THE UNIVERSITY OF CALGARY

Limit Cycles and Sinusoidal Oscillations in Digital Systems

by

Stephen David Worthington

A THESIS

SUBMITTED TO THE FACULTY OF GRADUATE STUDIES

IN PARTIAL FULFILLMENT OF THE REQUIREMENTS FOR THE

DEGREE OF MASTER OF SCIENCE

DEPARTMENT OF ELECTRICAL AND COMPUTER ENGINEERING

CALGARY, ALBERTA

SEPTEMBER, 1992

ISBN 0-315-83296-7

THE UNIVERSITY OF CALGARY

FACULTY OF GRADUATE STUDIES

The undersigned certify that they have read, and recommend to the Faculty of Graduate Studies for acceptance, a thesis entitled, "*Limit Cycles and Sinusoidal Oscillations in Digital Systems*", submitted by Stephen David Worthington in partial fulfillment of the requirements for the degree of Master of Science.

LE Turner

Dr. L. E. Turner, Supervisor and Chairman
Department of Electrical and
Computer Engineering

LT Bruton

Dr. L. T. Bruton
Department of Electrical and
Computer Engineering

Dr. T Chen
Department of Electrical and
Computer Engineering

Dr. P. Kwok
Department of Computer Science

Date: December 14, 1992

Abstract

Recursive digital systems have become commonplace in recent years. This thesis examines recursive digital systems and oscillations that can occur within them. Two very different kinds of oscillations are discussed.

The first, limit cycles, are caused by non-linear quantization operations within recursive filters. Because these oscillations are undesirable, it is important to be able to bound their amplitude for any given filter implementation. A general bound on the amplitude of limit cycles is presented, as well as a method of creating digital filters that do not display limit cycles at the output.

The second form of oscillation discussed is the discrete time sinusoid. Discrete time sinusoids are extensively used in many aspects of modern communication systems and are thus of considerable importance. Several methods of generating digital approximations to discrete time sinusoids are presented, including a structure based on an analog prototype. The effects of finite precision arithmetic on each of the structures is analyzed. Based on a comparison of the different structures, one is selected for implementation and an audio range oscillator is constructed. The oscillator generates quadrature sinusoids with 0.4 Hz resolution over the frequency range 0 to 52 kHz.

Acknowledgments

The author would like to thank Dr. L. E. Turner for the advice and guidance provided during the course of his studies.

The support provided by the Alberta Microelectronic Centre, in form of financial support and access to design and fabrication facilities, is gratefully acknowledged by the author.

The author would also like to thank the Natural Sciences and Engineering Research Council, and the Department of Electrical and Computer Engineering whose financial support in the form of a Post Graduate Scholarship and teaching assistantships allowed the author to undertake his studies.

The author would also like to thank Cam Patterson of the Alberta Microelectronic Centre, and Peter Graumann of the Department of Electrical and Computer Engineering for their assistance in preparing the hardware prototype of the oscillator.

To my parents, who encouraged me to learn,

and Liza, who stood beside me while I did.

Table of Contents

List of Tables

List of Figures

List of Symbols

A	amplitude of sinusoid
$\mathbf{A}$	state transition matrix
a	coefficient
$\mathbf{B}$	state matrix
b	coefficient
b_0	phase accumulator wordlength
b_2	ROM output wordlength
$B(\omega)$	frequency response
C	capacitance
$\mathbf{C}$	state matrix
c_i	filter coefficients
d	state matrix
e	quantization error
$\mathbf{e}_n$	difference between ideal and quantized next state
f_0	frequency of oscillation
$f(\Lambda)$	Jacobean of **A**
F_{clk}	sample frequency
$f()$	probability distribution function
$g()$	probability distribution function
g_n	unit sample response
h	quantization interval
h_n	unit sample response sequence
$H(z)$	transfer function

$\hat{H}(z)$	non-ideal transfer function
I	integer (Chapter 3)
I	maximum amplitude of $i(n)$ (Chapter 4)
i	current
$i(n)$	sine output
$I(z)$	Input sequence
L	inductance
m	integer
m_i	multiplier coefficient value
$ms1$	oscillator coefficient
$ms2$	oscillator coefficient
N	phase increment
n	integer
$N(z)$	Numerator
$O(z)$	Output sequence
p_i	oscillator poles
$\mathbf{S}$	energy function matrix
s_n	time domain sequence
$S(z)$	z-transformed sequence
$\mathbf{T}$	eignenvectors of **A**
T	sample period
t	time
u_n	input sequence (Chapter 3)
$U(z)$	z-transformed input sequence
V	initial voltage

V	maximum amplitude of $v(n)$
v	voltage
$v(n)$	cosine output
$\mathrm{V}(\mathbf{x})$	Lyapunov function
$\mathbf{v}_n$	limit cycle at states
$w()$	transformation (Chapter 2)
$\mathbf{w}_n$	limit cycle at outputs (Chapter 3)
X	maximum magnitude of x
x	multiplier input
$\mathbf{x}_n$	current state
$\mathbf{x}_{n+1}$	next state
$\mathbf{x}_0$	initial state
y	multiplier output
$\hat{y}$	quantized multiplier output
y_n	output sequence
$Y(z)$	z-transformed output sequence
z	discrete complex frequency variable
α	multiplier coefficient
Δf_0	frequency error
$\Delta V(\mathbf{x})$	change in Lyapunov function from current state to next state
$\Lambda_{\mathbf{A}}$	diagonal matrix of eigenvalues of **A**
λ_M	eigenvalue of **A** with greatest modulus
ϕ	phase shift
ϕ'	phase shift
ρ	largest movement in state space due to quantization

μ_i	eigenvalue of **S**
μ_j	largest eigenvalue of **S**
$\Omega_0 T$	normalized sinusoidal frequency
$\Omega_0' T$	normalized sinusoidal frequency
ωT	frequency of oscillation
$\|\bullet\|_1$	l_1 norm
$\|\bullet\|_2$	$l_2$norm
$[\bullet]_Q$	non-linear quantization operation
$\langle\bullet\rangle$	vector or matrix formed by taking the absolute value or modulus of each element

Chapter 1

Introduction

Digital filters were envisioned by Norbert Weiner and N. Levinson as early as the 1940's, but the lack of suitable hardware prevented their use [1]. With the invention of the digital computer, as well as the enormous reductions in hardware cost that have occurred as a result of the VLSI revolution, digital filters have become readily available and digital filtering has become increasingly more important in many areas [2]. Control systems, ranging from those operating nuclear power plants to those controlling automobile engines employ digital filters [3, 4, 5]. Modern telephone systems are almost entirely digital, and radio systems are making the transition to digital operation as well [6]. What was once impractical and of theoretical interest only has become commonplace. It is because of this ever growing importance of digital filters that the research described in this thesis was undertaken.

1.1 Digital Filters

Before discussing the research, however, it is necessary to present some definitions in order to clarify terminology. A discrete time system is defined as one in which time[†] (the independent variable) is quantized [7]. Discrete time signals are defined only at discrete points in time. Such signals are represented as sequences of numbers. The signal level

[†] For this thesis, the independent variable is considered to be time. In general, it can be any quantity.

itself can take on a continuum of values. Digital systems are those in which both the independent variable and the value of the signal are quantized [7]. The signal is defined only for discrete values of time and can only take on a finite set of amplitude values. A digital filter is a computational process by which a digital signal is transformed into a second digital signal termed the output [7]. A restriction imposed on the discrete time filters that will be considered in this thesis is that they are linear and shift invariant. A linear shift invariant (LSI) filter can be completely characterized by its unit sample response, denoted by h_n. The output is given by the convolution summation

$$y_n = \sum_{k=-\infty}^{\infty} u_k h_{n-k} \tag{1.1}$$

where u_n represents the input sequence and y_n represents the output sequence [2].

The convolution summation above can be more conveniently dealt with using the z-transform. The z-transform of a sequence s_n is given by [2]

$$S(z) = \sum_{n=-\infty}^{\infty} s_n z^{-n}. \tag{1.2}$$

The z-transform allows the convolution summation above to be expressed as [2]

$$Y(z) = H(z)U(z). \tag{1.3}$$

The LSI filter is completely characterized by either *H(z)* or h_n. The z-transform allows the frequency response of a signal or system to be characterized. If the substitution $z = e^{j\Omega T}$ is made, the steady state sinusoidal frequency response can be determined from its z-transform [2]. Note that the unit sample response, and the z-transform, characterize only linear, shift invariant systems.

Another method of describing a digital filter is the state space representation [8]. The state of a system is a minimum set of variables such that if their values are known at some initial time, and all inputs are known from this time onwards, then the behaviour of the system is completely defined. A recursive, linear, shift invariant filter can be described using the equations

$$\begin{aligned} \mathbf{x}_{n+1} &= \mathbf{A}\mathbf{x}_n + \mathbf{B}u_n \\ y_n &= \mathbf{C}\mathbf{x}_n + du_n \end{aligned} \tag{1.4}$$

assuming that the filter has only a single input and a single output. The state is represented by $\mathbf{x}_n$, the output by y_n, and the input by u_n. Of particular interest is the **A** matrix, known as the state transition matrix. If the eigenvalues of this matrix, or the poles of the filter transfer function, are all inside the unit circle (that is they have a modulus less than one), then the filter is stable [8]. The expression

$$\mathbf{x}_{n+1} = \mathbf{A}\mathbf{x}_n \tag{1.5}$$

is known as the autonomous state equation and it completely describes the state transitions that occur under zero input conditions [9]. If the filter is stable, the z-transform can be determined from the state space representation according to the relation [8]

$$H(z) = d + \mathbf{C}(z\mathbf{I} - \mathbf{A})^{-1}\mathbf{B}. \tag{1.6}$$

Digital filters can be implemented in hardware or software [10]. When digital filters are implemented in software, the filter designer may have the luxury of using floating point arithmetic. This is not the case for many hardware implementations. Because floating point functional units require very large amounts of hardware, fixed point arithmetic may have to be used. Fixed point, finite precision arithmetic leads to more compact functional units, however, fixed point, finite precision arithmetic introduces its errors in the filter, and these errors must be understood and characterized.†

† Floating point arithmetic also introduces errors, however these are beyond the scope of this thesis.

1.2 Digital Sinusoids

Discrete time sinusoids are an important part of many digital systems. Such systems include modems, digital radio, satellite communications systems as well as frequency analyzers and other laboratory equipment [11]. This thesis examines the problem of generating discrete time sinusoids for use in fixed point, finite precision digital systems.

A discrete time sinusoid is completely defined by

$$y(n) = A\sin\left(n\Omega_0 T + \phi\right) \tag{1.7}$$

where A is the amplitude, $\Omega_0 T$ is the ideal normalized frequency and ϕ is the phase. Because this is a discrete time signal, only time is quantized. The amplitude, frequency and phase of the signal can take on a continuum of values. In practice, when the signal is generated by a digital system with a finite number of respresentable values, it is quantized in value and time. Thus, a digital approximation to a discrete time sinusoid is given by

$$y'(n) = A'\sin\left(n\Omega_0{}' T + \phi'\right) \tag{1.8}$$

where A' represents the amplitude, $\Omega_0' T$ the actual normalized frequency and ϕ' the actual phase. Note that these quantities can be considered as changing from sample to sample due to the effects of quantizing the discrete time sinusoid. The manner and amount in which these values vary depends upon the means used to approximate the discrete time sinusoid. There is no such thing as a perfect digital sinusoid. Because the signal levels are quantized, the digital signal will always be an approximation of a discrete time sinusoid.

1.3 Research Goals

The purpose of this thesis is to study oscillations as they relate to recursive digital systems implemented using finite precision fixed point arithmetic. Two very different kinds of oscillations are considered. The first, found in digital filters, are highly undesirable oscillations known as limit cycles. These are oscillations caused by the nonlinearities inherent in finite precision fixed point arithmetic. They can be particularly troublesome in communications and control systems, where an oscillatory output is highly undesirable and even dangerous. The research into limit cycles is therefore presented with a view towards eliminating them. The goal of this research is to determine a general limit cycle bound that is applicable to all digital filter implementations and to determine ways of eliminating limit cycles from digital filters.

The second form of oscillation considered is a sinusoidal oscillation. Whereas limit cycles are undesirable artifacts of finite precision arithmetic, digital sinusoids are an integral part of many digital systems. The goal of the research is to study different methods of generating sinusoids, and to evaluate each method in terms of speed, accuracy and amount of hardware required for implementation. Based on the comparison of different means of generating sinusoids, a compact, high precision oscillator is to be constructed.

1.4 Overview

A review of finite precision arithmetic is presented in Chapter 2. Methods by which finite precision arithmetic can be modeled, as well as the effects due to finite precision arithmetic are discussed. An in depth examination of limit cycles is presented in Chapter 3. An explanation of how limit cycles can occur, as well as a general bound on the magnitude of limit cycles is presented. This chapter also discusses ways in which digital filters can be constructed such that they are free from limit cycles at the output. The problem of generating digital sinusoids is addressed in Chapter 4. Several techniques, including a new oscillator structure, are presented. The implementation of a digital sinusoidal oscillator is described in Chapter 5. Lastly, Chapter 6 summarizes the results found in the course of the research and discusses the degree to which the research goals were met.

Chapter 2

Finite Precision Effects

Because of the reduced complexity of fixed point arithmetic, digital filters often use fixed point binary representations of signals and coefficients. When fixed point arithmetic is used, signals and coefficients can be represented with only a finite degree of precision. The fact that only a limited set of numbers can be used in the filter causes changes to the filter's transfer function, as well as changes to the signals associated with the filter. This chapter presents fixed point binary representations of numbers, and discusses the types of errors inherent in fixed point arithmetic and their effects of these errors on filter performance and operation.

2.1 Representations of Fixed Point Numbers

There are many ways of representing numbers. One familiar representation is the base 10 integer format. For example, when the number 108 is written, it is understood to mean

$$1\times10^2+0\times10^1+8\times10^0=108. \qquad (2.1)$$

This notation can also be used for binary, or base 2, representations. For example, the number 108 can be written as

$$1\times2^6+1\times2^5+0\times2^4+1\times2^3+1\times2^2+0\times2^1+0\times2^0=1101100. \qquad (2.2)$$

Thus, 1101100 and 108 can be used to represent the same number. This scheme is easily extended to include rational numbers. With base 10 representations, a decimal point is added to represent the fractional portion of the number. For example, 45.375 represents

$$4\times10^1+5\times10^0+3\times10^{-1}+7\times10^{-2}+5\times10^{-3}=45.375. \qquad (2.3)$$

Likewise, a binary point can be added to represent fractional portions of the number. The previous example is represented as

$$1\times2^5+0\times2^4+1\times2^3+1\times2^2+0\times2^1+1\times2^0+0\times2^{-1}+1\times2^{-2}+1\times2^{-3}=101101.011. \tag{2.4}$$

For this discussion, it is assumed that fixed point arithmetic is used. That is, for any particular register or operator within the digital filter, the location of the binary point is fixed. The binary point does not move or "float". Note that although the location of the binary point is fixed for any register or operator, different registers and operators may be scaled so that the binary point is in different locations.

So far, only positive representations of numbers have been discussed. In digital filter implementations, it is important to be able to represent both positive and negative numbers. For most binary number formats, an extra bit is added to indicate the sign of the number. There are two common ways of doing this. The simplest is to simply use the sign bit to indicate that the number represented is negative if it is set. For example, the number -3.5 is represented as

$$-1\times\left(0\times2^2+1\times2^1+1\times2^0+1\times2^{-1}\right)=1011.1. \tag{2.5}$$

This is known as signed magnitude representation. Note that there are two representations for zero with signed magnitude representation.

Perhaps the most common representation is that known as two's complement. In this scheme, the power of two associated with the most significant bit is given a negative value. For example, -3.5 would be written as

$$1\times-2^3+1\times2^2+0\times2^1+0\times2^0+1\times2^{-1}=1100.1. \tag{2.6}$$

This system is far more commonly used than signed magnitude because it has several advantages when implementing digital filters in hardware [10]. These include, simplicity of construction for adders and multipliers, unique representation of zero, and correct computation of machine representable sums even if intermediate overflows occur. All the digital systems discussed in this thesis will be assumed to use two's complement representations of numbers.

2.2 Coefficient Quantization

The coefficients of a filter are chosen to meet a frequency response specification. The ideal filter transfer function is a function of the filter coefficients and can be written as

$$H(z) = f\left(c_0, c_1, c_2 \ldots c_n\right) \tag{2.7}$$

where $c_0, c_1, c_2 \ldots c_n$ are the filter coefficients. When the actual filter is implemented, the coefficient values must be quantized to a representable finite precision value. This modifies the transfer function of the filter, causing pole and zero locations to be shifted. The transfer function of the finite precision implementation is given by

$$\hat{H}(z) = f\left(\left[c_0\right]_{Q,}, \left[c_1\right]_{Q,}, \left[c_2\right]_{Q,} \cdots \left[c_n\right]_{Q,}\right). \tag{2.8}$$

The transfer function of the finite precision implementation must be examined to see if it still meets the original specification. If the poles and zeroes have shifted to such a degree that the filter response is unacceptable, then the coefficient word length must be increased. It is also possible that coefficient quantization can cause poles to be located on or outside the unit circle. In this case, the filter is not stable and will not operate properly. Again, the coefficient word length must be increased, the quantization method changed, or a different filter structure must be used.

Consider the notch filter shown in Figure 2.1. This filter is designed to have a narrow notch at 0.15 radians in the frequency response. The ideal frequency response is shown in Figure 2.2. Also shown in this figure is the frequency response that would occur if the filter were implemented using coefficients that have been quantized to eight fractional bits. That is, the quantized coefficient is represented using a binary number with eight bits following the binary point. It can be seen that the actual finite precision transfer function differs noticeably from the ideal response in that the notch frequency has increased. If the quantized coefficient response is unacceptable, then the number of fractional bits in the coefficient must be increased. If the number of fractional bits is increased to 16, then the actual frequency response and the ideal frequency response do not differ by more than 0.1 dB over the entire spectrum, except for the range from 0.149 radians to 0.151 radians. Although the frequency response with 16 fractional bits more closely matches the ideal response, it does not match it exactly. Only if the quantized coefficients exactly match the ideal coefficients will the quantized transfer function exactly match the ideal transfer function. For any filter implementation, the number of fractional bits required depends upon the original filter specifications. The final implementation of the filter with finite precision coefficients must meet this original specification.

2.3 Signal Overflow

Overflow occurs when the resulting value of a summation of two numbers exceeds the maximum number that can be represented in the finite precision system. Overflows in finite word length adders can be dealt with in two ways, saturation and two's complement overflow. Saturation implies that if a sum exceeds the maximum representable value, it is simply replaced with the maximum representable value of the same sign. Two's complement overflow implies that if a sum exceeds the representable value, the most significant bits (i.e. those that exceed the wordlength) are simply ignored. Examples of two's complement and saturation truncation are depicted in Figure 2.3.

