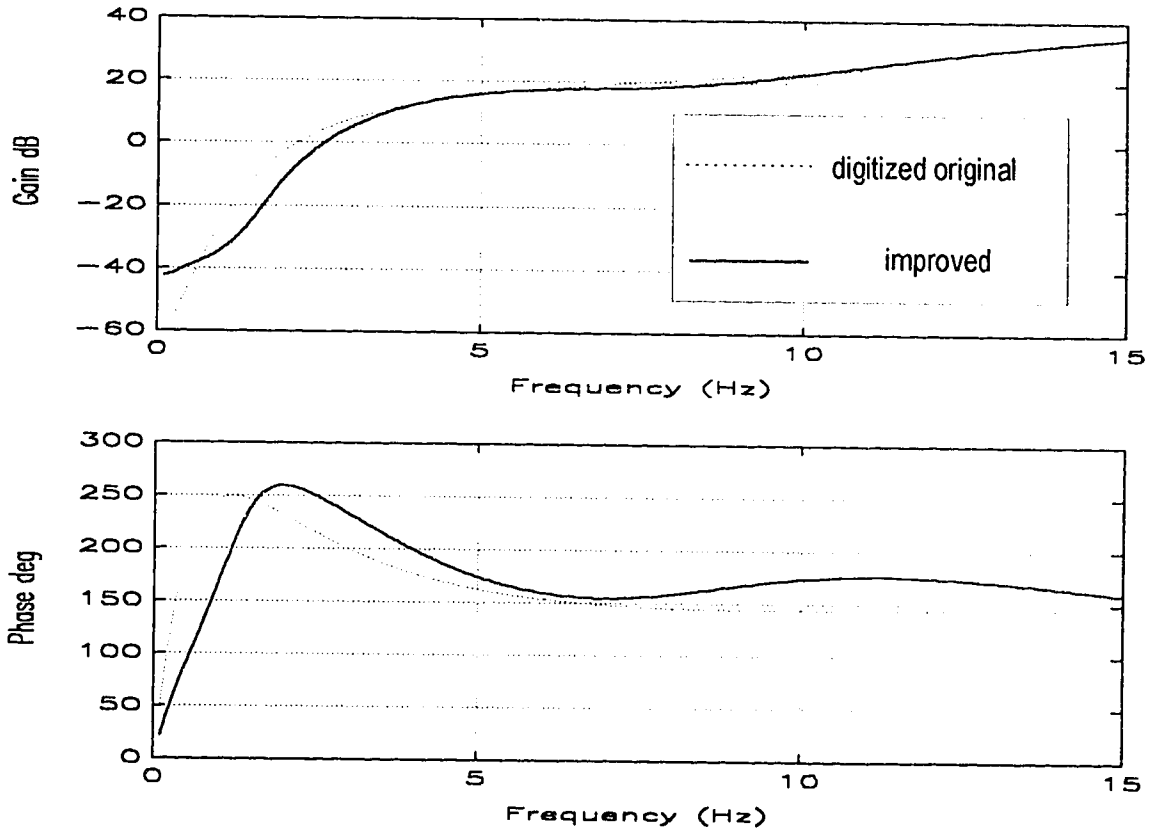


Figure 5-7: Frequency response of the improved digital filter (solid line) and the digitized original filter (dotted line). *Notice the extended range of low gain at frequencies 0-1Hz and a greater phase advance overall.*

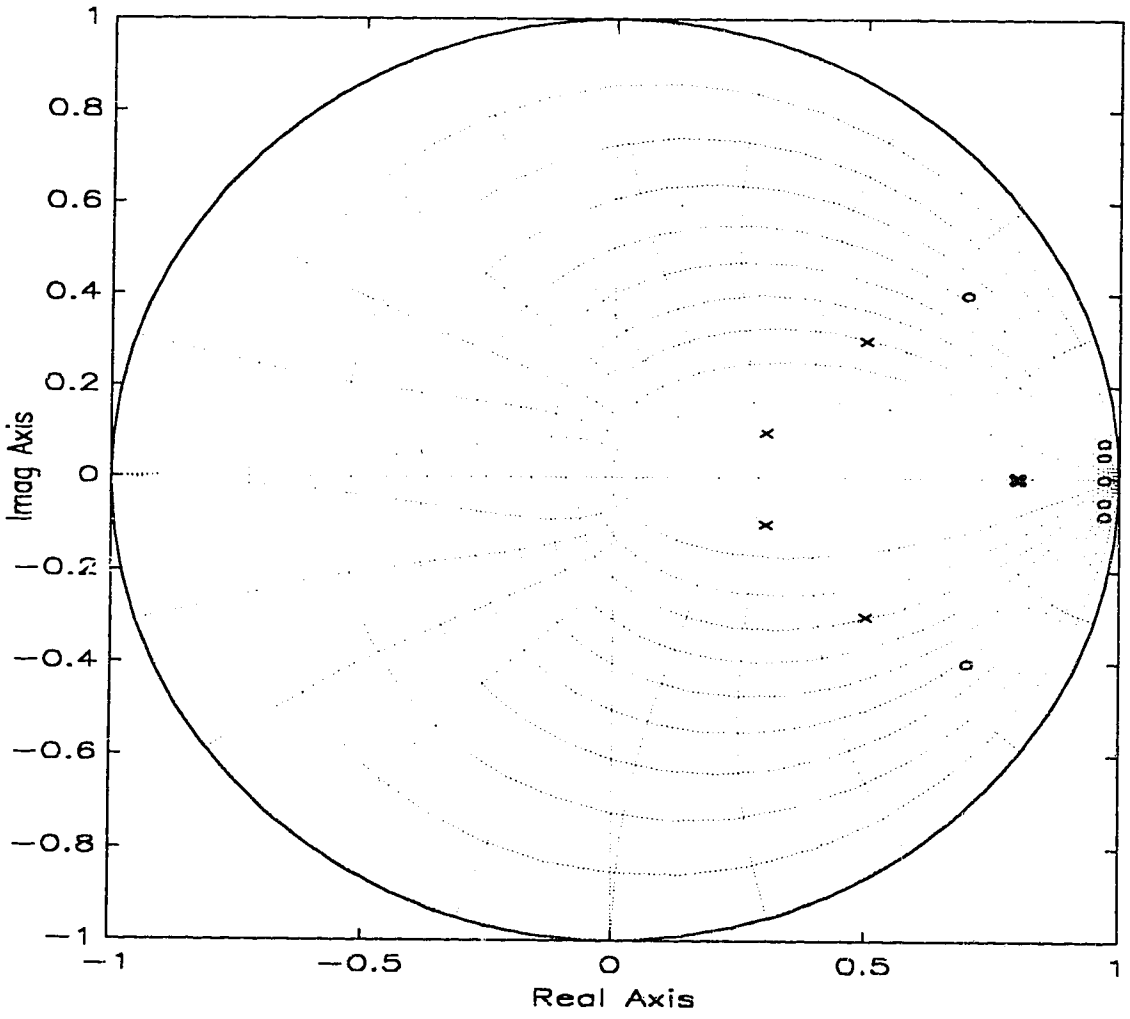


Figure 5-8: Pole-zero map of the improved digital filter

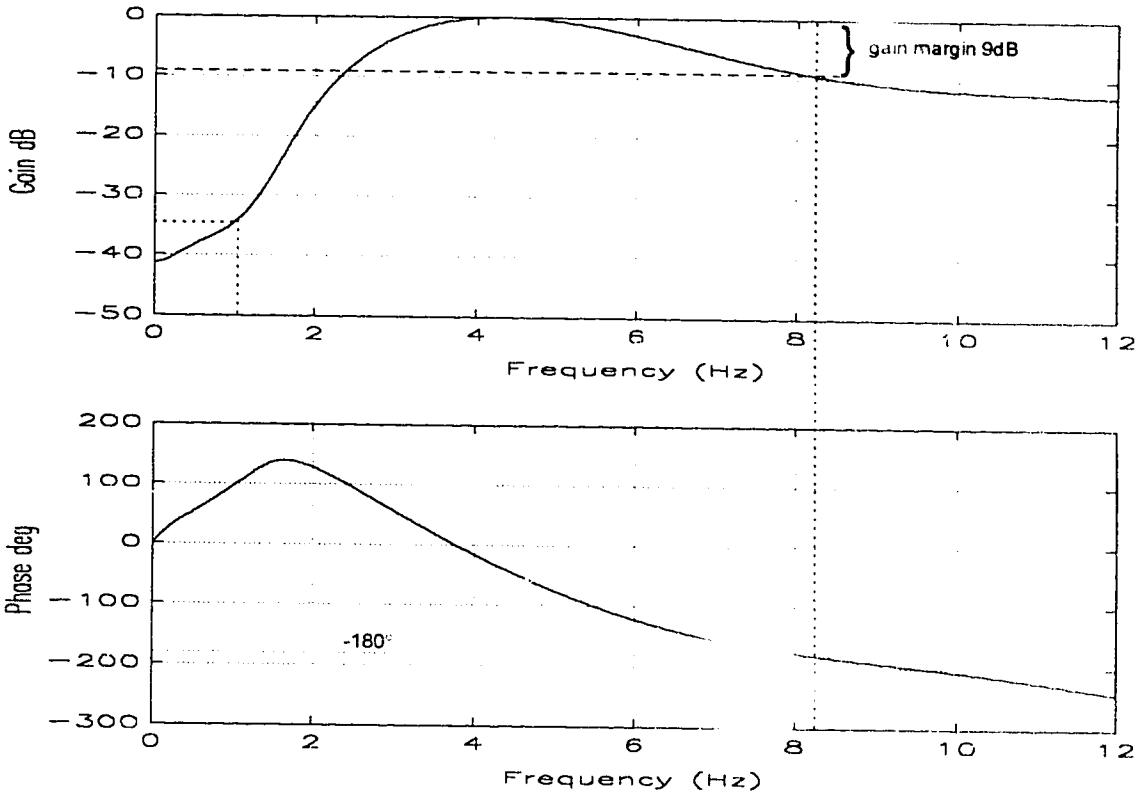


Figure 5-9: The open loop frequency response of the digitized wrist transfer function $D(z)$ with the improved digital filter

5.4.4 Performance analysis of the system

From the Block-diagram analysis section, the input-output relationship was found to be:

$$Y(\omega) = \frac{1}{1 + D(e^{j\omega T})H(e^{j\omega T})} X(e^{j\omega T})$$

where T is the sampling period.

Therefore, the tremor movement (Y) is suppressed to $\frac{1}{1+DH}$ of the original amplitude. It means that if the open loop gain DH is greater at a given frequency, the tremor suppression at that frequency will be more effective. However, the magnitude of DH is limited by the gain of H, the digital filter. Take the improved wrist filter as an example.

Figure 5-9 shows that the gain margin is about 9dB (2.8). This means, that at 4Hz where the peak gain is, the gain cannot be greater than 2.8. Gain above 2.8 will make the whole system unstable. Figure 5-10 shows that overall closed loop frequency response of the marginally stable tremor suppression system. For both of the digital filters, the suppression is greatest at around 4Hz. The improved digital filter suppresses a 4Hz tremor movement to about $\frac{1}{1+2.8}$ or 0.27 of its original amplitude. For comparison, the digitized original tremor only reduces the tremor to about 0.35 of its original amplitude. It can be observed from Figure 5-10 that the improved filter hardly affects movements from 0-1Hz. It also affects movements in the 0-2Hz range less compared with the original filter.

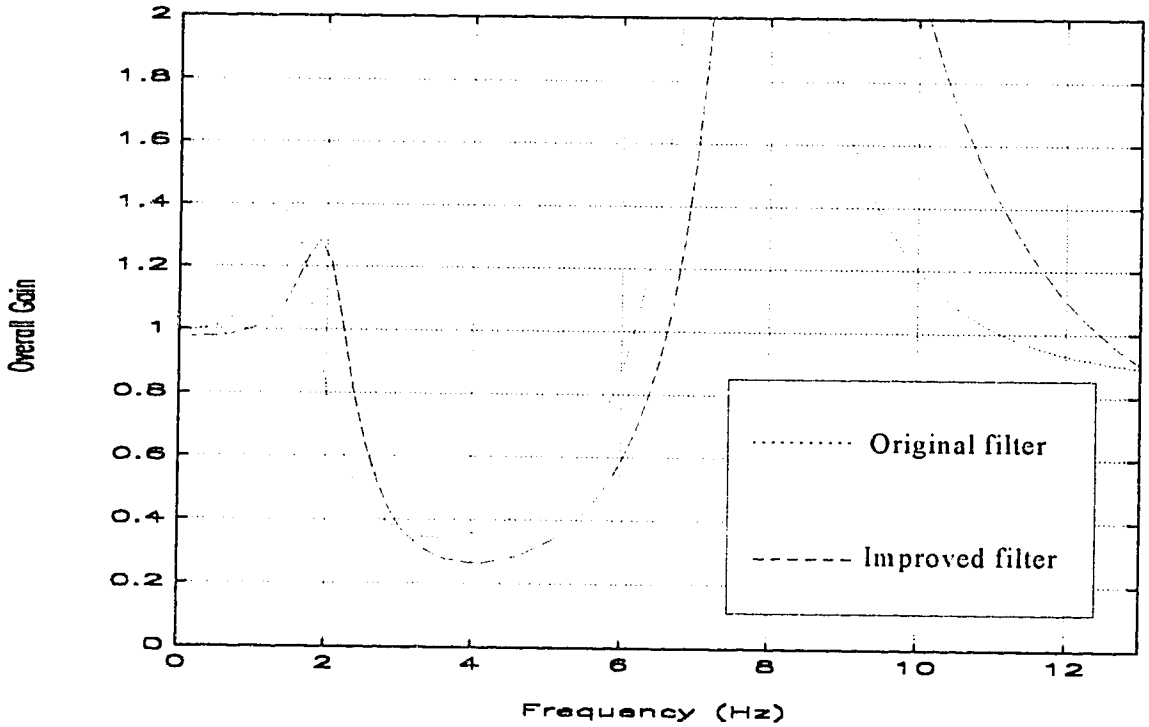


Figure 5-10: Overall closed loop frequency response of marginally stable tremor suppression systems with 1) the digitized original filter; 2) the improved digital filter.

5.5 Conditional filter for safety purpose

As seen from the above, the linear filter designed has a high-pass characteristic. The gain is very high at nyquist frequency, but the human muscle is expected to have virtually zero response at that frequency. In fact, it has been assumed that the highest frequency the muscle can produce is less than 15Hz, which is only a fraction of the Nyquist frequency used. Our frequencies of interest are from 0 to about 6Hz. Frequencies above this range are probably caused by internal noise of the filter and should be eliminated. Another cause for such high frequencies can be from the sensor itself. Unintentional shaking of the sensor or occasional noise in the sensor can contribute to signals which approach the nyquist frequency. Consider a possible implementation:

Sampling frequency: 100Hz

Nyquist frequency: 50Hz

Filter gain at 13Hz = 1

Filter gain at 8Hz = 0.25

Any input signals with input frequencies above 13Hz will have a saturated output of gain 1. As seen from the above realistic data, the ratio of the gain at 13Hz to the gain at 8Hz is 4 (not to mention the ratio of the gain at 50Hz to the gain of 8Hz is 28!). This means that the whole system can produce a stimulus 4 times that of the intended maximum output. This can be very uncomfortable or even painful to the user of this system.

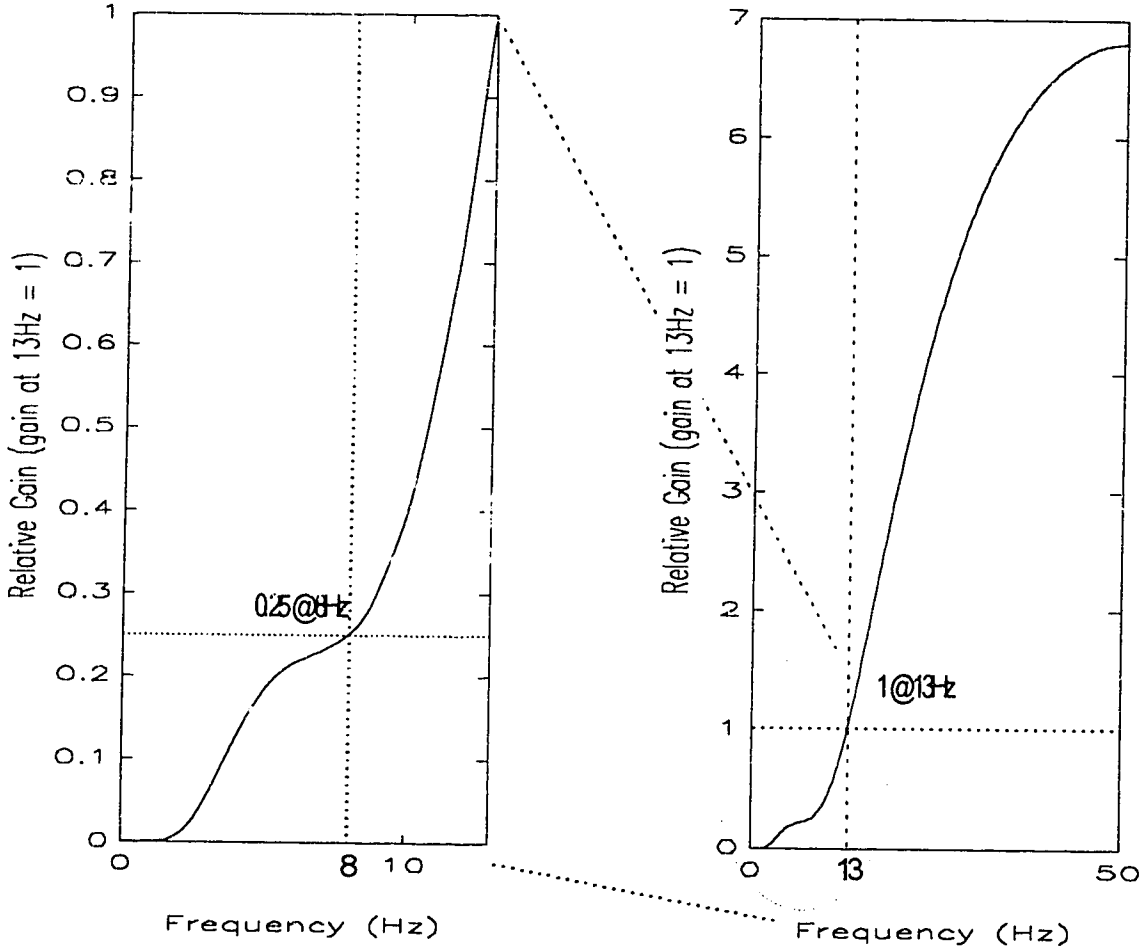


Figure 5-11: Left - the gain from frequencies 0 to 13Hz. 8Hz is the cut-off frequency for the non-linear filter. Right - the gain for frequencies up to the nyquist frequency. All gains above unity will saturate the filter.

A brute-force but effective method to eliminate this danger is to estimate the ratio of the output amplitude to the input amplitude and if the estimation is larger than 0.25, the stimulator will be turned off, in light of a possible harmful stimulus. Provided that the estimation is appropriate and approximately correct, this brute-force method becomes a two-ended needle:

- For a nearly static input, i.e. when the user is resting, the system will not produce any stimulus, eliminating the small calculation noise present in the output. (Since input $\rightarrow$ 0, it must also be true that output$\rightarrow$ 0 in order for output/input <0.25)

- At tremor frequency, the signal to noise ratio becomes higher. The conditional filter will not affect the output as long as the frequency of operation is within intended range. When the sensor is accidentally disturbed, a high frequency appears at the input and the conditional filter will immediately sense the abnormally high gain the filter produces. The stimulation will be immediately turned off for a pre-defined number of cycles, forbidding any dangerous stimulus to be generated. If the system goes into instability which occurs at a high frequency, this filter will destroy the rhythmic movement bringing the system back to stable state.

5.5.1 Estimation method

The amplitude of the input signals is estimated by calculating the difference between the minimum and maximum values of a history of about 30 samples. An input that approximates a sinusoidal wave is assumed. A window of 30 samples can estimate the amplitude of a sine wave of up to 1/(30*T). For a sampling frequency of 100Hz, this corresponds to around 3Hz. Below this frequency, the estimate is incorrect but will not affect the whole system since the gains at low frequencies are much smaller than 0.25. The conditional filter operates by looking at the estimated amplitude from the input signal and comparing it to the filter output. If the filter output is larger than one-fourth of the estimation, the stimulator is turned off for a pre-defined period of time.

References

Prochazka, A., Elek, J. and Javidan, M., "Attenuation of pathological tremors by functional electrical stimulation I: Method", Ann. of Biomed. Eng., vol 20, pp. 205-224, 1992a

Prochazka, A., Elek, J. and Javidan, M., "Attenuation of pathological tremors by functional electrical stimulation II: Clinical evaluation", Ann. of Biomed. Eng., vol 20, pp. 225-236, 1992b

Franklin, G., Powell, J. and Workman, M., "Digital Control of Dynamic Systems", 2nd edition, Addison Wesley, 1990

6. IMPLEMENTATION

6.1 Overview: the filter implementation

In chapter 4, $H(z)$ has been designed for the tremor suppression system. Moving from a theoretical domain to a practical domain is not always an easy task. For instance, there is a vast variety of hardware to choose from. Digital signal processor (DSP) is by far the best method to implement filters because of its flexibility. Calculations are done either using fixed-point or float-point mathematics, but most DSP's only support fixed-point mathematics.

In the section below, the method of filter implementation used in this paper is introduced. Efforts have been made to make the discussion as general as possible.

6.1.1 Fixed-point mathematics

In fixed-point mathematics, all numbers are normalized to a common range, usually from -1 to 1. Two's complement numbers are used. The convention is such that the radix point is just right of the most significant bit (b_0), which is also the sign bit:

$$x = b_0 \bullet b_1 b_2 b_3 \ldots . b_{n-1}$$

In this way, the range of the number x which can be represented is:

$$-1 \le x \le \frac{2^{n-1}-1}{2^{n-1}}$$

and the resolution of number x is: $\frac{1}{2^{n-1}}$

Arithmetic operations like addition and subtraction are done as normal two's complement operations. Multiplications are done as normal with an extra shift-left operation to move the radix point to the correct position.

6.1.2 Structure of Infinite Impulse Response Filter (IIR filter)

There are many different structures for implementing IIR filters. For the high order filters like the ones used in this paper, the simple direct form will be subjected to sensitivity problems. *[Oppenheim and Schafer, 1989, Chapter 6]*. Cascade form is the more common and preferable structure used for high order filters. The basic idea is to separate the filter into second-order sections and then cascade them in series. These second-order sections are obtained by pairing the poles and zeros together, although it is not always easy to find the best pairs. How to implement the second-order sections is yet another consideration. Different structures like direct form I, direct form II and their transposed forms are among the simplest. In particular, the direct form II (or its transpose) needs the least number of multiplications and storage. In this paper, the transposed direct form II is used (see Figure 6-1).

6.1.3 Constructing the filter

In general, the transfer function of a non-causal IIR filter can be written as:

$$H(z)=\prod_{i=1}^{N} H_i(z)=\prod_{i=1}^{N}\frac{G_i(z-m_{1,i})(z-n_i)}{(z-p_i)(z-q_i)}$$

where N is the number of second order sections in the filter.

If the total number of poles or zeros is odd, there will be a single first-order section. This section is treated as a second-order section by simply multiplying both the numerator and denominator by z, i.e. adding a pole and zero at the origin.

Each section consists of two zeros m_i & n_i, and two poles p_i & q_i. The two poles and zeros are chosen to have comparable distance from the origin. Together with an appropriate G_i, this can maximize the signal-to-noise ratio (S/N) but minimize overflow for the section.

For a system that handles magnitudes up to unity, G_i must be chosen such that the result at critical nodes[4] must between -1 and 1. To avoid overflow for all possible inputs, the sufficient condition will be:

$$G_i \le \frac{1}{\sum_{m=-\infty}^{\infty} |h_i[m]|}$$

where $h_i[m]$ is the impulse response series for the i-th section. Scaling the input in this way guarantees that overflow never occurs at any of the nodes. However, this method usually results in a very small G_i, reducing the S/N ratio a great deal. A preferable approach is to assume that the input is a narrowband signal which can be approximated as a sinusoidal signal. To avoid overflow for all possible sinusoidal inputs, the sufficient condition will be:

$$G_i \le \frac{1}{\max_{|\omega| \le \pi} |H_i(e^{j\omega})|}$$

[4] Depending on the hardware, overflow can occur at some summation nodes without causing any overflow at the output.

G_i from this method will be larger than that obtained from the previous one. The S/N ratio will generally be much better, but there is a slight but acceptable chance of output overflowing. (Overflow, if it occurs, can be mitigated through a separate procedure. See section 6.2.2.1)

After obtaining the appropriate G_i, the transfer function for each second-order section can be expanded to:

$$H_i(z) = \frac{b_{i0}z^2 + b_{i1}z + b_{i2}}{z^2 + a_{i1}z + a_{i2}}$$

where

$$a_{i1} = -(p_i + q_i)$$
$$a_{i2} = p_i \cdot q_i$$
$$b_{i0} = G_i$$
$$b_{i1} = -G_i \cdot (m_i + n_i)$$
$$b_{i2} = G_i \cdot m_i \cdot n_i$$

Now, let *x(n)* be the input and *y(n)* be the output of the *i*-th section. The difference equations for the second-order sections in the time domain will be:

$$y(n) + a_{i1}y(n-1) + a_{i2}y(n-2) = b_{i0}x(n) + b_{i1}x(n-1) + b_{i2}x(n-2)$$

After defining a couple of intermediate variables w_{i1} and w_{i2}, the equations become:

$$w_{i2}(n) = b_{i2}x(n) - a_{i2}y(n)$$
$$w_{i1}(n) = b_{i1}x(n) - a_{i1}y(n) + w_{i2}(n-1)$$
$$y(n) = b_{i0}x(n) + w_{i1}(n-1)$$

The set of difference equations can be calculated in fixed-point mathematics if all coefficients a_{i1}, a_{i2}, b_{i0}, b_{i1}, b_{i2} are less than one in absolute value. This is in general not true. For a stable

filter (i.e. $|p_i|<1$ and $|q_i|<1$), $|a_{i2}|$ will be less than 1, $|a_{i1}|$ will be less than 2. The b's, however, are not bounded. They depend highly on G_i, the overall section gain. To handle coefficients larger than unity, the largest coefficient in absolute value is first determined. If this value c is greater than 1, then all coefficients in the section are divided by 2^k where k is determined by:

$$\frac{c}{2^k} < 1 \text{ and } \frac{c}{2^{k-1}} \geq 1$$

During the calculation, the result will then be left-shifted k times to move the radix point back to the appropriate place.

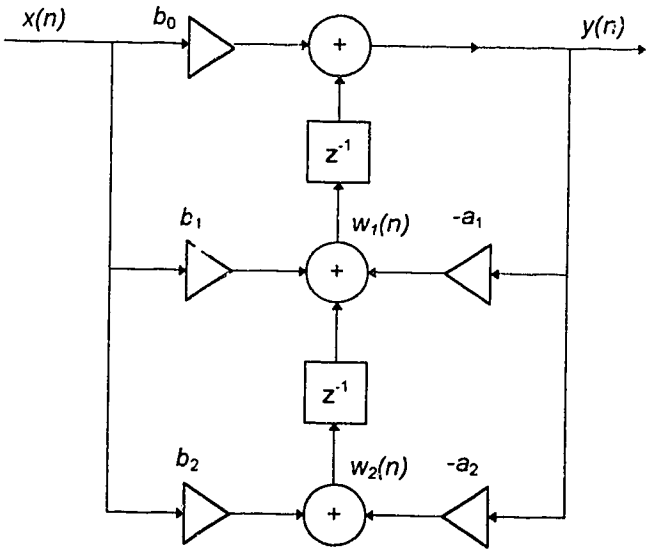


Figure 6-1: Transposed Direct Form II

6.2 Implementation using a digital signal processor

The implementation of the actual infinite impulse response (IIR) filter was done on two platforms: the 68HC16 evaluation board system and the 68HC11 bionic glove system (see section on bionic glove). The 68HC16 was selected as the digital signal processor primarily

68HC11's, with additional instructions for high precision fixed-point multiplications, additions and divisions. In particular, there is a 36-bit accumulator in the 68HC16 (normally used for 32-bit fractional value, 4 bits extra to handle overflow), allowing multiplications of 2 16-bit numbers without truncation. The large word size accumulator also allows intermediate results to overflow. If overflow does occur, the processor will allow saturation of the result, making the overall system more linear[5]. A single-instruction fractional multiplication and accumulation operation is also included for easy implementation of most common finite impulse response (FIR) and infinite impulse response filter algorithms.