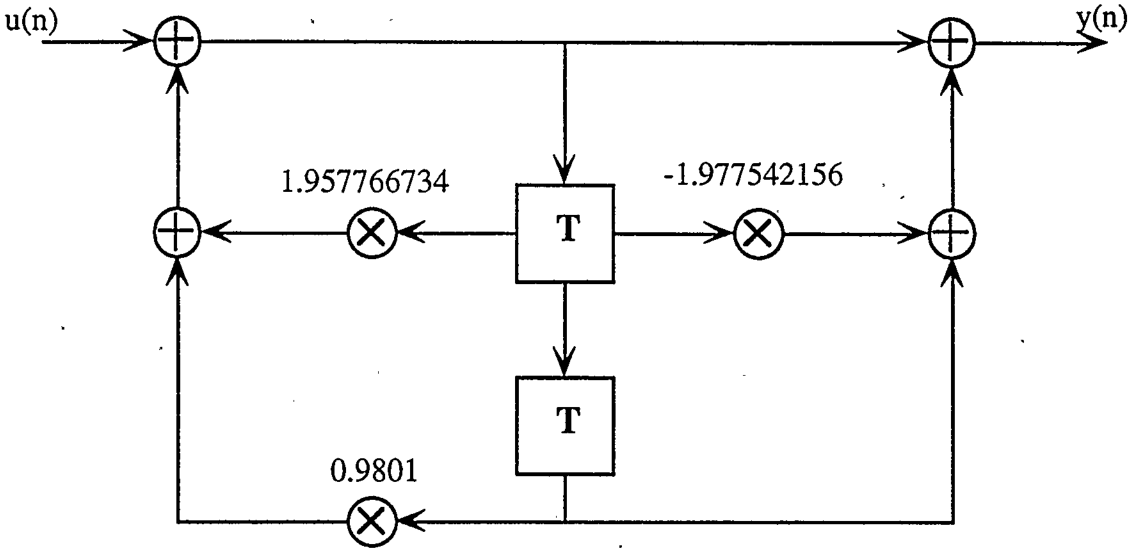

Figure 2.1 - Notch Filter

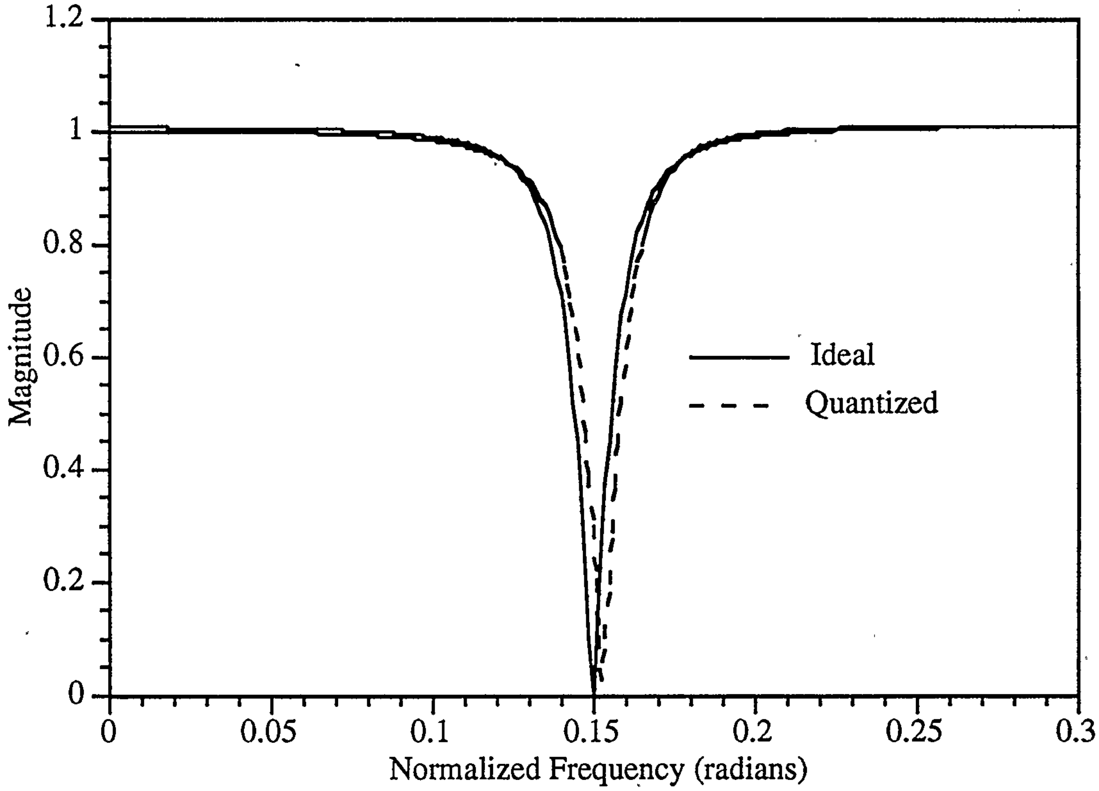

Figure 2.2 - Notch Filter Response (8 Bit Coefficients)

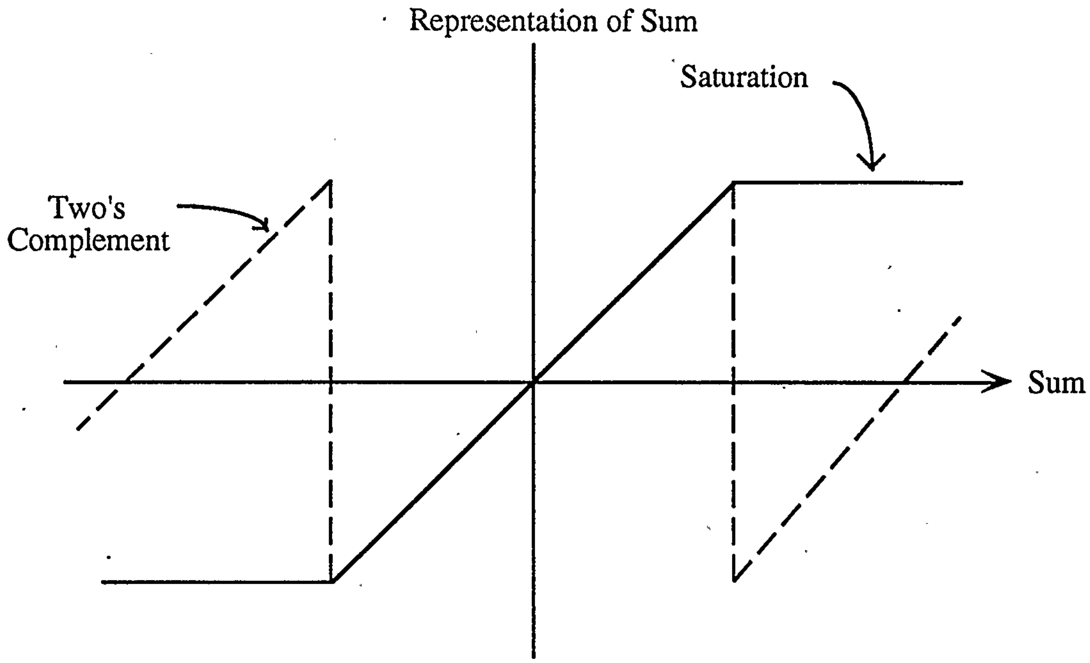

Figure 2.3 - Overflow Characteristics

These overflows can lead to large magnitude overflow oscillations, also termed overflow limit cycles, rendering the filter useless [12]. It is imperative that they be avoided. The simplest way of avoiding overflow oscillations is to ensure that overflow does not occur. One interesting feature of addition performed using two's complement overflow is that if the final summation is representable, then the addition can be performed correctly, even if overflows occur in intermediate operations [10]. Thus, it is not necessary to prevent overflows from occurring at the inputs of adders, if two's complement overflow is used. It is only necessary to prevent overflow from occurring at the points where signals are quantized. If an overflow error is quantized, it cannot be "unwrapped" or corrected at a later stage [2]. Thus, the signals entering quantizers cannot be allowed to overflow and for a system implemented using two's complement overflow such nodes represent the critical nodes in the system. The filter output is also considered a critical node, as overflows at the output cannot be allowed.

Some means of guaranteeing that overflows do not occur at the critical nodes within the filter must be found. One such method of doing so is to find a bound on the magnitude of signals entering critical nodes and to scale the signal so that the system is capable of representing such signals. If this is done, overflows cannot occur. One such bound is as follows: Recall that the output *y(n)* of a filter with unit sample response *g(n)* to an input sequence *u(n)* is given by

$$y(n)=\sum_{l=0}^{n} g(l)u(n-l). \tag{2.9}$$

If $g_i(n)$ is taken to be the unit sample response computed from the filter input to the critical node *i*, and *u(n)* is the filter input, a bound can be computed for the magnitude of the signal at the critical node. Let the input be bounded such that $|u(n)|<M$, then

$$\begin{aligned}|y_i(n)| &= \left|\sum_{l=0}^{n} g_i(l)u(n-l)\right| \\ &\le \sum_{l=0}^{n}|g_i(l)||u(n-l)| \\ &\le M\sum_{l=0}^{\infty}|g_i(l)|\end{aligned} \tag{2.10}$$

where $\sum_i |x(i)|$ is the l_1 norm, $\|x(i)\|_1$. For a stable filter, this summation will always converge. This computation can be performed from the input to every critical node and the signal scaled such that $M\times\max\left(\sum_{l=0}^{\infty}|g_i(l)|\right)$ does not exceed the value representable in the filter. Such a bound is guaranteed to avoid overflows at all critical nodes, regardless of the input signal. In practice, this bound is conservative, as signals of the type necessary to cause the output to approach its maximum are rare. In control theory, they are referred to as "bang-bang" inputs and must be carefully computed [12] and as such, are more contrived than practical.

2.4 Signal Quantization

When two binary numbers are multiplied together, to represent the product exactly will require a wordlength equal to the sum of the two input wordlengths, assuming that two's complement representation is used. Thus, if the two input words are eight bits long, then the product will require sixteen bits to be exactly represented. If signals were required to be exactly represented in a recursive filter, the wordlength would need to grow infinitely large, because of feedback within the filter. In a practical recursive filter implementation it is not possible to have arbitrarily large signal word lengths, so the product must be truncated in some fashion. One common method is to truncate the product such that it is represented with the same number of bits as the original input signal. Using this method, a 32 bit signal multiplied by a 16 bit coefficient is represented by a 32 bit product, even though the product requires 48 bits to be represented exactly. There are several means of truncating the product, namely rounding, magnitude truncation and two's complement truncation. Each method will be examined in turn.

When rounding is used, the product is quantized to the representable number that is closest to the infinite precision product. If the step size between values in the finite precision is given by h, then the greatest magnitude of error possible between the ideal and quantized product is given by $h/2$. In order to simplify discussion of quantization, let the operator $[\bullet]_Q$ represent the quantization operation. The operation of quantization by rounding can be written as

$$[x]_Q = h \times \mathrm{round}(x/h) \tag{2.11}$$

where *h* is the step size of the finite precision word and round(•) represents the operation of rounding to the nearest integer.

Another common type of quantization is magnitude quantization (also known as sign-magnitude quantization). Under this scheme, numbers are quantized to the nearest representable number that is smaller in magnitude than the ideal product. Using the notation introduced above, magnitude quantization is written as

$$[x]_Q = h \times \text{sign}(x)\text{floor}\left(|x/h|\right) \tag{2.12}$$

where floor(•) and sign(•) are defined as [9]

$$\text{floor}(x) = \text{greatest integer} \leq \text{x}$$
$$\text{sign}(x) = \begin{cases} 1, & x \geq 0 \\ -1, & x < 0. \end{cases} \tag{2.13}$$

The absolute value of the difference between the quantized product and the unquantized product will less than or equal to *h*.

The last type of quantization to be presented is two's complement truncation. This is the easiest to implement in hardware as it simply involves truncating the binary product at the desired wordlength and discarding the remainder of the product. The sign of the word is not considered during the truncation operation. The operation of two's complement truncation can be expressed as

$$[x]_Q = h \times \text{floor}(x/h). \tag{2.14}$$

The product is rounded towards minus infinity in all cases, thus the difference between the quantized product and the unquantized product will be in the range from 0 to *-h*. Rounding and two's complement truncation are very similar operations. Note that because

$$\text{round}(x) = \text{floor}\left(x + \tfrac{1}{2}\right) \tag{2.15}$$

and

$$\mathrm{floor}(x) = \mathrm{round}\left(x - \tfrac{1}{2}\right) \tag{2.16}$$

the operation of two's complement truncation is equivalent to performing rounding on a signal that has a constant of -1/2 added to it [12]. Also note that for all three types of quantization, the quantization operation will not cause the product to change sign.

2.4.1 Modeling Quantization at Multipliers

The effect of quantization due to multiplication can be exactly modeled as an ideal multiplier followed by an additive error signal [10]. This is shown below in Figure 2.4.

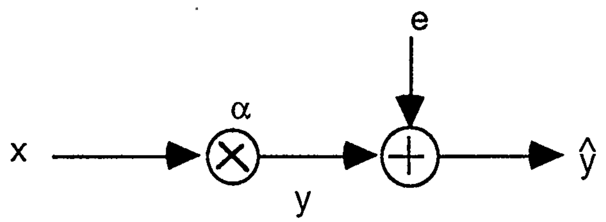

Figure 2.4 - Quantizing Multiplier Model

The error e associated with each x is a deterministic function of x. That is, a given value of x will always produce the same value of e, provided α remains constant. Because x represents a signal in a fixed point finite word length digital filter, it can only take on a limited number of values. This means that e and y will only take on a finite number of values. For this discussion, it is assumed that x is an integer value. Because the systems in question utilize fixed point arithmetic, this assumption can be made with no loss of generality. If x_l is the lowest value x can have and x_u is the upper value, then the number of possible values x can take is given by

$$R_x = x_u - x_l + 1. \tag{2.17}$$

Assume that α is less than one. Any multiplier can be represented by an integer multiply (which can be computed without error) summed with a quantizing fractional multiplier. In this case, the range of values *y* can take is given by

$$\alpha x_l \leq y \leq \alpha x_u. \tag{2.18}$$

The number of different possible values *e* can have is determined by α. Let

$$\alpha = \frac{a}{b} \qquad a, b \in \text{Integer} \tag{2.19}$$

where *a*<*b* and *a* and *b* do not have any common factors. Note that while an ideal filter implementation may have coefficients which are irrational numbers, these coefficients will be quantized to some rational value and thus any practical finite precision filter can have its coefficients expressed as above. Let the quantization operation be written as before

$$\hat{y} = \left[\frac{a}{b}x\right]_Q. \tag{2.20}$$

The output of the multiplier before quantization can be written as

$$I + \frac{m}{b}, \qquad 0 \leq m < b \tag{2.21}$$

where *I,m,b* are integers. This is an integer added to a rational fraction. Any unquantized product in a fixed point, finite word length system can be expressed this way. Thus,

$$\hat{y} = \left[\frac{a}{b}x\right]_Q = \left[I + \frac{m}{b}\right]_Q. \tag{2.22}$$

For some common quantization schemes, specifically rounding and two's complement truncation, the quantization operation depends only on the fractional part of the number and not the integer part. Because the quantization operation depends only on the fractional part of the product,

$$\hat{y} = \left[I + \frac{m}{b}\right]_Q = I + \left[\frac{m}{b}\right]_Q. \tag{2.23}$$

Thus, the value of the error, e, can be written as

$$\begin{aligned} e &= \hat{y} - y \\ &= \left[I + \frac{m}{b}\right]_Q - \left(I + \frac{m}{b}\right) \\ &= I + \left[\frac{m}{b}\right]_Q - \left(I + \frac{m}{b}\right) \\ &= \left[\frac{m}{b}\right]_Q - \frac{m}{b}. \end{aligned} \tag{2.24}$$

Since the quantization operation always yields the same value for any given value of m, and $0 \le m < b$ there can be at most b different values for the error e. If values in the range $0 \le \frac{m}{b} < 1$ are all quantized to the same value, then there will be exactly b unique values for e. For two's complement truncation, $\left[\frac{m}{b}\right]_Q = 0$ and there will be b unique values that e can take. For rounding, the values over the range $\frac{-1}{2} \le \frac{m}{b} < \frac{1}{2}$ will all be quantized to the same value. Thus, for rounding, there will be b unique values that e can take. With magnitude quantization, the quantization operation depends on the sign of the number. If the number range is divided into positive and negative sections, it can be shown using the arguments above that there are b distinct error values possible for each section. Thus, for the entire range of values there can be $2b$ distinct values of e.

2.4.2 Modeling Quantization Errors as Random Noise

Although the quantization errors are a deterministic function of the input to the multiplier, and thus statistically related, they can, under some circumstances, be modeled as uncorrelated random noise. Let the input to the multiplier be a random variable X with a discrete probability distribution $f(x)$. Note that X is a discrete random variable because the filter is a fixed point, finite precision system, and there is a finite set of values that X can take. Nothing is assumed about $f(x)$ at this point. Since the error due to quantization e is a function of x , it can be written as $e = w(x)$. This is a many-to-one transformation. There are only b different values for e and it can be shown that $w(x) = w(x+b)$ for two's complement truncation and rounding as follows:

$$\begin{aligned} w(x) &= \left[\frac{a}{b}x\right]_Q - \frac{a}{b}x \\ &= \left[I + \frac{m}{b}\right]_Q - \left(I + \frac{m}{b}\right) \\ &= I + \left[\frac{m}{b}\right]_Q - I - \frac{m}{b} \\ &= \left[\frac{m}{b}\right]_Q - \frac{m}{b} \end{aligned} \tag{2.25}$$

$$\begin{aligned} w(x+b) &= \left[\frac{a}{b}(x+b)\right]_Q - \frac{a}{b}(x+b) \\ &= \left[I + a + \frac{m}{b}\right]_Q - \left(I + a + \frac{m}{b}\right) \\ &= I + a + \left[\frac{m}{b}\right]_Q - I - a - \frac{m}{b} \\ &= \left[\frac{m}{b}\right]_Q - \frac{m}{b} \\ &\therefore w(x) = w(x+b). \end{aligned} \tag{2.26}$$

This can be used to derive the probability distribution for e. The complete set of values that e can take is given by

$$\begin{aligned} e_0 &= w(x_0) \\ e_1 &= w(x_1) \\ e_2 &= w(x_2) \\ &\vdots \\ e_{b-1} &= w(x_{b-1}). \end{aligned} \tag{2.27}$$

Recall that e can take on only b distinct values. Note that in general $e_0 < e_1 < \ldots e_{b-1}$ is not true. The probability that any given e will occur is equal to the sum of probabilities that an x yielding this e will be selected. Thus,

$$
\begin{aligned}
g(e_0) &= \sum_{n=-\infty}^{\infty} f(x_0 + nb) \\
g(e_1) &= \sum_{n=-\infty}^{\infty} f(x_1 + nb) \\
g(e_2) &= \sum_{n=-\infty}^{\infty} f(x_2 + nb) \\
&\vdots \\
g(e_{b-1}) &= \sum_{n=-\infty}^{\infty} f(x_{b-1} + nb).
\end{aligned}
\tag{2.28}
$$

It can be seen that the shape of *g(e)* depends on the shape of *f(x)*, however simulations have shown that some statements can be made about the distribution of *g(e)* [12]. If the frequency spectrum of *X* is relatively wide and the probability distribution function *f(x)* is broad relative to the quantization step size so that several quantization levels are crossed between samples of *X*, then *g(e)* will be uniformly distributed over its possible range [12]. These are reasonable conditions for most practical systems. This model breaks down only when *X* has a limited dynamic range, or an unusual probability distribution function. In most cases, the quantization errors will appear as a uniformly distributed discrete probability distribution.

2.4.3 Modeling Quantization Effects at the Output

It was shown that the effects of quantization can be modeled as random noise sources added to the filter at the quantizing nodes. If the effects of noise sources due to quantizing multipliers can be referred to the output, then they can be compared to the output signal. Two methods of doing this will be considered.

In the first method, it is assumed that the quantization errors are white noise that is added to the system. White noise is defined as a signal with a zero mean and a uniform power spectral density function. The quantization noise is assumed to be uniformly distributed over some range [12]. Recall that when rounding is used, the noise will be uniformly distributed over the range $-h/2$ to $h/2$. When magnitude quantization is used, this range will be from -h to h. Two's complement truncation does not meet the requirement of having a zero mean as the errors are uniformly distributed over the range -h to 0. The noise power, or variance, associated with a rounding quantizer can be shown to be $h^2/12$, while that associated with a magnitude truncation quantizer is $h^2/3$ [12]. According to [12], the output noise power due to the white noise from the quantizers is given by

$$\begin{aligned}\sigma_{out}^2 &= \sigma_e^2 \sum_i \|\mathrm{g}_i\|_2^2 \\ &= \sigma_e^2 \sum_i \left(\sum_{k=0}^{\infty} g_i^2(k) \right)\end{aligned} \tag{2.29}$$

where σ_e^2 represents the noise power at each quantizer and g_i represents the unit sample response from quantizer i to the output. The operator $\|\bullet\|_2$ represents the l_2 norm which is defined as

$$\|\mathrm{g}\|_2 = \sqrt{\sum_{k=0}^{\infty} g^2(k)}\,. \tag{2.30}$$

Thus, the total noise power expected at the output of the filter is equal to the sum of the squares of the l_2 norms from each quantizer to the output, multiplied by the noise power of the quantizers. It can be seen that a filter implemented using rounding quantization will have an output noise power that is expected to be only about a quarter of that found in the same filter implemented using magnitude quantization. The case of a filter that uses two's complement truncation is somewhat different. The noise power of two's complement truncation quantizers is equal to that of rounding quantizers, yet noise due to such quantizers does not have a zero mean. The total set of quantization errors due to two's complement truncation is a subset of those due to magnitude quantization, however. Thus, it is expected that the noise at the output due to two's complement truncation quantizers will be somewhere between that of rounding and that of magnitude quantization.

The estimate of output noise due to quantization is based on several assumptions about the nature of the signals. In particular, it is assumed that the quantization noise has a uniform distribution and that noise sources are uncorrelated. Because it is not possible to guarantee that this is true in all cases, the above estimate does not represent a guaranteed bound on the output noise. A guaranteed bound can be determined by considering the convolution summation of the filter response and the error signals. The signal at the output of the filter due to a single quantizer is simply the convolution summation of the quantization error and the unit sample response from the quantizer to the output. This is stated explicitly as

$$s_i(n) = \sum_{j=0}^{n} g_i(n-j)e_i(j) \tag{2.31}$$

where $s_i(n)$ is the output due to quantizer i, g_i is the unit sample response from quantizer i to the output and $e_i(n)$ is the error signal due to quantizer i. The magnitude of the output is bounded as follows:

$$\begin{aligned} |s_i(n)| &\le \sum_{j=0}^{n} |g_i(n-j)e_i(j)| = \sum_{j=0}^{n} |g_i(n-j)||e_i(j)| \\ &\le e\sum_{j=0}^{\infty} |g_i(j)| \end{aligned} \tag{2.32}$$

where e represents the bound on the quantizer error. For rounding, this bound will be $h/2$ while for both magnitude quantization and two's complement truncation it will be h. Because the system is a linear, shift invariant system, the bound on the output due to all quantization noise sources will be the sum of the individual bounds

$$|s(n)| \le e \sum_{i} \sum_{j=0}^{\infty} |g_i(j)|. \tag{2.33}$$

The above expression can be more compactly written as

$$|s(n)| \le e \sum_{i} \|\mathbf{g}_i\|_1 . \tag{2.34}$$

This bound is more pessimistic than the L_2 norm bound, and tends to give overly conservative estimates.

In this chapter, finite precision arithmetic was presented as were the effects of finite precision arithmetic. Ways in which these effects can be characterized were presented. In the next chapter, a special type of finite precision effect, the limit cycle, is presented and analyzed. Ways of eliminating and avoiding limit cycles are discussed and the bounds determined for limit cycles are compare to those determined for noise.

Chapter 3

Limit Cycles

In the previous chapter, finite precision arithmetic was introduced and it was shown that signal quantization can be modeled as noise that appears at the filter output. Product quantization can have other effects in recursive digital filters, including low level oscillations, termed limit cycles. Limit cycle oscillations can be particularly troublesome because they do not decay to zero and they are particularly difficult to characterize due to the non-linear operation of the quantization operations. Much work has been done to bound the amplitude of limit cycles [9, 13, 14, 15, 16, 17, 18, 19, 20]. Unfortunately, most of these bounds are for specific structures, such as wave digital filters, or are limited to second order sections. This chapter describes how limit cycles can occur in recursive digital filters. A general bound on the amplitude of limit cycles is derived and is compared to an earlier bound derived by Yakowitz and Parker [21] as well as the l_1 norm. Lastly, a method for designing filters that do not exhibit limit cycles at the output is presented.

3.1 How Limit Cycles Occur

Limit cycles are undesirable self sustaining oscillations caused by quantization operations within the filter. Limit cycles are different from the filter's response to a driving function. A stable filter's response to a driving function may be oscillatory, but it will eventually decay to zero after the driving function has been removed. Limit cycle oscillations on the other hand, continue indefinitely, even after the driving function has been removed.

Consider the filter shown in Figure 3.1. The filter is assumed to be at rest initially. If the input sequence to this filter is given by $u(n)=\{5,0,0,0,\ldots\}$, then the ideal output will be as shown in Figure 3.2. The envelope of the ideal response approaches zero as n approaches ∞. If the filter is implemented using finite precision arithmetic (integer arithmetic with two's complement truncation at the output of multipliers), then the output sequence will be as shown in Figure 3.2. It can be seen that the output is periodic with a period greater than one, not decaying and that the oscillations will continue indefinitely. The fact that the oscillations are periodic may be verified by examining the state of the filter over time. If the filter, under zero input conditions, reaches the same state twice, then the filter's response must be periodic. If the input sequence is changed to $u(n)=\{20,0,0,0,\ldots\}$, then the output sequences will be as shown in Figure 3.3. The ideal response is four times that of the previous input, while the output of the finite precision filter is radically different. Clearly, the non-linear nature of the quantization operations make limit cycles difficult to predict.

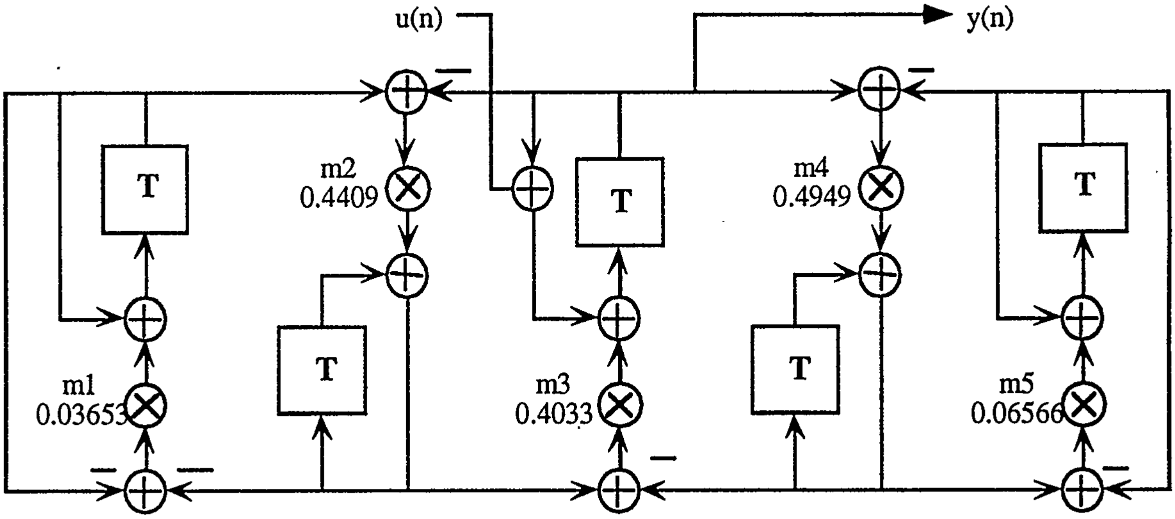

Figure 3.1 - Fifth Order LDI Filter

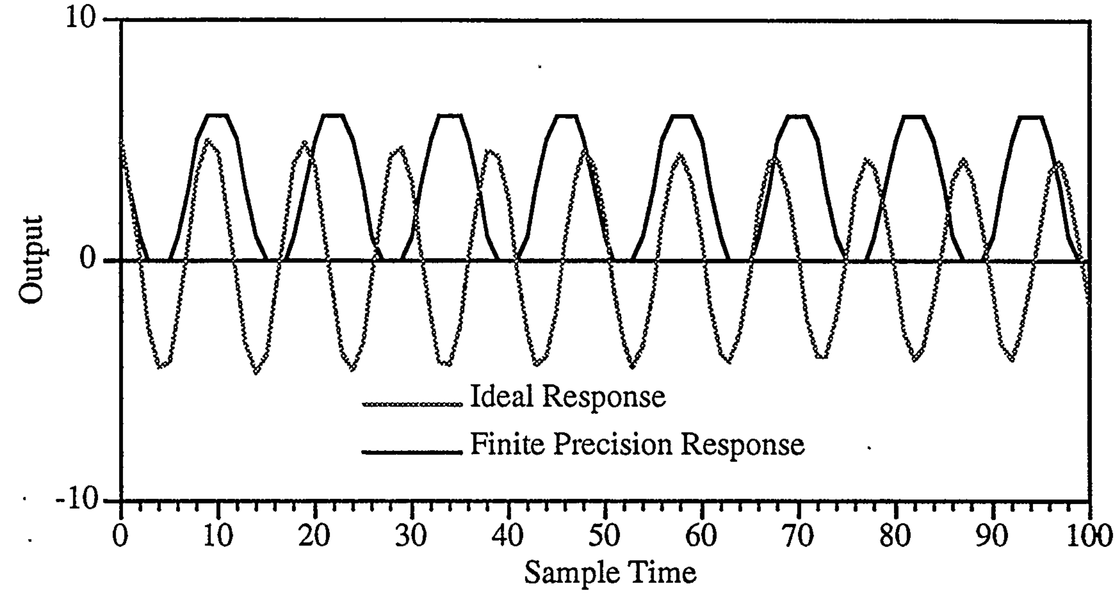

Figure 3.2 - Filter Output

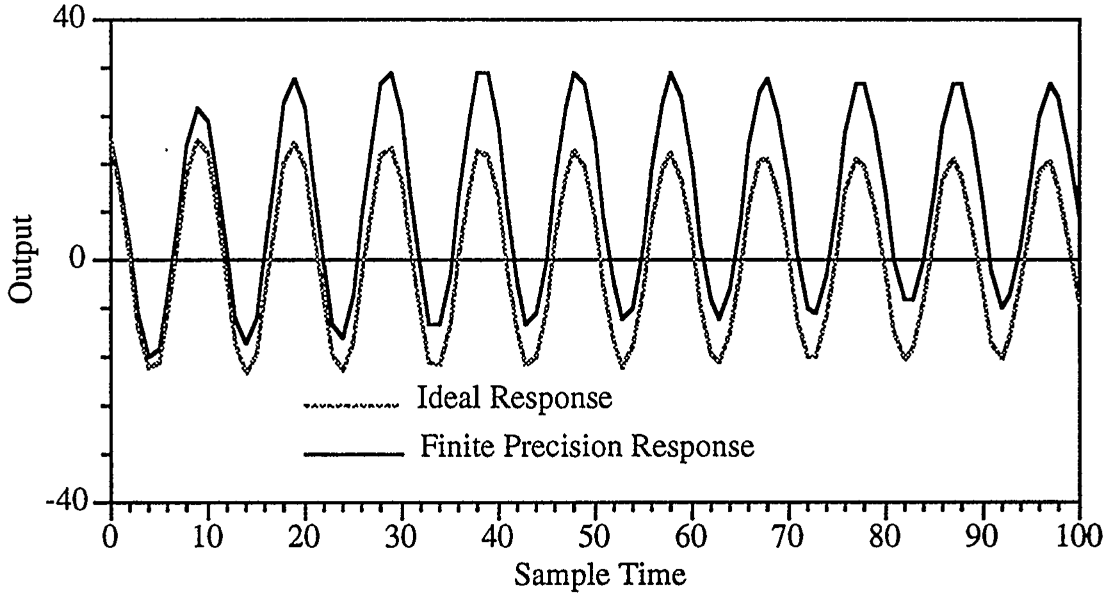

Figure 3.3 - Filter Output

3.2 Energy Functions and Limit Cycles

According to classical mechanical theory, a vibrating system is stable if the total energy in the system is decreasing until an equilibrium level is reached [3]. For physical systems, the amount of energy in a system is a real, measurable quantity. This is also true for simple electrical systems, such as passive filters. For purely mathematical systems, such as digital filters, there is no energy to measure, at least, not in a physical sense. Thus, while a digital filter may be constructed based on an analog prototype, the operation of the filter cannot be expressed in terms of energy like the prototype. The Russian mathematician, A.M. Lyapunov [3] introduced the concept of the Lyapunov function, a fictitious measure of energy, to describe mathematical systems in terms similar to physical systems. This allows the concepts of energy to be extended to purely mathematical systems. In fact, energy as defined for physical systems, represents a Lyapunov function for the system. In this discussion, when the term energy is applied to digital filters, it is a reference to the fictitious energy defined by the Lyapunov function for the filter in question.

A Lyapunov function is defined as a scalar function $V(\mathbf{x})$ which is positive definite and $V(\mathbf{x}_{n+1}) - V(\mathbf{x}_n)$ is negative definite for all n., where $\mathbf{x}$ is the state of the discrete time system [3]. If such a function exists, then the system is stable. Conversely, if a LSI system is stable, then a Lyapunov function must exist for it [22].

Consider a recursive discrete time filter under zero input conditions. The state transitions can be completely characterized by the autonomous state equation

$$\mathbf{x}_{n+1} = \mathbf{A}\mathbf{x}_n . \tag{3.1}$$

Provided **A** has unique eigenvalues, a Lyapunov function can be defined as [9]

$$V(\mathbf{x}) = \left\| \mathbf{T}^{-1}\mathbf{x} \right\|_2 \qquad (3.2)$$

where $\|\bullet\|_2$ is the Euclidean norm and **T** is the matrix of eigenvectors of **A** and **x** represents the state of the system. The function defined by (3.2) is positive definite. As well, $V(\mathbf{x}_{n+1}) - V(\mathbf{x}_n)$, is negative definite, as follows:

$$\begin{aligned} V(\mathbf{x}_{n+1}) &= V(\mathbf{A}\mathbf{x}_n) = \left\| \mathbf{T}^{-1}\mathbf{A}\mathbf{x}_n \right\|_2 \\ &= \left\| \mathbf{T}^{-1}\mathbf{A}\mathbf{T}\mathbf{T}^{-1}\mathbf{x}_n \right\|_2 \\ &= \left\| \Lambda_A \mathbf{T}^{-1}\mathbf{x}_n \right\|_2 \\ &\le \left\| \Lambda_A \right\| \left\| \mathbf{T}^{-1}\mathbf{x}_n \right\|_2 = |\lambda_M| V(\mathbf{x}_n) \\ &< V(\mathbf{x}_n), \qquad \text{if } |\lambda_M| < 1 \end{aligned} \qquad (3.3)$$

where Λ_A is the diagonal matrix of eigenvalues of **A**, and λ_M is the eigenvalue of **A** with the greatest modulus.

The quantization operations inherent in digital filters can affect the energy in the filter. Quantization causes the state of the filter to change from the ideal state to some representable state. Because there is an energy level associated with each point in the state space, this change in the state corresponds to a change in energy level. The change in energy level can be positive or negative, and this depends on both the filter's state and the type of quantization used. Thus, while an ideal stable filter will always have a decreasing level of energy, the finite precision version may have an increasing energy level, because of the effects of quantization. Figure 3.4 shows the energy levels as defined by (3.2) in response to the input sequence $u(n) = \{20,0,0,0,\ldots\}$ for the ideal and finite precision versions of the filter shown in Figure 3.1. It can be seen that the ideal version has an amount of energy that asymptotically approaches zero while the finite precision version has an energy level that increases, and then oscillates indefinitely.

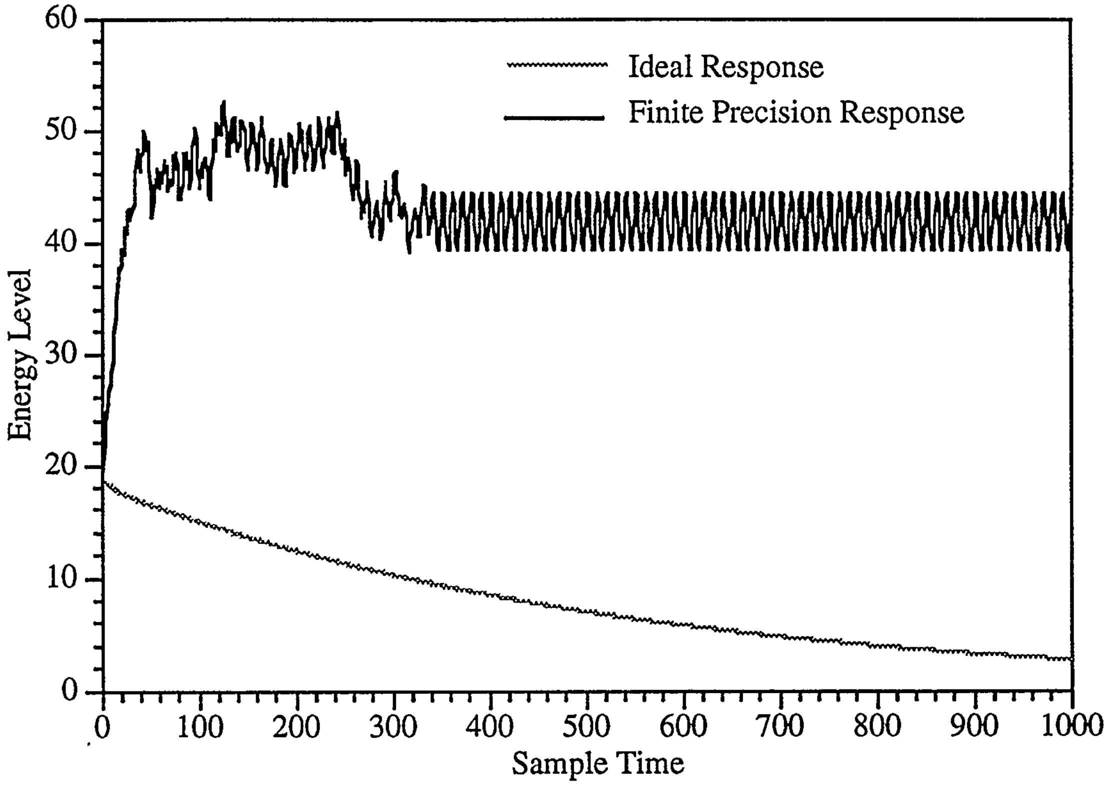

Figure 3.4 - Energy versus Time

With $V(\mathbf{x})$ defined in (3.2), it can be shown, that the maximum change in energy due to quantization is given by [9]

$$\Delta e = \sqrt{\mu_j}\ \rho \tag{3.4}$$

where μ_j is the largest eigenvalue of the matrix $\left(\mathbf{T}^{-1}\right)^{*}\mathbf{T}^{-1}$, and ρ is the length of the largest movement in state possible due to quantization. Since limit cycles can only occur in regions where quantization can increase the energy in the filter, limit cycles will be confined to the region where the change in energy due to quantization is as large as or larger than the loss in energy from one state to the next in the ideal filter. Outside this region, the filter's loss in energy will be greater than any possible addition of energy due to quantization and limit cycles cannot occur because the energy is always decreasing. Stated mathematically, limit cycles can only occur in the region where

$$V(\mathbf{x}_n) - V(\mathbf{x}_{n+1}) \le \Delta e \tag{3.5}$$

In order to simplify notation, rewrite $V(\mathbf{x})$ as

$$V(\mathbf{x}) = \sqrt{\mathbf{x}'\mathbf{S}\mathbf{x}} \tag{3.6}$$

where $\mathbf{S} = \left(\mathbf{T}^{-1}\right)^{*}\mathbf{T}^{-1}$. The change in energy from one state to the next, can then be written as

$$\begin{aligned} V(\mathbf{x}_n) - V(\mathbf{x}_{n+1}) &= V(\mathbf{x}_n) - V(\mathbf{A}\mathbf{x}_n) \\ &= \sqrt{\mathbf{x}'\mathbf{S}\mathbf{x}} - \sqrt{\mathbf{x}'\mathbf{A}'\mathbf{S}\mathbf{A}\mathbf{x}} \end{aligned} \tag{3.7}$$

Thus, limit cycles can only occur in the region where

$$\sqrt{\mathbf{x}'\mathbf{S}\mathbf{x}} - \sqrt{\mathbf{x}'\mathbf{A}'\mathbf{S}\mathbf{A}\mathbf{x}} \le \Delta e. \tag{3.8}$$

Consider a second order system with two's complement quantizers located at the states and a state transition matrix given by

$$\mathbf{A} = \begin{vmatrix} 0.6797 & 0.5546 \\ -0.5590 & 0.9922 \end{vmatrix}. \tag{3.9}$$

The maximum length of movement in the state space due to quantization is $\rho = \sqrt{2}$ [9], while the greatest eigenvalue of $\mathbf{S}$ has a modulus equal to 1.3901. Thus, the greatest change in energy due to quantization is equal to 1.67. Figure 3.5 shows a contour plot of $V(\mathbf{x}_n) - V(\mathbf{x}_{n+1})$ for the region around the origin. The dotted line is the curve corresponding to the contour $V(\mathbf{x}_n) - V(\mathbf{x}_{n+1}) = 1.67$. Any limit cycles that occur, must occur within this region. Unfortunately, while Δe is easy to compute, it is difficult to determine the curve where $\sqrt{\mathbf{x}'\mathbf{S}\mathbf{x}} - \sqrt{\mathbf{x}'\mathbf{A}'\mathbf{S}\mathbf{A}\mathbf{x}} \le \Delta e$. This can be done graphically for second order systems, but is computationally difficult to obtain for higher order systems.