The filter used for wrist tremor suppression has 10 zeros and 10 poles, as follows:

Zeros = [0.97, 0.97, 0.97±0.08i, 0.97±0.05i, 0.7±0.4i, 0.7±0.4i]

Poles = [0.8, 0.8, 0.8, 0.8, 0.8, 0.8, 0.5±0.3i, 0.3±0.1i]

and the sampling frequency is 100Hz.

6.2.1 Implementation on the 68HC16

68HC16 instructions for digital signal processing are designed to work with 16-bit fractional numbers with intermediate results up to 36-bit long. The on-chip analog-to-digital (A/D) converter has a maximum resolution of 10-bit. The transfer function coefficients are represented using 16-bit fractional numbers. This coefficient quantization has no noticeable

[5] Oscillation will occur if otherwise. e.g. Overflow limit cycle.

16-bit resolution.

The less conservative method for scaling individual sections is employed. Since the tremor movements closely resemble sinuscidal waves, it is safe to use the frequency response as the guideline to scale the sections.

The intended useful frequency range of the above filter is from 0Hz to around 10Hz. However, the filter has very high gain towards 50Hz. If the filter is scaled to have gain of 1 at 50Hz, the output in the useful range will be unacceptably low. So, when implementing the filter, a gain of 1 is assigned to a frequency around 10Hz instead. Higher-frequency input components, which should not appear theoretically, will saturate the filter. This scheme, apart from increasing the signal-to-noise ratio at low frequencies, will also minimize dangerously high output from the filter resulting from unexpected movement artifacts turbulence.

Using the method discussed above, the 10th order filter is separated into second order sections, with the pairing of poles and zeros as follows:

Filter stage	Zero pairs		Pole pairs	
1	0.7+0.4i	0.7-0.4i	0.3+0.1i	0.3-0.1i
2	0.7+0.4i	0.7-0.4i	0.5+0.3i	0.5-0.3i
3	0.8	0.8	0.97	0.97
4	0.8	0.8	0.97+0.05i	0.97-0.05i
5	0.8	0.8	0.97+0.08i	0.97-0.08i

Table 6-1: Poles and zeros pairing for the improved digital filter

ratio of each section. Figure 6-2 shows the frequency response plots of the individual second-order sections ($|H_1(z)|$, $|H_2(z)|$ ..) and the plots at the end of the sections including all previous sections ($|H_1(z)H_2(z)|$, $|H_1(z)H_2(z)H_3(z)|$ ..) with coefficient quantization taken into account.

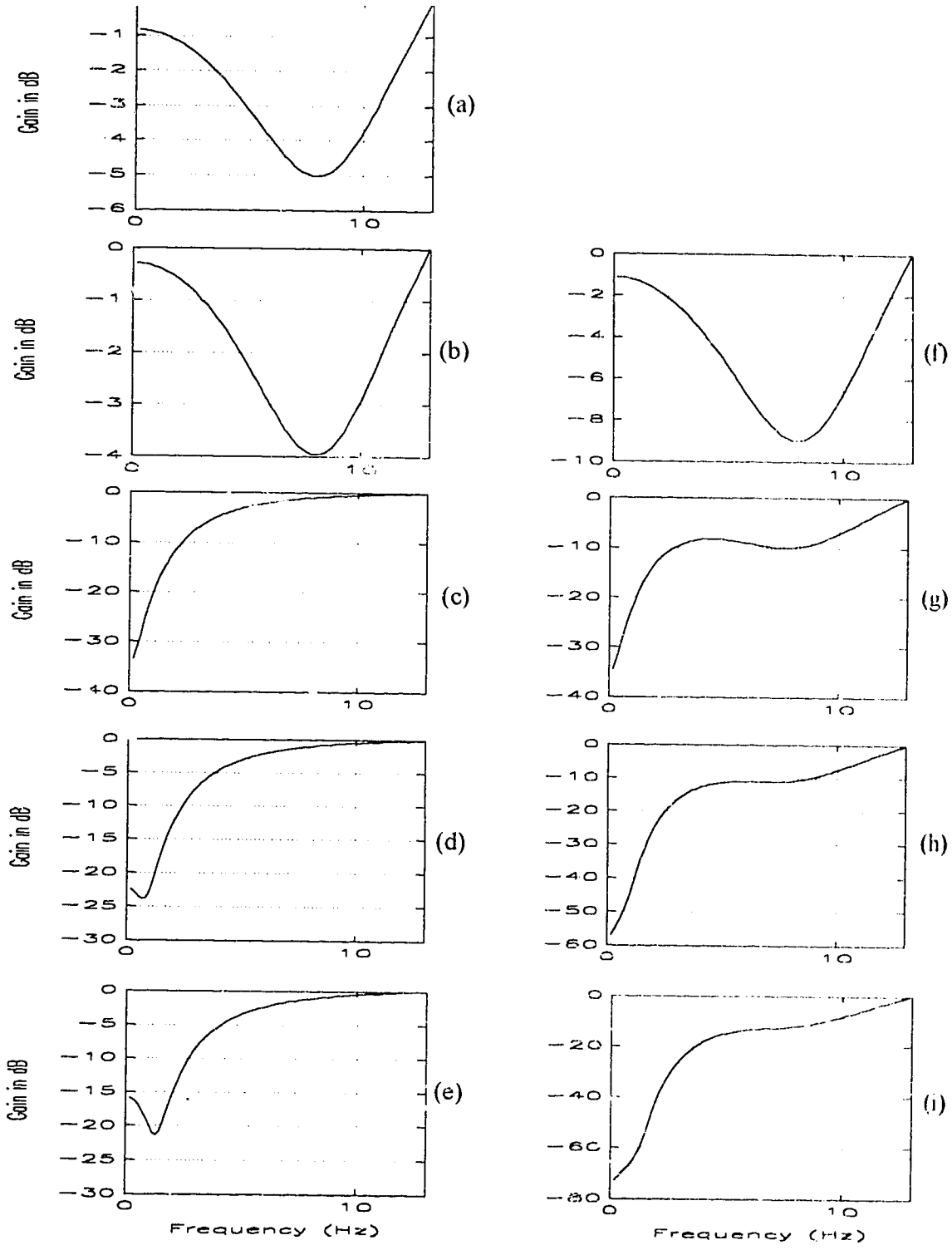


Figure 6-2: Frequency response plots at different stages of the cascaded system. *Graphs (a) to (e) are the frequency response plots of the individual sections. Graphs (f) to (i) are the frequency response plots at the end of the sections including all previous sections.*

6.2.2 Implementation on the 68HC11

The multiplication instruction in the HC11 is very limited. It only supports 8-bit by 8-bit signed integer multiplication. This is simply not sufficient to represent the coefficients to give the desired frequency response (see Figure 6-3 & Figure 6-4)

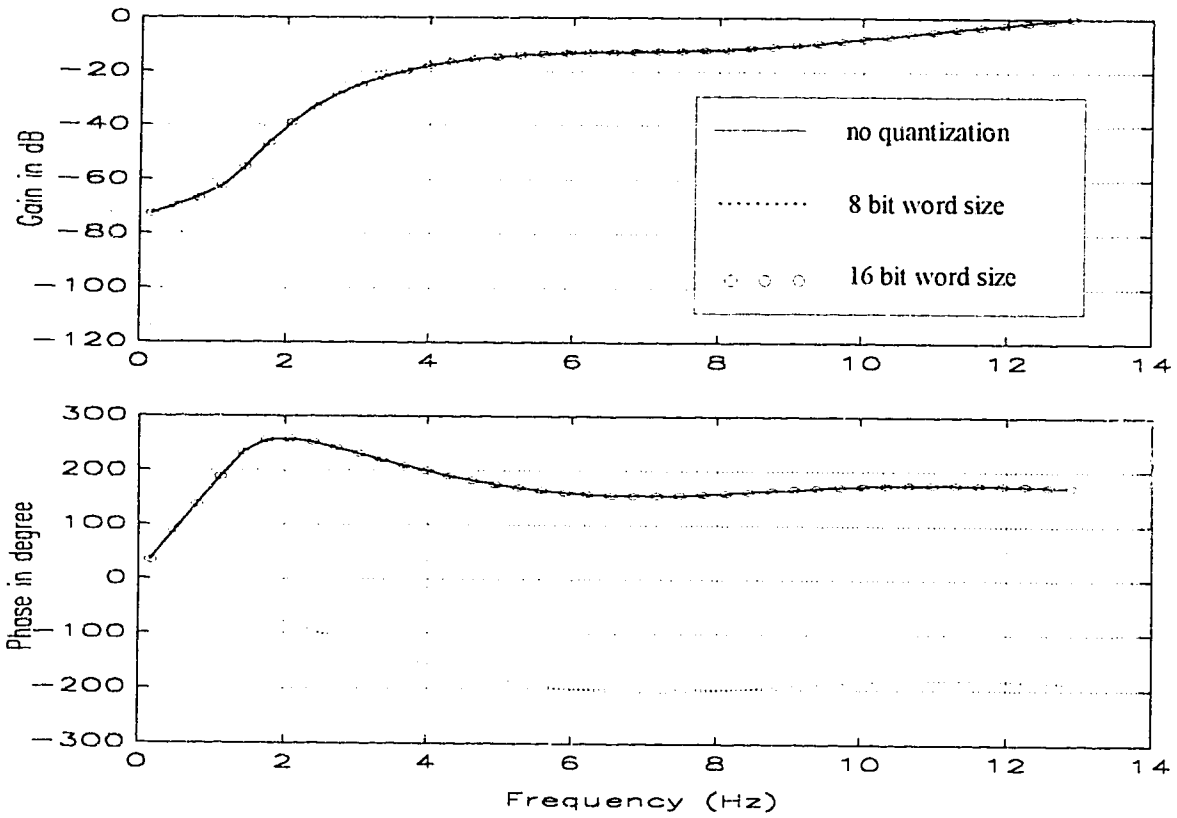


Figure 6-3: Effect of coefficient quantization on frequency response

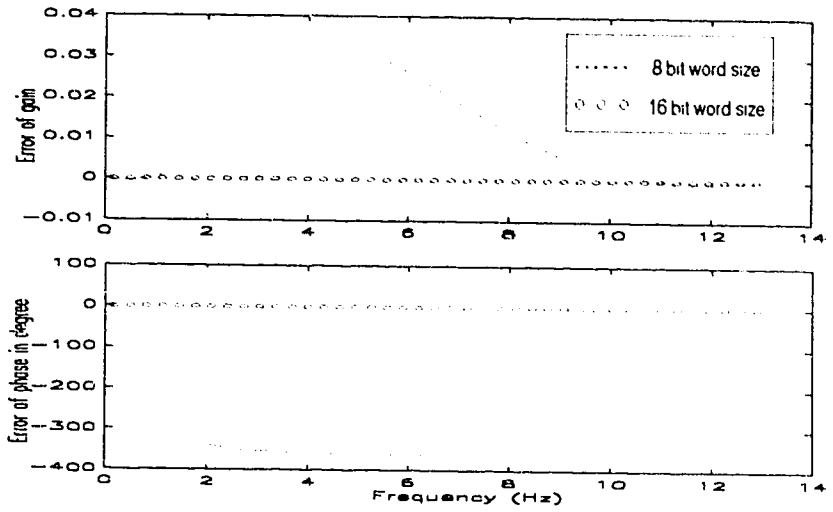


Figure 6-4: Difference from original frequency response after coefficient quantization

In order to preserve the precision of the coefficients and the intermediate results, 16-bit by 16-bit multiplications are necessary. This is accomplished by performing the double precision multiplication as a series of single precision multiplications and additions. The algorithm is as follows:

A 32-bit virtual accumulator which made up of 4 continuos 8-bit memory locations is first setup in memory to hold the multiplication result, and to facilitate additions of intermediate results. To multiply two 16-bit two's complement numbers AB and CD (A and C are the most significant bytes, C and D are the lower bytes), a formula derived from a manual multiplication is used:

$$AB \bullet CD = A \cdot C \cdot \$10000 + B \cdot C \cdot \$100 + A \cdot D \cdot \$100 + B \cdot D + sign(AB) \cdot CD \cdot \$10000 + sign(CD) \cdot AB \cdot \$10000$$

In the above formula, all multiplications are 8-bit by 8-bit, and the multiplications with hexadecimal numbers \$10000 and \$100 are actually byte shifts which are done implicitly by manipulating the appropriate bytes in the 32-bit accumulator. The multiplications of the 16-bit

operand with a sign are done by conditionally negating the operand. Therefore, the formula needs only four 8-bit by 8-bit multiplications.

The operation will give a 32-bit result. For a fractional multiplication, the answer has to be shifted left once to place the radix point at the correct place.

6.2.2.1 Overflowing Consideration

Because of the complexity of implementing the extra 4 bits in the accumulator (as is the case in the 68HC16), the overflowing condition is taken care of explicitly. Natural overflowing (which is basically 'do nothing' approach) is not preferable because it potentially gives the filter a non-linear behavior. Saturation is the preferred way to handle overflow, and this is the method employed, which is the same for the 68HC16 implementation.

The only possible overflowing condition for a multiplication of two 16-bit fractions is when the two fractions are both -1. The result 1 cannot be represented. With conditional testing instructions, this overflowed results will be replaced with $\frac{32767}{32768}$. For all the additions, overflowed results are replaced with either -1 or $\frac{32767}{32768}$.

To make sure that the algorithm developed above will indeed produce a correct result, a verification program was designed for the 68HC16. Since 68HC16's instructions are basically the superset of 68HC11's, a paraphrase of the algorithm used in the 68HC11 was added to the original 68HC16 program. Both algorithms were fed with the same set of numbers from the input signals, and the results were compared. It was found that the results matched bit to bit for all input signals, except when a very high frequency component was in the input signal. In

this case, the 68HC16 algorithm handled intermediate overflowing differently from the 68HC11 algorithm. It was concluded that the 68HC11 algorithm developed is exact and correct and could be used in the final bionic glove implementation.

6.3 Length transducer as an angle sensor

Although angle sensor is generally a simple device (typically a potentiometer), it is not easy to arrange a portable configuration to measure the wrist angle. A more convenient yet simple way of indirectly measuring the angle of a joint is to measure the distance between two points across the joint. As long as these two points are carefully selected, the length can be viewed as a direct linear translation of the angle of the joint, as discussed in section 6.3.1.

6.3.1 Experimental configuration

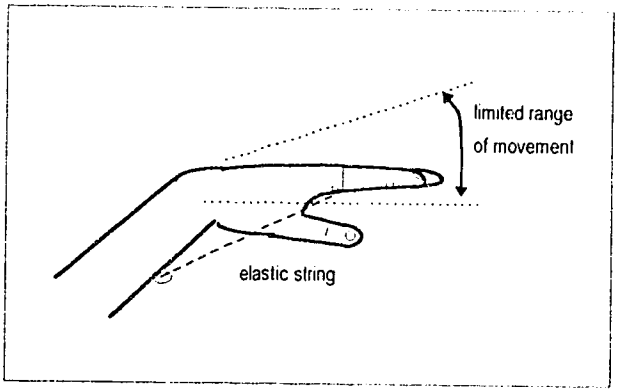


Figure 6-5: Length transducer experimental setup

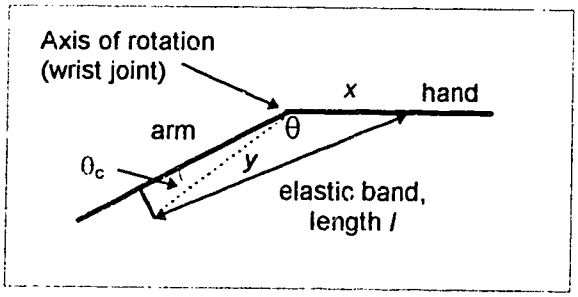


Figure 6-6: A simplified model of the experimental length transducer configuration

Figure 6-5 shows the length transducer placement for most of the experiments reported in this thesis. To find a mathematical relationship between the wrist angle and length, a simplified model is found, as shown in Figure 6-6. The relationship between l and θ can then be derived from the model as follows:

$$l = \sqrt{x^2 + y^2 - 2 \cdot x \cdot y \cdot \cos(\theta - \theta_c)}$$

Assuming x is approximately equal to y, and θ_c is small, the relationship between the length and wrist angle is graphed in Figure 6-7.

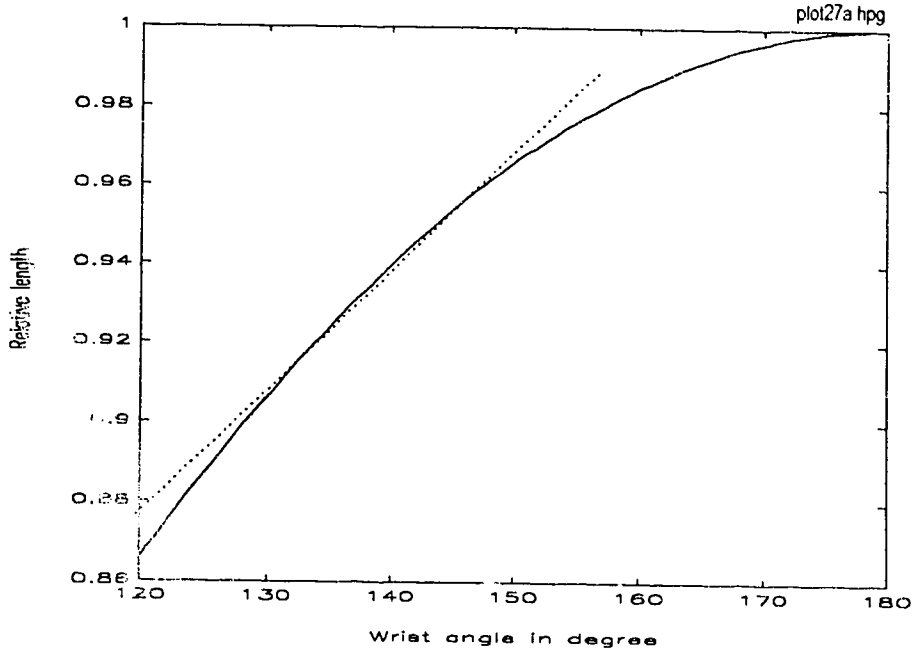


Figure 6-7: Length to wrist angle relationship with the experimental setup. *Notice that the relationship is quite non-linear close to 180 degrees. The useful range of this transducer is very limited.*

From the graph, we can see that the relationship is quite non-linear. During the actual experiments, only a very limited range of wrist angles was used. (about a total of 30 degrees movement around a wrist angle of 140 degrees). In this case, the system still behaves quite linearly. However, to be able to put the system into practice, a much wider range of linearity is required, not to mention a more robust structure. A second length transducer configuration is designed to be more portable. This configuration will give a nearly linear translation of the joint angle, over a wider range of wrist angles.

6.3.2 Proposed portable configuration

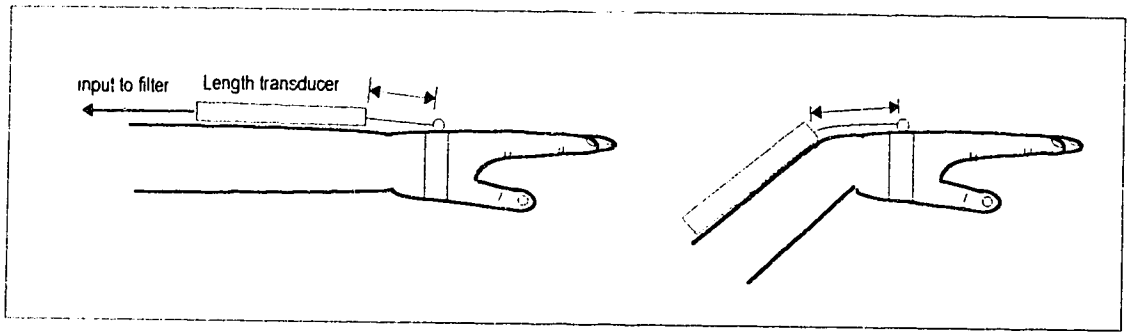


Figure 6-8: A portable length sensor placement on the arm

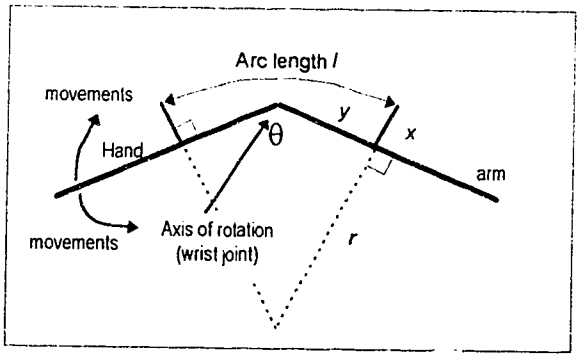


Figure 6-9: A simplified model for the portable length transducer configuration.

From the diagram, the arc length l, which correspond to the reading from the length transducer, can be modeled as:

$$l = (r + x)(\pi - \theta)$$
$$= \left[x + y \cdot \tan(\frac{\theta}{2})\right] \cdot (\pi - \theta)$$

Theoretically, depending the ratio of x to y, the equation can become very non-linear, as shown in Figure 6-11. However, the points can be selected such that x/y is approximately equal to 1.

In this case, the angle to length relationship is nearly linear, as shown in Figure 6-10. [refer to chapter 8, section on recorded results]

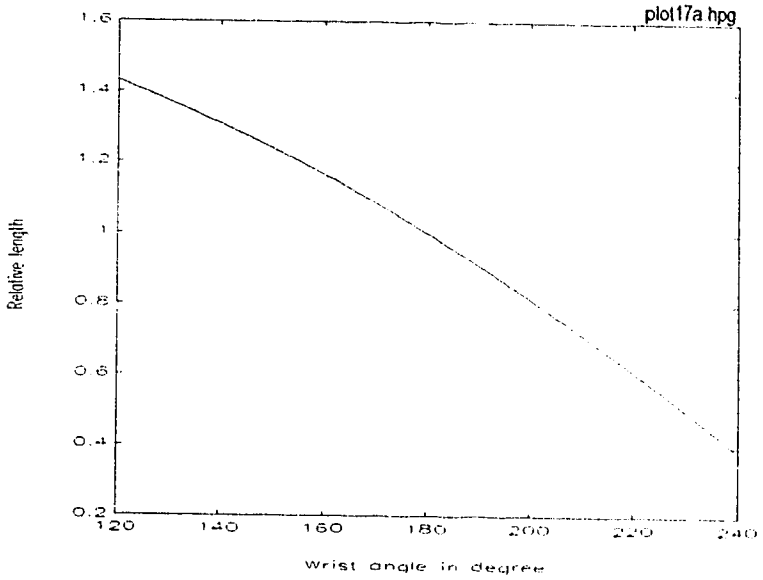


Figure 6-10: Relationship between the angle and length with $y/x = 1$

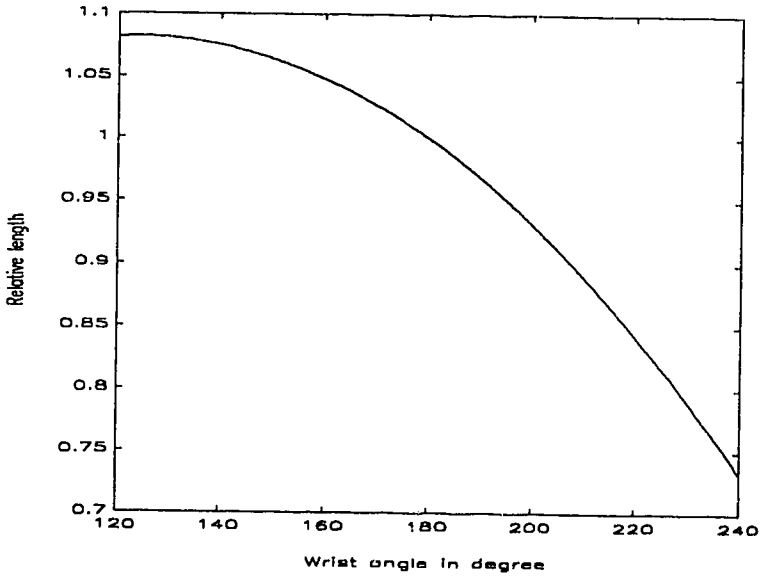


Figure 6-11: Relationship between the angle and length with $y/x = 3$

There are two variations in the types of portable length transducer structures employed in the system.

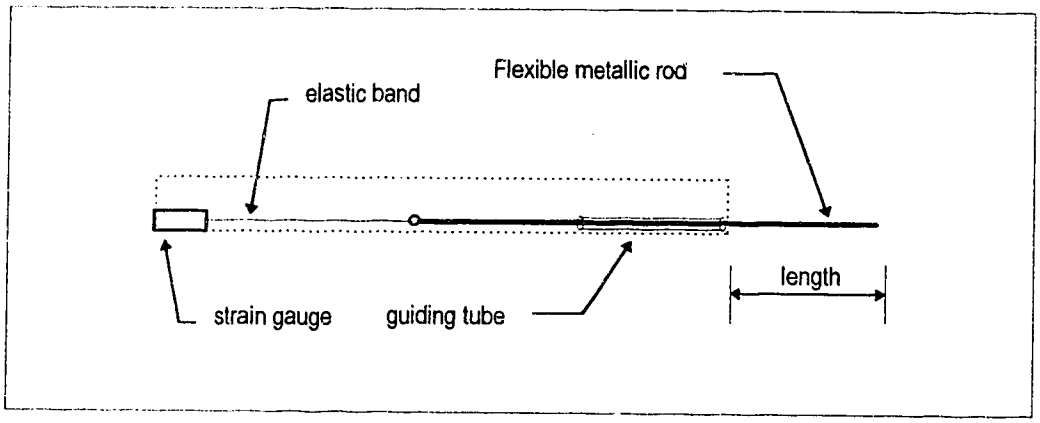


Figure 6-12 Structure of the length transducer using an elastic string

The first is basically a strain gauge attached to an elastic string, as shown in Figure 6-12. The strain gauge will translate strain on a piece of metal linearly into voltage levels. The whole device then makes use of the formula:

$$x = \frac{F}{k}$$

where x is the elongation of the elastic string from the resting length, and k is a constant. Thus the voltage level is directly proportion to the length difference of the elastic string from its resting length.

A second length transducer basically replaces the elastic band with a modified linear voltage differential transducer or LVDT, as shown in Figure 6-13.

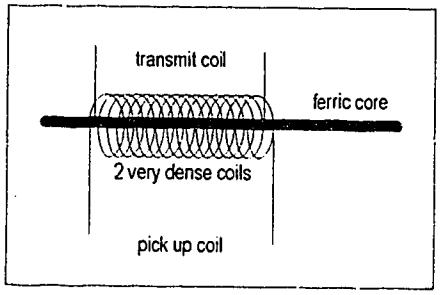


Figure 6-13: Structure of LVDT

The basic principle is that the transmit coil will create a fast varying magnetic field, which induces a current in the pick-up coil. The amplitude of the induced current will be affected by the amount of ferric material in the coil. As long as the coil is dense and even, the current induced will be linearly proportional to the length of metallic rod in the coil. The robustness of this transducer makes it preferable over the previous one, whose elastic string is subjected to wear thus degrading the linearity. However, the need for a high frequency oscillation (in the kilo-Hertz range) in the LVDT makes the output signals particularly noisy. There has been no success in utilizing the LVDT so far.

7.1 Experimental setup

The following figure (Figure 7-1) shows the typical experimental setup of the electrodes on the patients.

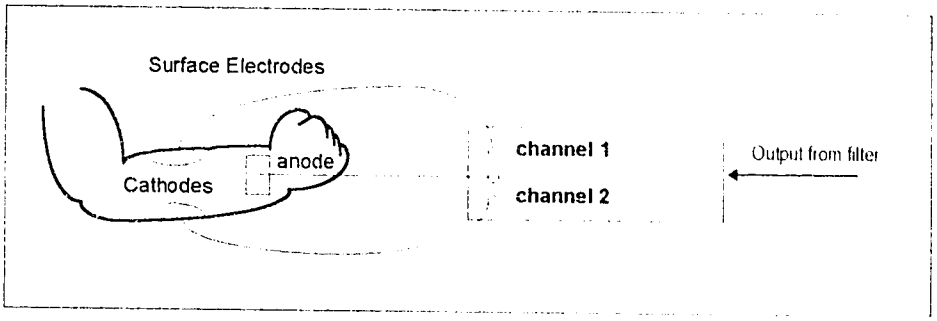


Figure 7-1: Surface electrode and stimulator setup

7.1.1 Electrode placement

The first step in setting up the whole system is to place the electrodes at the appropriate locations. Often much time is spent just to find the best location which produces the greatest contraction of the correct muscle without causing too much sensation. (The skin sensors will also be stimulated where muscle fibres are.) The anode can be placed almost anywhere (even at the leg) since it only serves as the return path for the current. However, the return path should never cross the body, which could potentially allow current to flow through the heart and cause heart arrest.