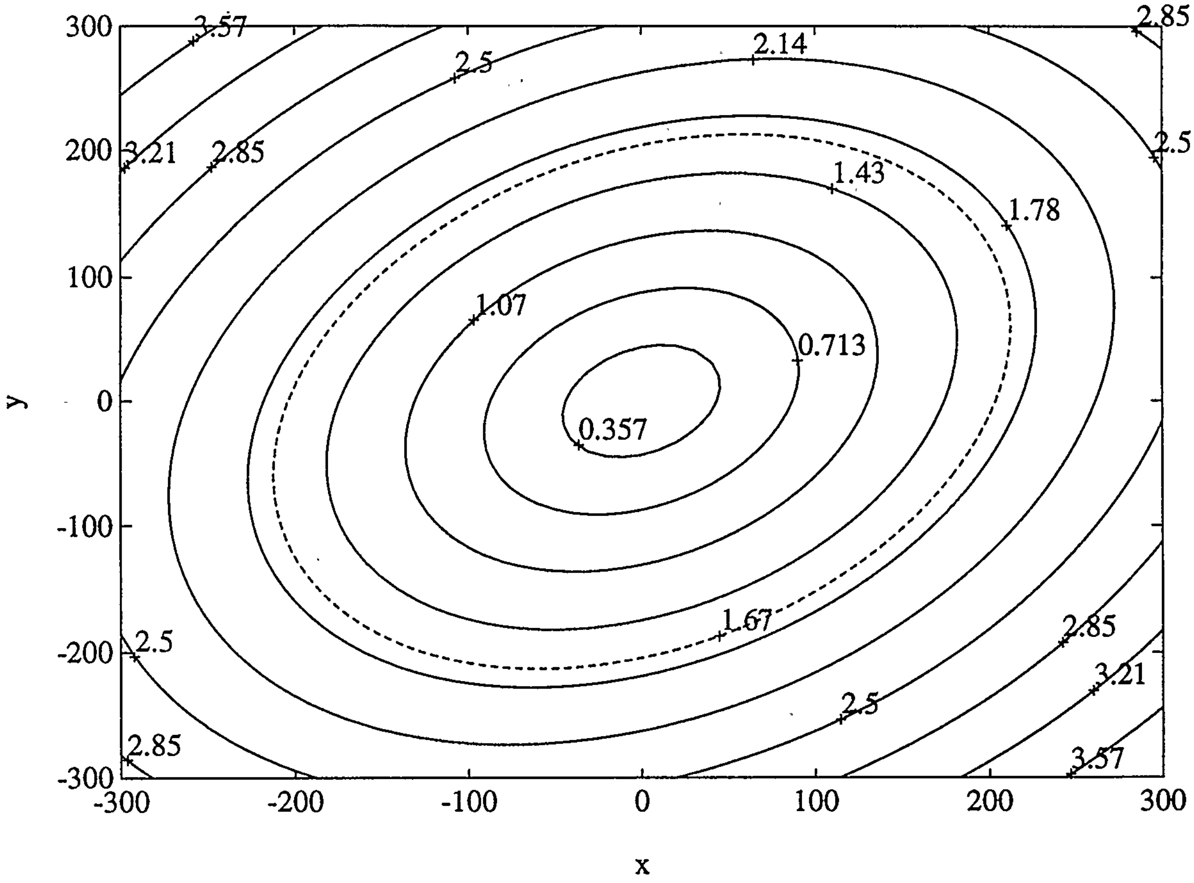

Figure 3.5 - Change in V(x) versus State for Second Order Filter

3.3 A General Limit Cycle Bound

A linear shift invariant digital filter can be described by the state difference equations [8]

$$\begin{aligned} \mathbf{x}_{n+1} &= \mathbf{A}\mathbf{x}_n + \mathbf{B}u_n \\ y_n &= \mathbf{C}\mathbf{x}_n + du_n. \end{aligned} \tag{3.10}$$

The quantization of any signal in the digital filter implementation is a non-linear operation which can be exactly modeled as an additive error input [10]. That is, the quantization of signal $g(n)$ results in a digital signal $\hat{g}(n)$ where

$$\hat{g}(n) = Q[g(n)] \tag{3.11}$$

or

$$\hat{g}(n) = g(n) + e_g(n), \qquad |e_g(n)| < h \tag{3.12}$$

where h is the quantization step size. Thus, a finite precision recursive digital filter can be modeled as a linear shift invariant system with multiple inputs;

$$\begin{aligned} \hat{\mathbf{x}}_{n+1} &= \mathbf{A}\hat{\mathbf{x}}_n + \mathbf{B}u_n + \mathbf{B}_e\mathbf{E}_n \\ y_n &= \mathbf{C}\hat{\mathbf{x}}_n + du_n + \mathbf{D}_e\mathbf{E}_n \end{aligned} \tag{3.12}$$

where $\mathbf{E}_n$ is a vector of additive error sources, $\mathbf{B}_e$ is a matrix describing the gain from the error sources to the states, and $\mathbf{D}_e$ is a similar matrix describing the gain from the error sources to the output. Since only the signals due to the quantization operations are of interest, the principle of superposition can be applied and the input u_n can be set to zero. If the filter input is set to zero, the state transition equation becomes

$$\hat{\mathbf{x}}_{n+1} = \mathbf{A}\hat{\mathbf{x}}_n + \mathbf{B}_e\mathbf{E}_n. \tag{3.14}$$

If the filter is assumed to be initially at rest (i.e $\hat{\mathbf{x}}_0 = \overline{0}$), the state vector $\hat{\mathbf{x}}_n$ is given by

$$\hat{\mathbf{x}}_n = \sum_{j=1}^{n} \mathbf{A}^{n-j}\mathbf{B}_e\mathbf{E}_{j-1} \qquad n > 0. \tag{3.15}$$

In this discussion, the same notation given in [21] is used. If $\mathbf{W}$ and $\mathbf{W}'$ are both real vectors, then $\mathbf{W}<\mathbf{W}'$ means that each element of $\mathbf{W}$ is less than the corresponding element of $\mathbf{W}'$. If $\mathbf{W}$ is a matrix, then $\langle\mathbf{W}\rangle$ denotes the matrix formed by taking the absolute value or modulus of each element in $\mathbf{W}$. If $\mathbf{E}_n$ is bounded such that $\langle \mathbf{E}_n \rangle \le \mathbf{E}$, then

$$\langle \mathbf{x}_n \rangle = \left\langle \sum_{j=1}^{n} \mathbf{A}^{n-1-j} \mathbf{B}_e \mathbf{E}_n \right\rangle \le \sum_{j=0}^{\infty} \langle \mathbf{A}^j \rangle \langle \mathbf{B}_e \rangle \mathbf{E}. \tag{3.16}$$

Let Λ_A be a diagonal matrix of distinct eigenvalues of $\mathbf{A}$ and $\mathbf{T}$ be a matrix of the eigenvectors such that

$$\mathbf{AT} = \mathbf{T}\Lambda_A \tag{3.17}$$

If this is the case, then

$$\sum_{i=0}^{\infty} \mathbf{A}^i = \mathbf{T}\left(\sum_{i=0}^{\infty} \Lambda_A{}^i \right)\mathbf{T}^{-1} \tag{3.18}$$

and

$$\sum_{i=0}^{\infty} \langle \mathbf{A}^i \rangle \le \sum_{i=0}^{\infty} \langle \mathbf{T} \rangle \langle \Lambda_A{}^i \rangle \langle \mathbf{T}^{-1} \rangle = \langle \mathbf{T} \rangle \left(\sum_{i=0}^{\infty} \langle \Lambda_A{}^i \rangle \right) \langle \mathbf{T}^{-1} \rangle \tag{3.19}$$

Because Λ_A is a diagonal matrix of eigenvalues, it can be shown that

$$\sum_{i=0}^{\infty} \langle \Lambda_A{}^i \rangle = \begin{bmatrix} (1-|\lambda_1|)^{-1} & 0 & 0 & \cdots & 0 \\ 0 & (1-|\lambda_2|)^{-1} & 0 & \cdots & 0 \\ 0 & 0 & (1-|\lambda_3|)^{-1} & \cdots & 0 \\ \vdots & \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & 0 & \cdots & (1-|\lambda_N|)^{-1} \end{bmatrix} \tag{3.20}$$

where N is the system order. In order to simplify notation, let

$$f(\Lambda_A) = \sum_{i=0}^{\infty} \langle \Lambda_A{}^i \rangle \tag{3.21}$$

Thus,

$$\sum_{j=0}^{\infty} \langle \mathbf{A}^j \rangle \le \langle \mathbf{T} \rangle f(\Lambda_A) \langle \mathbf{T}^{-1} \rangle. \tag{3.22}$$

Combining (3.16) and (3.22) yields the closed form bound .

$$\langle \mathbf{x}_n \rangle \le \langle \mathbf{T} \rangle f(\Lambda_A) \langle \mathbf{T}^{-1} \rangle \langle \mathbf{B}_e \rangle \mathbf{E}. \tag{3.23}$$

The quantization error signals observed at the output will be bounded in amplitude by

$$\langle y_n \rangle \le \langle \mathbf{C} \rangle \sum_{j=0}^{\infty} \langle \mathbf{A}^j \rangle \langle \mathbf{B}_e \rangle \mathbf{E} + \langle \mathbf{D}_e \rangle \mathbf{E} \tag{3.24}$$

which can be written in closed form as

$$\langle y_n \rangle \leq \langle \mathbf{C} \rangle \langle \mathbf{T} \rangle f(\Lambda_A) \langle \mathbf{T}^{-1} \rangle \langle \mathbf{B}_e \rangle \mathbf{E} + \langle \mathbf{D}_e \rangle \mathbf{E}. \tag{3.25}$$

Together (3.23)-(3.25) are termed the General Bound (GB), because they allow bounds on quantization errors to be computed for any filter. The term closed formed GB refers to (3.23) and (3.25), while the term iterative GB refers to the iterative expressions (3.22) and (3.24).

If the quantization operations are performed at the states, then the state transition equation is given by

$$\hat{\mathbf{x}}_{n+1} = \mathbf{A}\hat{\mathbf{x}}_n + \mathbf{E}_n. \tag{3.26}$$

This is equivalent to setting $\mathbf{B}_e$ in (3.14) equal to the identity matrix, **I.** Because $\langle \mathbf{I} \rangle = \mathbf{I}$, (3.23) and (3.25) simplify to

$$\langle \mathbf{x}_n \rangle \leq \langle \mathbf{T} \rangle f(\Lambda_A) \langle \mathbf{T}^{-1} \rangle \mathbf{E} \tag{3.27}$$

and

$$\langle y_n \rangle \leq \langle \mathbf{C} \rangle \langle \mathbf{T} \rangle f(\Lambda_A) \langle \mathbf{T}^{-1} \rangle \mathbf{E} + \langle \mathbf{D}_e \rangle \mathbf{E} \tag{3.28}$$

respectively. Note that (3.27) and (3.28) are identical to (15) and (17) reported in [21] (Yakowitz and Parker). Thus, (3.27) and (3.28) will be referred to as the Yakowitz-Parker (YP) bound. The YP bound is a special case of the GB reported here.

3.4 Comparing the General Bound to the l_1 Norm

In Chapter 2, the l_1 norm was introduced as a means of bounding quantization errors. In this section, the l_1 norm will be related to the GB, and the YP bound. The l_1 norm is the summation of the absolute values of the unit sample response of the filter due to input error signals modeling the effects of quantization. The time domain response of the filter is given by [8]

$$y_n = \begin{cases} \mathbf{D}_e\mathbf{E}_n & \text{if } n = 0 \\ \sum_{j=0}^{n-1} \mathbf{C}\mathbf{A}^{n-1-j}\mathbf{B}_e\mathbf{E}_j + \mathbf{D}_e\mathbf{E}_n. & n > 0 \end{cases} \tag{3.29}$$

Thus,

$$|y_n| \le \sum_{j=0}^{\infty} \langle \mathbf{C}\mathbf{A}^j\mathbf{B}_e \rangle \mathbf{E} + \langle \mathbf{D}_e \rangle \mathbf{E} \tag{3.30}$$

for all n. By examining (3.30), it can be seen that it provides a tighter bound on than the iterative General Bound because

$$\sum_{j=0}^{\infty} \langle \mathbf{C}\mathbf{A}^j\mathbf{B}_e \rangle \mathbf{E} + \langle \mathbf{D}_e \rangle \mathbf{E} \le \langle \mathbf{C} \rangle \left(\sum_{j=0}^{\infty} \langle \mathbf{A}^j \rangle \right) \langle \mathbf{B}_e \rangle \mathbf{E} + \langle \mathbf{D}_e \rangle \mathbf{E}. \tag{3.31}$$

The l_1 norm also provides a tighter bound than the closed form General Bound (and hence the Yakowitz-Parker bound) because

$$\langle \mathbf{C} \rangle \left(\sum_{j=0}^{\infty} \langle \mathbf{A}^j \rangle \right) \langle \mathbf{B}_e \rangle \mathbf{E} + \langle \mathbf{D}_e \rangle \mathbf{E} \le \langle \mathbf{C} \rangle \langle \mathbf{T} \rangle f(\Lambda_A) \langle \mathbf{T}^{-1} \rangle \langle \mathbf{B}_e \rangle \mathbf{E} + \langle \mathbf{D}_e \rangle \mathbf{E} \tag{3.32}$$

and hence

$$\sum_{j=0}^{\infty} \langle \mathbf{C}\mathbf{A}^j\mathbf{B}_e \rangle \mathbf{E} + \langle \mathbf{D}_e \rangle \mathbf{E} \le \langle \mathbf{C} \rangle \langle \mathbf{T} \rangle f(\Lambda_A) \langle \mathbf{T}^{-1} \rangle \langle \mathbf{B}_e \rangle \mathbf{E} + \langle \mathbf{D}_e \rangle \mathbf{E}. \tag{3.33}$$

Thus, the l_1 norm provides a tighter bound on the magnitude of limit cycles than both the iterative and closed form General Bound. The General Bound, however, has the advantage of providing a closed form solution, which the l_1 norm does not.

3.5 Convergence

One difficulty associated with using the l_1 norm or the iterative General Bound is determining the number of iterations required for convergence. This problem can be addressed by using the Lyapunov function defined in Section 3.2. It was shown that

$$V(\mathbf{x}_{n+1}) \le |\lambda_M| V(\mathbf{x}_n) \tag{3.34}$$

where $|\lambda_M|$ is the eigenvalue of **A** with the largest modulus. This can be extended to show that [9]

$$V(\mathbf{x}_{n+k}) \le |\lambda_M|^k V(\mathbf{x}_n) \tag{3.35}$$

and

$$V(\mathbf{x}_n) \to 0 \qquad \text{as } n \to \infty. \tag{3.36}$$

Recall that

$$V(\mathbf{x}) = \sqrt{\mathbf{x}^T\mathbf{S}\mathbf{x}}\,. \tag{3.37}$$

and it can be shown that **S** is a real symmetric matrix [9]. Thus, **S** can be decomposed as follows [9]:

$$\mathbf{S} = \mathbf{Q}\Lambda_S\mathbf{Q}^T \tag{3.38}$$

where **Q** is an orthogonal norm-preserving real matrix of normalized eigenvectors of **S** and Λ_S is the diagonal matrix of real eigenvalues of **S**. This can be used to define a new coordinate system

$$\mathbf{v} = \mathbf{Q}^T\mathbf{x} \tag{3.39}$$

so that

$$\begin{aligned} V(\mathbf{x})^2 &= \mathbf{x}^T\mathbf{S}\mathbf{x} = \mathbf{x}^T\mathbf{Q}\Lambda_S\mathbf{Q}^T\mathbf{x} = \mathbf{v}^T\Lambda_S\mathbf{v} \\ &= \mu_1 v_1^2 + \mu_2 v_2^2 + \cdots + \mu_n v_n^2 \end{aligned} \tag{3.40}$$

where $\mu_1, \mu_2, \cdots, \mu_n$ are the eigenvalues of **S** [9]. Thus, $V(\mathbf{x}) = K$ describes an n-dimensional hyper-ellipse, where n is the order of the filter.

As well, $V(\mathbf{x})$ has the important property

$$V(\alpha\mathbf{x}) = \alpha V(\mathbf{x}) \tag{3.41}$$

where $\alpha \ge 0$, because

$$\begin{aligned}V(\alpha\mathbf{x}) &= \left\|\mathbf{T}^{-1}\alpha\mathbf{x}\right\| \\ &= \sqrt{\left(\mathbf{T}^{-1}\alpha\mathbf{x}\right)^T\left(\mathbf{T}^{-1}\alpha\mathbf{x}\right)} \\ &= \sqrt{\alpha^2\left(\mathbf{T}^{-1}\mathbf{x}\right)^T\left(\mathbf{T}^{-1}\mathbf{x}\right)} \\ &= \alpha\sqrt{\left(\mathbf{T}^{-1}\mathbf{x}\right)^T\left(\mathbf{T}^{-1}\mathbf{x}\right)} \\ &= \alpha\left\|\mathbf{T}^{-1}\mathbf{x}\right\| = \alpha V(\mathbf{x}).\end{aligned} \tag{3.42}$$

The summations in both the iterative GB and the l_1 norm are equivalent to summing the absolute values of the unit sample response of the filter [2]. When computing the unit sample response of a filter, the filter is assumed to be initially at rest. At time $n=0$, the unit sample occurs, and as $n \to \infty$, the filter states and output asymptotically approach zero. Let the value of the Lyapunov function immediately after the impulse has occurred be given by

$$V(\mathbf{x}_1) = K. \tag{3.43}$$

When $n>0$, the output of the filter is given by

$$y_n = \mathbf{C}\mathbf{x}_n. \tag{3.44}$$

Because y_n is a scalar, (3.44) can be considered the dot product of the vectors $\mathbf{C}^T$ and $\mathbf{x}_n$. Refer to Figure 3.6 for a graphical representation. Note that although the figure represents the second order case, this discussion is not limited to second order filters. Let $\mathbf{x}_1'$ be the point on the curve $V(\mathbf{x}_1) = K$ that yields the largest possible value of y. No assumption is made as to whether $\mathbf{x}_1$ is actually equal to $\mathbf{x}_1'$, but because $\mathbf{x}_1'$ represents the point on the curve $V(\mathbf{x}_1) = K$ that yields the maximum possible output, we can say

$$|y_1| \le |\mathbf{C}\mathbf{x}_1'| \tag{3.45}$$

and in general

$$|y_n| \le |\mathbf{C}\mathbf{x}_n'|. \tag{3.46}$$

Using (3.45) and (3.46), after n iterations, the point that at which the maximum output can occur is given by

$$\mathbf{x}_n' \le |\lambda_M|^{n-1}\mathbf{x}_1'. \tag{3.47}$$

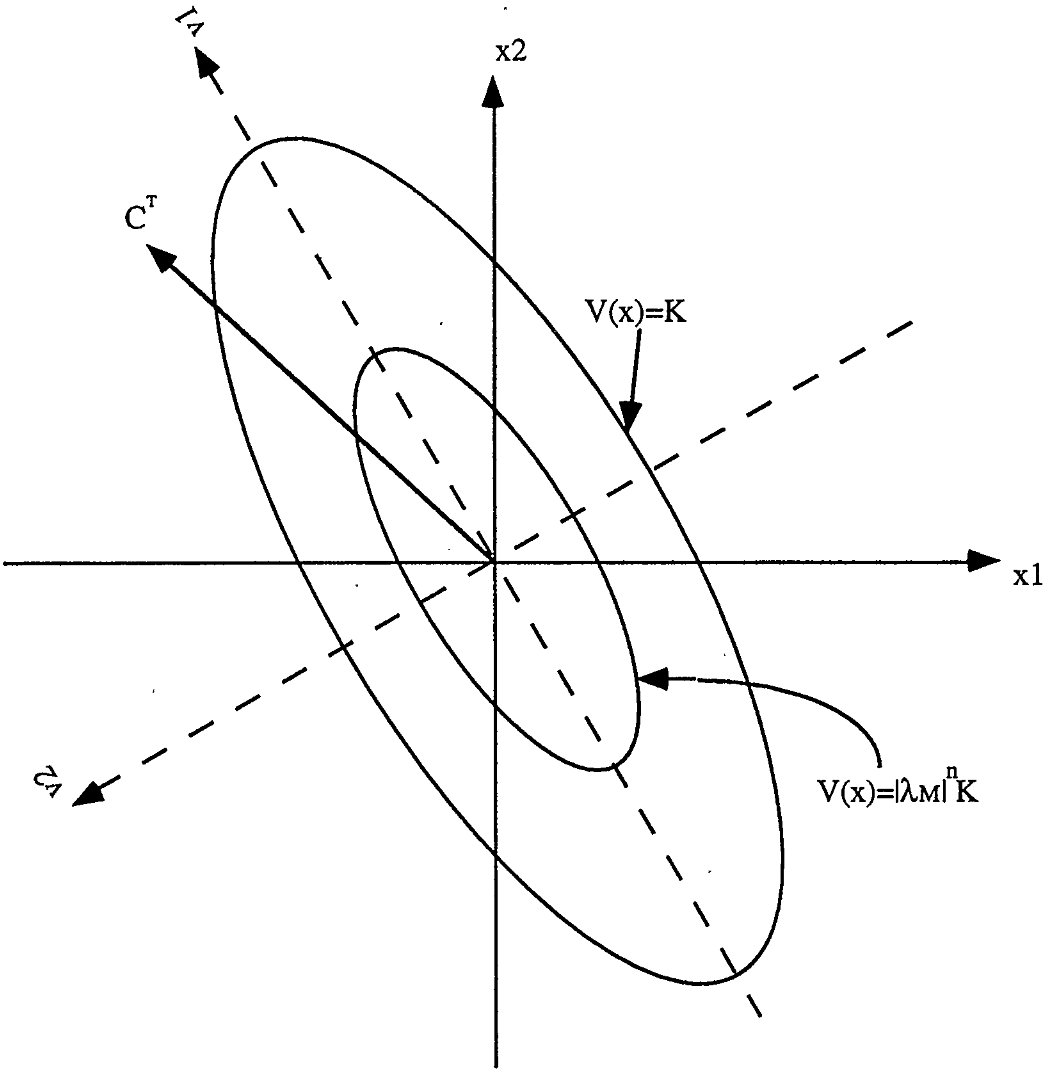

Figure 3.6 - Geometry of V(x)

The maximum possible output that can occur is thus given by

$$|y_n| \le |\mathbf{Cx}'_n| \le |\lambda_M|^{n-1} |\mathbf{Cx}'_1|. \tag{3.48}$$

If the maximum possible output is denoted by $y_{\max}$,

$$|y_{\max}| = |\mathbf{Cx}'_1|. \tag{3.49}$$

This leads to

$$\frac{|y_n|}{|y_{\max}|} \le \frac{|\lambda_M|^{n-1} |\mathbf{Cx}_1|}{|\mathbf{Cx}_1|} = |\lambda_M|^{n-1}. \tag{3.50}$$

After N iterations, the envelope of the unit sample response is less than $|\lambda_M|^{N-1}|y_{\max}|$. Thus,

$$\begin{aligned} &|y_1| \le |y_{\max}| \\ &|y_2| \le |\lambda_M|^1 |y_{\max}| \\ &|y_3| \le |\lambda_M|^2 |y_{\max}| \\ &\vdots \\ &|y_n| \le |\lambda_M|^{n-1} |y_{\max}|. \end{aligned} \tag{3.50}$$

The error after N iterations will be equal to the difference between the infinite summation and the finite summation as given below:

$$\text{error} = \sum_{\text{n=0}}^{\infty} |y_n| - \sum_{\text{n=0}}^{N-1} |y_n| = \sum_{\text{n=N}}^{\infty} |y_n|. \tag{3.52}$$

Furthermore,

$$\sum_{\text{n=N}}^{\infty} |y_n| \le \sum_{\text{n=N}}^{\infty} |\lambda_M|^{n-1} |y_{\max}| = |y_{\max}| \sum_{\text{n=N}}^{\infty} |\lambda_M|^{n-1} \tag{3.53}$$

and

$$\sum_{n=N}^{\infty} |\lambda_M|^{n-1} = \sum_{n=N-1}^{\infty} |\lambda_M|^{n} = \sum_{n=0}^{\infty} |\lambda_M|^{n} - \sum_{n=0}^{N-2} |\lambda_M|^{n}. \tag{3.54}$$

It can be shown that [23]

$$\sum_{n=0}^{\infty} |\lambda_M|^{n} = \frac{1}{1 - |\lambda_M|} \tag{3.55}$$

and

$$\sum_{n=0}^{N-2} |\lambda_M|^{n} = \frac{1 - |\lambda_M|^{N-1}}{1 - |\lambda_M|}. \tag{3.56}$$

Thus, (3.52), (3.53), (3.54), (3.55) and (3.56) can be combined to show that

$$\text{error} \le |y_{\max}| \frac{|\lambda_M|^{N-1}}{1 - |\lambda_M|}. \tag{3.57}$$

This yields an expression that can be used to determine the maximum error in the summation after N iterations. It is necessary to compute $|y_{\max}|$ and this can be done in a straightforward fashion. Recall that

$$\begin{aligned} y_{\max} &= \mathbf{C}^T \cdot \mathbf{x}_1' \\ &= \left\|\mathbf{C}^T\right\|_2 \cdot \left\|\mathbf{x}_1'\right\|_2 \cos\theta \\ &\le \left\|\mathbf{C}^T\right\|_2 \cdot \left\|\mathbf{x}_1'\right\|_2 \end{aligned} \tag{3.58}$$

where θ is the angle between $\mathbf{C}^T$ and $\mathbf{x}_1'$ [24]. Since the l_2 norm has the property of distance in a Euclidian space, $\left\|\mathbf{x}_1'\right\|_2$ will be less than or equal to the length of the longest axis of the n-dimensional hyper-ellipse. With $V(\mathbf{x})$ defined as (3.40), the length of the longest axis is given by $\frac{V(\mathbf{x}_1)}{\sqrt{\mu_s}}$ [23] where μ_s is the eigenvalue of $\mathbf{S}$ which has the smallest square root. Because $\mathbf{S}$ is a positive definite symmetric real matrix, all of its eigenvalues will be positive real numbers [9]. Thus,

$$y_{\max} \le \left\|\mathbf{C}^T\right\| \frac{V(\mathbf{x}_1)}{\sqrt{\mu_s}}. \tag{3.59}$$

The number of iterations required to compute the summation with an error less than a specified value can be determined by rearranging (3.57) to yield

$$N = \frac{\ln\left(\frac{\left(1-\left|\lambda_M\right|\right)\text{error}}{y_{\max}}\right)}{\ln\left(\left|\lambda_M\right|\right)} + 1. \tag{3.60}$$

Computing N iterations will guarantee that the error in the summation is less than the value specified.

3.6 Examples

The General Bound developed in the preceding sections has been applied to the fifth order Chebychev lowpass LDI filter [9] shown in Figure 3.7 and the sixth order Elliptic bandpass LDI filter [25] shown in Figure 3.8.

The fifth order Chebychev filter has a quantization operation located at the input of each state. The results of computing the closed form GB, the iterative GB and the l_1 norm are summarized in Table 3.1. It can be seen that the closed form GB yields the poorest bound, while the iterative GB and the l_1 norm yield equivalent results. This is expected because $\mathbf{C}=\mathbf{I}$ and $\mathbf{B}_e=\mathbf{I}$. Note that because of the location of the quantization operations, the closed form GB is equivalent to the YP bound. The iterative results required 49 iterations for the summations in the l_1 norm and the iterative GB to converge with an error of less than 0.01. Table 3.1 also includes results reported in [9] for this particular filter. It can be seen that both the iterative GB and the l_1 norm provide tighter bounds than those reported in [9] for states S1,S3,S5 and the output.

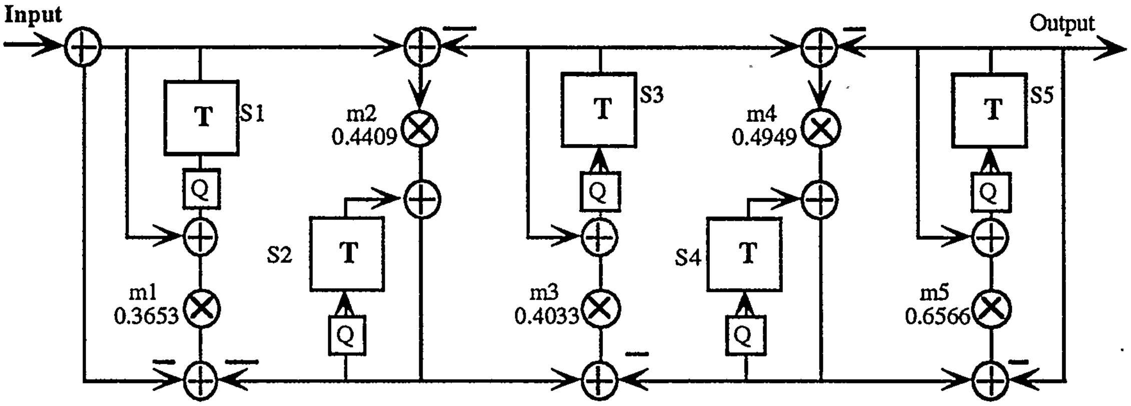

Figure 3.7 - Fifth Order Chebychev Lowpass LDI Filter

Table 3.1 - Quantization Error Bounds for Figure 3.7

State	Closed Form GB	Iterative GB	l_1 norm	Green/Turner Bound [9]
S1	26	10	10	32
S2	34	16	16	10
S3	35	15	15	22
S4	53	16	16	13
S5	54	14	14	33
Output	54	14	14	33

The sixth order elliptic filter has quantization operations located at the outputs of the multipliers. The results of computing the closed form GB, the iterative GB and the l_1 norm are summarized in Table 3.2. It can be seen that the closed form GB provides the poorest bound while the l_1 norm provides the tightest. The iterative results required 94 iterations for the summations in the l_1 norm and the iterative GB to converge with an error of less than 0.01. Note that the method reported in [9] is not applicable to filters with the quantization operations located anywhere but the states and thus is not applicable to this filter.

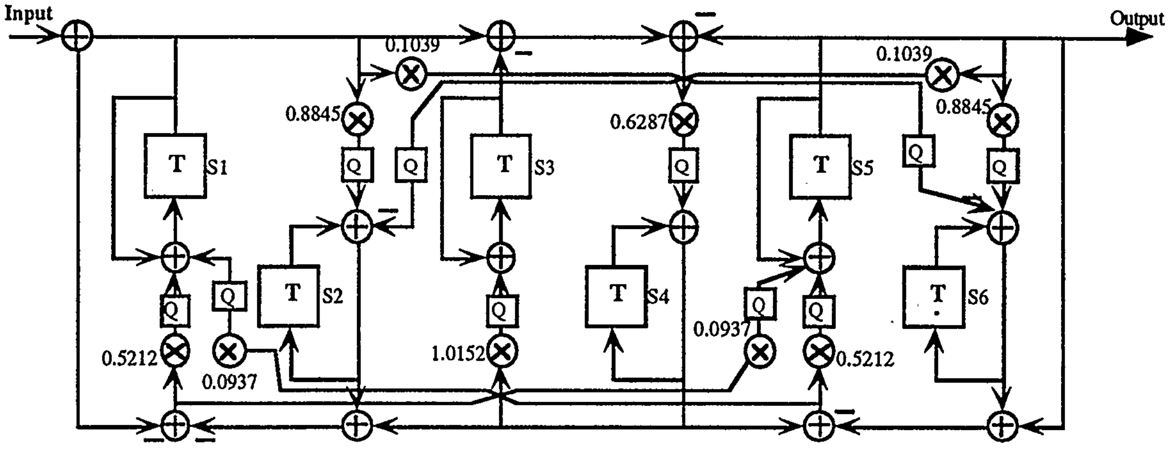

Figure 3.8 - Sixth Order Elliptic Bandpass LDI Filter

Table 3.2 - Quantization Error Bounds for Figure 3.8

State	Closed Form GB	Iterative GB	l_1 norm
S1	97	35	24
S2	151	71	48
S3	117	49	32
S4	138	55	36
S5	138	55	36
S6	97	35	24
Output	267	108	42

3.7 Using the Limit Cycle Bound

The limit cycle bounds given above can be used to create filter implementations guaranteed to be free from limit cycles at the output. Suppose the sixth order elliptic band pass filter depicted in Figure 3.8 is to be implemented. For the purposes of this example, assume that the filter will use two's complement truncation for quantization of the products. According to the l_1 norm, limit cycles (and quantization noise) will be no greater than 42 times the quantization interval at the output. Thus, any limit cycle oscillations will be limited in magnitude to the six least significant bits of the output word. If the input to the filter is shifted left by six bits, and the output shifted right by six bits, then no limit cycle oscillations will be present at the filter output. Note that it is possible for limit cycle oscillations to occur inside the filter, but these oscillations will not be observable at the output of the filter. Further, because the l_1 norm bounds all forms of quantization noise, quantization will not introduce more than one bit of error in the filter output.

Although the filter can be scaled such that limit cycles and quantization noise are not observable at the output, a penalty in the form of increased wordlength must be paid. In the above example, the filter's internal wordlength would have to be increased by six bits. Whether this is feasible or not depends upon the technology used to implement the filter. For bit serial systems, the filter's internal wordlength may be much greater than the data input wordlength due to the nature of bit serial implementations [26]. If this is the case, the necessary extra bits are already present, and it is simply a matter of correctly scaling the input and output. Many modern DSP chips such as the Motorola 56000 use thirty-two bit arithmetic for integer operations. Data words this large often allow effective scaling so that overflows do not occur and limit cycles and noise are not present at the output. In all cases, however, whether or not the extra wordlength required for scaling is available depends upon the technology used to implement the filter and the application at hand.

3.8 Filter Structure and Limit Cycles

In the derivation of the GB it was shown that the error due to quantization depends on the structure of the filter, and not just the transfer function of the filter. Thus, changing the structure of the filter will change the bounds for limit cycle oscillations.

Consider the filter shown in Figure 3.9. The filter is implemented using integer arithmetic, and two's complement truncation with the quantizers located at the states. The iterative GB for the quantization errors seen at output of this structure is equal to 12. If the filter is rearranged as shown in Figure 3.10, the iterative GB at the output is 3. Although the two systems have the same transfer function, they will behave differently because of the effects of finite precision arithmetic. If the two structures are examined strictly in terms of limit cycle oscillations, then the second structure is superior, as the magnitude of any possible limit cycles is lower. Other finite precision effects must also be considered however. The first filter requires three extra bits of wordlength to prevent overflow, while the second requires four bits. The total wordlength, to prevent overflow and to prevent limit cycles from appearing at the output is lower for the second structure. Therefore, it is the preferable implementation, from the point of view of wordlength required. Although different filter structures can implement the same transfer function, they may display very different performance when implemented using finite precision arithmetic. Limit cycles, along with other aspects of finite precision arithmetic must be carefully considered when implementing digital filters.