As discussed in chapter 3, functional electrical stimulation of muscle exhibits a threshold property. The stimulator must output this threshold level to each muscle even if the filter output is zero. This threshold varies from muscle to muscle, and also depends on the electrode placement. Therefore, the stimulation threshold for the extensor and the flexor must be set separately each time before the filter is operational.

When adjusting the thresholds, the filter output must be set to zero. This can easily be achieved by setting the gain of the filter to zero. The threshold for the muscle on the first channel can then be found by slowly increasing the stimulation output from zero until the subject senses the slightest contraction. The contraction is usually accompanied by a small sensation on the skin, and sometimes even visible vibrations of the muscle around the electrode. After setting the threshold of one channel, the same procedure can be applied to the other muscle on the second channel. It is best to leave the first channel at threshold stimulation because stimulation may drop when both channels are turn on. This means that when the threshold adjustment for the second channel is done, a re-adjustment of the first channel may be required because the stimulation may now be slightly below threshold. When the subject agrees that both channels are at threshold stimulation, the filter can be put into operation.

The next procedure is to set the gain for each channel. The subject is asked to mimic a tremor movement. The gain on the first channel is slowly increased until the subject signals a comfortable attenuation without the system going into instability. With the first channel still on, the gain on the second channel can be adjusted with the same procedure. Further

adjustments on the gains of both channels may be required to achieve a satisfactory result. The main objective is to have the subject makes his own judgement and finds the optimal effectiveness and comfort level of the system.

7.2 Trials with the 68HC16 system

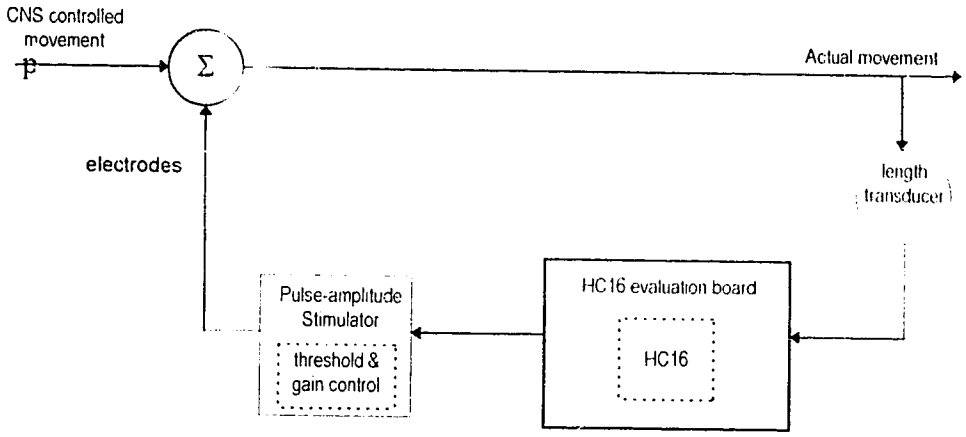


Figure 7-2: The 68HC16 system experimental setup

The 68HC16 system serves as a more ideal experimental setup in preparation for a practical portable system: the bionic glove. The 68HC16 valuation board serves only as the filter, and a separate analog stimulator which converts voltage output into pulse-amplitude modulated stimulation signal is employed as the stimulator. The [illegible] filter is the strain gauge which is placed below the arm with the elastic string end tied to the middle finger (as discussed in chapter 6). The movement of the hand is quite limited.

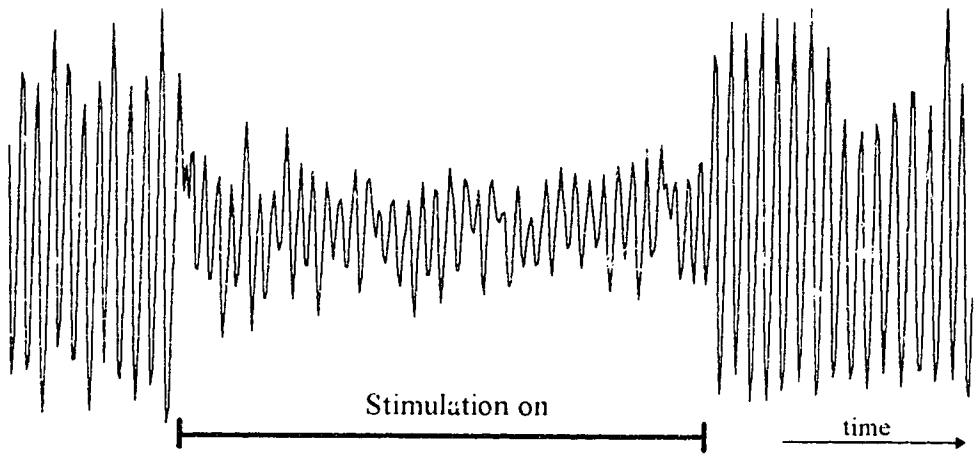


Figure 7-3: 68HC16 tremor suppression system operation on a normal subject performing a tremor-like 4Hz movement. This signal was recorded from the length transducer directly.

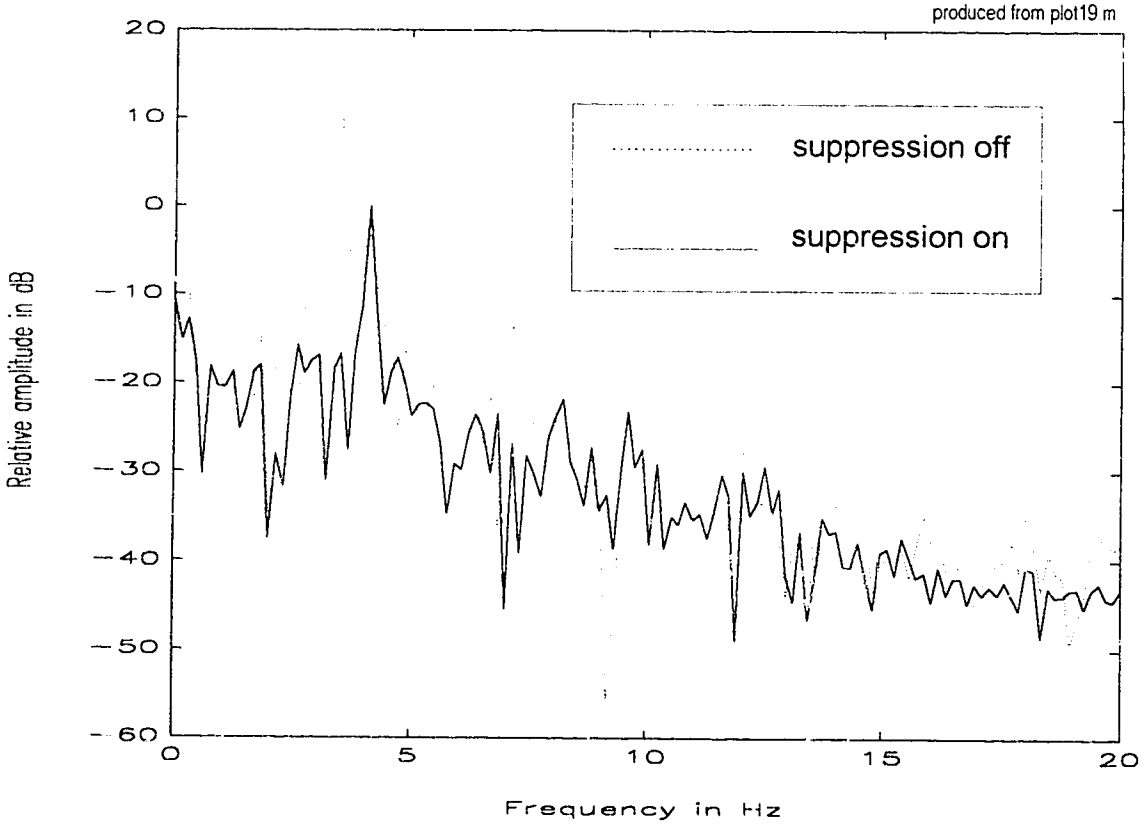


Figure 7-4: Fourier transform plots from the length signal before the after the 68HC16 tremor suppression system was turned on. *As seen from the plot, an attenuation of around 10dB is achieved. This corresponds to a 1:3.1 attenuation.*

The performance of the system was satisfactory. As shown in Figure 7-3, an attenuation of more than 1:3 was achieved. The Fourier spectrum graph in Figure 7-4 reveals some interesting information. When the system was off, the graph shows that the normal subject was indeed performing a tremor-like movement at around 4Hz. However, when the system was turned on, the peak of the frequency was shifted to near 5Hz. This phenomenon agrees with a previous statement that the stimulation does affect the central nervous system in some way that the analysis has not taken into account. Fortunately, the performance of the system is actually very close to the expected result from calculations, which predict that the suppression should be close to but no more than 1:3.8.

7.3 Trials with the bionic glove system (68HC11 system)

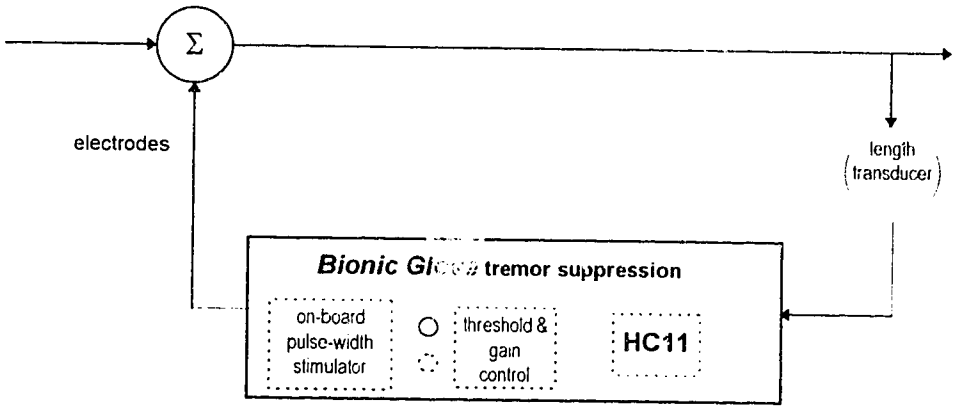


Figure 7-5: 68HC11 tremor suppression system experimental setup

The bionic glove system is a more practical implementation for the tremor suppression system. However, the ability of the hardware is quite limited. The 68HC11 inside the system has only 8-bit A/D converter, compared to a 10-bit A/D converter found in the 68HC16 system. The processor is slower, and it has to handle not only the filter calculations, but also the stimulation of the muscles. Despite all these limitations, the trial

of the system on a normal subject has been done successfully. The setup is done similarly to the 68HC16 trials, with the length sensor placed below the arm. A non-rechargeable battery is used, which supplies a more constant current than the rechargeable battery intended for use with the bionic glove.

The attenuation of the tremor by this 68HC11 system is more than 1:3 as shown in Figure 7-6, which is comparable to the 68HC16 system. The same frequency shift occurs when the system is turned on.

The selectivity of the system is also found to be satisfactory. When the subject was doing a slow movement (around 1/4 Hz), the suppression system did not attenuate this movement, as shown in Figure 7-8. The subject reported that his movement was smooth and unhindered. When the subject was mimicking an intended movement mixed with tremors (shown in Figure 7-9), the suppression system attenuated only the tremor movements. The slow intended movement was only slightly disrupted (shown in Figure 7-10). Overall the subject reported the system to be smooth and no uncomfortable stimulation was produced.

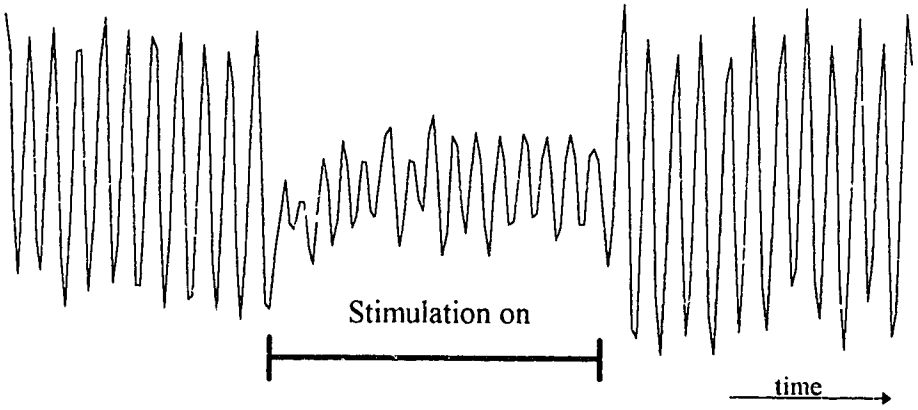


Figure 7-6: 68HC11 tremor suppression system working on a normal subject performing a tremor-like 4Hz movement. This signal was recorded directly from the length transducer.

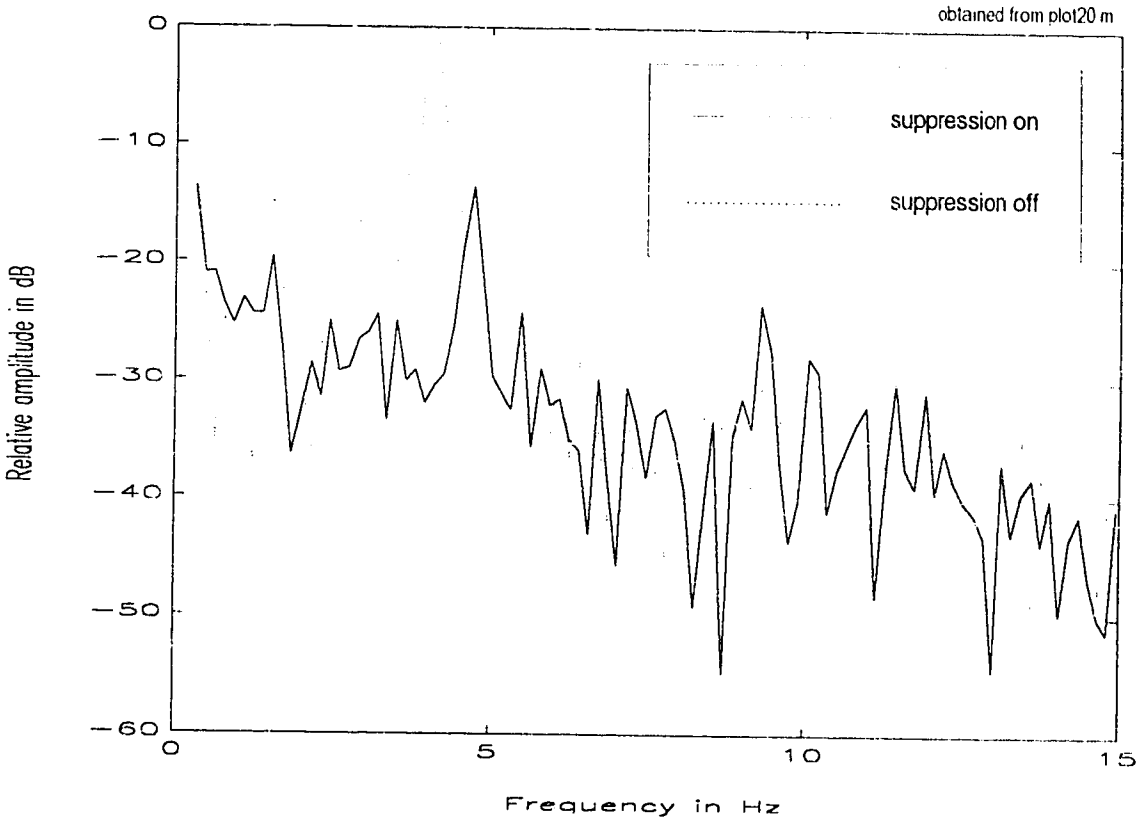


Figure 7-7: Fourier transform plots from the length signal before and after the 68HC11 tremor suppression system was turned on for a movement similar to one shown in Figure 7-6.

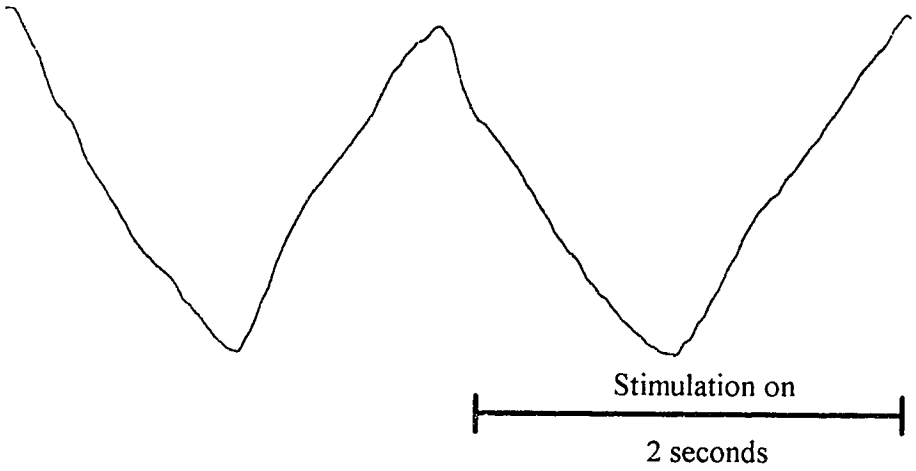


Figure 7-8: 68HC11 tremor suppression system on a normal subject performing a slow movement. The signal was recorded from the length transducer directly.

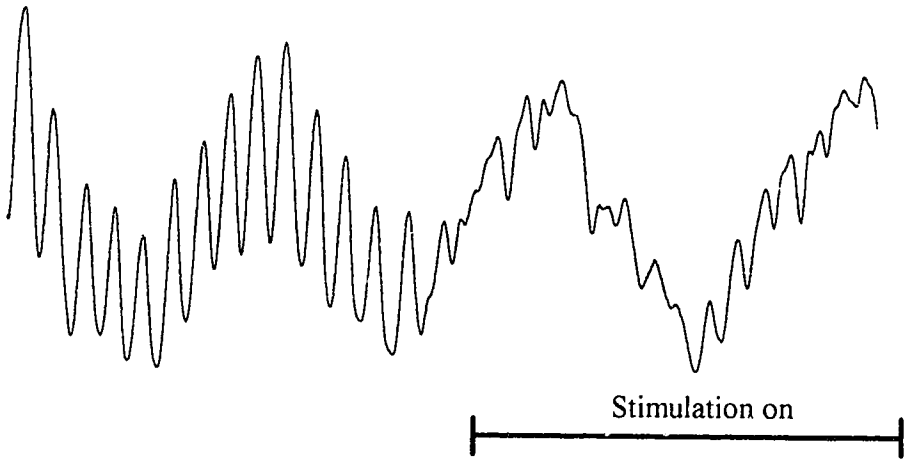


Figure 7-9: 68HC11 tremor suppression system on a normal subject performing a tremor-like 4Hz movement mixed with a slow intended movement. The signal was recorded directly from the length transducer.

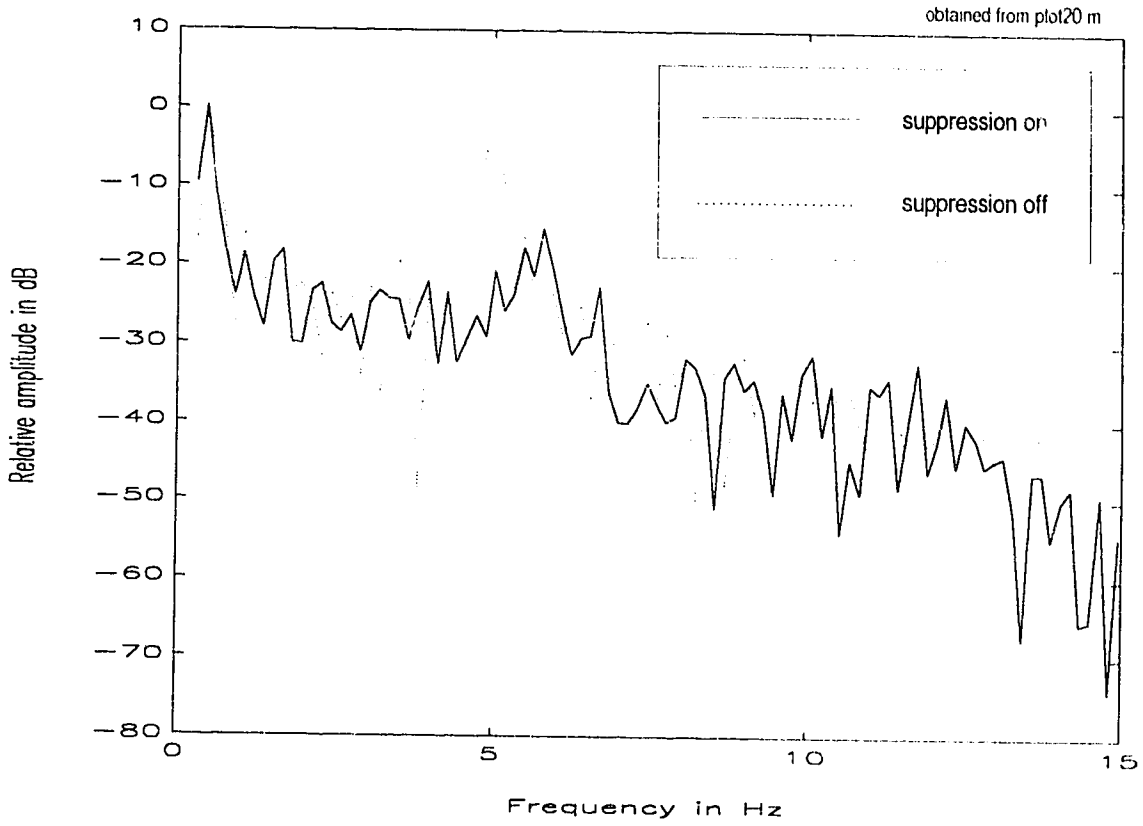


Figure 7-10: Fourier transform plots from the length signal before and after the 68HC11 tremor suppression system was turned on for a movement similar to the one shown in Figure 7-9. *Notice the intentional low frequency component of the signal, which was not attenuated.*

7.4 The final implementation

Although both the 68HC16 system and the 68HC11 system were tested experimentally on normal subjects with satisfactory results, the proposed final implementation was not tested on patients. This final implementation includes the bionic glove box and a portable length transducer discussed before. Experiments done with this setup failed on normal subjects. The system became unstable and even violent even with a small filter gain. Since the major difference between the experimental setup and the proposed final implementation is the length transducer, the transducer is suspected to be the cause of the failure. In the following chapter, the suspected cause of failure and other weaknesses of the system are discussed, and solutions are suggested to allow further research to be carried on.

8. DISCUSSION

8.1 Problem analysis

8.1.1 Length transducer construction

The length transducer used in the experiments is not very portable. As discussed in the previous chapter, this experimental setup allows only a limited range of movement. A more serious potential problem is that the elastic string is subject to external disturbance, introducing high frequency oscillation to the length signal. This could be a very serious problem because the filter is high-pass in nature. Any high frequency components can drive the system into instability. The portable length transducer proposed is designed to solve the mentioned problems. A proper placement allows the wrist to move freely without limitation. A plastic box protects the structure from external disturbance. A guiding tube ensures that the movement is one-dimensional. However, this portable length transducer introduces problems of its own, some of which are shown in Figure 8-1.

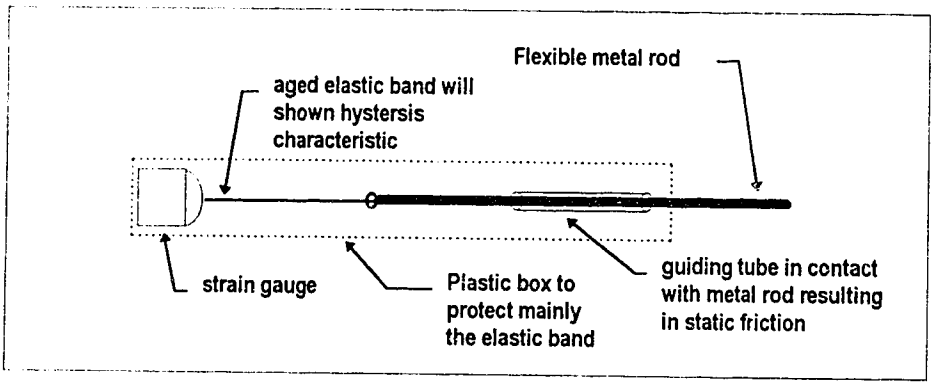


Figure 8-1: Weaknesses of the portable length transducer. *The construction of this portable length transducer has some weaknesses. However, until a better length/angle sensor can be found, these problems must be dealt with carefully.*

8.1.1.1 Hysteresis associating with elastic band

Figure 8-2 shows a plot of length to wrist angle relationship recorded from a transducer with aged elastic band. The plot clearly shows a hysteresis relationship. Since the data was recorded with the static friction problem to be described in section 8.1.1.2, the curve is not very smooth. The effect of this hysteresis can be estimated by using describing function analysis [Slotine and Li, 1991]. The describing function N(u) is defined as:

$$N(u) = \frac{b}{u} + \frac{a}{u} j ,$$

$$a = \frac{\Delta}{\pi \cdot u}, b = \frac{1}{\pi} \int_0^{2\pi} f(u \cdot \sin wt) \cdot \sin wt \cdot dwt$$

where Δ is the area inside the loop and $f(x)$ is the mean function of the relationship. The describing function for this relationship will be:

$$N(u) = 1 + \frac{\Delta}{\pi \cdot u^2} j$$

Basically, the imaginery part is $1/\pi$ of the ratio of the area inside the loop to the size of the plot in Figure 8-2. As seen from the plot, the ratio is so small that the describing function approximates a constant; that is, $f(x)$ is very close to a linear function.

A spring can be used to replace the elastic band which potentially eliminates or at least reduces the hysteresis problem. Unless the area of the loop becomes too large, the effect of the hysteresis will not be noticeable.

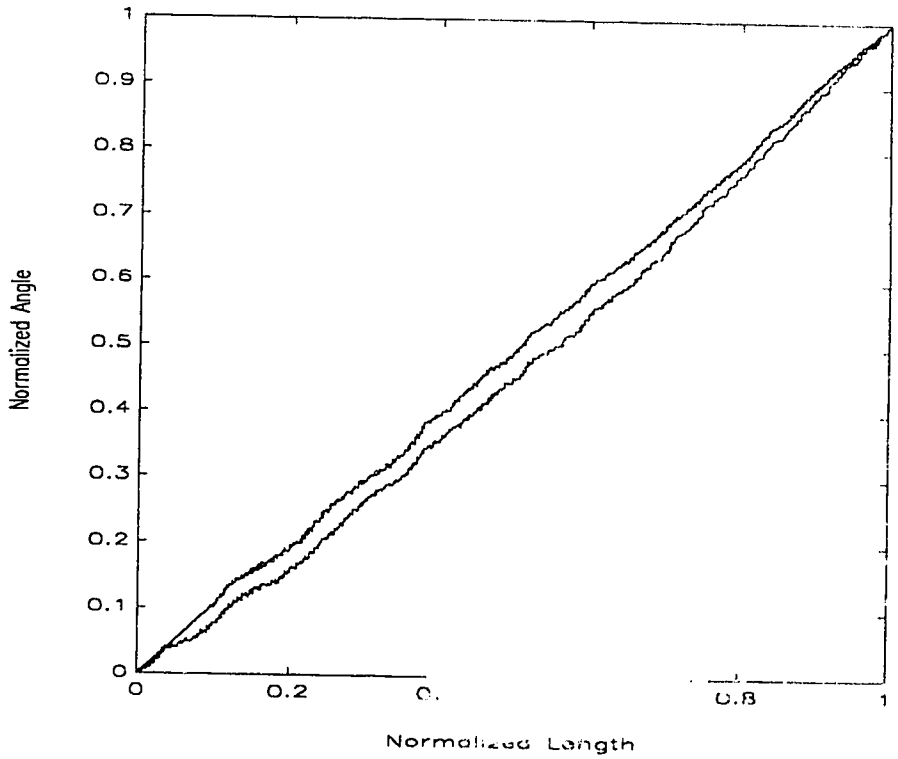


Figure 8-2: A plot showing a hysteresis relationship between the length and the wrist angle.