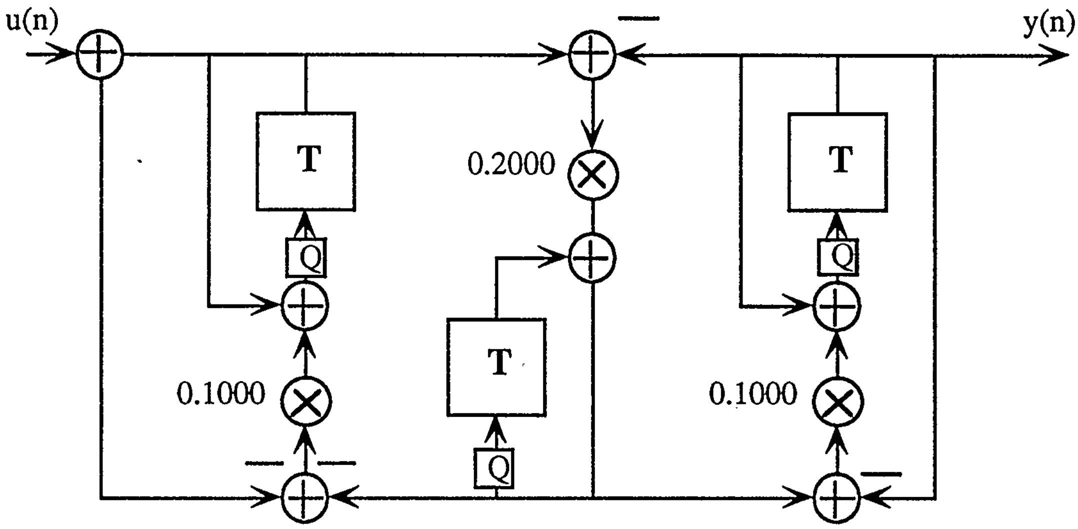

Figure 3.9 - Third Order LDI Filter

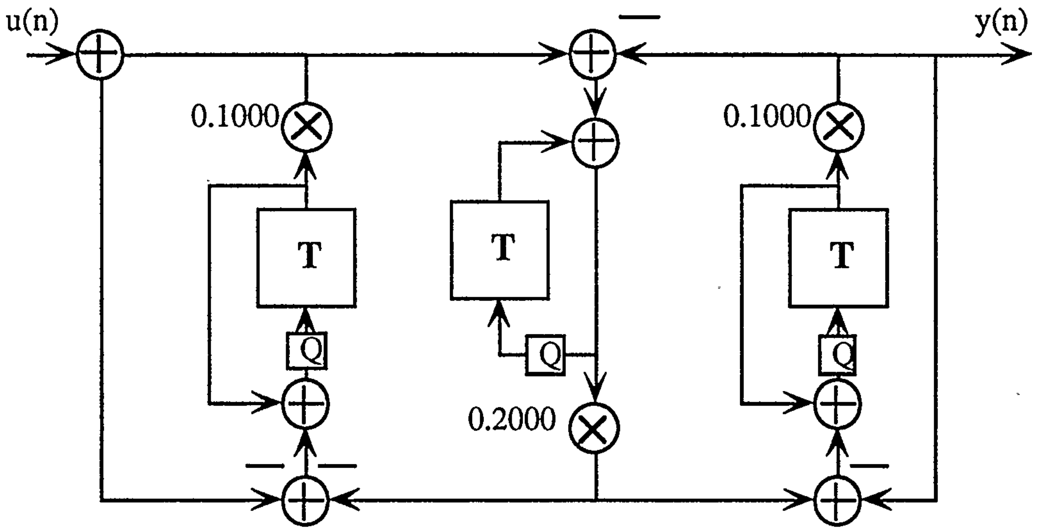

Figure 3.10 - Rearranged Third Order LDI Filter

Chapter 4

Discrete Time Sinusoids

Many digital signal processing applications such as digital radios, satellite communication systems, modems, frequency analyzers and laboratory equipment require discrete time sinusoids. Because of the quantized nature of these digital systems, true discrete time sinusoids cannot be generated. Rather, digital approximations to discrete time sinusoids are used. These digital approximations to discrete time sinusoids will be termed digital sinusoids. Although the term is not strictly accurate, it is a convenient way of describing the digital approximations and use of this term does not introduce any errors or ambiguities into the discussion. This chapter is concerned with ways of generating digital sinusoids in real time. There are two main categories of digital oscillators, those in which the sinusoid is computed ahead of time and stored in a look-up table and those in which an algorithm is used to compute the sinusoid as it is generated. The performance, advantages and disadvantages of each method are examined.

4.1 ROM Based Oscillators

One method of generating a fixed point digital approximation to a discrete time sinusoid is to use a phase accumulator in conjunction with a read only memory (ROM) [2]. A discrete time sinusoid is a sine function of a time varying phase. The phase increases linearly with respect to time and can be viewed as a straightforward addition operation, in which the phase is incremented at each sample time. The digital hardware required to implement this is an accumulator. The most straightforward method of implementing the sine function is to use a look-up table method in which the phase is used as an index into a table of sinusoid values. This is implemented in hardware as a ROM in which the phase is the address and the data stored in the ROM is the sinusoidal value corresponding to each phase. This is shown schematically in Figure 4.1. This structure is used in many commercially available digital sinusoidal oscillators [27, 28, 29].

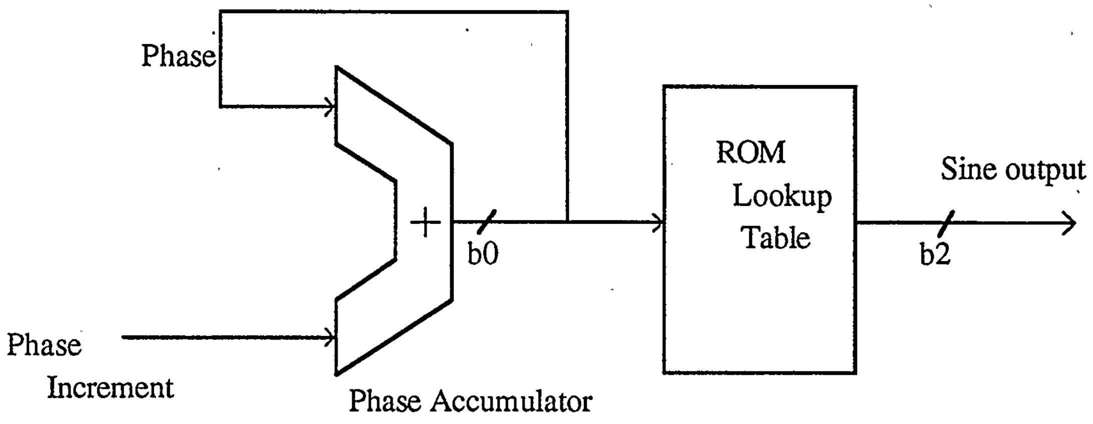

Figure 4.1 - ROM Oscillator

One advantage of oscillators of this type is that their performance is easy to characterize. Let the width of the phase accumulator (in bits) be denoted by b_0. The frequency of oscillation will be given by

$$f_0 = \frac{N}{2^{b_0}} \times F_{clk} \tag{4.1}$$

where N is the phase increment (an integer value) and F_{clk} is the sample frequency of the oscillator. The normalized frequency of oscillation is given by

$$\Omega_0 T = \frac{N}{2^{b_0}} \times 2\pi \tag{4.2}$$

Note that this type of oscillator can operate at frequencies up to one half of the sample frequency. The resolution with which the frequency can be set is determined by the width of the phase accumulator, b_0. The interval between possible frequencies is given by

$$\Delta f_0 = \frac{1}{2^{b_0}} \times F_{clk} \tag{4.3}$$

If the maximum the frequency can be in error is one bit at the phase accumulator input, then the maximum error in the sinusoidal frequency is given by Δf_0.

The variations in amplitude in the ROM output can be viewed as noise added on top of the sinusoidal signal. Because the ROM values are computed in advance, the maximum error that they will have is less than one least significant bit. Thus, the noise added will always be less than one least significant bit for any sample and will be bounded by

$$|e_n| < \frac{1}{2^{b_2-1}} \tag{4.4}$$

where e_n. When $|e_n|$ is expressed in dB, it is termed spectral purity [27]. This oscillator structure is interesting in that the errors in frequency and amplitude are completely separate; they are controlled by different elements within the oscillator circuitry.

4.1.1 Example

The HSP45106 [27] is an example of a commercially available ROM based digital sinusoidal oscillator. Implemented in single chip form, it features sample rates of up to 40 MHz, both sine and cosine outputs with spectral purity greater than -90 dB, and frequency resolution of less than 0.01 Hz at 40 MHz operation [27].

The phase accumulator is 32 bits wide in the HSP45106. This yields a frequency resolution of 0.0093 Hz at a 40 MHz sample rate, according to (4.3). The outputs are 16 bits, thus according to (4.4) the magnitude of the maximum error at the output is 30.52×10^{-6} or -90.3 dB. These figures agree with those published by the manufacturer.

4.1.2 Practical Concerns

One limitation of this type of oscillator is the large ROM size required. The ROM size is given by

$$ROM \;\; size = b_2 \times 2^{b_1}. \tag{4.5}$$

Because

$$\sin(\varphi) = -\sin(\varphi + \pi) = \sin(\pi - \varphi) = -\sin(2\pi - \varphi) \tag{4.6}$$

ROM size can be reduced by storing only one quarter of the sinusoid and computing the output value for each sample. ROM size can also be reduced by only storing a limited set of values in the ROM and computing the output using interpolation. Both of these methods require extra hardware, but the cost of this is often offset by the savings in reduced ROM size. Interpolation will reduce the accuracy of the output.

It can be seen that ROM based digital sinusoidal oscillators offer high performance that is easily characterized. The greatest drawback of this method is the large amount of ROM storage required for the look-up table.

4.2 Algorithmic Oscillators

Sinusoidal oscillators can also be designed using second order discrete time systems. Consider the second order transfer function

$$\frac{O(z)}{I(z)} = \frac{N(z)}{z^2 + cz + 1}. \tag{4.7}$$

The poles are located at

$$p_{1,2} = \frac{-b \pm \sqrt{b^2 - 4}}{2}. \tag{4.8}$$

It can be shown that if $-2 < b < 2$, then the poles will be complex conjugates located on the unit circle. The unit sample response of a filter with two complex conjugate poles on the unit circle will be sinusoidal and the oscillations will continue indefinitely [2]. The frequency of oscillation is given by the angle of the poles, and the phase of the oscillation is determined by the zeros of the transfer function [2].

4.2.1 Direct Form Oscillator

A straightforward way of implementing a filter with poles on the unit circle is to use a second order direct form (DF2) section [10] as shown in Figure 4.2. The difference equation is given by [30]

$$y_n = by_{n-1} - y_{n-2}. \tag{4.9}$$

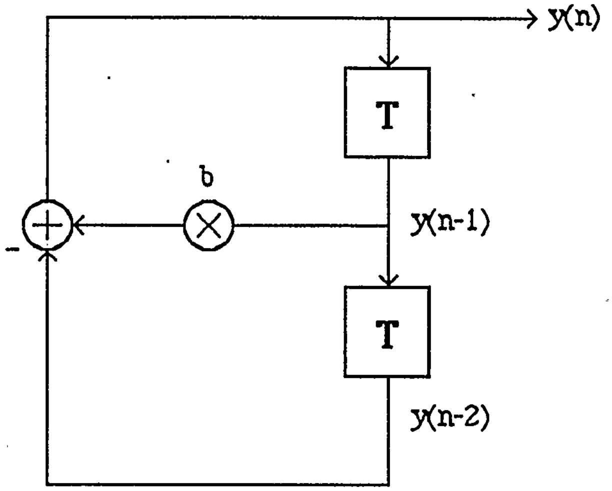

Figure 4.2 - Direct Form Oscillator

If the delay corresponding to y_{n-1} has its initial condition set to one and the other delay is set to zero, then the z transform of the signal y_n will be

$$Y(z) = \frac{1}{z^2 - bz + 1} \tag{4.10}$$

which is equivalent to a time domain sinusoid with a frequency given by [2]

$$\Omega_0 T = \cos^{-1}\left(\frac{b}{2}\right) \tag{4.11}$$

provided that $-2 < b < 2$. If b is outside this range, then the response will be a hyperbolic sinusoid which will grow without bound [31].

This structure is suited for use as an oscillator because the poles are always on the unit circle, even when the coefficients are quantized. The poles are given by

$$p_{1,2} = \frac{b}{2} \pm \frac{j}{2}\sqrt{b^2 - 4}. \tag{4.12}$$

With $-2 < b < 2$, $|p_{1,2}| = 1$, meaning that even though the value of b will change slightly when it is quantized, the quantization will not move the poles off of the unit circle.

4.2.2 Practical Concerns

There are two major practical problems with the DF2 oscillator, both related to finite precision arithmetic effects. The first concerns coefficient sensitivity. If the coefficient is assumed to be a finite precision, fixed point number, then the number of values it can assume is limited. From (4.12) it can be seen that the real part of the pole locations are equal to $b/2$ and that the poles are always on the unit circle. The frequency of oscillation is given by the angle between the pole and the real axis, which is equal to $\cos^{-1}(b/2)$. Poles near the positive z axis correspond to low frequencies, while those on the imaginary axis correspond to a frequency of $\Omega_0 T = \pi/2$ or 4 samples per sinusoidal cycle. Unfortunately, the possible pole locations are not uniformly distributed around the unit circle. Figure 4.3 shows the possible pole locations for the oscillator with four fractional bits for the coefficient. It can be seen that there are fewer poles near the positive real axis and more near the imaginary axis, meaning that there are fewer coefficients corresponding to low frequencies. Thus, the high sensitivity to coefficient quantization at low frequencies requires that the number of fractional bits for the coefficients be increased, increasing filter size and cost.

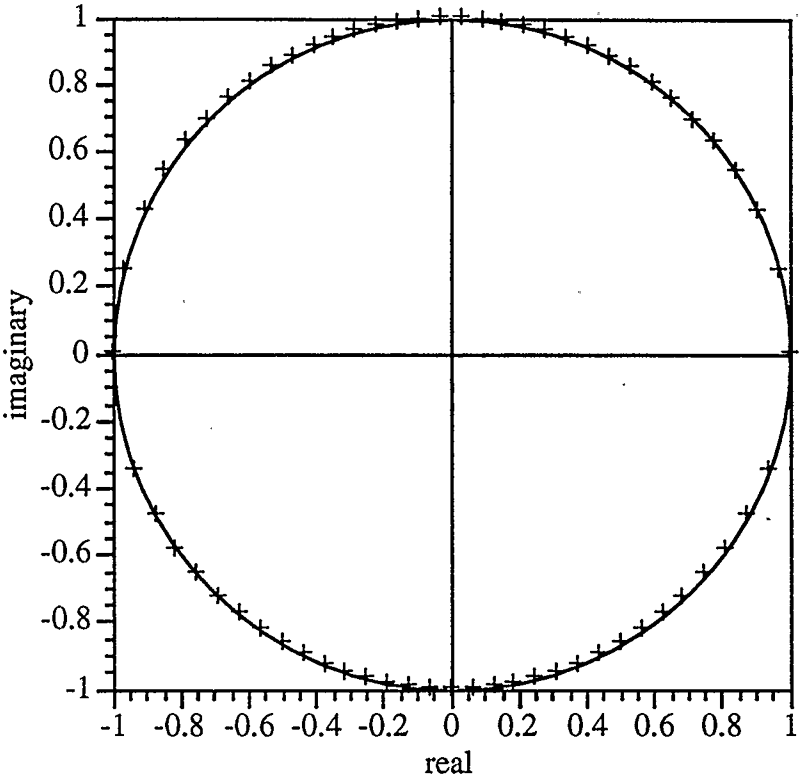

Figure 4.3 - Pole Locations for DF2

The second difficulty is caused by signal quantization. Because the oscillator will be implemented with finite precision arithmetic, the signals within the oscillator will be quantized and this can cause changes in the amplitude and frequency of oscillation [30, 32]. Several solutions to this problem have been proposed. One method suggested in [2] is to reset the oscillator to its initial conditions every *mn* samples where $\frac{n}{m} = \frac{f_o}{F_s}$.

The drawback of this method is that it requires external hardware and does not address the problem of the errors which build up before the reset occurs. Another solution proposed in [30] is to select initial conditions and coefficients which guarantee that the oscillator is absolutely periodic and of known period length. Unfortunately, this is a trial and error procedure and is only feasible for oscillators with small word length and a small number of fractional bits in the coefficient. Other structures have been proposed [32, 33] which attempt to reduce the errors inherent in the structure, but none can guarantee the continued operation of the oscillators.

4.2.3 Normal Form

In addition to the DF2 structure, other second order systems with poles on the unit circle can be used to construct sinusoidal oscillators. A second order normal form structure is shown in Figure 4.4.

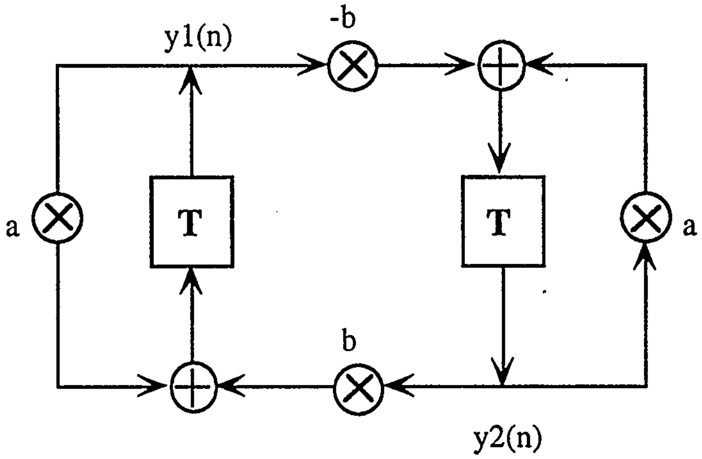

Figure 4.4 - Normal Form Oscillator

For this structure, the pole locations are given by

$$p_{1,2} = a \pm jb. \qquad (4.13)$$

Since the poles must be exactly on the unit circle for the circuit to oscillate indefinitely, $a^2 + b^2$ must equal one exactly. If $a^2 + b^2 < 1$, then the response will be a decaying sinusoid which will asymptotically approach zero. If $a^2 + b^2 > 1$, the response will grow without bound. Because of coefficient quantization, it will be virtually impossible to find quantized values for a and b that yield the desired frequency of operation and exactly satisfy $a^2 + b^2 = 1$. Thus, this structure is not suited for use as a sinusoidal oscillator.

4.2.4 Wave Digital Oscillator

Another possible way of generating a second order digital structure with poles on the unit circle is to create a wave digital filter from an analog prototype. Because wave digital filters are mapped from the continuous domain into the discrete domain using the bilinear transform [34, 35], an analog prototype with two poles on the imaginary s-plane axis will yield a discrete time circuit with poles on the unit circle. The circuit shown in Figure 4.5 is such a circuit and it can be shown that the poles are located at $\pm j\frac{1}{\sqrt{LC}}$ in the s-plane. The resonant frequency of this circuit is determined by the pole location, and will be $\omega_0 = \frac{1}{\sqrt{LC}}$.

The analog prototype is converted to a digital circuit by describing the propagation of voltage waves within the circuit and expressing these as digital signals according to the technique described in [35]. This yields the digital circuit shown in Figure 4.6.

The coefficients *ms1* and *ms2* are given by

$$\begin{aligned} ms1 &= \frac{2T^2}{T^2 + 4LC} \\ ms2 &= \frac{8LC}{T^2 + 4LC} \end{aligned} \tag{4.14}$$

where T is the inverse of the sample frequency of the discrete time system.

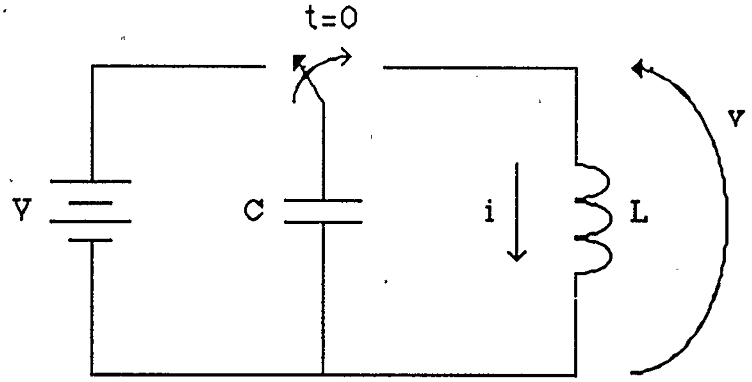

Figure 4.5 - Analog Prototype

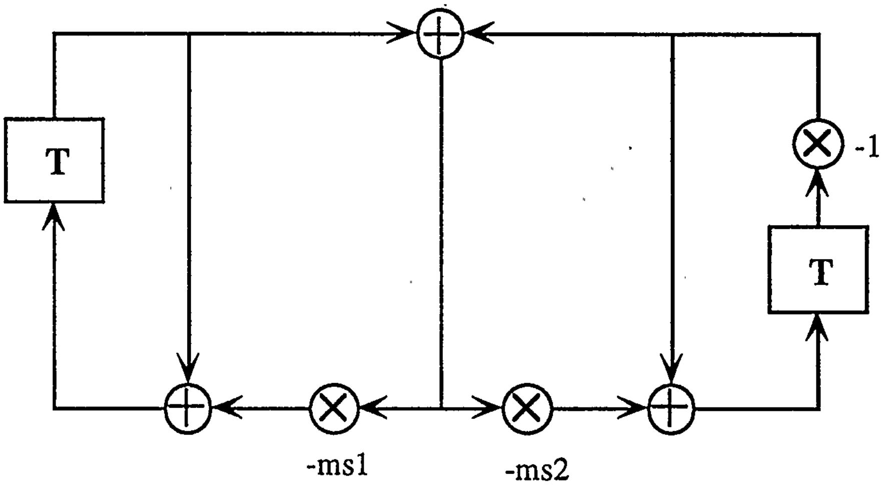

Figure 4.6 - Wave Digital Oscillator

The frequency of oscillation of the analog prototype is given by $\omega_0 = \frac{1}{\sqrt{LC}}$ which corresponds to a discrete time frequency of $\Omega_0 T = \frac{2}{T}\tan^{-1}\left(\frac{T}{2\sqrt{LC}}\right)$.

The characteristic equation for this structure is given by

$$D(z) = z^2 - (ms2 - ms1)z - (1 - ms1 - ms2). \tag{4.15}$$

If the poles of this structure are to be located on the unit circle, then $(1-ms1-ms2)$ must equal -1 exactly. Ideally, 1-*ms*1-*ms*2 will equal -1, but in practice, when *ms1* and *ms2* are quantized their values will change and this will cause the poles to move off of the unit circle. With the poles not exactly on the unit circle, the amplitude of oscillations will not remain constant. Therefore, this structure is unsuited for use as an oscillator.

4.4 The LDI Oscillator

Another method of creating a digital circuit from an analog prototype is using the lossless discrete integrator (LDI) transform [36]. The circuit shown in Figure 4.5 can be described by the voltage-current signal flowgraph shown in Figure 4.7.

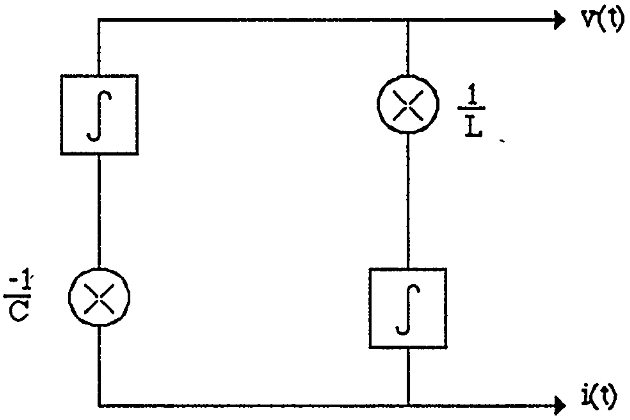

Figure 4.7 - Analog Prototype

This analog circuit is transformed to a digital circuit by replacing the continuous integrators (equal to $\frac{1}{s}$) with lossless discrete integrators [36] as shown in Figure 4.8. This is equivalent to making the substitution $\frac{1}{s} \rightarrow \frac{z^{\frac{1}{2}} - z^{-\frac{1}{2}}}{T}$ [36].

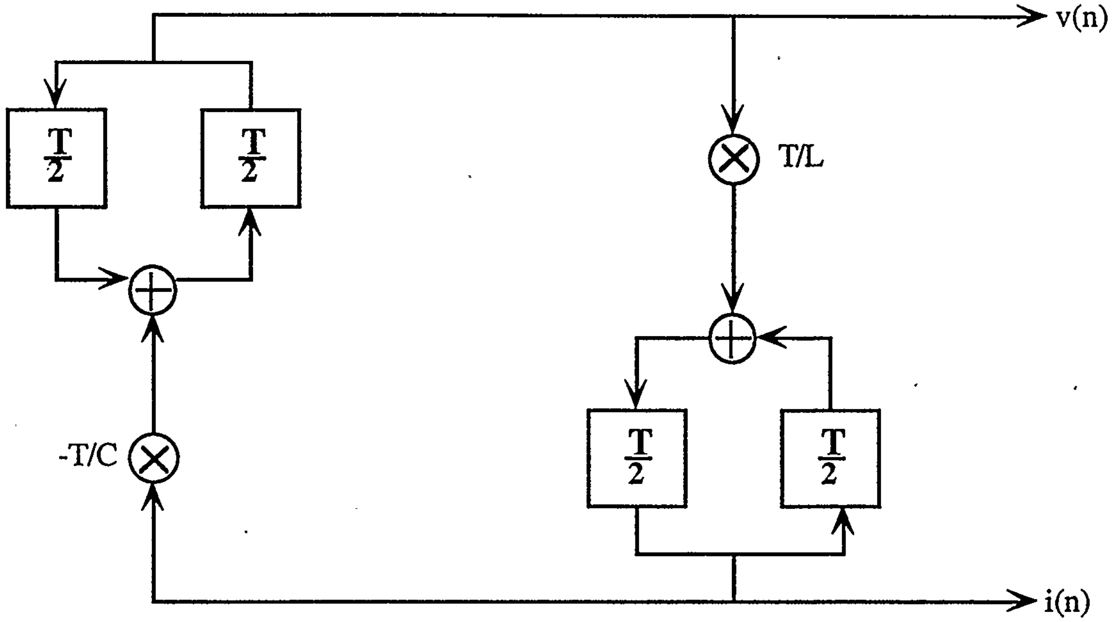

Figure 4.8 - Transformed Analog Prototype

The half delay elements can be eliminated by signal flowgraph manipulation as shown in Figure 4.9. Note that the LDI transformation is an exact one in this case, because there are no lossy elements [36].

The relationship between continuous and discrete time frequencies is given by [36]

$$\Omega T = 2\sin^{-1}\left(\frac{\omega T}{2}\right) \tag{4.16}$$

so that the frequency of oscillation is

$$\Omega_0 T = 2\sin^{-1}\left(\frac{T}{2\sqrt{LC}}\right) = 2\sin^{-1}\left(\frac{\omega_0 T}{2}\right). \tag{4.17}$$

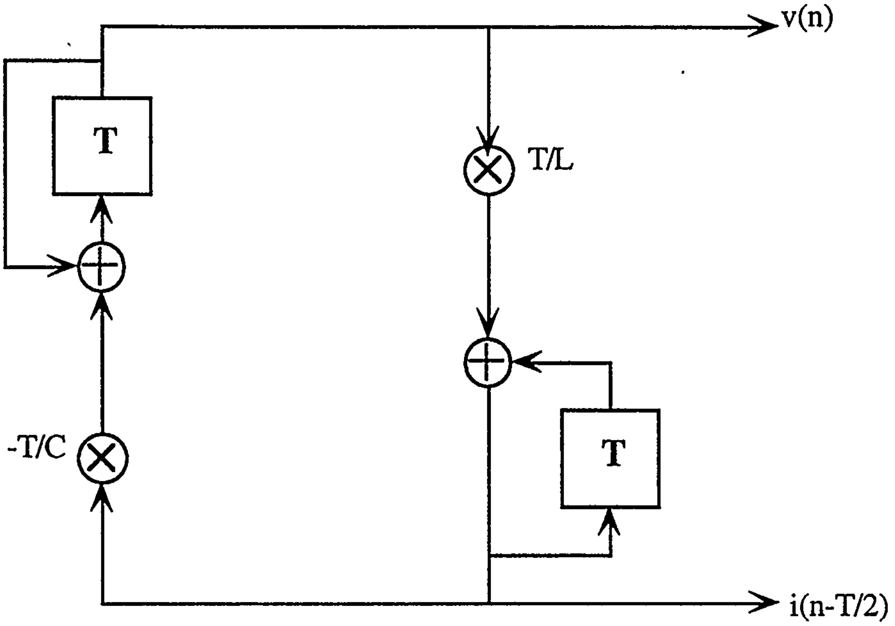

Figure 4.9 - LDI Oscillator

The state transition matrix for this system is given by

$$\mathbf{A} = \begin{vmatrix} 1-ab & b \\ -a & 1 \end{vmatrix} \tag{4.18}$$

where $a = \dfrac{T}{C}$ and $b = \dfrac{T}{L}$. The characteristic equation is given by

$$D(z) = z^2 - (2-ab)z + 1. \tag{4.19}$$

The fact that the last term in (4.19) is always unity indicates that when the poles are complex conjugates, they will fall exactly on the unit circle. The pole locations are given by

$$p_{1,2} = \frac{(2-ab) \pm \sqrt{(2-ab)^2 - 4}}{2}. \tag{4.20}$$

The poles will be complex conjugate pairs when

$$|2-ab| < 2 \tag{4.21}$$

or

$$0 < ab < 4. \tag{4.22}$$

If the same quantization scheme is used, there is a greater number of possible pole locations for the LDI oscillator than for the DF2 oscillator due to the fact that there are two multipliers in the LDI oscillator as opposed to one for the DF2, resulting in a greater number of possible stable coefficients. If coefficients with four fractional bits are used and a and b are equal, then the poles will be distributed as shown in Figure 4.10. From this, it can be seen that there are many more possible pole locations near the point $z=1$, thus a large number of low frequency oscillations can be implemented.

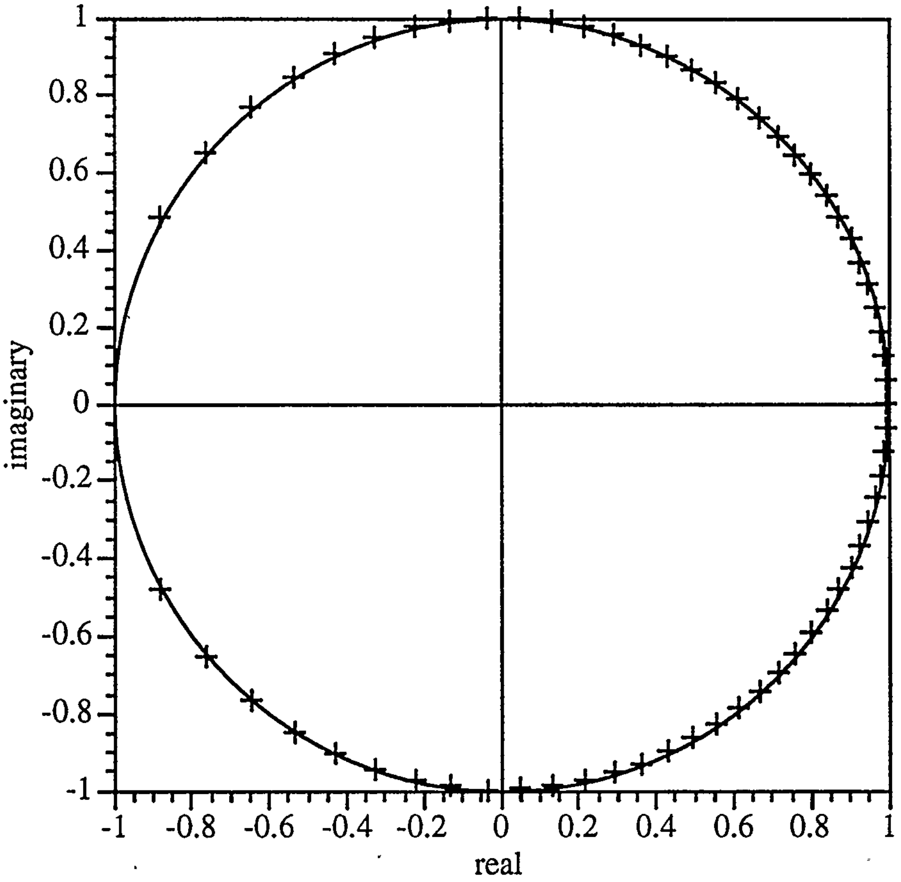

Figure 4.10 - Possible Pole Locations for LDI Oscillator

4.5 Oscillator Theory

According to classical oscillator theory, an oscillator can be viewed as a frequency selective network connected in a feedback configuration [37]. Such a structure is shown in Figure 4.11.

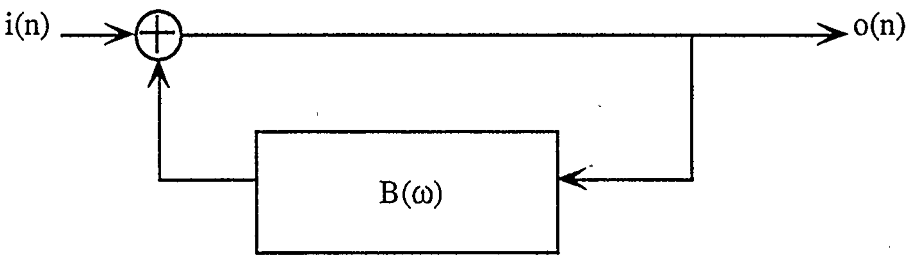

Figure 4.11 - General Feedback Oscillator

The output of the oscillator is denoted by *o(n)* and the input (an impulse to start the oscillations) is given by *i(n)*. If sinusoidal oscillations are to be sustained, the gain around the loop must be unity and the phase shift around the loop zero at the frequency of oscillation [37]. According to [37], this condition holds even if the frequency selective network includes non-linear elements, provided the effect of the non-linearities are not too great. In classical oscillator theory, it is assumed that the effects of the non-linearities do not dominate the circuit operation and that the circuit operates almost like a linear circuit. If a gross non-linearity is included, then the output will be far from sinusoidal. This same method will be used to analyze the LDI oscillator. First, the oscillator will be treated as an ideal circuit, free from non-linear quantization operations. Next, the effects of the quantization operations will be modeled under the assumption that the behaviour of the circuit does not differ greatly from the ideal circuit.

Assuming that the circuit is ideal, the oscillations will occur at the frequency where

$$B(\omega) = 1. \tag{4.23}$$

The LDI oscillator can be implemented in the form shown in Figure 4.11 if the frequency selective network is drawn as shown in Figure 4.12. This is the LDI oscillator redrawn with the feedback loop broken at the output of the forward Euler integrator. Assume that ideal arithmetic is used. This means that linear analysis techniques can be used. The transfer function of the oscillator is given by

$$\frac{O(z)}{I(z)} = \frac{1}{1 - B(z)} \tag{4.24}$$

where

$$B(z) = -ab\frac{z}{(z-1)^2}. \tag{4.25}$$

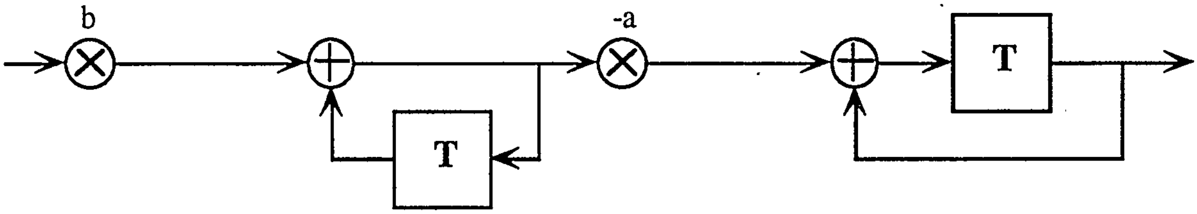

Figure 4.12 - LDI Frequency Selective Block

Sinusoidal oscillations will occur when $B(z) = 1$. First, consider the phase of *B(z)*, given by

$$\begin{aligned}\arg\left[B\left(e^{j\Omega T}\right)\right] &= \arg\left[-ab\frac{e^{j\Omega T}}{\left(e^{j\Omega T}-1\right)^2}\right] \\ &= \arg[-ab]+\arg\left[e^{j\Omega T}\right]-\arg\left[\left(e^{j\Omega T}-1\right)^2\right] \\ &= \pi+\Omega T-2\arg[\cos\Omega T-1+j\sin\Omega T] \\ &= \pi+\Omega T+2\tan^{-1}\left(\cot\frac{\Omega T}{2}\right) \\ &= \pi+\Omega T+2\left(\frac{\pi}{2}-\frac{\Omega T}{2}\right) \\ &= 2\pi.\end{aligned} \tag{4.26}$$

The phase shift is always 2π, provided $0 < ab < 4$. This leads to this important conclusion that the frequency of oscillation is determined solely by the magnitude characteristic of *B(z)*. The magnitude response of the frequency selective section is given by

$$\begin{aligned}\left|B\left(e^{j\Omega T}\right)\right| &= ab\frac{\left|e^{j\Omega T}\right|}{\left|\left(e^{j\Omega T}-1\right)^2\right|} \\ &= ab\frac{1}{(\cos\Omega T-1)^2+\sin^2\Omega T} \\ &= ab\frac{1}{2-2\cos\Omega T}.\end{aligned} \tag{4.27}$$

The frequency of oscillation, $\Omega_0 T$ is the point at which the magnitude response is unity. Thus, the frequency of operation can be solved for as follows:

$$\begin{aligned}&\left|B\left(e^{j\Omega_0 T}\right)\right| = 1 \\ &ab\frac{1}{2-2\cos\Omega_0 T} = 1 \\ &\Omega_0 T = \cos^{-1}\left(\frac{2-ab}{2}\right).\end{aligned} \tag{4.28}$$

This is equivalent to (4.17) as follows:

$$\begin{aligned}
\Omega_0 T &= \cos^{-1}\left(\frac{2-ab}{2}\right) \\
\cos(\Omega_0 T) &= \frac{2-ab}{2} \\
\cos(\Omega_0 T) &= 1-\frac{ab}{2} \\
\frac{ab}{2} &= 1-\cos(\Omega_0 T) \\
\frac{ab}{2} &= 2\sin^2\left(\frac{\Omega_0 T}{2}\right) \\
\frac{\sqrt{ab}}{2} &= \sin\left(\frac{\Omega_0 T}{2}\right) \\
\Omega_0 T &= 2\sin^{-1}\left(\frac{\sqrt{ab}}{2}\right) \\
\Omega_0 T &= 2\sin^{-1}\left(\frac{T}{2\sqrt{LC}}\right).
\end{aligned} \tag{4.29}$$

The next step is to consider the effects of implementing the oscillator using finite precision arithmetic. For this discussion, it will be assumed that the quantizers are located at the outputs of the multipliers and that the quantization operations will be magnitude truncation, rounding or two's complement truncation. The frequency selective network implemented using finite precision arithmetic is shown in Figure 4.13.

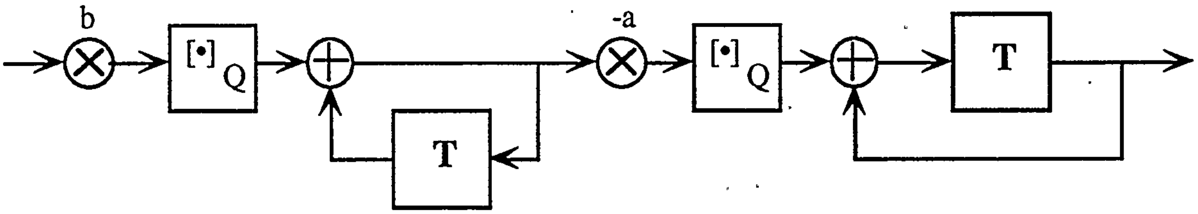

Figure 4.13 - Model of Finite Precision Frequency Selective Network

It is assumed that the amplitude of the signals in the circuit are many times greater than the quantization step size. Because the amplitude of the signals are many times greater than the quantization step size, the circuit will be assumed to operate in an almost linear fashion. If the amplitude of the signal in the oscillator is not sufficiently large then this model can no longer be considered valid.

Because the phase shift through the frequency selective network is independent of the coefficients chosen, the phase shift will be equal to 2π, even if multiplier outputs are quantized. This means that the quantizing operations will affect only the gain of the frequency selective network, and not the phase.