8.1.1.2 Non-linearity

Input non-linearity is one of the potential problems the system may have. Experiments using the experimental length transducer have been successful mainly because of the small range of tremor movements. The proposed portable length transducer has a more linear

length-angle relationship which allows a much larger range of operation without concern about the non-linearity problem. Figure 8-3 shows the relationship between recorded length signal and recorded angle signal. The curve is found to be fairly linear. Using least square curve fitting, a cubic curve approximating the recorded data is found as shown in Figure 8-4 (solid line). This curve turns out to match very closely to the theoretically calculated curve discussed in chapter 6. Although the relationship is not perfectly linear, the non-linearity is small enough to allow a faithful translation of wrist angle to length signal (see Figure 8-7).

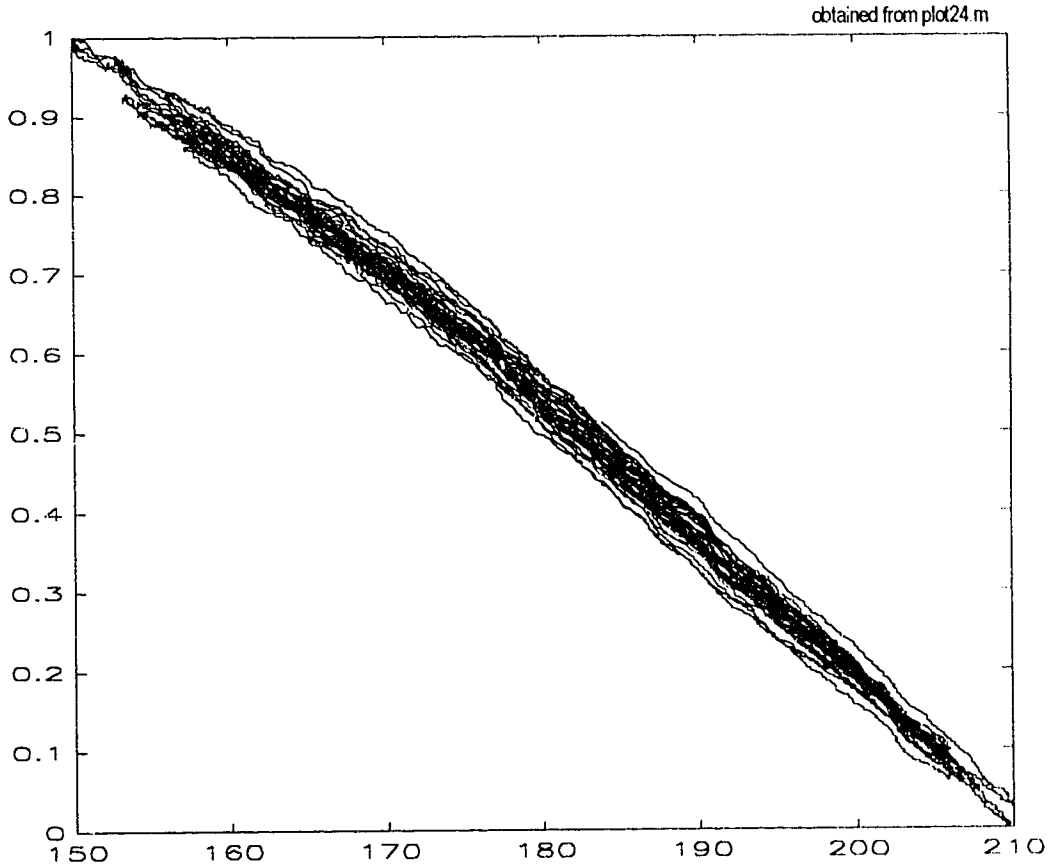


Figure 8-3: Actual recorded length to wrist angle relationship.

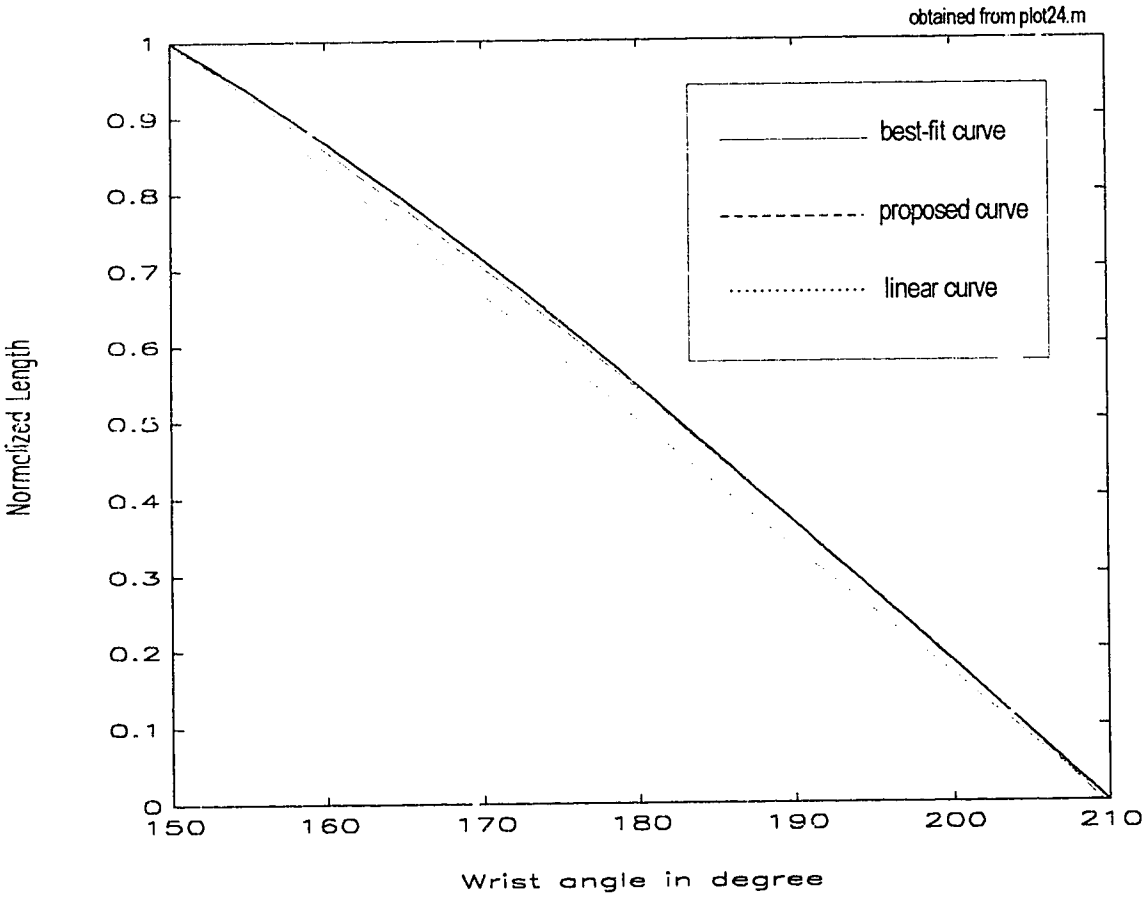


Figure 8-4: A best fit cubic curve for the recorded data using least square algorithm. *The best fit cubic curve is shown as the solid line. The relationship is nearly but not quite linear. Notice that this curve obtained from actual data agrees with the theoretical relationship proposed in chapter 6 for x/y = 1 (shown as dash line in the plot)*

8.1.1.3 Static friction causing random noise

The use of the guiding tube introduces a problem into the system. The flexible metal rod is in contact with the guiding tube at all time. If the surface on the rod or inside the tube is rough, the difference between the dynamic friction and static friction becomes very large. This will ultimately introduce random noise into the length signal, as shown in Figure 8-5. The Fourier spectrum plot of the length signal reveals that high amplitude second and third harmonics are present (shown in Figure 8-6). The harmonics can be explained by the fact that the static friction frequently stops the metal rod from moving momentarily. This will create a step-like length signal which contains high frequency harmonics. These harmonics and randomness are suspected to cause experiments to fail with the bionic glove system. In experiments conducted with the portable length transducer and the 68HC11 bionic glove system, the system reacted violently even with slow movements. With the subject mimicking tremor movements, the system quickly became unstable with a very small filter gain. Signals recorded directly from the A/D converter in the bionic glove system confirms that there are large amplitude harmonics present. As can be seen from Figure 8-8, the Fourier transform plot of that recorded signal closely resembles the plot obtained from the length transducer with large static friction. Unfortunately, this static friction problem was not realized when the experiments were conducted. An easy fix to this problem would be to apply lubricants to the metallic rod. A properly constructed and lubricated length transducer gives a faithful translation of the wrist angle, as shown in Figure 8-7.

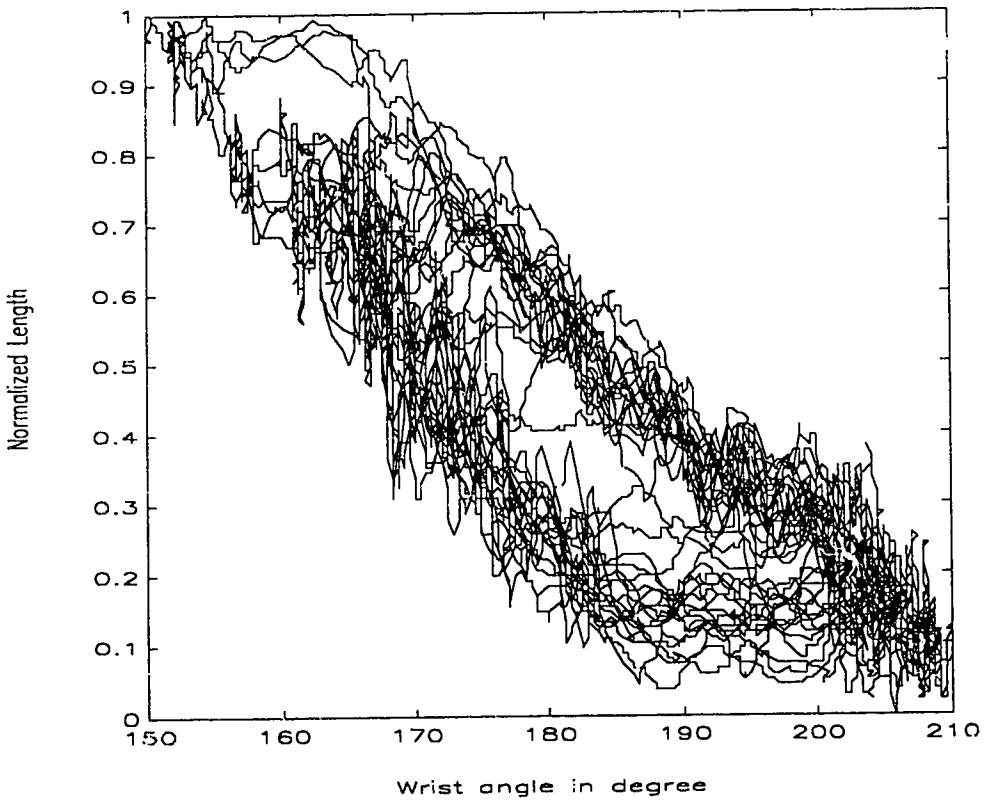


Figure 8-5: Relationship between length and wrist angle with static friction. *With large static friction between the metal rod and the guiding tube, the length to angle relationship became very non-linear and chaotic.*

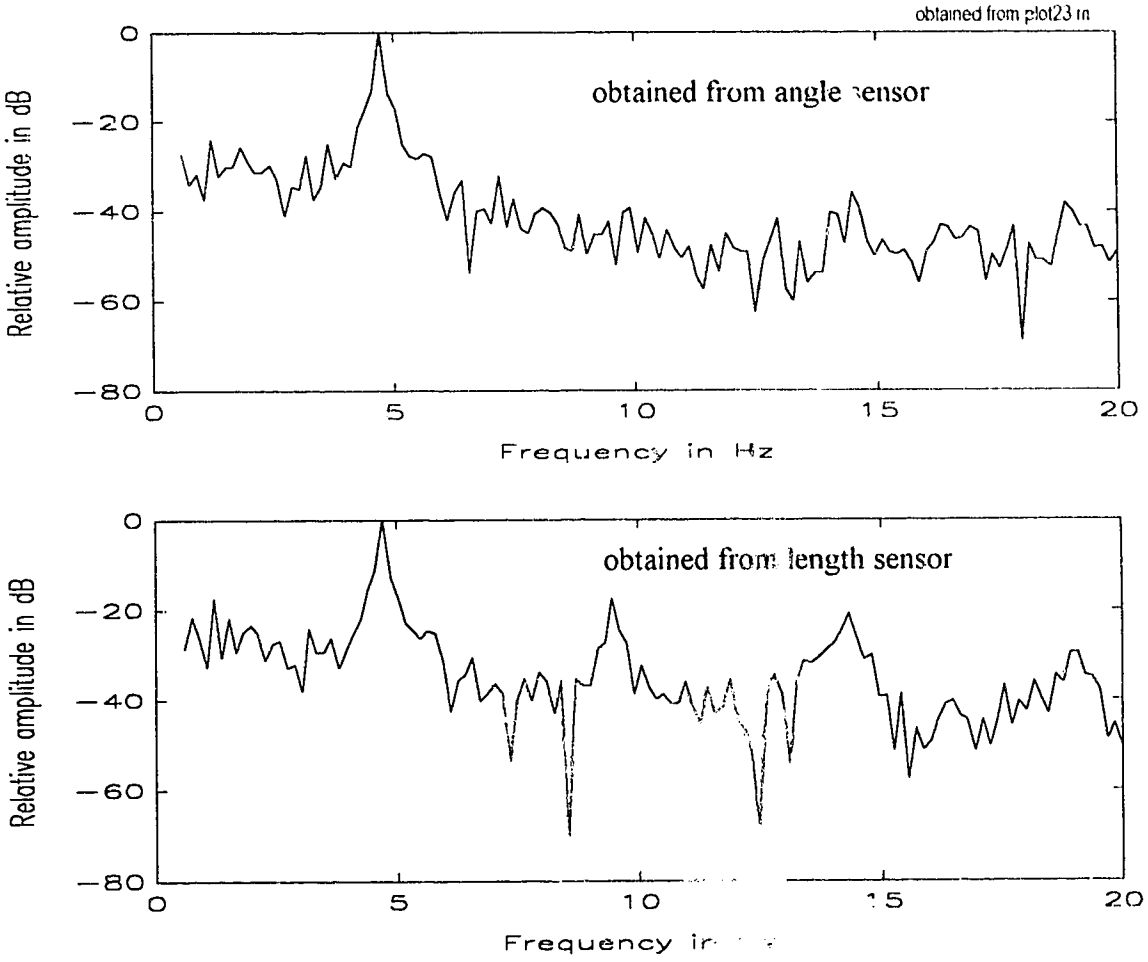


Figure 8-6: Fourier spectrum plots showing the effect of large static friction on the length transducer. *The top Fourier spectrum is obtained from the signal of an angle sensor at the wrist. A normal subject was mimicking a 4Hz to 5Hz tremor movement, which is shown as a peak around the frequency range. The Fourier spectrum at the bottom is obtained from the length transducer signal collected simultaneously from the same tremor movement. Notice that high amplitude harmonics exist in the length transducer signal. These harmonics are nearly non-existing when the length transducer is properly lubricated (Figure 8-7).*

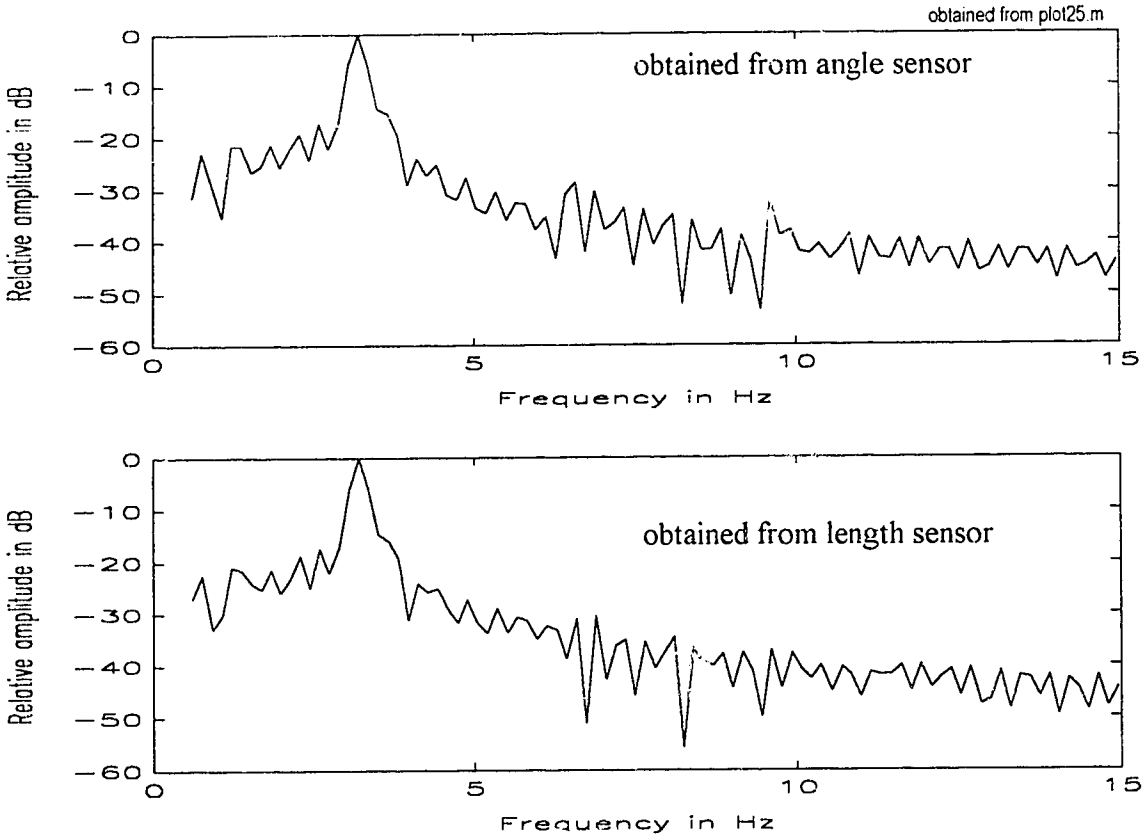


Figure 8-7: Fourier spectrum plots obtained from a properly constructed and lubricated length transducer. *A properly constructed and lubricated length transducer gives a faithful translation of the wrist angle. The top plot shows the Fourier transform plot from the angle sensor signal and the bottom is for the length sensor. These signals were recorded from a person mimicking a 4-Hz tremor. The two plots are nearly identical.*

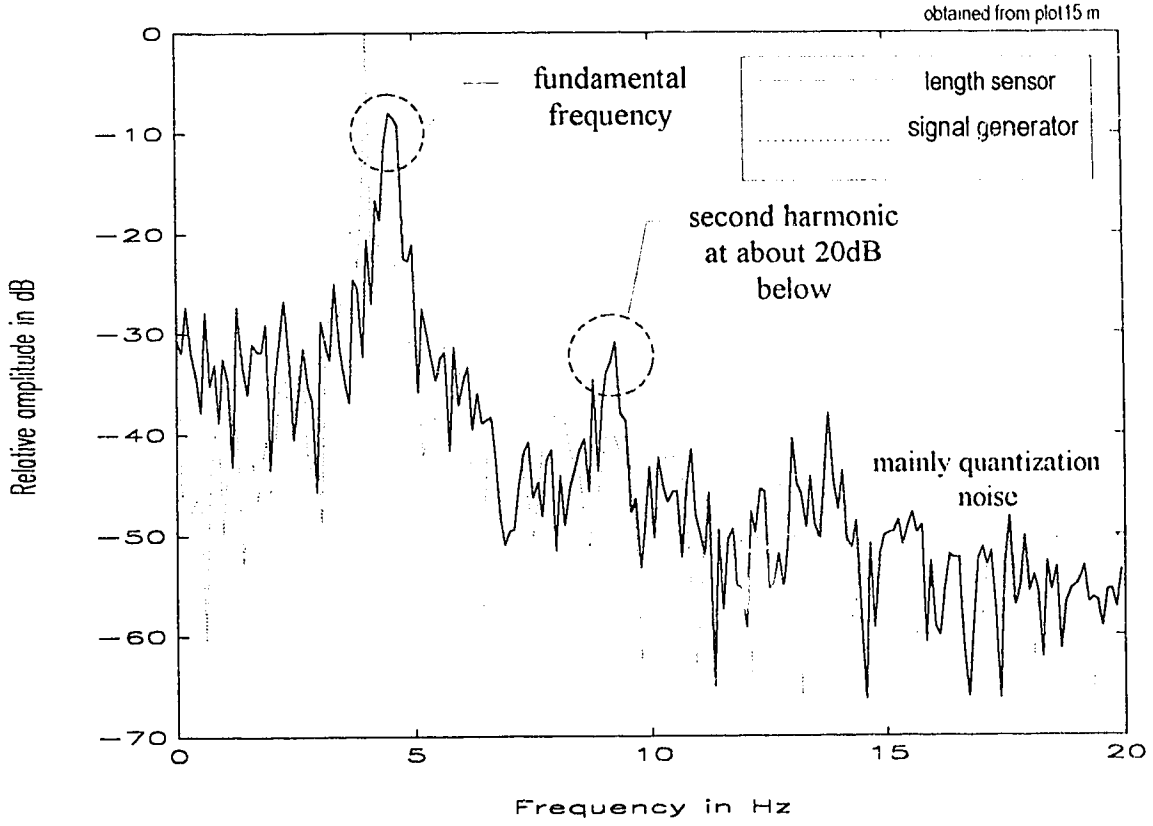


Figure 8-8: The Fourier spectrum plots of the signals the 68HC11 internal A-to-D converter saw. *1) solid line is the spectrum for the input signal from the potentiometer length sensor. 2) dotted line is the spectrum from a signal generator input at 4Hz. The quantization noise for both inputs are comparable. This proves that the length transducer does not introduce any more high frequency noise. The unwanted harmonics can be explained by the static friction discussed above.*

8.1.2 Signal-to-noise ratio

The bionic system uses the 68HC11 internal A/D converter for the length input signal. This A/D converter is only 8-bit, which corresponds to a theoretical signal-to-noise ratio (S/N) of around 41dB from the simplified formula:

$S/N = 6b - 7.1$ *[Kuc, 1988, Chapter 10]*

where b is the number of bits the A/D converter has. This agrees with the quantization noise recorded directly from the A/D converter, as shown in Figure 8-9. From that spectrum plot for the input signal, the quantization noise is found to be at around -45dB. The signal-to-noise ratio becomes a problem at the filter output. From the plot, the signal-to-noise ratio is reduced to around 30dB with the feedback loop not closed. The signal-to-noise ratio is further reduced once the feedback loop is closed. This, however, is the nature of this system because:

1. The useful signal is at a very low frequency (around 4 Hz) where the filter has a relatively low gain (around -30dB). This signal will be attenuated at the filter output.

2. This already attenuated output will cause a reduction of the input signal externally (external relative to the filter, i.e. tremor counteracted by the functional electrical stimulation). This causes a further attenuation of the useful input to the filter in the feedback loop (with a theoretical attenuation of around 10dB).

3. The filter is high pass. Quantization noise, unfortunately, is also dominating this high frequency range where no useful signal is found. This means most of the noise will pass through the filter unattenuated.

Clearly this system in nature leads to a greatly reduced signal-to-noise ratio. This can potentially cause dangerous high frequency stimulation. During the experiments done with the 68HC11 system, the length signal was adjusted to get the largest range. This means that the range of movements was limited to avoid clipping.

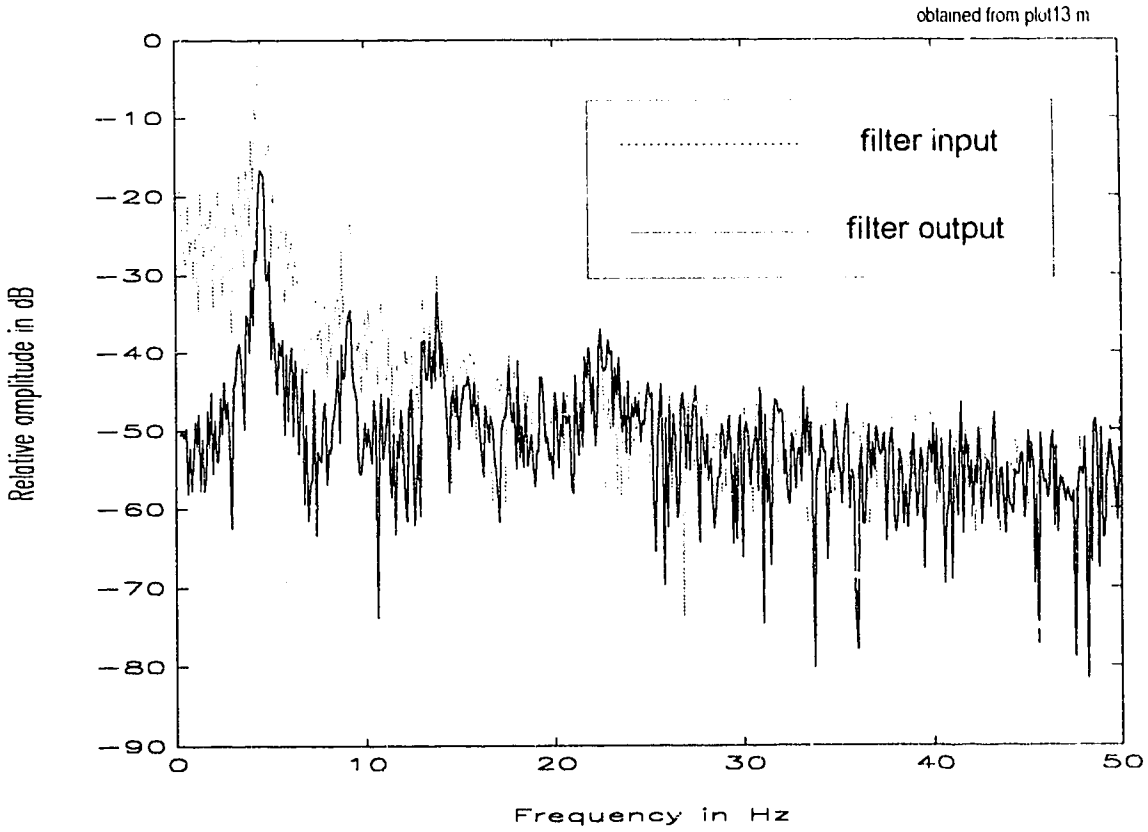


Figure 8-9: Fourier spectrum plots for both the input and output of the 68HC11 digital filter in an open loop system. *These are numbers directly obtained from the 68HC11 program. Notice that the signal-to-noise ratio decreases significantly because the high frequency components were never attenuated.*

8.1.3 Cross-over frequency

Another fundamental problem of the system is that the cross-over frequency of the filter unfortunately coincides with the second harmonic of the tremor frequency which is at

around 8Hz. This means that the second harmonic will be amplified dramatically by the closed-loop system if the system is close to instability. Figure 8-10 shows three curves:

1. The solid line is the Bode plot of the overall closed-loop gain of the system, i.e. the transfer function from x to y in Figure 4-1. The system is just below instability.

2. The Fourier spectrum plot of an input sample with static friction noise. It is shown as dotted line. The second harmonic is very apparent. This is the input.

3. The dash line shows the theoretical output of the closed-loop system with input signal mentioned in 2. This is the output y.

It is clear that the system is nearly unusable with such a large high frequency movement introduced into the system. Of course, it is possible to reduce the closed-loop gain at cross-over frequency by reducing the filter gain, but this will also degrade the system performance with a smaller tremor suppression.

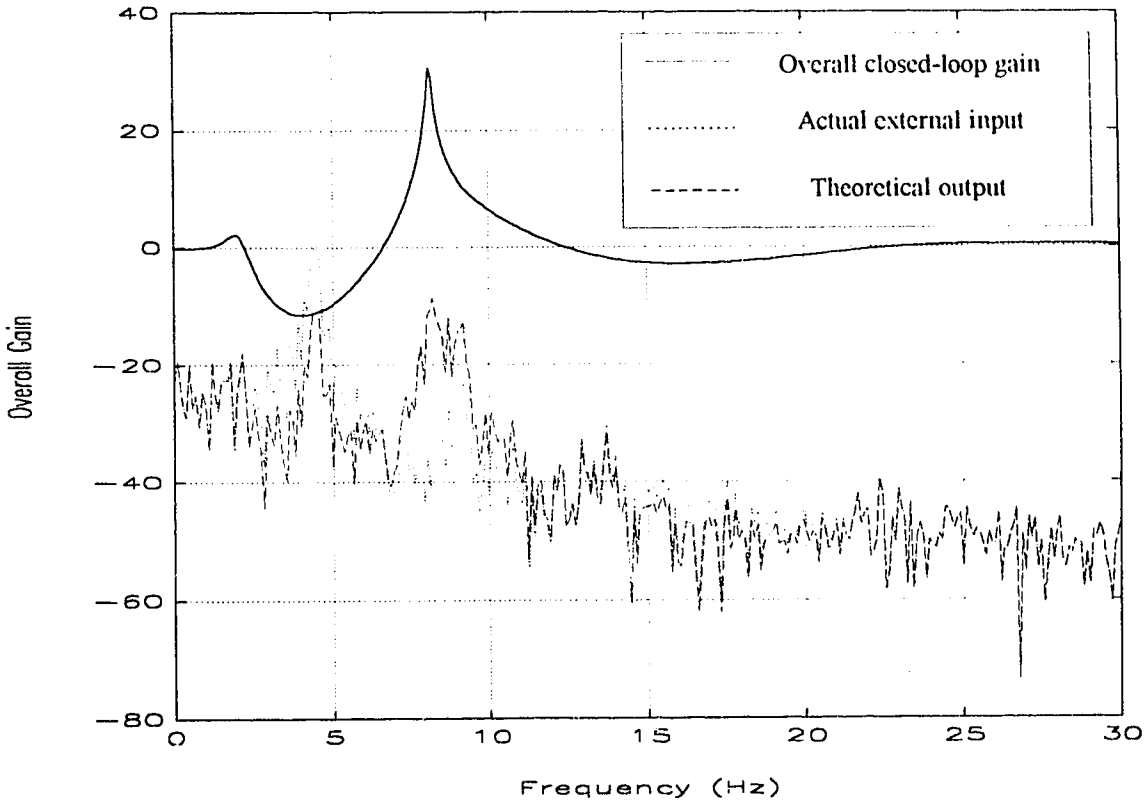


Figure 8-10: Overall closed-loop gain of the system and sample input and output. *The system is just below the instability margin. Notice that the second harmonic is amplified dramatically.*

8.1.4 Possible solutions

The worst problem encountered in the experiments conducted was the suspected static friction problem discussed in section 8.1.1.3. Fortunately, the solution to this problem is simply to lubricate the moving parts in the portable length transducer properly. This transducer will otherwise improve the system by allowing nearly unrestricted wrist movements without worrying about non-linearity. Another potential problem discussed in section 8.1.2 is a small signal-to-noise ratio at the output. The root to this problem is the insufficient A/D quantization word length the bionic glove system (68HC11) provides. To solve this problem, hardware change must be made to the system to use an external A/D converter with a longer word length. For example, a 12-bit A/D converter is readily available and is relatively inexpensive. This will increase the signal-to-noise ratio by 24dB, enough to solve the problem. Once the above steps are done, other problems associated with the fundamental design of the system will be solved at the same time because high frequency noise and unwanted harmonics are gone.

8.2 The evolution of the original analog device

The original tremor suppression system designed by Prochazka et al. is a large and heavy device which has to sit on a table. The length transducer limits the range of wrist movements and is subject to external disturbance. To change the filter parameters, a matrix of resistors and capacitors has to be replaced one-by-one carefully. The system has now evolved into a small portable system which sits on top of the arm. All the signal filtering which was done by an analog circuit is now all calculated by the micro-controller

in the portable system. Notice that the micro-controller in the portable device is not designed to do any intensive calculations like filtering. However, with careful and intelligent assembly language programming and a wise choice of algorithm, this is made possible even with the micro-controller also controlling the electrical stimulation at the same time. Changing parameters of the filter is no longer a painful task of replacing resistors and capacitors on the circuit; it can now be accomplished easily by reprogramming the micro-controller. Increasing the order of the filter will not result in a more complicated circuit but only a slightly longer parameter list within the micro-controller and a longer calculation period. Unfortunately, adaptive or nonlinear feedback control algorithm was not used, mainly due to the difficulty of finding a suitable design. Even if an algorithm were found, the usual complexity of such algorithms would require likely much more processing power than available with the micro-controller used in the portable system.

In addition to the bulkiness of the analog system, the deficiency of the old length transducer structure is also a factor that limits the old system to only experimental usage. The wrist movements were confined because, firstly, the length-to-angle relationship is nearly linear only in a small range, and secondly the elastic string on the length transducer may touch the body or clothes if the range of wrist movements is too large. The new portable length transducer solves both problems by the use of a protective case and a more suitable position. The whole system is small and robust, and is now ready to be tested on patients in real-life situations.

8.3 Useful tools

In developing and testing different digital filter designs during the research, MATLAB programs were written to automatically create parameter files from any valid filter transfer function. There is a program for the 68HC16 evaluation system and one for the 68HC11 bionic glove system. The parameter file can then be inserted into a master assembly language program (again, one for each system) to produce the required filtering program. In the case of the bionic glove system, the master program also contains codes to do electrical stimulation The process from generating a parameter file to putting the program into the system takes only about 10 minutes. Notice that the above procedures can also be applied to general purpose filter design, not just the tremor suppression system.

During the research, a graphical data acquisition program (DASA2) was developed which is intended to be used for general purposes in the laboratory. A variant of this program is designed specifically to sample data from the bionic glove tremor suppression system. The filter program in the system sends the value of the internal A/D converter and the filter output through an optical serial link to the acquisition program, which displays the data in real-time. Most of the graphs shown in this thesis use the data sampled with this acquisition program.

8.4 Conclusions

In this thesis, different means of suppressing tremor using digital feedback control were studied and experimented. The use of mechanical braking has been considered but no suitable device was found to effectively implement the idea. A system using functional

electrical stimulation was transformed and improved from an analog system to a digital implementation. Although no experiment using this digital implementation was conducted on patients, those done with normal subjects have been successful. Satisfactory suppression and comfortable stimulation were reported. With the small size and lightness of the bionic glove system, it would be feasible to use the system outside an experimental environment. Patients can potentially carry the device with them during almost all normal daily activities and only turn on the system when necessary. In essence, the ultimate goal sought by this thesis research has been reached already: a portable, self-contained tremor suppression system using function electrical stimulation has been designed. Problems have been identified and analyzed, and the system is now ready to be tested on patients after suggested steps to elimitate remaining problems are implemented.

References

Kuc, R., "Introduction to Digital Signal Processing", McGraw-Hill Book Company, 1988

Slotine, J. E. and Li, W., "Applied Nonlinear Control", Prentice-Hall, 1991

APPENDIX A

Bionic Glove Tremor Suppression System: User Manual.

The Bionic Glove: wrist tremor suppression

Introduction

The bionic glove is a portable stimulator designed in-house[1]. It is driven by a Motorola 68HC11[2], with 256 bytes of internal RAM and 2 Kbytes of on-chip EEPROM. The circuit has a DC-DC converter, an optical serial interface and 4 independent, time-multiplexed output channels. It uses an internal rechargeable 9V battery, and a special length sensor called the LVDT[3].

The stimulation method

With the bionic glove, there is a lot of flexibility on the stimulation method. The amplitude and pulse width for each channel can be controlled independently. However, the amplitude has only 6-bit of resolution, and the pulse width has a 0.5μs step resolution. The two parameters must be coordinated to get the maximum controllable range for the muscle.

The physical specifications

Weight: 135g (including rechargeable battery)

Dimension: 5cm x 1.7cm x 11.8cm

---

[1] Designed by Michel Gauthier and Zoltan Kenwell at University of Alberta

[2] Please refer to Motorola HC11 databook for specifications and details

[3] Developed by Zoltan Kenwell.

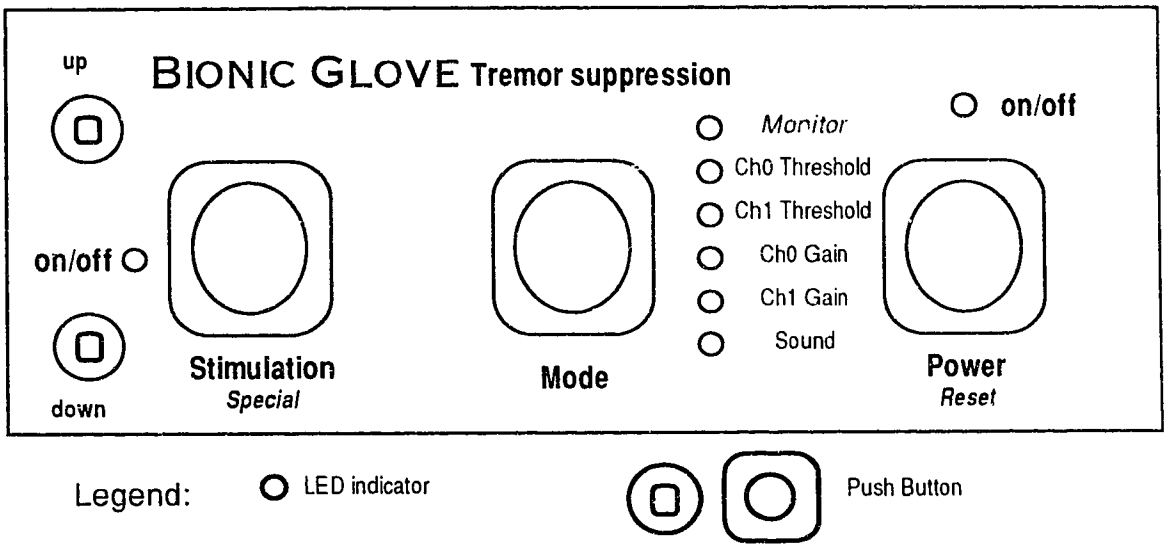


Figure 1: Panel Face

The buttons

There are a total of 5 push buttons labeled *Stimulation*, *Mode*, *Power*, *Up* and *Down*. They function as follow:

- *Power*: Toggle the power of the whole unit on and off. The LED above the switch lights if the power is on.
- *Stimulation*: Toggle the stimulator on and off. The LED above the switch flashes if the stimulator is on. This button, when held, also allows the unit to enter special operations. For details, please read the section "Special Modes" on page 1.
- *Mode*: Change the mode of operation. It also toggles the sound on and off. The LEDs beside the switch will indicate the current mode and the status of the sound output.
- *Up* and *Down*: Used for adjusting the level of the parameters. These buttons have an auto-repeat function.

The Mode indicators

There are five valid modes:

- Monitor mode, for debugging purpose
- Channel 0 threshold adjustment
- Channel 1 threshold adjustment
- Channel 0 gain adjustment
- Channel 1 gain adjustment

A sixth LED indicates the status of the sound output. If the LED is lighted, the stimulator will generate sound which loudness is proportional to the strength of the stimulus.

Changing modes

A flashing light on the Mode indicators represents the current mode. After pressing the Mode button, a steady light will appear. The position of the light will cycle through all the available modes when the mode button is clicked. When the desired mode LED is lighted, wait about a second and it will become a flashing light which indicates that the new mode has been selected. If the light is cycled to the sound status LED, the sound output will be toggled on and off.

Changing levels

There are 4 parameters that must be adjusted: thresholds and gains for both channels. These can be increased and decreased in the corresponding mode by pressing the up and down buttons. When the up/down buttons are pressed, the Mode Indicator LEDs will become a scale indicator. When a single LED is lighted, the vertical position of it indicates the approximate value of the parameters on a full scale (6 LEDs). When there is no LED lighted, this indicates a zero value. When all the LEDs are lighted, this indicates a maximum value. Clicking the up/down buttons will increase/decrease the level by one at a time. If the buttons are held pressed, the level will increase/decrease continuously.

Special Modes

The tremor suppression glove unit has 4 built-in mode-toggles for experimental and debugging purposes. These are:

- *Pulse amplitude/width control*: instead of using pulse width as the varying output, the amplitude of the pulse can be used. In each mode, the unit will remember a different set of thresholds and gains, so that if the unit is switched between the two modes of control, no re-calibration is needed.
- *conditional filter on/off*: The conditional filter can be turned on and off by toggling this mode. Warning: *Without the conditional filter, any high frequency disturbance can potentially cause a high power output, resulting in pain or even injury.*
- *by-pass filter*: When this mode is toggled on, the output will follow the input directly without transformation. This mode is only for testing purpose only. The user is advised not to change this mode.
- *Use 25Hz/33Hz Stimulation*: This mode switches between using 25Hz or 33Hz as the stimulation frequency. This has a small effect on the comfort level so the user should choose one to get the best but comfortable suppression.

To toggle these special modes

Press the stimulation button and keep it down. After clicking the mode button once, the mode indicators will show the special modes currently switched on. Clicking the mode button then will cycle a steady light through the different modes. After waiting for about one second, the mode where the steady light is will be toggled, and the new current status will be shown, and the special modes will be in effect immediately. Releasing the stimulation button will revert the unit to normal panel display.

There is yet another set of parameters that can be adjust. With pulse width control, the amplitude of the pulse output is fixed at a certain voltage. It is sometimes desirable to adjust this voltage so that the stimulation has a more linear effect. This level can be adjust when the unit is in the two threshold adjustment modes. To adjust the level, depress the stimulation button and keep it down, and decrease or increase the level with the down/up button. Releasing the stimulation button will revert the unit to normal operations.

Calibration

The unit must be calibrated to suit a particular individual when it is first used. Minor adjustments may be needed in subsequent uses, but all the parameters will be remembered if the system is turned off by pressing the power button, even after the battery has been disconnected.

There are a total of six parameters that can be adjusted. Four of these must be adjusted: the threshold and the gain for both channels.

To adjust the threshold, set the mode to the appropriate one. Make sure the level is low to start with. Slowly increase the level until a contraction of the muscle is produced. However, if the level reaches the mid-range without causing a contraction, the amplitude of the pulse should be increase instead. Increase the amplitude until the muscle contraction. In any case, threshold level is the level which is just about to cause a muscle contraction. Use the pulse width level adjust to fine tune the unit to such a level.

After adjusting the threshold for both channels, the gain should then be adjusted. Increase the gain until a comfortable but effective suppression is achieved, or a level just below instability is found. Repeat the adjustment for all four parameters to achieve the best suppression.

Resetting the parameters

When the unit needs to be re-calibrated for a new patient or new setup, the parameters should be reset. This can be done quickly by holding the stimulation button while clicking the Power button. All parameters will revert to zeros and the user can then proceed to a calibration.

APPENDIX B

Listing of the 68HC11 assembly code running in the bionic glove system.

Pseudo Code for 68HC11 code

```
main() {
        /* Initialization */
        Initialize I/0;
        Clear memoriy to zeros;
        Setup timer and interrupt;

        /* Wait for interrupt */
        LOOP forever;
}

/* The following routine handle interrupt ocurring each sampling period */

Interrupt_service_routine() {

        Clear interrupt and setup next interrupt;
        Get current button status;
        If (power button is pressed) goto Power_down();
        If (no pending mode) {
                show current mode;
        } else if (mode still pending) {
                show current mode;
        } else if (pending mode timeout) {
                Change pending mode to current mode;
                show new mode;
        }
        Flash LED;
        Prepare magnitude on both channels to threshold only;
        If (current mode = Ch0gain or Ch1gain) {
                Filter_calculate();
                /* Subroutine will return the amplitude and which channel to */
                /* stimulate */
                Prepare channel stimulation according to filter result;
        }

        Start stimulation sequence;
        /* timing is not interrupt driven. 68HC11 has facility to control */
        /* timing on external pins */

        Process_key();

        return from interrupt;
}
```

```
Power_down() {
        Save parameters to EEPROM;
        Cut off power;
        Stop processor;
}

/* Main user interface */
Process_key() {
        If (mode key is pressed) goto change_mode();
        if (stim key is pressed) toggle stimulation;
        if (power & stim pressed) clear parameters;
        if (current mode does not need A/B buttons) return;
        if (not A or B button) return;

        Change level according to mode and A/B button;
        Show approximate scale of level on LEDs;
        return;
}

change_mode() {
        If (in pending mode) change pending mode;
        else go into pending mode;
        set pending timeout period;
}

filter_calculation() {
        Read Length transducer;
        store as x(n);
        for (I = 1 to last section) {
                y(n) = bi0 * x(n) + wi1(n-1);
                wi1(n) = bi1 * x(n) + ai1 * y(n) + wi2(n-1);
                wi2(n) = bi2 * x(n) + ai2 * y(n);
        }
        store y(n) from last section;
        determine which channel to stimulate;
        return;
}
```

```
American Automation Cross-Assembler 68HC11 9.04.11
Assembly date: 23 Jan 1995
Assembly time: 20:59:42
Options in effect:expand format list object errorfile uppercase

                ; File: tremor.s
                ;
                ;*****************************************************************
                ; Include definitions of all internal registers
                ;*****************************************************************
                        include register.s
                ;
                ;*****************************************************************
                ;     68HC11E Standard Register Addresses
                ;*****************************************************************
                ;
00001000 PORTA  EQU     $1000           ; I/O Port A
00001002 PIOC   EQU     $1002           ; Parallel I/O Control Register
00001003 PORTC  EQU     $1003           ; I/O Port C
00001004 PORTB  EQU     $1004           ; Output Port B
00001005 PORTCL EQU     $1005           ; Alternate Latched Port C
00001007 DDRC   EQU     $1007           ; Data Direction Register for Port C
00001008 PORTD  EQU     $1008           ; I/O Port D
00001009 DDRD   EQU     $1009           ; Data Direction Register for Port D
0000100A PORTE  EQU     $100A             Input Port E
0000100B CFORC  EQU     $100B             Compare Force Register
0000100C OC1M   EQU     $100C             Action Mask Register
0000100D OC1D   EQU     $100D           ; OC1 Action Data Register
0000100E TCNT   EQU     $100E           ; Timer Counter Register
00001010 TIC1   EQU     $1010           ; Input Capture 1 Register
00001012 TIC2   EQU     $1012           ; Input Capture 2 Register
00001014 TIC3   EQU     $1014           ; Input Capture 3 Register
00001016 TOC1   EQU     $1016           ; Output Compare 1 Register
00001018 TOC2   EQU     $1018           ; Output Compare 2 Register
0000101A TOC3   EQU     $101A           ; Output Compare 3 Register
0000101C TOC4   EQU     $101C           ; Output Compare 4 Register
0000101E TOC5   EQU     $101E           ; Output Compare 5 Register
0000101E TI4O5  EQU     $101E           ; Output Compare 5 Input Capture 4 Register
00001020 TCTL1  EQU     $1020           ; Timer Control Register 1
00001021 TCTL2  EQU     $1021           ; Timer Control Register 2
00001022 TMSK1  EQU     $1022           ; Timer Interrupt Mask Register 1
00001023 TFLG1  EQU     $1023           ; Timer Interrupt Flag Register 1
00001024 TMSK2  EQU     $1024           ; Timer Interrupt Mask Register 2
00001025 TFLG2  EQU     $1025           ; Timer Interrupt Flag Register 2
00001026 PACTL  EQU     $1026           ; Pulse Accumulator Control Register
00001027 PACNT  EQU     $1027           ; Pulse Accumulator Count Register
00001028 SPCR   EQU     $1028           ; SPI Control Register
00001029 SPSR   EQU     $1029           ; SPI Status Register
0000102A SPDR   EQU     $102A           ; SPI Data Register
0000102B BAUD   EQU     $102B           ; SCI Baud Rate Control
0000102C SCCR1  EQU     $102C           ; SCI Control Register 1
0000102D SCCR2  EQU     $102D           ; SCI Control Register 2
0000102E SCSR   EQU     $102E           ; SCI Status Register
0000102F SCDR   EQU     $102F           ; SCI Data (Read RDR Write TDR)
00001030 ADCTL  EQU     $1030           ; A/D Control Register
00001031 ADR1   EQU     $1031           ; A/D Result Register 1
00001032 ADR2   EQU     $1032           ; A/D Result Register 2
00001033 ADR3   EQU     $1033           ; A/D Result Register 3
00001034 ADR4   EQU     $1034           ; A/D Result Register 4
00001035 BPROT  EQU     $1035           ; EEPROM Block Protect Register
00001039 OPTION EQU     $1039           ; System Configuration Options

0000103A COPRST EQU     $103A           ; Arm/Reset COP Timer Circuit
0000103B PPROG  EQU     $103B           ; EEPROM Program Control Register
0000103C HPRIO  EQU     $103C           ; Highest Priority Interrupt Bit Int and Misc
0000103D INIT   EQU     $103D           ; RAM and I/O Mapping Register
0000103E TEST1  EQU     $103E           ; Factory TEST Control Register
0000103F CONFIG EQU     $103F           ; COP, ROM, and EEPROM Enables
                ;
                ;*****************************************************************
                ; Compiler Constants
                ;*****************************************************************
                ;
FFFFFFFF true   equ     -1
00000000 false  equ     0
                ;
                eemversion:
FFFFFFFF        equ     true            ; standalone version with 2K EEPROM
                saveparam:
FFFFFFFF        equ     true            ;
                crystal80000:
FFFFFFFF        equ     true            ; 8MHz (2MHz instruction cycle)
                crystal49152:
00000000        equ     false           ;
                crystal36864:
00000000        equ     false           ;
                analysismode:
00000000        equ     false           ; this mode will allow filter
                ;                                 ; output at different stages
                ;                                 ; to be seen on the SCI port
                ;
                ;*****************************************************************
                ; Addresses
                ;*****************************************************************
                ;
                ; 2k EEPROM version of 68HC11 memory map:
                ;
                ; 0xFFFF +------------------+ } interrupt vectors
                ;        | EEPROM area      |
                ; 0xF800 +------------------+ <- start of application program
                ;        | unused           |
                ; 0x0100 +------------------+ <- top of stack (grow down)
                ;        | Internal RAM     |
                ; 0x0000 +------------------+
                ;
0000F800 prog   equ     $F800           ;
000000FF stack  equ     $FF             ; top of the stack
                variables:
00000000        equ     $0              ; variables storage
0000FFD6 EEVecTab equ   $FFD6           ; Interrupt vector table
                ;
                ;*****************************************************************
                ; Program Constants
                ;*****************************************************************
                ;
                ;
                ; * Masks for Output compare registers
                ;       Please refer to HC11 user manual for details;
                ; OC2 - Output Compare 2 used as a real time clock
                ;
00000040 SET_OC2 EQU    $40             ; Setup OC2 interrupt for TFLG1
00000040 ACK_OC2 EQU    $40             ; Acknowledge OC2 interrupt for TFLG1
00000040 FOR_OC2 EQU    $40             ; Force, for CFORC
```

```
                ALL_OC2  EQU     $C0             ; for TCTL1
                TOG_OC2  EQU     $40             ; for TCTL1
                CLR_OC2  EQU     $80             ; for TCTL1
00000000        PUL_OC2  EQU     $C0             ; for TCTL1
                ;
00000020        SET_OC3  EQU     $20             ; Setup OC3 Interrupt
00000020        ACK_OC3  EQU     $20             ; Acknowledge OC3 Interrupt
00000020        FOR_OC3  EQU     $20             ;
00000030        ALL_OC3  EQU     $30             ;
00000010        TOG_OC3  EQU     $10             ;
00000020        CLR_OC3  EQU     $20             ;
00000030        PUL_OC3  EQU     $30             ;
                ;
00000010        SET_OC4  EQU     $10             ; Setup OC4 Interrupt
00000010        ACK_OC4  EQU     $10             ; Acknowledge OC4 Interrupt
00000010        FOR_OC4  EQU     $10             ;
0000000C        ALL_OC4  EQU     $0C             ;
00000004        TOG_OC4  EQU     $04             ;
00000008        CLR_OC4  EQU     $08             ;
0000000C        PUL_OC4  EQU     $0C             ;
                ;
00000008        SET_OC5  EQU     $08             ; Setup OC5 Interrupt
00000008        ACK_OC5  EQU     $08             ; Acknowledge OC5 Interrupt
00000008        FOR_OC5  EQU     $08             ;
00000003        ALL_OC5  EQU     $03             ;
00000001        TOG_OC5  EQU     $01             ;
00000002        CLR_OC5  EQU     $02             ;
00000003        PUL_OC5  EQU     $03             ;
                ;
0000101A        TOC_DCDC EQU     TOC3            ;DC-DC CONVERTER PULSES
00000020        SET_DCDC EQU     SET_OC3         ;
00000020        ACK_DCDC EQU     ACK_OC3         ;
00000020        FOR_DCDC EQU     FOR_OC3         ;
00000030        ALL_DCDC EQU     ALL_OC3         ;
00000010        TOG_DCDC EQU     TOG_OC3         ;
00000020        CLR_DCDC EQU     CLR_OC3         ;
00000030        PUL_DCDC EQU     PUL_OC3         ;
                ;
0000101C        TOC_PULS EQU     TOC4            ;STIMULUS PULSES
00000010        SET_PULS EQU     SET_OC4         ;
00000010        ACK_PULS EQU     ACK_OC4         ;
00000010        FOR_PULS EQU     FOR_OC4         ;
0000000C        ALL_PULS EQU     ALL_OC4         ;
00000004        TOG_PULS EQU     TOG_OC4         ;
00000008        CLR_PULS EQU     CLR_OC4         ;
0000000C        PUL_PULS EQU     PUL_OC4         ;
                ;
0000101E        TOC_BFET EQU     TOC5            ;BOSFET SWITCH PULSES
00000008        SET_BFET EQU     SET_OC5         ;
00000008        ACK_BFET EQU     ACK_OC5         ;
00000008        FOR_BFET EQU     FOR_OC5         ;
00000003        ALL_BFET EQU     ALL_OC5         ;
00000001        TOG_BFET EQU     TOG_OC5         ;
00000002        CLR_BFET EQU     CLR_OC5         ;
00000003        PUL_BFET EQU     PUL_OC5         ;
                ;
00000080        FOC1     EQU     $80             ;
                ;
                ;* INPUT CAPTURE FLAGS           ;
                ;                                ;
00000004        ACK_IC1  EQU     $04             ;
00000002        ACK_IC2  EQU     $02             ;
00000001        ACK_IC3  EQU     $01             ;
                ;
FFFFFFFF                 if      crystal60000    ;
00004E20        ICI_CNT  EQU     20000           ; 10 ms time constant (100Hz)
                                                   E/1 0.5uS per interrupt)
00000096        IPI_DLY  EQU     150             ; delta t, BFET->pulse period
                endif                            ;
                ;
                ;* Transmission Characters
                ;
00000004        ETX      EQU     $04             ; End of Transmission
00000011        XON      EQU     $11             ; Continue Transmission
00000013        XOFF     EQU     $13             ; Stop Transmission
0000001B        ESC      EQU     $1B             ; Escape Character
0000000D        CR       EQU     $0D             ; Carriage Return
0000000A        LF       EQU     $0A             ; Line Feed
                ;
                ;
                ;*****************************************************************
                ;*
                ;* ANALOG INPUTS
                ;*
                ;*****************************************************************
                ;
00000001        EBUT     EQU     %00000001       ; ADCTL values
00000002        BATTV    EQU     %00000010       ;
00000000        Haddr    EQU     %00000000       ;
                ;
                ; * values for sensing analog buttons A and B
                ;
000000E0        NAB      EQU     $e0             ; Neither A nor B
000000A0        YB       EQU     $a0             ; B pressed
00000060        YA       EQU     $60             ; A pressed
                                                 ; below -> A & B pressed

                ;************************************
                ;* RAM SECTION OF THE PROGRAM
                ;************************************
0000         D          section data,variables  ; start of variable definitions
0000                    org     variables
0000            RAMSTART:
                ;
                ;*****************************************************************
                ;*
                ;* Jamnan's 32 bit IIR accumulator (for 16x16 bit multiplications)
                ;*
                ;*****************************************************************
                ;
0000 0001       ovbit    ds      1               ; indicates if overflow occurs
0001 0001       acc0     ds      1               ; Jamnan's accumulator
0002 0001       acc1     ds      1               ;
0003 0001       acc2     ds      1               ;
0004 0001       acc3     ds      1               ;
                ;
                ;*****************************************************************
                ; Variables
                ;*****************************************************************
                ;
                ;
0005 0002       chstatus ds      2               ; status for all two channels
```

```
                                                ; -1 off (zero output)
                                                ; 0 base (threshold stimulation)
                                                ; 1 base+stimulus
0007            pulsewidth:
     0002               ds      2               ; Pulse Width
0009 0002       DCDCmod  ds     2               ;
                ;*****************************************************************
                ;
                ;********************* SYSTEM INFORMATION *********************
                ;
                ;*****************************************************************
                ;
                ; Explanation of pending modes:
                ;     The user is allowed a short period of time to decide which
                ;     mode to switch to. The user can press the mode button
                ;     multiple times until the desired mode is reached.
                ;     After a short period, this pending mode will then
                ;     become the operating mode.
                ;
                ; * Debug -  These modes are normally not visible to user
                ;
000B            Debugpend:
     0001               ds      1               ; Pending mode for special functions
                ;
                d_amplitude:
     00000080           EQU     %10000000       ; 1 = use amplitude modulation
                ;                               ; 0 = use pulse width modulation
                d_nonlinear:
     00000040           EQU     %01000000       ; 1 = conditional filter off
     00000010   d_25Hz  EQU     %00010000       ; 1 = stimulation at 25Hz
                ;                               ; 0 = stimulation at 33Hz
                d_invalidmode:
     0000000F           EQU     %00001111       ;
                d_validmode:
     000000F0           EQU     %11110000       ;
                ;
000C            Debugflag:
     0001               ds      1               ; indicate debug mode is invoked
                ;
                d_changemode:
     00000001           EQU     %00000001       ; tell process_key not to toggle stim
     00000002   d_ABkey EQU     %00000010       ; using A/B buttons
     00000004   d_reset EQU     %00000100       ;
                ;                               ;
                ; * MODE                        ;
                ;                               ;
000D 0001       Mode     ds     1               ; Mode of Operation system use only
                ;                               ; bit definition- see below
                ;                               ;
                ; * Pending                     ;
                ;                               ;
000E 0001       Pending  ds     1               ; pending mode user still selecting
                ;                               ; bit definition - see below
                ;                               ;
                ; * LED                         ; Lights on panel
                ;                               ;
000F 0001       LED      ds     1               ; LED, PORTD on 68HC11
                ;                               ;
     00000080   normal   equ    %10000000       ; Normal operation LED
     00000080   monitor  equ    %10000000       ; Monitor Mode LED
                                                ; monitor - for filter analysis only

     00000040   ChObase  equ    %01000000       ; Channel 0 Theshold Mode/LED
     00000020   Ch1base  equ    %00100000       ; Channel 1 Theshold Mode/LED
     00000010   ChOgain  equ    %00010000       ; Channel 0 Gain Adjustment/LED
     00000008   Ch1gain  equ    %00001000       ; Channel 1 Gain Adjustment/LED
     00000004   Sound    equ    %00000100       ; Sound On/Off Mode/LED
     00000002   StimLED  equ    %00000010       ; Stim On/Off Mode/LED
     00000001   PowerLED equ    %00000001       ; Power ON LED
                ;                               ;
     00000078   ABadjust equ    %01111000       ;

     000000FC   full_LED equ    %11111100       ;
     00000007   clrmode  equ    %00000111       ;
                invalidmode:
     00000003           equ     %00000011       ;
     000000F8   allmode  equ    %11111000       ;
                ;                               ;
                ;                               ;
                ; DIGITAL OUTPUTS               ; Used for Multiplexing & Power Switch
                ; PORTB                         ;
                ;                               ;
0010 0001       digital  ds     1               ;
     00000080   ChOSw    equ    %10000000       ; turn on channel
     00000040   Ch1Sw    equ    %01000000       ;
     00000008   Clicker  equ    %00001000       ; turn on clicker
     00000001   PwrSw    equ    %00000001       ; turn off power
                ;                               ;
     000000C8   AllSw    equ    %11001000       ;
                ;                               ;
                ; * BTN                         ;
                ;                               ;
0011 0001       Btn      ds     1               ; BTN - BUTTONS (MODE & STIM)
                ;                               ;
     00000004   Stim_B   equ    %00000100       ;  IC1 - STIM BUTTON
     00000002   Mode_B   equ    %00000010       ;  IC2 - MODE BUTTON
     00000001   Pwr_B    equ    %00000001       ;  IC3 - POWER BUTTON
     00000080   A_But    equ    %10000000       ; A button
     00000040   B_But    equ    %01000000       ; B button
                ;                               ;
                ; * OC1_M                       ;
                ;                               ;
0012 0001       OC1_M    ds     1               ; shadow register for oc1m
                ;                               ;
     00000080   OC1D7    equ    %10000000       ; TOC1 PIN
     00000040   OC1D6    equ    %01000000       ; TOC2 PIN
     00000080   LVDT_b   equ    %10000000       ; TOC1 PIN - LVDT POWER PIN
     00000020   DCDC_b   equ    %00100000       ; TOC3 PIN - DCDC POWER PIN
     00000010   PULS_b   equ    %00010000       ; TOC4 PIN - STIMULUS PULSE BIT
     00000008   BFET_b   equ    %00001000       ; TOC5 PIN - BOSFET ENABLE
     000000F8   OC1T05   equ    %11111000       ; All OC pins
                ;                               ;
                ; * StimWait                    ; period before stimulation starts
                ;                               ;
0013 0001       StimWait ds     1               ; turn off stimulations
0014 0002       StimCh   ds     2               ; used to tell current channel to stimulate
                totalchannel:
     00000003           equ     3               ; stimulate every 3 samples (100 3=33Hz
                ;                               ;
                ; * flash                       ;
                ;                               ;
0016 0001       FLASH    ds     1               ; To conserve power used for LED
                pendingflash:
     00000003           equ     3               ; fast flash 3x every n TOC2 period
```

```
                normalflash:
      0000000F          equ     15              ; slow flash
                ;                               ;
                ; * oldbtn                      ;
                ;                               ;
0017 0001       oldbtn  ds      1               ; remember old buttons for repeat
                ;                               ;
                ; * repeat                      ;
                ;                               ;
0018 0001       repeat  ds      1               ; for use with A & B buttons
                                                ; rate of repeat
                ;                               ;
                ; * TCT1                        ;
0019 0001       TCT1    ds      1               ; shadow register for tctl1
                ;                               ;
                ; * Wait                        ;
                ;                               ;
001A 0001       Wait    ds      1               ; use for pending mode
                                                ; time to wait before pending becomes
                                                ; current mode
                waitperiod:
      00000064          EQU     100             ; wait for one second
                ;                               ;
                ;                               ;
      00000000          if      analysismode    ;
                        endif                   ;

001B 0002       Stimulus ds     2               ; 16-bit resolution result
001D 0002       S256    ds      2               ; scaled stimulus (normally within 256)
001F 0002       S1024   ds      2               ; scaled stimulus (normally with 1K)
0021 0001       Bs      ds      1               ; store EBUT reading
0022 0002       Hs      ds      2               ; store Handaddr reading
                                                ;
0024            tx_start:                       ; Copied from EEPROM
0024            rx_start:                       ;
                                                ;
0024 0002       Base    ds      2               ; Base for chan 0 and 1
0026 0002       Gain    ds      2               ; Gain for chan 0 and 1
0028            default_amp:
     0002               ds      2               ; default amplitude for 0 & 1
002A            default_pulse:
     0002               ds      2               ; default pulsewidth for 0 & 1
002C 0001       Debug   ds      1               ; allow store debug mode
                                                ;
002D            tx_end:                         ;
002D 0002       rx_end: ds      2               ;
                                                ;
      00000009  Rx_Count equ    rx_end-rx_start ;
                ;
                ; The followings are used for the non-linear conditional filter
                ;
                ;
002F 0002       xindex  ds      2               ;
0031 0001       xmin    ds      1               ;
0032 0001       xmax    ds      1               ;
0033 0001       xrange  ds      1               ;
      00000020  xbufsize EQU    32              ;
0034            xbuffer:                        ;
0034 0020               ds      xbufsize        ;
0054            rd_buff                         ;
0054            IIRbuffer:                      ; Jamnan's IIR buffer
0054            targetAddr:                     ; For EEprom programming
```

```
0054            _program:                                ; RAM code starting point
0054 0026               ds      36                       ; just enough for 6 sections
                                                         ; 3 words each 6 sections 1 output
                                                         ; (3x6+1)*2
007A 0001       ram_end: ds     1                        ;
                ;* * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * *
* * * * *

                ;
                ;**********************************************************************
                ; Main Program
                ;**********************************************************************
                ;
F800         P          section program,prog
F800                    org     prog
                ;
                ;**********************************************************************
                ; Initialize System
                ;**********************************************************************
                ;
F800 04         main:   db      ETX                      ; This instruction simply gives an etx
                                                         ; character in case the screen data is
                                                         ; corrupted The opcode is for LSRD
                                                         ;
                                                         ; The following two registers must be
                                                         ; written to within the first 64 clock cycles
                                                         ;
F801 0F                 sei                              ; mask all maskable interrupts
F802 8E00FF             lds     #stack                   ; stack pointer
F805 8693               ldaa    #$93                     ; adpu, dly, irqe, cop
F807 B71039             staa    OPTION                   ;* Option Register
                                                         ;
F80A 8617               ldaa    #%00010111               ; CONFIG PROTECTED,
F80C B71035             staa    BPROT                    ;* Block Protect Register allowing the
                                                         ; EEPROM to be written to in operation
                                                         ;      BPROT
                                                         ;  b0 - x800->x9FF
                                                         ;  b1 - xA00->xBFF
                                                         ;  b2 - xC00->xDFF
                                                         ;  b3 - xE00->xFFF
                                                         ;  b4 - CONFIG REGISTER
                                                         ;
     FFFFFFFF           if      crystal80000             ;
F80F 8600               ldaa    #$00                     ; E/1 0.500 uS per timer tick
                        endif                            ;
     00000000           if      crystal49152             ;
                        endif                            ;
     00000000           if      crystal36864             ;
                        endif                            ;
F811 B71024             staa    TMSK2                    ; timer prescaler
                                                         ;
F814 86FF               LDAA    #$FF                     ; This maintains the battery power
F816 B71004             STAA    portb                    ;
                                                         ;
                ;
                ;**********************************************************************
                ; Copy Vector Table to RAM for buffalo monitor interrupt vectors
                ;**********************************************************************
                ;
     000000C4   SCI_V   equ     $C4                      ; Serial Port Interrupt
     000000C7   SPI_V   equ     $C7                      ; Serial Peripheral Interface Interrupt
     000000CA   PAIE_V  equ     $CA                      ; Pulse Accumulator Input Edge Interrupt
     000000CD   PAO_V   equ     $CD                      ; Pulse Accumulator Overflow Interrupt
```

```
000000D0  TO_V     equ     $D0                 ; Timer Overflow Interrupt
000000D3  TOC5_V   equ     $D3                 ; Timer Output Compare 5 Interrupt
000000D6  TOC4_V   equ     $D6                 ; Timer Output Compare 4 Interrupt
000000D9  TOC3_V   equ     $D9                 ; Timer Output Compare 3 Interrupt
000000DC  TOC2_V   equ     $DC                 ; Timer Output Compare 2 Interrupt
000000DF  TOC1_V   equ     $DF                 ; Timer Output Compare 1 Interrupt
000000E2  TIC3_V   equ     $E2                 ; Timer Input Capture 3 Interrupt
000000E5  TIC2_V   equ     $E5                 ; Timer Input Capture 2 Interrupt
000000E8  TIC1_V   equ     $E8                 ; Timer Input Capture 1 Interrupt
000000EB  RTI_V    equ     $EB
          ;
          ; Real Time Interrupt
          ;
000000EE  IRQ_V    equ     $EE                 ; Maskable Interrupt
000000F1  XIRQ_V   equ     $F1                 ; Non-Maskable Interrupt
000000F4  SWI_V    equ     $F4                 ; Software Interrupt
000000F7  ILL_V    equ     $F7                 ; Illegal Opcode Interrupt
000000FA  COP_V    equ     $FA                 ; Computer Operating Properly Watchdog Int
000000FD  CLK_V    equ     $FD                 ; Clock Monitor Interrupt
          ;*****************************************************************
          ; Clear RAM
          *****************************************************************
          ;
F819      init_variables.                     ;
F819 CE0000       ldx     #RAMSTART           ;
F81C      clr_ram:                            ;
F81C 6F00         clr     0,x                 ;
F81E 08           inx                         ;
F81F 8C007A       cpx     #ram_end            ;
F822 23F8         bls     clr_ram             ;
          ;*****************************************************************
          ; Recover Non-Volatile Data
          *****************************************************************
          ;
                                              ;
F824 18CEFE6E     ldy     #ex_start           ; y=EX_Start ( Start of EEPROM data )
F828 CE0024       ldx     #tx_start           ; x=TX_Start ( Start of TxData data )
F82B      update:                             ; Repeat
F82B 18A600       ldaa    0,y                 ;  A:=(y);
F82E A700         staa    0,x                 ;  (x):=A;
F830 1808         iny                         ;  x:=x+1;
F832 08           inx                         ;  y:=y+1;
F833 8C002D       cpx     #tx_end             ; Until x=tx_end;
F836 25F3         blo     update              ;
          ;*****************************************************************
          ; Other initializations
          *****************************************************************
          ;
F838 80F8AF       JSR     INIT_SPI            ; * Initialize the SPI
F83B      OnSCI:                              ; * Initialize serial communications
          FFFFFFFF  if      crystal80000      ;
F83B 8630         ldaa    #$30                ;
                  endif                       ;
          00000000  if      crystal49152      ;
                  endif                       ;
          00000000  if      crystal38864      ;
                  endif                       ;
```

```
F83D B7102B       staa    BAUD                ; set for baud 9600
F840 8600         ldaa    #$00                ;
F842 B7102C       staa    SCCR1               ; clear word length bit
F845 8608         ldaa    #$08                ;
F847 B7102D       staa    SCCR2               ; enable transmitter
                  ;                           ;
                  ;* SETUP PORTC AS LED OUTPUT PORT
                  ;                           ;
F84A 86FF         LDAA    #$FF                ;
F84C B71003       STAA    PORTC               ;
F84F B71007       STAA    DDRC                ; CONFIGURE PORTC FOR ALL OUTPUTS
                                              ;
F852 8601         ldaa    #powerLED           ; turn on power light
F854 43           coma                        ;
F855 B71003       staa    portc               ;
F858      Power_Loop:
F858 B61000       ldaa    porta               ;
F85B 8501         bita    #1                  ;
F85D 26F9         bne     Power_Loop          ; Wait for Power Button to be released
                                              ;
                  ;                           ;
                  ;* Initialize IC1, IC2 & IC3 ;
                  ;                           ;
F85F 8615         LDAA    #%00010101          ; CAPTURE RISING EDGE OF IC1, IC2 IC3
F861 B71021       STAA    TCTL2               ;
F864 8607         LDAA    #ACK_IC1+ACK_IC2+ACK_IC3
F866 B71023       STAA    TFLG1               ;
                  ;                           ;
                  ;* Initialize TOC2 Interrupt ;
                  ;                           ;
F869 FC100E       LDD     TCNT                ; Get Current Timer value
F86C FD1018       STD     TOC2                ; Store Value in TOC2 Register
F86F 8640         LDAA    #SET_OC2            ; Value to cause interrupt on OC2
F871 B71023       STAA    TFLG1               ; Clear any previous interrupt on OC2
F874 B71022       STAA    TMSK1               ; Allow interrupt on OC2
F877 4F           CLRA                        ;
F878 B71020       STAA    TCTL1               ; disconnect timer from output pins
F87B 86F8         ldaa    #OC1T05             ;
F87D B7100D       staa    OC1D                ; set pins to high when OC1 occurs
                                              ;
F880 8680         ldaa    #LVDT_b             ; set OC1 to control LVDT
F882 9712         staa    OC1_M               ;
F884 B7100C       staa    OC1M                ;
F887 B7100B       staa    CFORC               ; force OC1 action turn LVDT on
                  ;                           ;
                  ; Initialize variables      ;
                  ;                           ;
F88A 4F           clra                        ;
F88B 5F           clrb                        ;
F88C 9711         staa    Btn                 ; clear button
F88E DD2F         std     xindex              ;
F890 971A         staa    Wait                ;
F892 970C         staa    Debugflag           ;
                  ;                           ;
                  ; setup starting mode       ;
                  ;                           ;
          00000000  if      analysismode      ;
                  else                        ;
F894 8680         ldaa    #normal             ;
                  endif                       ;
                                              ; turn on Power Light & operating mode light
F896 970E         staa    Pending             ;
```

```
F898 8A04            oraa    #sound            ;
F89A 970D            staa    Mode              ; set the system to sound on
F89C 970F            staa    LED               ;
                                               ;
F89E 860F            ldaa    #normalflash      ;
F8A0 9718            staa    FLASH             ; set up flash period
                                               ;
F8A2 8664            ldaa    #100              ; disable the stimulator for the first second
                                               ; for safety purpose
F8A4 9713            staa    StimWait          ;
F8A6 CC0003          ldd     #totalchannel     ;
F8A9 DD14            std     StimCh            ;
     00000000        if      analysismode      ;
                     endif                     ;
                                               ;
F8AB 0E              cli                       ; get ready for interrupt
;-------------------------------------------------------------------------
; Program Stuck here. Wait for interrupts

;vvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvv
                ;                              ;
F8AC            WAI_LP:                              ;
F8AC 3E                 WAI                          ; Infinite wait loop. The program waits here
F8AD 20FD               bra     WAI_LP               ;
                ;                              ;
                ;****************************************************************
                ; Subroutine definitions
                ;****************************************************************
                ;
                ;
                ;
                ;--------------------------------------------------------------------
                ; Include external Subroutines
                ;
                        include SPI.s
                ;----------------------------
                ; Initialize controller SPI port D.
                ;
F8AF            INIT_SPI:
F8AF 862F               LDAA    #$2F            ;-,-,1,0;1,1,1,1
                ;
                ;SS*-Hi, SCK-Lo, MOSI-Hi
F8B1 B71008             STAA    PORTD
                ;So SS stays high when DDR5 set
                ;       --------------------;
F8B4 863A               LDAA    #$3A            ;-,-,1,1;1,0,1,0
F8B6 B71009             STAA    DDRD
                ;SS*, SCK, MOSI - Outs
                ;MISO, TxD, RxD - Ins
                ;       --------------------
                ;DDRD5=1 so SS* pin is a general purpose output
                ;
                ;  7     6    5    4    3    2    1    0
                ; SPIE  SPE DWOM MSTR CPOL CPHA SPR1 SPR0
                ;  0     1    0    1    0    1    1    0
F8B9 8656               LDAA    #%01010110
F8BB B71028             STAA    SPCR
                ;SPI on as Master,CPHA=1,CPOL=0,E/16 Clk rate
F8BE 39                 RTS

                ; ------------------- S P I  BUFFER ---------------------------
                ;
```

```
                      ; Purpose:   D contains the two bytes to be transmitted to the D.A
                      ;            A is the Tone generator value and B is the intensity value
                      ;
                      ; Pre:       Valid display data values in RAM.
                      ; Post:      LED display now shows the buffer contents.
                      ; Registers: A,B,X,Y (all).
                      ;
                      ;
      00000020 SS       EQU    $20                     ;SLAVE SELECT BIT
      00000080 SPIF     EQU    $80                     ;SPI FLAG - Set When Conversion Complete
      00000008 PORTD_   EQU    $08                     ;
      0000002A SPDR_    EQU    $2A                     ;
      00000029 SPSR_    EQU    $29                     ;
               ;                                       ;
F8BF           D_A:                                    ;
F8BF 3C                 PSHX                           ;
F8C0 CE1000             LDX    #$1000                  ;Point at on-chip registers
F8C3 1D0820             BCLR   PORTD_,X,SS             ;Drive SS* low, enable display.
F8C6 04                 LSRD                           ;
F8C7 04                 LSRD                           ;
F8C8 6D29               TST    SPSR_,X                 ;This performs a read operation of the status
F8CA A72A               STAA   SPDR_,X                 ;Write SPI data reg to start xfer
F8CC 1F2980FC SPI_LP1:  BRCLR  SPSR_,X,SPIF,SPI_LP1;Loop to wait for SPIF
F8D0 E72A               STAB   SPDR_,X                 ;(when SPIF set, SPSR neg)
F8D2 1F2980FC SPI_LP2:  BRCLR  SPSR_,X,SPIF,SPI_LP2;Loop to wait for SPIF
F8D6 1C0820             BSET   PORTD_,X,SS             ;Drive SS* high
F8D9 38                 PULX                           ;
F8DA 39                 RTS                            ;Return to main pgm.

                        include tre1sub.s

               ;*************************************
               ;*
               ;* The following is code to be copied into ram for
               ;* EEPROM Programming
               ;*
               ;*************************************
      FFFFFFFF          if     saveparam
F8DB           SourceAddr:                             ;
F8DB           program:                                ;
F8DB 37                 pshb                           ;
F8DC E600               ldab   0,x                     ;
F8DE 53                 comb                           ;
F8DF 2714               beq    prgm
F8E1 C616               ldab   #$16                    ;
F8E3 F7103B             stab   pprog                   ;
F8E6 E700               stab   0,x                     ;
F8E8 C617               ldab   #$17                    ;
F8EA 2601               bne    erase1                  ;
F8EC 5F                 clrb                           ;
F8ED F7103B    erase1:  stab   pprog                   ;
F8F0 8D1D               bsr    dly10                   ;
F8F2 7F103B             clr    pprog                   ;
F8F5 16        prgm:    tab                            ;
F8F6 53                 comb                           ;
F8F7 2714               beq    prgm_done               ;
F8F9 C602               ldab   #$02                    ;
F8FB F7103B             stab   pprog                   ;
F8FE A700               staa   0,x                     ;
F900 C603               ldab   #$03                    ;
F902 2601               bne    prgm1                   ;
```

```
F904 5F                 clrb
F905 F7103B     prgm1:  stab    pprog
F908 8D05               bsr     dly10
F90A 7F103B             clr     pprog
F90D            prgm_done:
F90D 33                 pulb
F90E 39                 rts

F90F            dly10:
F90F 3C                 pshx
     FFFFFFFF           if      crystal80000
F910 CE0005             ldx     #3333
                        endif
     00000000           if      crystal49152
                        endif
F913 09         wait10  dex
F914 26FD               bne     wait10
F916 38                 pulx
F917 39                 rts


F918            End_Code:


                ;**************************************

F918            chgbyte:
F918 36                 psha
F919 A100               cmpa    0,x             ; If Data is the Same
F91B 2704               beq     chgbyt3         ; then skip programming
F91D 0F                 sei
F91E 9D54               jsr     _program
F920 0E                 cli
F921            chgbyt3:
F921 32                 pula
F922 39                 rts

F923            Copy_Params:
F923 3C                 pshx
F924 183C               pshy
F926 36                 psha
F927 CEFE6E             ldx     #ex_start
F92A 18CE0024           ldy     #rx_start
F92E            copy_loop:
F92E 18A600             ldaa    0,y
F931 8DE5               bsr     chgbyte
F933 08                 inx
F934 1808               iny
F936 8CFE77             cpx     #ex_end
F939 25F3               blo     copy_loop
F93B 32                 pula
F93C 1838               puly
F93E 38                 pulx
F93F 39                 rts
                        endif

                ;-------------------------------------------------------
                ;
                ; Stim + power button combination
                ; special key sequence to set all thresholds/gains to zero
                ;
F940            clr_parameter:
```

```
F940 5F                 clrb
F941 4F                 clra
F942 DD24               std     base
F944 DD26               std     gain
F946 140C04             bset    DebugFlag,d_reset
F949 39                 rts
                ;-------------------------------------------------------
                ;
                ; Process any key pressed
                ;
F94A            Process_key:
F94A 12110289           brset   Btn,mode_b,change_mode          ; Mode button toggled
F94E 1211047C           brset   Btn,stim_b,toggle_stim          ; Stim button toggled
F952 121101EA           brset   Btn,pwr_b,clr_parameter         ; Stim+Power
F956 130D785C           brclr   Mode,ABadjust,end_process_key   ; if not adjustment modes, ret
F95A 1311C058           brclr   Btn,A_but+B_but,end_process_key ; if not A or B, then go back
                ;-------------------------------------------------------
                ;
                ; A or B button is pressed
                ; it is either of the four possible modes
                ;
F95E CE0000             ldx     #0                      ; first assume changing threshold levels
                                                        ; x is used as the index into the variable
                                                        ; to change. The offset is ch0base
F961 B61000             ldaa    porta                   ;
F964 8504               bita    #Stim_b                 ; if stim button is not held
F966 2614               bne     change_level            ; goto normal change level routine
                ;
                ; if in debug change mode, quit
                ;
F968 120C014A           brset   Debugflag,d_changemode,end_process_key
                ;
                ; This is the special change level
                ; either change the default amplitude or the default pulsewidth:
                ; 1) In amplitude mode, change default pulsewidth
                ; 2) In pulsewidth mode, change default amplitude
                ;
                ; The next line makes sure when stim button is released the stimulation mode
                ; will not be toggled
                ;
F96C 140C02             bset    Debugflag,d_ABkey
                ;
                ; valid only in this two modes: ch0 & ch1 base modes
                ;
F96F 130D8043           brclr   Mode,ch0base+ch1base,end_process_key
                ;
                ; Now determine which variable to change
                ; first assume default_amp
                ;
F973 CE0004             ldx     #(default_amp-Base)
                ;
                ; if stimulation mode is amplitude modulation
                ; use default_pulse
                ;
F976 132C8002           brclr   Debug,d_amplitude,change_level
F97A 08                 inx                             ;
F97B 08                 inx                             ; default_pulse = default_amp + 2
                ;-------------------------------------------------------
                ;
                ; Change the level of the variable pointed by x, according to
                ; A or B button
                ; A and B have auto-repeating
```

```
                ;
F97C            change_level:                   ;
F97C 9611               ldaa    Btn             ;
F97E 16                 tab                     ;
F97F 9817               eora    oldBtn          ;
F981 D717               stab    oldBtn          ; save Btn to oldBtn
F983 9411               anda    Btn             ; find changed button
F985 C60A               ldab    #10             ; wait time for first A,B pressed
F987 85C0               bita    #A_but+B_but    ; any changed key?
F989 2607               bne     next1           ; new key pressed
F98B 7A0018             dec     repeat          ; time to auto repeat key?
F98E 2626               bne     end_process_key ;
F990 C603               ldab    #3              ; used as auto repeat period
F992            next1:                          ;
F992 D718               stab    repeat          ;
                ;
                ; The following found the mode number
                ; ch0base = 0
                ; ch1base = 1
                ; ch0gain = 2
                ; ch1gain = 3
                ;
F994 960D               ldaa    Mode            ;
F996 48                 lsla                    ;
F997 2B09               bmi     found_index     ;
F999 08                 inx                     ;
F99A 48                 lsla                    ;
F99B 2B05               bmi     found_index     ;
F99D 08                 inx                     ;
F99E 48                 lsla                    ;
F99F 2B01               bmi     found_index     ;
F9A1 08                 inx                     ;
F9A2            found_index:                    ;
F9A2 E624               ldab    Base,x          ;
                ;
                ; Increase or decrease?
                ; A = decrease
                ; B = increase
                ;
F9A4 12114006           brset   Btn,B_but,increase
F9A8            decrease:                       ; decrease level, x points to the variable
F9A8 5D                 tstb                    ; already zero?
                                                ; do not decrease if already 0
F9A9 2707               beq     call_display_scale
F9AB 5A                 decb                    ;
F9AC 2004               bra     call_display_scale
F9AE            increase:                       ;
F9AE 5C                 incb                    ;
                                                ; wrap around?
F9AF 2601               bne     call_display_scale
F9B1 5A                 decb                    ; yes, make it stay at $FF
F9B2            call_display_scale:             ;
F9B2 E724               stab    Base,x          ; store the value first
F9B4 8D49               bsr     display_scale   ; show aproximate value on LEDs
F9B6            end_process_key:                ;
F9B6 39                 rts                     ;
                ;                               ;
                ;-----------------------------------------------
                ;
                ; change operation mode (or debug mode)
                ;
F9B7            change_mode:                    ;
                ;
                ; check for special debug mode change sequence
                ; Stim pressed + mode button
F9B7 B61000             ldaa    porta           ;
F9BA 8504               bita    #Stim_b         ;
F9BC 2720               beq     change_debug    ; if stimulation button is held
                                                ; change debug mode.
F9BE 8664               ldaa    #waitperiod     ; setup time for a mode to be pending
F9C0 971A               staa    Wait            ;
F9C2 74000E             lsr     Pending         ; shift pending mode
F9C5 130E0304           brclr   Pending,invalidmode,end_change_mode
                ;                               ;
                ; last mode is reached. Wrap back to 1st mode
                ;                               ;
     00000000           if      analysismode    ;
                        else                    ;
F9C9 8680               ldaa    #normal         ; 1st mode
                        endif                   ;
                                                ;
F9CB 970E               staa    Pending         ; save mode
F9CD            end_change_mode:
F9CD 39                 rts
                ;-----------------------------------------------
                ;
                ; take action when stimulate button is released.
                ;
F9CE            toggle_stim:
     00000000           if      analysismode
                        endif
F9CE 960C               ldaa    Debugflag       ; if in any of the debug mode
F9D0 2606               bne     quit_debug      ; ignore button release
                                                ;
F9D2 960D               ldaa    Mode            ;
F9D4 8802               eora    #StimLED        ; toggle stimulation
F9D6 970D               staa    Mode            ;
F9D8            quit_debug:                     ;
F9D8 4F                 clra                    ;
F9D9 970C               staa    Debugflag       ;
F9DB 971A               staa    Wait            ; stop Pending
F9DD 39                 rts                     ;
                ;-----------------------------------------------
                ;
                ; change the debug mode. Called from change_mode
                ;
F9DE            change_debug:
                ;
                ; debug mode
                ; do not enter changemode if in some other modes
                ;
                ; second time in change debug, shift to next mode
                ;
F9DE 120C010B           brset   Debugflag,d_changemode,shift_debug
                ;
                ; in some other modes already, so quit
                ;
F9E2 120C0208           brset   Debugflag,d_ABkey,end_change_debug
                ;
                ; make sure other parts of the program knows that
                ; change_debug is active
                ;
```

```
F9E6 140C01             bset    Debugflag,d_changemode
F9E9 7F000B             clr     DebugPend
F9EC                end_change_debug:
F9EC 39                 rts
                    ;-----------------------------------------------------------
                    ;
                    ; shift debug mode to the next one
                    ;
F9ED                shift_debug:                    ;
F9ED 8664               ldaa    #waitperiod         ;
F9EF 971A               staa    Wait                ; setup pending time
F9F1 960B               ldaa    DebugPend           ;
F9F3 2705               beq     top_debug           ; if pending mode is zero, use 0x80
F9F5 44                 lsra                        ; otherwise, shift right
F9F6 84F0               anda    #d_validmode        ; check to see if mode is valid
F9F8 2002               bra     end_shift_debug     ; yes, then store and quit
F9FA                top_debug:                      ;
F9FA 8680               ldaa    #$80                ; use 0x80 as first debug mode
F9FC                end_shift_debug                 ;
F9FC 970B               staa    DebugPend           ; store new pending mode
F9FE 39                 rts                         ;
                                                    ;
     00000000           if      analysismode        ;
                        endif
                    ;-----------------------------------------------------------
                    ;
                    ;       display_scale
                    ;
                    ; This display the current level graphically on the mode LED
                    ; a is the scale value, b = LED
                    ;
F9FF                display_scale:
F9FF 36                 psha                        ; store a somewhere
FA00 86FC               ldaa    #full_LED           ; use all LEDs
FA02 C1FF               cmpb    #$ff                ; top value?
                                                    ; yes, then use all LEDs
FA04 271F               beq     display_scale_end
FA06 8680               ldaa    #$80                ; use top LED
FA08 C1D0               cmpb    #$d0                ; value > 0xd0? yes, use top
FA0A 2219               bhi     display_scale_end
FA0C 44                 lsra                        ; etc...
FA0D C1A0               cmpb    #$a0                ;
FA0F 2214               bhi     display_scale_end
FA11 44                 lsra                        ;
FA12 C180               cmpb    #$80                ;
FA14 220F               bhi     display_scale_end
FA16 44                 lsra                        ;
FA17 C160               cmpb    #$60                ;
FA19 220A               bhi     display_scale_end
FA1B 44                 lsra                        ;
FA1C C130               cmpb    #$30                ;
FA1E 2205               bhi     display_scale_end
FA20 44                 lsra                        ;
FA21 5D                 tstb                        ;
FA22 2601               bne     display_scale_end
FA24 4F                 clra                        ; turn off all LEDs if value = 0
FA25                display_scale_end:
FA25 970F               staa    LED                 ;
FA27 8601               ldaa    #1                  ;
FA29 9716               staa    flash               ; do not flash graphical scale
FA2B 32                 pula                        ; restore a
FA2C 39                 rts                         ;
```

```
                    ;-----------------------------------------------------------
                    ;                               ;
FA2D                jmp_Quit_pulse:                 ; for relative calls that are too long
FA2D 7EFAE6             jmp     Quit_pulse          ;
                    ;                               ;
                    ;                               ;
                    ;-----------------------------------------------------------
                    ;                               ;
                    ; setup the stimulation         ;
                    ; x = channel                   ;
                    ;                               ;
FA30                Setup_Stim:                     ;
FA30 7F0010             clr     digital             ;
FA33 4F                 clra                        ;
FA34 5F                 clrb                        ;
FA35 DD1D               std     S256                ;
FA37 DD1F               std     S1024               ;
FA39 8C0002             cpx     #2                  ;
FA3C 2CEF               bge     jmp_Quit_pulse      ; idle cyle, do not stimulate
FA3E 6D05               tst     chstatus,x          ;
FA40 2BEB               bmi     jmp_Quit_pulse      ; if 0xff, channel is off
FA42 271A               beq     setup_stim2         ; if 0x0, use threshold stimulation only
                    ;
                    ; calculate: gain x stimulus
                    ; use MAC 16bit x 8bit, acc2 = acc3 = 0, both positive only
                    ; radix pt of Gain is before MSBit, i.e.      .xxxx xxxx
                    ; radix pt of pulsewidth is after MSBit  i.e x.xxx xxxx
                    ;
FA44                modulation:                     ;
FA44 E626               ldab    Gain,x              ;
FA46 961B               ldaa    Stimulus            ; MSB
FA48 3D                 mul                         ;
FA49 DD01               std     acc0                ;
                                                    ;
FA4B 961C               ldaa    Stimulus+1          ; LSB
FA4D E626               ldab    Gain,x              ;
FA4F 3D                 mul                         ;
FA50 16                 tab                         ; use only MSB
FA51 4F                 clra                        ;
FA52 D301               addd    acc0                ; calculate upper word ONLY
                                                    ; estimated range 0 - 16384
FA54 04                 lsrd                        ;
FA55 04                 lsrd                        ;
FA56 04                 lsrd                        ;
FA57 04                 lsrd                        ; use d/16, range = 1024
FA58 DD1F               std     S1024               ;
FA5A 04                 lsrd                        ;
FA5B 04                 lsrd                        ; use d/4, range = 256
FA5C DD1D               std     S256                ; store modulated pulsewidth
                                                    ;
FA5E                setup_stim2:                    ;
FA5E 4F                 clra                        ;
FA5F E628               ldab    default_amp,x       ;
FA61 1320800D           brclr   Debug,d_amplitude,store_amplitude
                                                    ; pulsewidth modulation, use fixed amplitude
                    ;                               ;
                    ; for amplitude modulation      ;
                    ;                               ;
FA65 E624               ldab    Base,x              ; use threshold+stimulus
FA67 D31D               addd    S256                ;
FA69 2601               bne     valuecheck1         ;
FA6B 5D                 tstb                        ; disallow value 0
```

```
FA6C                valueok1:                               ;
FA6C 4D                     tsta                            ; see if > 255
FA6D 2703                   beq     store_amplitude         ; smaller, ok.
FA6F CC00FF                 ldd     #$ff                    ; use 0xff
FA72                store_amplitude                         ;
FA72 BDF8BF                 jsr     D_A                     ;
FA75 E62A                   ldab    default_pulse,x         ;
FA77 4F                     clra                            ;
FA78 05                     lsld                            ;
FA79 05                     lsld                            ; use default_pulse * 4
FA7A 122C8011               brset   Debug,d_amplitude,store_pulse
                    ;                                       ;
                    ; for pulsewidth modulation             ;
                    ;                                       ;
FA7E E624                   ldab    Base,x                  ;
FA80 4F                     clra                            ;
FA81 D31F                   addd    S1024                   ;
FA83 2601                   bne     valueok2                ;
FA85 5C                     incb                            ; disallow value 0
                                                            ; pulsewidth of 0 will become a long pulse
FA86                valueok2:                               ;
FA86 1AB3FE79               cpd     maxpulse                ; limit the pulsewidth to maxpulse
FA8A 2F03                   ble     store_pulse             ;
FA8C FCFE79                 ldd     maxpulse                ; use maxpulse
FA8F                store_pulse:                            ;
FA8F DD07                   std     pulsewidth              ;
FA91 05                     lsld                            ;
FA92 05                     lsld                            ;
FA93 05                     lsld                            ;
FA94 DD09                   std     DCDCmod                 ; turn on DCDC converter for 8 times pulsewidth
                                                            ;
FA96 5F                     clrb                            ;
FA97 D712                   stab    OC1_M                   ;
FA99 D719                   stab    TCT1                    ;
                                                            ;
FA9B                Set_Digital_Outputs:                    ;
FA9B 5F                     clrb                            ;
FA9C 130D020B               brclr   Mode,stimLED,nostim
FAA0 8C0001                 cpx     #1                      ;
FAA3 2704                   beq     use_chan1               ;
FAA5 CA80                   orab    #Ch0Sw                  ; turn on ch0
FAA7 2002                   bra     nostim                  ;
FAA9                use_chan1:                              ;
FAA9 CA40                   orab    #Ch1Sw                  ; turn on ch1
FAAB                nostim:                                 ;
FAAB 130D0402               brclr   Mode,sound,nosound
FAAF CA08                   orab    #clicker                ; turn on clicker
FAB1                nosound:                                ;
FAB1 D710                   stab    digital                 ;
                    ;---------------------------------------------------
                    ;                                       ;
                    ; setup all the output compare pins timing
                    ;                                       ;
FAB3 B61020                 ldaa    TCTL1                   ; Force BFET to come on
FAB6 8A03                   oraa    #PUL_BFET               ;
FAB8 B71020                 staa    TCTL1                   ;
FABB 8608                   ldaa    #FOR_BFET               ;
FABD B7100B                 staa    CFORC                   ;
                                                            ;
FAC0 FC100E                 ldd     TCNT                    ;
FAC3 C30098                 addd    #IPI_DLY                ; delta T for BFET
FAC6 FD1016                 std     TOC1                    ; pulse will begin after BFET delay
FAC9 D307                   addd    pulsewidth              ;
FACB FD101C                 std     TOC_PULS                ; raises pulse as TOC1 drop at TOC1+pulsewidth
FACE FD101E                 std     TOC_BFET                ;
FAD1 D309                   addd    DCDCmod                 ;
FAD3 FD101A                 std     TOC_DCDC                ;
                                                            ;
FAD6 141230                 bset    OC1_M,PULS_b+DCDC_b
                                                            ; OC1 raise pulse & DCDC
FAD9 14192A                 bset    tct1,CLR_PULS+CLR_BFET+CLR_DCDC
FADC 9612                   ldaa    OC1_M                   ;
FADE B7100C                 staa    OC1M                    ;
FAE1 9619                   ldaa    tct1                    ;
FAE3 B71020                 staa    TCTL1                   ;
                                                            ;
FAE6                Quit_Pulse:                             ;
FAE6 D610                   ldab    digital                 ;
FAE8 53                     comb                            ;
FAE9 F71004                 stab    PORTB                   ;
                                                            ;
FAEC 39                     rts                             ;
                    ;                                       ;
                    ;***************************************************
                    ; TOC2 Interrupt Service Routine
                    ;
                    ; Interrupts are determined by stimulation frequency
                    ;***************************************************
                    ;
FAED                TOC2_ISR:                               ;{$ Interupt Service Routine }
                                                            ;  { Reset Timer Output Compare 2 .
FAED 0F                     sei                             ; disable further interrupt
FAEE CE1003                 ldx     #portc                  ; store portc to x, can use shorter (faster)
                                                            ; instructions below
FAF1 1D0001                 bclr    0,x,PowerLED            ; signal the beginning of TOC2_ISR
FAF4 8640                   ldaa    #ACK_OC2                ; A:=ClrOC2; { Get the Mask Value }
FAF6 B71023                 staa    TFLG1                   ; TFLG1:=A; { Clear Interrupt Flag }
                    ;                                       ;
                    ; setup for next interrupt              ;
                    ;                                       ;
FAF9 FC1018                 ldd     TOC2                    ; D:=D+ICI_CNT; { Add InterCycleInterval }
FAFC C34E20                 addd    #ICI_CNT                ; 10ms INTERVAL
FAFF FD1018                 std     TOC2                    ; TOC2:=D; { Store In Compare Register }
                                                            ;
FB02                start_AtoD:                             ;
FB02 8600                   ldaa    #Haddr                  ; start A/D converter
                                                            ; read length transducer
FB04 B71030                 staa    adctl                   ;
FB07 4F                     clra                            ;
FB08 B7100C                 staa    OC1M                    ; stop OC1M
                                                            ;
                    ;---------------------------------------------------
FB0B                Check_Buttons:                          ; Check buttons while waiting for conversion
FB0B B61023                 ldaa    TFLG1                   ;
FB0E 8407                   anda    #Mode_B+Stim_B+PWR_B
F310 9711                   staa    Btn                     ;
FB12 B71023                 staa    TFLG1                   ; clear register
                    ;
                    ; A & B buttons reading, use A/D reading from last cycle
                    ;
FB15                checkAB_but:                            ;
FB15 9621                   ldaa    Bs                      ; this is read from last interrupt.
FB17 81E0                   cmpa    #NAB                    ;
```

```
FB19 2210              bhi     end_check_but       ;
FB1B 81A0              cmpa    #YB                 ;
FB1D 2505              blo     checkA              ;
FB1F 141140            bset    Btn,B_But           ; found that B was pressed
FB22 2007              bra     end_check_but       ;
FB24                checkA:                        ;
FB24 8160              cmpa    #YA                 ;
FB26 2503              blo     end_check_but       ;
FB28 141180            bset    Btn,A_But           ; found that A was pressed
                                                   ;
FB2B                end_check_but:                 ;
                                                   ; when power button is clicked
FB2B 1311010A          brclr   Btn,Pwr_B,flash_LED
FB2F B61000            ldaa    porta               ; then check to see if stimulation button
FB32 8504              bita    #Stim_b             ; is clicked too.
FB34 2703              beq     flash_LED           ; If yes, this means special modes
FB36 7EF702            jmp     power_down          ; else power down request
                    ;--------------------------------------------------------
                    ;
                    ; take care of LED display and all the pending modes/ current mode
                    ;
FB39                flash_LED:                     ;
FB39 4F                 clra                      ; a = 0
FB3A D60D               ldab    Mode              ; b = operating mode
                                                  ; if Wait==0 this means that
                                                  ; pending mode is not active
                                                  ; goto show operating mode
FB3C 131AFF2F           brclr   Wait,$ff,operating
FB40 7A001A             dec     Wait              ; decrease pending wait
FB43 2719               beq     newmode           ; Just turn 0? Change mode (b = Mode)
                    ;                         ;
                    ; In Pending Mode!        ;
                    ;                         ;
FB45 7A0016             dec     Flash             ;
FB48 263A               bne     output_LED        ; a = 0 still
                    ;
                    ; show current mode when Flash turns zero
                    ;
FB4A 8603               ldaa    #pendingflash     ; a = flash, setup for next LED display
FB4C D60E               ldab    Pending           ; b = Pending
                                                  ; if not debug mode change, show normal mode
FB4E 130C012C           brclr   Debugflag,d_changemode,store_LED
FB52 D60B               ldab    DebugPend         ; show pending debug mode
                                                  ;
FB54 2028               bra     store_LED         ; output Pending to LED, store pendingflash
                                                  ;
                    ;--------------------------------------------------------
FB56                newdebug:
FB56 D62C               ldab    Debug             ;
FB58 D80B               eorb    DebugPend         ; toggle debug mode bit
FB5A D72C               stab    Debug             ;
FB5C 2019               bra     operating2        ;
                    ;--------------------------------------------------------
FB5E                newmode:                       ;
                                                   ; is it in debug mode? yes goto setup newdebug
FB5E 120C01F4           brset   Debugflag,d_changemode,newdebug
                                                   ; is it sound pending mode?
                                                   ; if yes, means that sound is to be toggled
FB62 120E0402           brset   Pending,Sound,toggle_sound
FB66 C407               andb    #clrmode           ; b = Mode, now clear all operating modes
FB68                toggle_sound:                  ;
FB68 D80E               eorb    Pending            ; setup new mode or toggle sound mode
```

```
FB6A 17                 tba                        ;
FB6B 84F8               anda    #allmode           ;
FB6D 970E               staa    Pending            ;
                    ;--------------------------------------------------------
                    ;
                    ; no pending mode, b = current mode
                    ;
FB6F                operating:                     ;
FB6F D70D               stab    Mode               ; Mode = b
                                                   ; debug mode only
                                                   ; if not debug change mode,
                                                   ; then display current mode
FB71 130C0102           brclr   Debugflag,d_changemode,operating2
FB75 D62C               ldab    Debug              ; else display debug mode
FB77                operating2:                    ;
FB77 7A0016             dec     Flash              ;
FB7A 2608               bne     output_LED         ;
FB7C 860F               ldaa    #normalflash       ;
                    ;--------------------------------------------------------
                    ;
                    ; a = flash period, b = new LED display
                    ;
FB7E                store_LED:
FB7E 9716               staa    Flash              ;
FB80 980F               ldaa    LED                ; use LED
FB82 D70F               stab    LED                ; store Pending into LED
                    ;--------------------------------------------------------
                    ;
                    ; a = current LED display
                    ;
FB84                output_LED:
FB84 8A01               oraa    #PowerLED          ; Power LED on (TOC2_ISR is still running)
FB86 43                 coma                       ;
FB87 A700               staa    0,x                ; write to LED, x = portc
                    ;                          ;
                    ; run mode operations:     ;
                    ;                          ;
     00000000           if      analysismode       ;
                        endif                      ;
FB89 4F                 clra                       ;
FB8A 5F                 clrb                       ;
FB8B DD05               std     chstatus           ; set both channels to base
only
FB8D 8F                 xgdx                       ; set x = 0
FB8E 130D9808           brclr   Mode,ch0gain+ch1gain+normal,finish_filter
                    ;
                    ; only mode left: ch0gain, ch1gain, normal
                    ;
                    ; ********** run filter **********
                    ;
FB92 8D54               bsr     filter_start       ;
FB94 DD1B               std     Stimulus           ;
FB96 8601               ldaa    #1                 ; enable channel x
FB98 A705               staa    chstatus,x         ;
                                                   ;
FB9A                finish_filter:                 ;
FB9A 9613               ldaa    StimWait           ; check if stim should be off temporarily
FB9C 2705               beq     stimulate          ;
FB9E 4A                 deca                       ;
FB9F 9713               staa    StimWait           ;
FBA1 8F05               clr     chstatus,x         ; turn stimulus
FBA3                stimulate:                     ;
```

```
FBA3 CE0003             ldx     #totalchannel       ;
FBA6 132C1001           brclr   Debug,d_25Hz,stimulate1
FBAA 08                 inx                         ;
FBAB                stimulate1:                     ;
FBAB DC14               ldd     StimCh              ;
FBAD 2701               beq     store_stimch        ; if StimCh ==0, reset stimch
FBAF 8F                 xgdx                        ;
FBB0                store_stimch:                   ;
FBB0 09                 dex                         ;
FBB1 DF14               stx     StimCh              ;
FBB3 BDFA30             jsr     setup_stim          ;
                    ;-------------------------------------------------
                    ;
FBB6                end_TOC2_ISR:
FBB6 8601               ldaa    #EBUT               ; start reading A,B buttons
FBB8 B71030             staa    adctl               ;
FBBB BDF94A             jsr     process_key         ;
FBBE                wait_for_ccf1:                  ; WAIT FOR CONVERSION COMPLETE FLAG
FBBE B61030             ldaa    adctl               ;
FBC1 2AFB               bpl     wait_for_ccf1       ;
FBC3 B61031             ldaa    ADR1                ; read A & B button
FBC6 9721               staa    Bs                  ; save A & B buttons analog reading
FBC8 BDFE2D             jsr     outputSCI           ; output input/output to SCI port
FBCB CE1003             ldx     #portc              ;
FBCE 1C0001             bset    0,x,PowerLED        ; Signal the END of TOC2_ISR
FBD1 3B                 rti                         ;

     00000000           if      analysismode
                        endif                       ;
                    ;
                    ; ****************************************
                    ; Power_down
                    ; ****************************************
                    ;                               ;
FBD2                Power_Down:                     ;
FBD2 0F                 sei                         ;
FBD3 4F                 clra                        ;
FBD4 B71022             staa    TMSK1               ; no more interrupt
FBD7 43                 coma                        ; a = $ff
FBD8 B71004             staa    portb               ; turn off all LED
FBDB B71003             staa    portc               ; turn off all DEVICE except power
FBDE                wait_for_release:               ;
FBDE B61000             ldaa    porta               ;
FBE1 8501               bita    #Pwr_B              ;
FBE3 26F9               bne     wait_for_release    ;
FBE5 7EFEBE             jmp     stopit              ;
                    ;*****************************************************************
                    ; IIR Filter Code :
                    ;       return D as result, x as channel number (0 or 1)
                    ;*****************************************************************
                    ;
FBE8                filter_start:                   ;
                                                    ;
FBE8 18CE0054           ldy     #IIRbuffer          ; y = IIR buffer
                                                    ;
FBEC                wait_for_ccf:                   ; WAIT FOR CONVERSION COMPLETE FLAG
FBEC B61030             ldaa    adctl               ;
FBEF 2AFB               bpl     wait_for_ccf        ;
                                                    ;
FBF1 5F                 clrb                        ;
FBF2 D700               stab    ovbit               ; set overflow bit to zero

FBF4 B61031             ldaa    ADR1                ; HAND SENSOR
                                                    ; use the average of the 4 A/D ports
FBF7 BB1032             adda    ADR2                ;
FBFA C900               adcb    #0                  ;
FBFC BB1033             adda    ADR3                ;
FBFF C900               adcb    #0                  ;
FC01 BB1034             adda    ADR4                ;
FC04 C900               adcb    #0                  ;
FC06 56                 rorb                        ; carry bit is 0
FC07 46                 rora                        ;
FC08 56                 rorb                        ;
FC09 46                 rora                        ;
FC0A 56                 rorb                        ; D is the average of ADR's
                                                    ;
FC0B 8080               suba    #$80                ; convert to signed value
FC0D DD54               std     IIRbuffer           ; store a 10-bit number
FC0F DD22               std     Hs                  ; save sensor value
                                                    ;
                    ;                               ;
                    ; store in circular buffer, for use in conditional filter
                    ;                               ;
FC11 DE2F               ldx     xindex              ;
FC13 A734               staa    xbuffer,x           ;
FC15 08                 inx                         ;
FC16 8C0020             cpx     #xbufsize           ;
FC19 2603               bne     cont1               ;
FC1B CE0000             ldx     #0                  ;
FC1E                cont1:                          ;
FC1E DF2F               stx     xindex              ;
                                                    ;
                    ;-------------------------------------------------
                    ;
                    ; the actual filter calculation starts here
                    ;
FC20 CEFE7B             ldx     #IIRparam           ;
FC23 E600               ldab    0,x                 ; no of sections
FC25 08                 inx                         ; x = IIRcoeff
                                                    ;
FC26                loopIIR:                        ;
FC26 37                 pshb                        ; save no of sections
                                                    ;
                    ;                               ;
                    ;************ MAC i.e. 16x16 multiplication
                    ;                               ;
FC27 A601               ldaa    1,x                 ;
FC29 18E601             ldab    1,y                 ;
FC2C 3D                 mul                         ;
FC2D DD03               std     acc2                ;
FC2F A600               ldaa    0,x                 ;
FC31 18E600             ldab    0,y                 ;
FC34 3D                 mul                         ;
FC35 DD01               std     acc0                ;
FC37 A600               ldaa    0,x                 ;
FC39 18E601             ldab    1,y                 ;
FC3C 3D                 mul                         ;
FC3D D302               addd    acc1                ;
FC3F DD02               std     acc1                ;
FC41 9601               ldaa    acc0                ;
FC43 8900               adca    #0                  ;
FC45 9701               staa    acc0                ;
                                                    ;
FC47 A601               ldaa    1,x                 ;
```

```
FC49 18E600            ldab    0,y
FC4C 3D                mul
FC4D D302              addd    acc1
FC4F DD02              std     acc1
FC51 DC01              ldd     acc0
FC53 8900              adca    #0

FC55 186D00            tst     0,y
FC58 2A02              bpl     label11
FC5A A300              subd    0,x
FC5C            label11:
FC5C 6D00              tst     0,x
FC5E 2A03              bpl     label12
FC60 18A300            subd    0,y
FC63            label12:
FC63 780003            lsl     acc2
FC66 59                rolb
FC67 49                rola
                ;
                ;************ END of MAC
                ;
FC68 2809              bvc     noverflow1      ; only case of overflowing -1*-1 = 1
FC6A CC7FFF            ldd     #$7fff          ; use this overflow value (32767:32768)
FC6D D700              stab    ovbit           ; indicate overflow occurred
FC6F 2402              bcc     noverflow1
FC71 43                coma                    ; use this overflow value (-1)
FC72 53                comb
FC73            noverflow1:
FC73 18E302            addd    2,y             ; w1(n-1)
FC76 291D              bvs     overflow1
FC78 DD01              std     acc0

FC7A E60A              ldab    $a,x
FC7C 2713              beq     noshift
FC7E 2000              bra     shift
FC80            shift:
FC80 780004            lsl     acc3
FC83 790003            rol     acc2
FC86 790002            rol     acc1
FC89 790001            rol     acc0
FC8C 2907              bvs     overflow1
FC8E 5A                decb
FC8F 26EF              bne     shift
FC91            noshift:
FC91 DC01              ldd     acc0
FC93 2007              bra     store1
FC95            overflow1:
FC95 CC7FFF            ldd     #$7fff
FC98 2402              bcc     store1
FC9A 43                coma
FC9B 53                comb
FC9C            store1:
FC9C 18ED08            std     8,y             ; y(n)
                ;
                ;************ MAC
                ;
FC9F A603              ldaa    3,x
FCA1 18E601            ldab    1,y
FCA4 3D                mul
FCA5 DD03              std     acc2
FCA7 A602              ldaa    2,x
```

```
FCA9 18E600            ldab    0,y
FCAC 3D                mul
FCAD DD01              std     acc0
FCAF A602              ldaa    2,x
FCB1 18E601            ldab    1,y
FCB4 3D                mul
FCB5 D302              addd    acc1
FCB7 DD02              std     acc1
FCB9 9601              ldaa    acc0
FCBB 8900              adca    #0
FCBD 9701              staa    acc0

FCBF A603              ldaa    3,x
FCC1 18E600            ldab    0,y
FCC4 3D                mul
FCC5 D302              addd    acc1
FCC7 DD02              std     acc1
FCC9 DC01              ldd     acc0
FCCB 8900              adca    #0

FCCD 186D00            tst     0,y
FCD0 2A02              bpl     label21
FCD2 A302              subd    2,x
FCD4            label21:
FCD4 6D02              tst     2,x
FCD6 2A03              bpl     label22
FCD8 18A300            subd    0,y
FCDB            label22:
FCDB DD01              std     acc0
                ;
                ;************ MAC
                ;
FCDD A605              ldaa    5,x
FCDF 18E607            ldab    7,y
FCE2 3D                mul
FCE3 D303              addd    acc2
FCE5 DD03              std     acc2
FCE7 DC01              ldd     acc0
FCE9 C900              adcb    #0
FCEB 8900              adca    #0
FCED DD01              std     acc0
FCEF A604              ldaa    4,x
FCF1 18E606            ldab    6,y
FCF4 3D                mul
FCF5 D301              addd    acc0
FCF7 DD01              std     acc0
FCF9 A604              ldaa    4,x
FCFB 18E607            ldab    7,y
FCFE 3D                mul
FCFF D302              addd    acc1
FD01 DD02              std     acc1
FD03 9601              ldaa    acc0
FD05 8900              adca    #0
FD07 9701              staa    acc0
FD09 A605              ldaa    5,x
FD0B 18E606            ldab    6,y
FD0E 3D                mul
FD0F D302              addd    acc1
FD11 DD02              std     acc1
FD13 DC01              ldd     acc0
FD15 8900              adca    #0
```

```
FD17 186D06          tst     6,y
FD1A 2A02            bpl     label31
FD1C A304            subd    4,x
FD1E            label31:
FD1E 6D04            tst     4,x
FD20 2A03            bpl     label32
FD22 18A306          subd    6,y
FD25            label32:
FD25 780003          lsl     acc2
FD28 59              rolb
FD29 49              rola
                     ;
                     ;************ END of MAC
                     ;

FD2A 2809            bvc     noverflow2      ; chance of overflow (acc has 2 numbers added)
FD2C CC7FFF          ldd     #$7fff
FD2F D700            stab    ovbit
FD31 2402            bcc     noverflow2
FD33 43              coma
FD34 53              comb

FD35            noverflow2:
FD35 18E304          addd    4,y
FD38 2807            bvc     store2

FD3A CC7FFF          ldd     #$7fff
FD3D 2402            bcc     store2
FD3F 43              coma
FD40 53              comb
FD41            store2:
FD41 18ED02          std     2,y
                     ;
                     ;************ MAC
                     ;

FD44 A607            ldaa    7,x
FD46 18E601          ldab    1,y
FD49 3D              mul
FD4A DD03            std     acc2
FD4C A606            ldaa    6,x
FD4E 18E600          ldab    0,y
FD51 3D              mul
FD52 DD01            std     acc0
FD54 A606            ldaa    6,x
FD56 18E601          ldab    1,y
FD59 3D              mul
FD5A D302            addd    acc1
FD5C DD02            std     acc1
FD5E 9601            ldaa    acc0
FD60 8900            adca    #0
FD62 9701            staa    acc0

FD64 A607            ldaa    7,x
FD66 18E600          ldab    0,y
FD69 3D              mul
FD6A D302            addd    acc1
FD6C DD02            std     acc1
FD6E DC01            ldd     acc0
FD70 8900            adca    #0

FD72 186D00          tst     0,y
FD75 2A02            bpl     label41
FD77 A306            subd    6,x
```

```
FD79            label41:
FD79 6D06            tst     6,x
FD7B 2A03            bpl     label42
FD7D 18A300          subd    0,y
FD80            label42:
FD80 DD01            std     acc0
                     ;
                     ;************ MAC
                     ;

FD82 A609            ldaa    9,x
FD84 18E607          ldab    7,y
FD87 3D              mul
FD88 D303            addd    acc2
FD8A DD03            std     acc2
FD8C DC01            ldd     acc0
FD8E C900            adcb    #0
FD90 8900            adca    #0
FD92 DD01            std     acc0
FD94 A608            ldaa    8,x
FD96 18E606          ldab    6,y
FD99 3D              mul
FD9A D301            addd    acc0
FD9C DD01            std     acc0
FD9E A608            ldaa    8,x
FDA0 18E607          ldab    7,y
FDA3 3D              mul
FDA4 D302            addd    acc1
FDA6 DD02            std     acc1
FDA8 9601            ldaa    acc0
FDAA 8900            adca    #0
FDAC 9701            staa    acc0

FDAE A609            ldaa    9,x
FDB0 18E606          ldab    6,y
FDB3 3D              mul
FDB4 D302            addd    acc1
FDB6 DD02            std     acc1
FDB8 DC01            ldd     acc0
FDBA 8900            adca    #0

FDBC 186D06          tst     6,y
FDBF 2A02            bpl     label51
FDC1 A308            subd    8,x
FDC3            label51:
FDC3 6D08            tst     8,x
FDC5 2A03            bpl     label52
FDC7 18A306          subd    6,y
FDCA            label52:
FDCA 780003          lsl     acc2
FDCD 59              rolb
FDCE 49              rola
                     ;
                     ;************ END of MAC
                     ;

FDCF 2809            bvc     store3
FDD1            overflow3:
FDD1 CC7FFF          ldd     #$7fff
FDD4 D700            stab    ovbit
FDD6 2402            bcc     store3
FDD8 43              coma
FDD9 53              comb
FDDA            store3:
```

```
FDDA 18ED04            std     4,y
FDDD C60B              ldab    #$b             ; $b = 11
FDDF 3A                abx                     ; use next set of cooefficients
FDE0 C606              ldab    #$6
FDE2 183A              aby                     ; use next set of buffer space
FDE4 33                pulb                    ; get number of sections left
FDE5 5A                decb                    ; finish all sections?
FDE6 2703              beq     exitIIR         ; yes, then quit
FDE8 7EFC28            jmp     loopIIR         ; do next section

FDEB            exitIIR:

                                               ; if conditional filter is not on,
                                               ; skip the following
FDEB 122C4023          brset   Debug,d_nonlinear,end_nonlinear

                ; conditional filter: start searching for maximum and minimum input

FDEF CE0020            ldx     #xbufsize
FDF2 A633              ldaa    xbuffer-1,x
FDF4 9731              staa    xmin
FDF6 9732              staa    xmax
FDF8            cmploop:
FDF8 09                dex
FDF9 2710              beq     cmpend

FDFB A633              ldaa    xbuffer-1,x
FDFD 9131              cmpa    xmin
FDFF 2C02              bge     cmpnext
FE01 9731              staa    xmin
FE03            cmpnext:
FE03 9132              cmpa    xmax
FE05 2FF1              ble     cmploop
FE07 9732              staa    xmax
FE09 20ED              bra     cmploop

FE0B            cmpend:
FE0B 9632              ldaa    xmax
FE0D 9031              suba    xmin
FE0F 44                lsra                    ; use range/2, unsigned number
FE10 9733              staa    xrange

FE12            end_nonlinear:
FE12 CE0000            ldx     #0              ; use channel 0
FE15 18EC00            ldd     0,y             ; output
FE18 2A06              bpl     check_result
FE1A 4F                clra                    ; get absolute value
FE1B 5F                clrb
FE1C 18A300            subd    0,y
FE1F 08                inx                     ; use channel 1 if negative

FE20            check_result:                  ; conditional filter on?

FE20 122C4008          brset   Debug,d_nonlinear,end_filter
FE24 9133              cmpa    xrange          ; see if stimulation should be off
FE26 2304              bls     end_filter      ; no, then return
FE28 861E              ldaa    #30             ; disable stimulator for 10 stimulation periods
FE2A 9713              staa    StimWait
FE2C            end_filter:

FE2C 39                rts

                ;--------------------------------------------------------------
                ; Output to SCI port - input and filter output

FE2D            outputSCI:
FE2D            waitempty1:
FE2D B6102E            ldaa    SCSR
FE30 8580              bita    #$80
FE32 27F9              beq     waitempty1      ; wait SCI output buffer to be empty
FE34 9623              ldaa    Hs+1
FE36 B7102F            staa    SCDR
FE39            waitempty2:
FE39 B6102E            ldaa    SCSR
FE3C 8580              bita    #$80
FE3E 27F9              beq     waitempty2      ; wait SCI output buffer to be empty
FE40 9622              ldaa    Hs
FE42 B7102F            staa    SCDR

FE45 18CE0054          ldy     #IIRbuffer      ; find the output from the corresponding stage
FE49 C608              ldab    #8
     00000000          if      analysismode
                       else
FE4B B6FE7B            ldaa    IIRparam        ; get the number of stage
FE4E            check_stage:
FE4E 183A              aby
FE50 4A                deca
FE51 26FB              bne     check_stage
                       endif

FE53            waitempty3:
FE53 B6102E            ldaa    SCSR
FE56 8580              bita    #$80
FE58 27F9              beq     waitempty3
FE5A 18A601            ldaa    1,y
FE5D B7102F            staa    SCDR
FE60            waitempty4:
FE60 B6102E            ldaa    SCSR
FE63 8580              bita    #$80
FE65 27F9              beq     waitempty4
FE67 18A600            ldaa    0,y
FE6A B7102F            staa    SCDR
FE6D 39                rts

                ;**************************************************************
                ; NON-VOLATILE EEDATA
                ;**************************************************************

FE8E            ex_start:                      ; Save the parameters in here

FE8E 00         ex_Base  db     0
FE8F 00                  db     0
FE70 00         ex_Gain  db     0
FE71 00                  db     0
FE72            ex_default_amp
FE72 98                  db     150             ; default voltage output
```

```
FE73 96                 db      150             ;
FE74            ex_default_pulse
FE74 7D                 db      125             ;
FE75 7D                 db      125             ;
FE76 00         ex_Debug db     0               ;
FE77            ex_end:                         ;
FE77            maxresult:
     2000               dw      8192            ;
FE79 0400       maxpulse dw     1024            ; 500us max pulse
                                                ;
FE7B            IIRparam:                       ; Filter parameter definitions
FE7B 05                 db      $00005          ; no of sections
FE7C            IIRcoeff:
FE7C            coef:
                ;
                ; filter from wrist5.m
                ;
FE7C 4766               dw      $04766          ; b0 = 0.557789
FE7E 9985               dw      $099[illegible] ; b1 = -0.800627
FE80 3B81               dw      $03[illegible]  ; -a1 = 0.464888
FE82 3396               dw      $[illegible]    ; b2 = 0.403003
FE84 F000               dw      $[illegible]    ; -a2 = -0.125
FE86 01                 db      $00001          ; no of left shifts
FE87 2F78               dw      $02F78          ; b0 = 0.370841
FE89 8949               dw      $0B949          ; b1 = -0.552449
FE8B 1CF4               dw      $01CF4          ; -a1 = 0.226207
FE8D 224C               dw      $0224C          ; b2 = 0.267932
FE8F F800               dw      $0F800          ; -a2 = -0.0625
FE91 02                 db      $00002          ; no of left shifts
FE92 2D31               dw      $02D31          ; b0 = 0.353069
FE94 AA61               dw      $0AA61          ; b1 = -0.668925
FE96 2C02               dw      $02C02          ; -a1 = 0.343801
FE98 28C9               dw      $028C9          ; b2 = 0.318645
FE9A F052               dw      $0F052          ; -a2 = -0.1225
FE9C 02                 db      $00002          ; no of left shifts
FE9D 367A               dw      $0367A          ; b0 = 0.425595
FE9F 965E               dw      $0965E          ; b1 = -0.825247
FEA1 6496               dw      $06496          ; -a1 = 0.78583
FEA3 3342               dw      $03342          ; b2 = 0.400442
FEA5 D70A               dw      $0D70A          ; -a2 = -0.32
FEA7 01                 db      $00001          ; no of left shifts
FEA8 3B92               dw      $03B92          ; b0 = 0.465384
FEAA 94C7               dw      $094C7          ; b1 = -0.837691
FEAC 6000               dw      $06000          ; -a1 = 0.75
FEAE 3040               dw      $03040          ; b2 = 0.376961
FEB0 DC00               dw      $0DC00          ; -a2 = -0.28125
FEB2 01                 db      $00001          ; no of left shifts
FEB3 000B               ds      11              ;
                                                ;
FEBE            IIRparam_end:
                ;
                ;*****************************************************************
                ; Emergency Stop
                ; Any other Interrupt Service Routine
                ;*****************************************************************
                ;
                                                ;
FEBE            SCI_ISR                         ; Serial Port Interrupt
FEBE            SPI_ISR                         ; Serial Peripheral Interface Interrupt
FEBE            PAIE_ISR                        ; Pulse Accumulator Input Edge Interrupt
FEBE            PAO_ISR                         ; Pulse Accumulator Overflow Interrupt
FEBE            TO_ISR                          ; Timer Overflow Interrupt
FEBE            TOC5_ISR                        ; Timer Output Compare 5 Interrupt
FEBE            TOC4_ISR                        ; Timer Output Compare 4 Interrupt
FEBE            TOC3_ISR                        ; Timer Output Compare 3 Interrupt
                ;
                ;TOC2_ISR                       ; Timer Output Compare 2 Interrupt
                ;
FEBE            TOC1_ISR                        ; Timer Output Compare 1 Interrupt
FEBE            TIC3_ISR                        ; Timer Input Capture 3 Interrupt
FEBE            TIC2_ISR                        ; Timer Input Capture 2 Interrupt
FEBE            TIC1_ISR                        ; Timer Input Capture 1 Interrupt
FEBE            RTI_ISR                         ; Real Time Interrupt
FEBE            IRQ_ISR                         ; Maskable Interrupt
FEBE            XIRQ_ISR                        ; Non-Maskable Interrupt
FEBE            SWI_ISR                         ; Software Interrupt
FEBE            ILL_OP_ISR                      ; Illegal Opcode Interrupt
FEBE            COP_ISR                         ; Computer Operating Properly Watchdog Int
FEBE            CLK_ISR                         ; Clock Monitor Interrupt
                                                ;
FEBE            STOPIT:
                ;
                ; This routine will stop the microprocessor (save parameter first)
                ;*****************************************************************
                ;
                ; Copy EEPROM programming code to RAM
                ;*****************************************************************
                ;
     FFFFFFFF           if      saveparam
FEBE CEF8DB             ldx     #SourceAddr     ; address of vector table in EEPROM
FEC1 18CE[illegible]54  ldy     #Target*Addr    ; RAM address for vector table
                                                ;
FEC5            copy_code:                      ; Copy Vector table from EEPROM to
                                                ; RAM address
FEC5 A600               ldaa    0,x             ;
FEC7 18A700             staa    0,y             ;
FECA 08                 inx                     ;
FECB 1808               iny                     ;
FECD 8CF918             cpx     #End_Code       ;
FED0 25F3               blo     copy_code       ;
                                                ;
FED2 CEFE6E             ldx     #ex_start       ;  y:=EX_Start.{ Start of EEPROM data }
FED5 18CE0024           ldy     #tx_start       ;  x:=TX_Start.{ Start of TxData data }
FED9            update1:                        ;  Repeat
FED9 18A600             ldaa    0,y             ;   A:=(y);
FEDC 9D54               jsr     _program        ;   write to eeprom
FEDE 1808               iny                     ;   x:=x+1;
FEE0 08                 inx                     ;   y:=y+1;
FEE1 8CFE77             cpx     #ex_end         ;  Until x=ex_end;
FEE4 25F3               blo     update1         ;
                        endif                   ;
FEE8            shut_down:                      ;
FEE8 86FE               LDAA    #$FE            ; turn off power to system
FEEA B71004             STAA    PORTB           ;
FEED 86FF               LDAA    #$FF            ; turn off all LED
FEEF B71003             STAA    PORTC           ;
FEF2 8650               LDAA    #$50            ; Stop-Enable; IRQ, XIRQ-Off
FEF4 06                 TAP                     ;
FEF5 20F1               bra     shut_down       ; In Case Continue by XIRQ
FEF7 3B                 RTI                     ;
                                                ;
                ;*****************************************************************
                ; Interrupt Jump Vector Table
```

```
                ;*****************************************************************
                ;
                ;
FFD6                    org     EEVECTAB            ;
FFD6            vector_table:                       ;
FFD6 FEBE       SCI     dw      SCI_ISR             ; Serial Port Interrupt
FFD8 FEBE       SPI     dw      SPI_ISR             ; Serial Peripheral Interface Interrupt
FFDA FEBE       PAIE    dw      PAIE_ISR            ; Pulse Accumulator Input Edge Interrupt
FFDC FEBE       PAO     dw      PAO_ISR             ; Pulse Accumulator Overflow Interrupt
FFDE FEBE       TO      dw      TO_ISR              ; Timer Overflow Interrupt
FFE0 FEBE       TOC5I   dw      TOC5_ISR            ; Timer Output Compare 5 Interrupt
FFE2 FEBE       TOC4I   dw      TOC4_ISR            ; Timer Output Compare 4 Interrupt
FFE4 FEBE       TOC3I   dw      TOC3_ISR            ; Timer Output Compare 3 Interrupt
FFE6 FAED       TOC2I   dw      TOC2_ISR            ; Timer Output Compare 2 Interrupt
FFE8 FEBE       TOC1I   dw      TOC1_ISR            ; Timer Output Compare 1 Interrupt
FFEA FEBE       TIC3I   dw      TIC3_ISR            ; Timer Input Capture 3 Interrupt
FFEC FEBE       TIC2I   dw      TIC2_ISR            ; Timer Input Capture 2 Interrupt
FFEE FEBE       TIC1I   dw      TIC1_ISR            ; Timer Input Capture 1 Interrupt
FFF0 FEBE       RTI_I   dw      RTI_ISR             ; Real Time Interrupt
FFF2 FEBE       IRQ_I   dw      IRQ_ISR             ; Maskable Interrupt
FFF4 FEBE       XIRQ_I  dw      XIRQ_ISR            ; Non-Maskable Interrupt
FFF6 FEBE       SWI_I   dw      SWI_ISR             , Software Interrupt
FFF8 FEBE       ILL_OP  dw      ILL_OP_ISR          ; Illegal Opcode Interrupt
FFFA FEBE       COP_I   dw      COP_ISR             ; Computer Operating Properly Watchdog Int
FFFC FEBE       CLK_I   dw      CLK_ISR             ; Clock Monitor Interrupt
FFFE F800       RESET   dw      PROG                ; Reset Vector
                                                    ;
10000                             END               ;
There were: 0000 Errors

SECTIONS USED:

 Section          Relocation   Alignment  Type     Begins      Ends
 DATA             absolute                         0000        007A
 PROGRAM          absolute                         F800        FFFF

Assembly time: 0 min, 3 sec
```

APPENDIX C

Listing of the 68HC16 assembly code running in the evaluation system.

```
        include "equates.asm"
;
;   this one works with the HC16 EVB using the on-board D/A
;   includes SCI port communication routines to talk to the outside
;   included by another file where the coefficients are.
;   This is for use with tf2cas16.m generated coefficient files
;

freq    equ     $064            ; 100Hz sampling frequency

;
; Do all 68HC16 initialization
;
        org     $0200
bank:
        ldab    #$00
        tbsk
        ldab    #$0f
        tbek
SRAM:
        lds     #$33fe
        ldab    #$00
        stab    $fb05
        ldd     #$3000
        std     $fb06
        clr     $fb00
clk:
        ldab    #$7f
        stab    $fa04
        clr     $fa21
        ldd     #$0001
        std     $fc08
        ldd     #$000c
        std     $fc0a
xyz:
        clrd
        tbxk
        tbyk
        tbzk
        ldz     #0
        ldx     #$2100
clrloop:
        std     0,x
        aix     #2
        cpx     #$2600
        bne     clrloop
mask:
        ldaa    #$0ff
        ldab    #$0ff
        tdmsk
adc:
        clr     ADCMCR
        ldab    #$080
        stab    ADCTL0
dac:
        ldab    #%00001010      ;PCS0/SS and MOSI used by QSPI
        stab    QPAR
        ldab    #%00001110      ;PCS0/SS MOSI and SCK used as
outputs
        stab    QDDR
comd:
        ldab    #%11100001      ;cont, bitse, dt
        stab    $fd40           ;command 0
        ldab    #%11100000
        stab    $fd41
        ldd     #$0100
        std     SPCR2
brate:
        ldab    #$0F
        stab    $0fc19
        stab    $0fc1b
        ldab    #$080
        stab    SPCR0
;
; Set up the periodic timer interrupt to generate an interrupt
; 100 times / second.
;
timer:
        ldaa    #00
        ldab    #($8000/freq/$4)
        std     PITR
        ldd     #ISR
        std     $70,z
        ldaa    #$06            ; interrupt level 7
        ldab    #$38            ; interrupt vector $38
        std     PICR
        clr     ADCTL1

        orp     #$10
;
```

```
; setup the serial communication port
;
sciport:
        ldd     #$37            ;!48
        std     SCCR0
        ldd     #%001000000001000
        std     SCCR1
        ldd     #SCIISR
        std     $80,z
        std     $82,z
        ldaa    #$07            ; interrupt level 6
        ldab    #$40            ; interrupt vector $40
        std     QUILR
        ldd     #$08f
        std     QMCR            ; setup interrupt Arbitration ID

;
        andp    #$ff1f          ; enable interrupts
;
; program stuck here and wait for interrupts
;
iloop:
        bra     iloop
;
; Interrupt service routine - for periodic timer
;
ISR:
        pshm    D,E,X,Y,Z,K     ; save all registers
loopADC:
        ldaa    ADSTAT          ; wait for A/D conversion to complete
        bpl     loopADC

        ldd     $f720           ; get ADC result
        ldy     #$2100
        std     0,y             ; y points to buffer
        ldx     #$2000
        ldd     0,x             ; D = number of sections
        inx
        inx                     ; x points to coefficient
        ldhi                    ; load first two operands in registers
;
; the actual IIR filter algorithm
;
loop6816:
        pshb                    ; save number of sections left to
do
        lde     2,y             ; acc = w1(n-1)
        tem
        mac     2,0             ; acc = w1(n-1)+b0*x(n)
        ldd     8,x             ; get the number of left shifts
shift6816:
        decb
        bmi     noshift6816
        aslm
        jmp     shift6816
noshift6816:
        tmet                    ;
        ste     6,y             ; y(n) = acc
        lde     4,y             ; acc = w2(n-1)
        tem
        mac     2,6
        mac     2,-6            ; acc = b1*x(n)+a1*y(n) + w2(n-1)
        tmet
        ste     2,y             ; w1(n) = acc
        clrm
        mac     2,6
        mac     4,0             ; acc = b2*x(n)+a2*y(n)
        tmet
        ste     -2,y            ; w2(n) = acc

        pulb                    ; check number of sections left
        decb
        beq     exit6816        ; no more, then exit
        jmp     loop6816

exit6816:
        lde     0,y             ;load the signal from last section
        ste     $fd20           ;store data to be transmitted

output:
        ldaa    #$80            ; send output to D/A
        staa    SPCR1
        clr     ADCTL1          ; start next A/D conversion
        pulm    D,E,X,Y,Z,K
        rti
;
; The following handles the communication
;
SCIISR:
```

```
        pshm    D,X,Y,K
        ldy     #$2f00          ; SCI variables
        ldab    #$f
        tbek
        decw    0,y             ; no of char in buffer
        bpl     sendchar
        ldd     #%0010000000001000
        std     SCCR1
sendchar:
        ldx     4,y             ; current output data index
        ldd     SCSR            ; to acknowledge interrupt
        ldab    0,x             ; get data
        std     SCDR
        xgdx
        incb
        std     4,y
        pulm    D,X,Y,K
        rti

; character in A
SEND:
        pshm    D,X,Y,K
        ldy     #$2f00
        ldab    #$f
        tbek

        ldx     2,y
        staa    0,x
        xgdx
        incb
        std     2,y
        inc     0,y
        bne     sendcont
        ldd     #%0010000010001000
        std     SCCR1
sendcont:
        pulm    D,X,Y,K
        rts

;
;       2000    Coefficients
;       2100    IIR values
;       2f00    SCI variable
;       3000    SCI buffer
```

```
;
; The coefficients for the IIR filter
;
        org     $2000
        dc.w      $00005; no of sections
coef:
        dc.w      $071BC; b0 = 0.888557
        dc.w      $0B42D; b1 = -0.592371
        dc.w      $0E668; -a1 = -0.199942
        dc.w      $00CA3; b2 = 0.0987286
        dc.w      $0FD71; -a2 = -0.0199885
        dc.w      $00001; no of left shifts
        dc.w      $033A9; b0 = 0.40361
        dc.w      $09CA5; b1 = -0.776206
        dc.w      $0F332; -a1 = -0.100029
        dc.w      $02FC5; b2 = 0.373197
        dc.w      $0FEB8; -a2 = -0.0100058
        dc.w      $00002; no of left shifts
        dc.w      $03A5B; b0 = 0.455899
        dc.w      $08F14; b1 = -0.882199
        dc.w      $06DF9; -a1 = 0.859166
        dc.w      $036A1; b2 = 0.426803
        dc.w      $0D0C2; -a2 = -0.369086
        dc.w      $00001; no of left shifts
        dc.w      $03A23; b0 = 0.454181
        dc.w      $08ED0; b1 = -0.884279
        dc.w      $06E4E; -a1 = 0.861761
        dc.w      $03718; b2 = 0.430423
        dc.w      $0D079; -a2 = -0.371319
        dc.w      $00001; no of left shifts
        dc.w      $040AB; b0 = 0.505226
        dc.w      $0D4E3; b1 = -0.336817
        dc.w      $06E24; -a1 = 0.860468
        dc.w      $0072F; b2 = 0.0561362
        dc.w      $0D09D; -a2 = -0.370199
        dc.w      $00001; no of left shifts
```

APPENDIX D

Listing of the MATLAB program which generates the filter paramter file for 68HC11.

```
function [cz,cp,qz,qp,ndout,ddout] = tf2cas11(nd,dd,fn,w)
% TF2CAS11 TRANSFER FUNCTION TO CASCADE FORM I
ASM FILE
%       TF2CAS11(ND,DD,FN,W) TRANSFORMS THE TF ND/DD
TO
%       FILE FN.
%       [CZ,CP,QZ,QP] = TF2CAS(ND,DD,FN) WILL RETURN
%       SECOND ORDER CASCADED SECTIONS
%       COEFFICIENTS AND QUANTIZED COEFFICIENTS
%       IF W IS GIVEN, THE FREQUENCY RESPONSE
BETWEEN
%       0 AND W WILL HAVE A MAXIMUM GAIN OF 1.  W IS
DEFAULT TO PI
%

if nargin<3,
   error('Invalid number of parameters.');
end

if nargin==3,
   w = pi;
end
% Find the poles and zeros
rnd = roots(nd);
rdd = roots(dd);
% Separate the poles and zeros into real and complex
zc = [];
zr = [];
pc = [];
pr = [];

for i=1:size(rnd),
   if real(rnd(i))==rnd(i),
      zr = [zr; rnd(i)];
   else
      zc = [zc; rnd(i)];
   end
end

for i=1:size(rdd),
   if real(rdd(i))==rdd(i),
      pr = [pr; rdd(i)];
   else
      pc = [pc; rdd(i)];
   end
end

% rearrange the poles and zeros
z = [sort(zc); sort(zr)];
p = [sort(pc); sort(pr)];

zn = length(z);
pn = length(p);
% If order is odd, add an extra pole and zero at 0
if rem(zn,2)==1,
   z = [z;0];
   zn = zn+1;
end

if rem(pn,2)==1,
   p = [p;0];
   pn = pn+1;
end

if zn~=pn,
   error('Numerator and denominator must have equal size.');
end

% separate the filter into sections
cz = [];
cp = [];
tz = 1;
tp = 1;
```

```
% section polynomial

for i=1:2:zn,
   sz = poly(z(i:i+1));
   sp = poly(p(i:i+1));
   w1 = [0.01:0.01:w];
   % maximum gain at w
   % get frequency response
   [h] = freqz(conv(tz,sz),conv(tp,sp),w1);
   sz = sz / max(abs(h));
   % scale filter according to maximum gain from 0 to w
   % the maximum gain is scaled to unity
   % Form new section
   tz = conv(tz,sz);
   tp = conv(tp,sp);
   cp = [cp;sp];
   cz = [cz;sz];
end

% Now make sure all the coefficients are within +/- 1
[m,n] = size(cp);
shift= [];
newcz = [];
newcp = [];
qz = [];
qp = [];
ndout = 1;
ddout = 1;
for i=1:m,
   shift1 = 0;
   cz1 = cz(i,:);
   cp1 = cp(i,:);
   % shift coefficent (/2) until it is smaller than +/- 1
   while (max(abs(cz1))>=1)|(abs(cp1(2))>=1)|(abs(cp1(3))>=1),
      shift1 = shift1+1;
      cz1 = cz1/2;
      cp1 = cp1/2;
   end
   shift = [shift;shift1];
   newcz = [newcz;cz1];
   newcp = [newcp;cp1];
   % now Quantize coefficient to 16 bits
   ndout = conv(ndout,quantize(cz1,16));
   ddout = conv(ddout,quantize(cp1,16));
   qz = [qz;quantize(cz1,16)];
   qp = [qp;quantize(cp1,16)];
end

%delete old file
eval(['delete ' fn]);

% Save no of sections
fprintf(fn,['      db     $0' int2hex(m) '; no of sections\n']);
fprintf(fn,'coef:\n');

% Store all the coefficients
for i=1:m,
   fprintf(fn,['      dw     $0' int2hex(newcz(i,1)*32768) '; b0 =
%g\n'],newcz(i,1));
   fprintf(fn,['      dw     $0' int2hex(newcz(i,2)*32768) '; b1 =
%g\n'],newcz(i,2));
   fprintf(fn,['      dw     $0' int2hex(newcp(i,2)*(-32768)) '; -a1 =
%g\n'],-newcp(i,2));
   fprintf(fn,['      dw     $0' int2hex(newcz(i,3)*32768) '; b2 =
%g\n'],newcz(i,3));
   fprintf(fn,['      dw     $0' int2hex(newcp(i,3)*(-32768)) '; -a2 =
%g\n'],-newcp(i,3));
   fprintf(fn,['      db     $0' int2hex(shift(i)) '; no of left shifts\n']);
end

% show the frequency response after the quantization
[h1,w] = freqz(nd,dd,1000);
[h2,w] = freqz(ndout,ddout,1000);
clg
```

```
subplot(211);
semilogy(w,abs(h1),w,abs(h2));
subplot(212);
plot(w,angle(h1),w,angle(h2));
pause
clg
subplot(211);
semilogy(w,abs(h1)-abs(h2));
subplot(212);
plot(w,angle(h1)-angle(h2));
pause
```