Consider the effects of using magnitude quantization. This reduces the magnitude of the output of the multiplier. Figure 4.14 shows the bounds on the output of a quantizing multiplier driven by a sine wave.

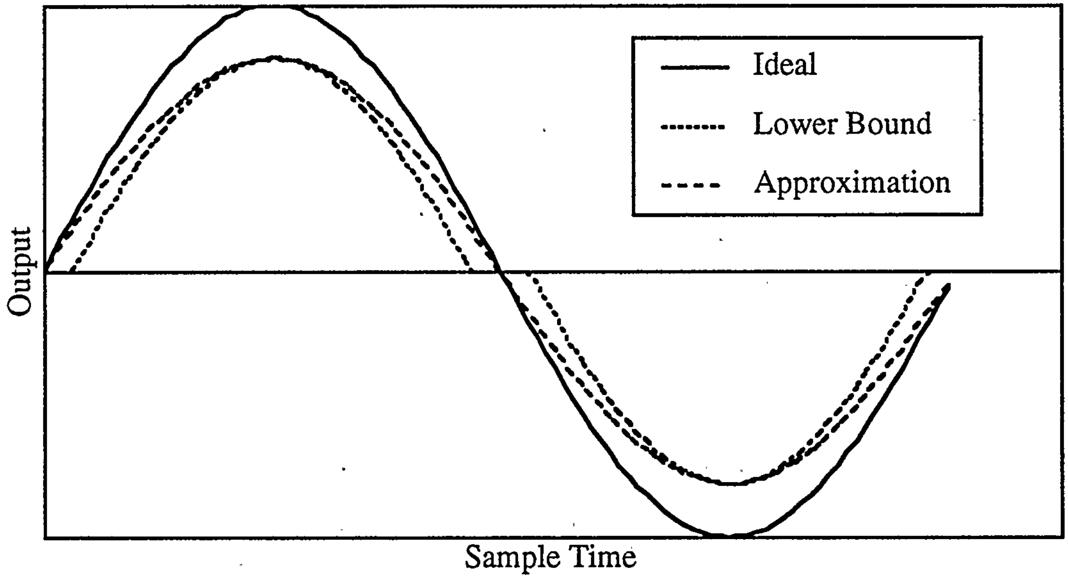

Figure 4.14 - Sinusoid After Magnitude Quantization

With a magnitude quantizing multiplier, the actual output will fall somewhere in between the ideal output and the lower bound given by the dotted line. For this analysis, assume that the output of the magnitude quantizing multiplier will be a sinusoid with an amplitude that falls between the ideal amplitude and the maximum lower bound value. Thus, the worst case quantization error will be approximated by the dotted line labeled "approximation" in Figure 4.14.

This approximation allows the effects of the non-linear quantization operation to be modeled as another multiplier in series with the original multiplier. The value of this multiplier is determined by the magnitude of the waveform being quantized. With magnitude quantization, the maximum the signal can be reduced by is one quantization interval, given by h. If the amplitude of the ideal output is given by X, then the amplitude of the approximation to the worst case output will be given by

$$X\left(1-\frac{h}{X}\right). \tag{4.30}$$

If V and I are the amplitudes of the two quadrature outputs (taken from the forward and backward integrator respectively), the amplitude of the signals at the output of the quantizing multipliers are given by

$$Vb\left(1-\frac{h}{Vb}\right) \tag{4.31}$$

and

$$Ia\left(1-\frac{h}{Ia}\right) \tag{4.32}$$

respectively. This leads to an expression for the gain of the frequency selective network when the approximation to the worst case quantization is used, namely

$$\left|B'\left(e^{j\Omega T}\right)\right| = a\left(1-\frac{h}{Vb}\right)b\left(1-\frac{h}{Ia}\right)\frac{1}{2-2\cos\Omega T}. \tag{4.33}$$

This in turn yields an estimate for the worst case frequency of oscillation

$$\Omega_0{}'T = \cos^{-1}\left(\frac{2 - ab\left(1 - h/_{Vb}\right)\left(1 - h/_{Ia}\right)}{2}\right). \tag{4.34}$$

Thus, the actual frequency of operation of the LDI oscillator implemented with magnitude quantizing multipliers is expected to fall between the ideal frequency, Ω_0T, and the estimated worst case frequency, Ω_0'T. Notice that the worst case estimate for the frequency of oscillation will be lower than ideal frequency because $\left(1-\frac{h}{Vb}\right)<1$ and $\left(1-\frac{h}{Ia}\right)<1$. Also note that the expressions for the worst case frequency are only valid when $Vb >> h$ and $Ia >> h$. This condition is met however, when values of *V* and *I* are chosen to be many times greater than the quantization interval, as stated before.

4.5.1 Example

An implementation of the LDI oscillator using magnitude quantization was constructed to test the above estimate for the worst case frequency of oscillation. The oscillator was implemented using the Texas Instruments TMS32010 and was created such that *a=b* and *V=I*. The testing was performed with the oscillator operating in real time and generating digital outputs which were converted to analog signals. Table 4.1 lists the ideal frequency of oscillation, the actual measured frequency and the estimate for the worst case frequency for a variety of coefficients and signal magnitudes. In all cases, where *Vb>>h* and *Ia>>h* the actual frequency of oscillation falls in between the ideal frequency and the estimated worst case frequency.

Table 4.1 - Ideal, Worst Case and Actual Frequencies of Oscillation

	Coefficient (a=b)				
V,I	0.1000	0.2000	0.3000	0.4000	0.5000
4	0.0159 *	0.0319 *	0.0479 0.0080 *0.0300*	0.0641 0.0239 *0.0400*	0.0804 0.0399 *0.0700*
8	0.0159 *	0.0319 0.0119 *0.018*	0.04793 0.02789 *0.0400*	0.0641 0.0440 *0.0530*	0.0804 0.0600 *0.074*
16	0.0159 0.0060 *0.0100*	0.0319 0.0219 *0.0259*	0.0479 0.0379 *0.0421*	0.0641 0.0540 *0.0560*	0.0804 0.0702 *0.0802*
32	0.0159 0.0109 *0.0140*	0.0319 0.0269 *0.0300*	0.0479 0.0429 *0.0460*	0.0641 0.0590 *0.0634*	0.0804 0.0753 *0.0804*
64	0.0159 0.0134 *0.0150*	0.0319 0.0294 *0.0300*	0.0479 0.0454 *0.0479*	0.0641 0.0616 *0.0641*	0.0804 0.0779 *0.0804*
128	0.0159 0.0147 *0.0150*	0.0319 0.0306 *0.0319*	0.0479 0.0467 *0.0479*	0.0641 0.0628 *0.0641*	0.0804 0.0792 *0.0804*
256	0.0159 0.0153 *0.0159*	0.0319 0.0313 *0.0319*	0.0479 0.0473 *0.0479*	0.0641 0.0635 *0.0641*	0.0804 0.0798 *0.0804*
512	0.0159 0.0156 *0.0159*	0.0319 0.0316 *0.0319*	0.0479 0.0476 *0.0479*	0.0641 0.0638 *0.0641*	0.0804 0.0801 *0.0804*
1024	0.0159 0.0158 *0.0159*	0.0319 0.0317 *0.0319*	0.0479 0.0478 *0.0479*	0.0641 0.0639 *0.0641*	0.0804 0.0803 *0.0804*

Each table entry lists ideal, worst case and *actual* frequency in that order. An asterisk indicates that the model is not valid for this entry because Vb, Ia are not >> h.

Figure 4.15 shows the frequency versus magnitude curve for a single set of coefficients. From this graph, it can be seen that as the signal magnitude (effectively the word length) increases, both the actual frequency of oscillation and the estimated worst case frequency approach the ideal frequency. This is what is expected to occur because as the word length increases, the effects due to quantization are reduced. As the word length increases to infinity, the errors will approach zero. It can be seen that the model suggested above is consistent with this idea. Further, it can be seen that the model predicts lower frequencies of operation when magnitude quantization is used. Again, this is consistent with what is expected. The multipliers are connected directly to the integrators. The multiplier output represents the change in the value of the integrators for each cycle. Magnitude quantization tends to reduce the magnitude of this change, which corresponds to a reduction in frequency. The model proposed yields results which are consistent with this notion as well. Because the model is consistent with these qualitative arguments and the observed behaviour of the prototypes match this model, this model will be used to estimate the worst case frequency of operation for a given set of coefficients and initial conditions.

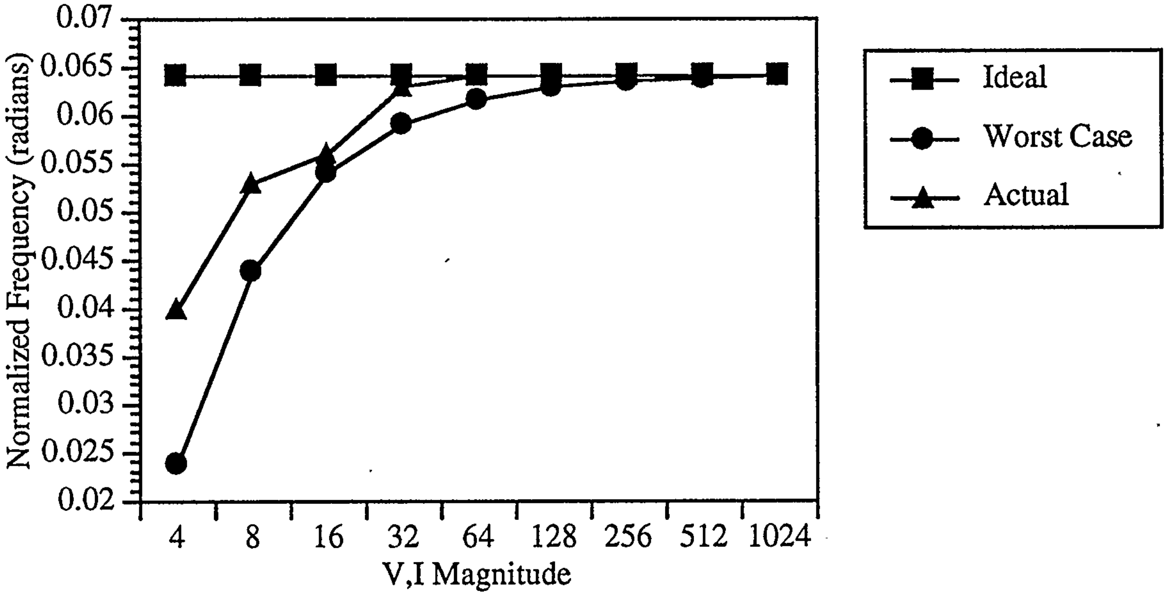

Figure 4.15 - Frequency Bounds versus Signal Magnitude

4.5.2 Rounding Multipliers

As in the case of magnitude quantizing multipliers, rounding multipliers can decrease the magnitude of the signal being multiplied, but unlike magnitude quantizing multipliers, they can also increase it. This will lead to an estimate for an upper bound on the frequency, as well as a lower bound. Consider the effect of quantizing a sine wave by rounding. The result will be bounded as shown in Figure 4.16. There will be both an upper bound and a lower bound.

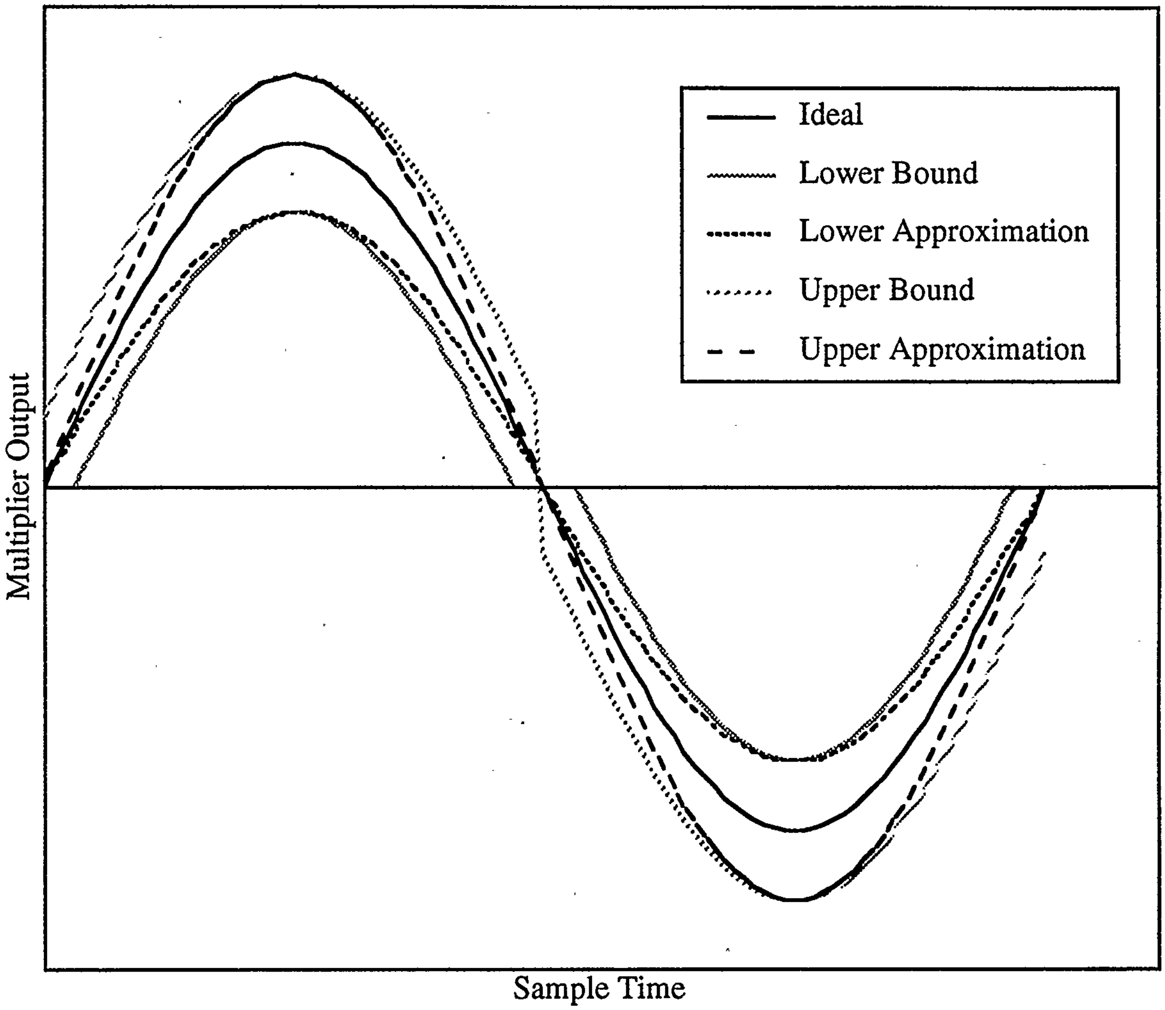

Figure 4.16 - Output of Rounding Multiplier

If the amplitude of the ideal output is X, the amplitude of the lower bound will be X-$h/2$. Thus, the amplitude of the sinusoidal approximation to the lower bound will be given by

$$X\left(1-\frac{h}{2X}\right). \tag{4.35}$$

Likewise, the amplitude of the approximation to the upper bound will be given by

$$X\left(1+\frac{h}{2X}\right). \tag{4.36}$$

Using the same arguments as in the previous section, expressions for both the upper and lower frequencies of operation can be derived. The expression for the lower frequency bound, denoted by $\Omega_l T$ is given by

$$\Omega_l T = \cos^{-1}\left(\frac{2-ab\left(1-{}^{h}\!/_{2Vb}\right)\left(1-{}^{h}\!/_{2Ia}\right)}{2}\right) \tag{4.37}$$

while $\Omega_u T$, the upper frequency bound is given by

$$\Omega_u T = \cos^{-1}\left(\frac{2-ab\left(1+{}^{h}\!/_{2Vb}\right)\left(1+{}^{h}\!/_{2Ia}\right)}{2}\right). \tag{4.38}$$

4.5.3 Example

As before, a prototype oscillator was constructed using the TMS32010 to test the proposed model. In this case, the oscillator was implemented using rounding as the quantization scheme and the coefficients were chosen such that *a*=*b* and *V*=*I*. Table 4.2 lists the ideal frequency of oscillation, the estimated minimum and maximum frequencies, as well as the actual measured frequency of oscillation. It can be seen that in all cases, where *Vb*>>*h* and *Ia*>>*h* the measured frequency of operation falls in between the upper and lower frequency estimates.

Table 4.2 - Ideal, Upper, Lower and Measured Frequencies of Oscillation

V,I	Coefficients				
	0.100	0.200	0.300	0.400	0.500
4	0.0159 * * *	0.0319 * * *	0.0479 0.0279 0.0682 *0.0455*	0.0641 0.0439 0.0845 *0.0714*	0.0804 0.0600 0.1012 *0.0703*
8	0.0159 * * *	0.0319 0.0219 0.0419 *0.0333*	0.0479 0.0379 0.0580 *0.0453*	0.0641 0.0540 0.0743 *0.0624*	0.0804 0.0702 0.0907 *0.0769*
16	0.0159 0.0109 0.0209 *0.0172*	0.0319 0.0269 0.0369 *0.0319*	0.0479 0.0429 0.0530 *0.0453*	0.0641 0.0590 0.0692 *0.0624*	0.0804 0.0753 0.0856 *0.0779*
32	0.0159 0.0134 0.0184 *0.0160*	0.0319 0.0294 0.0344 *0.0319*	0.0479 0.0454 0.0504 *0.0476*	0.0641 0.0616 0.0666 *0.0624*	0.0804 0.0779 0.0830 *0.0804*
64	0.0159 0.0147 0.0172 *0.0160*	0.0319 0.0306 0.0331 *0.0319*	0.0479 0.0467 0.0492 *0.0476*	0.0641 0.0628 0.0654 *0.0639*	0.0804 0.0791 0.0817 *0.0804*
128	0.0159 0.0153 0.0165 *0.0160*	0.0319 0.0313 0.0325 *0.0319*	0.0479 0.0473 0.0486 *0.0479*	0.0641 0.0635 0.0647 *0.0641*	0.0804 0.0798 0.0811 *0.0804*
256	0.0159 0.0156 0.0162 *0.0160*	0.0319 0.0316 0.0322 *0.0319*	0.0479 0.0476 0.0482 *0.0479*	0.0641 0.0638 0.0644 *0.0641*	0.0804 0.0801 0.0808 *0.0804*
512	0.0159 0.0158 0.0161 *0.0160*	0.0319 0.0317 0.0320 *0.0319*	0.0479 0.0478 0.0481 *0.0479*	0.0641 0.0639 0.0643 *0.0641*	0.0804 0.0803 0.0806 *0.0804*
1024	0.0159 0.0158 0.0160 *0.0159*	0.0319 0.0318 0.0320 *0.0319*	0.0479 0.0478 0.0480 *0.0479*	0.0641 0.0640 0.0642 *0.0641*	0.0804 0.0804 0.0805 *0.0804*

Each table entry lists ideal, worst case upper, worst case lower and *actual* frequency in that order. An asterisk indicates that the model is not valid for this entry because Vb, Ia are not >> h.

4.5.4 Two's Complement Quantization

If two's complement truncation operations are placed at the outputs of the multipliers, the result will be similar to that of placing rounding quantizers at the multiplier output. That is, there will be an upper and a lower bound on the magnitude of the signals at the multiplier outputs, and hence an upper and a lower bound on the frequencies of oscillation. Consider the effect of quantizing a sine wave by two's complement truncation. The result will be bounded as shown in Figure 4.17.

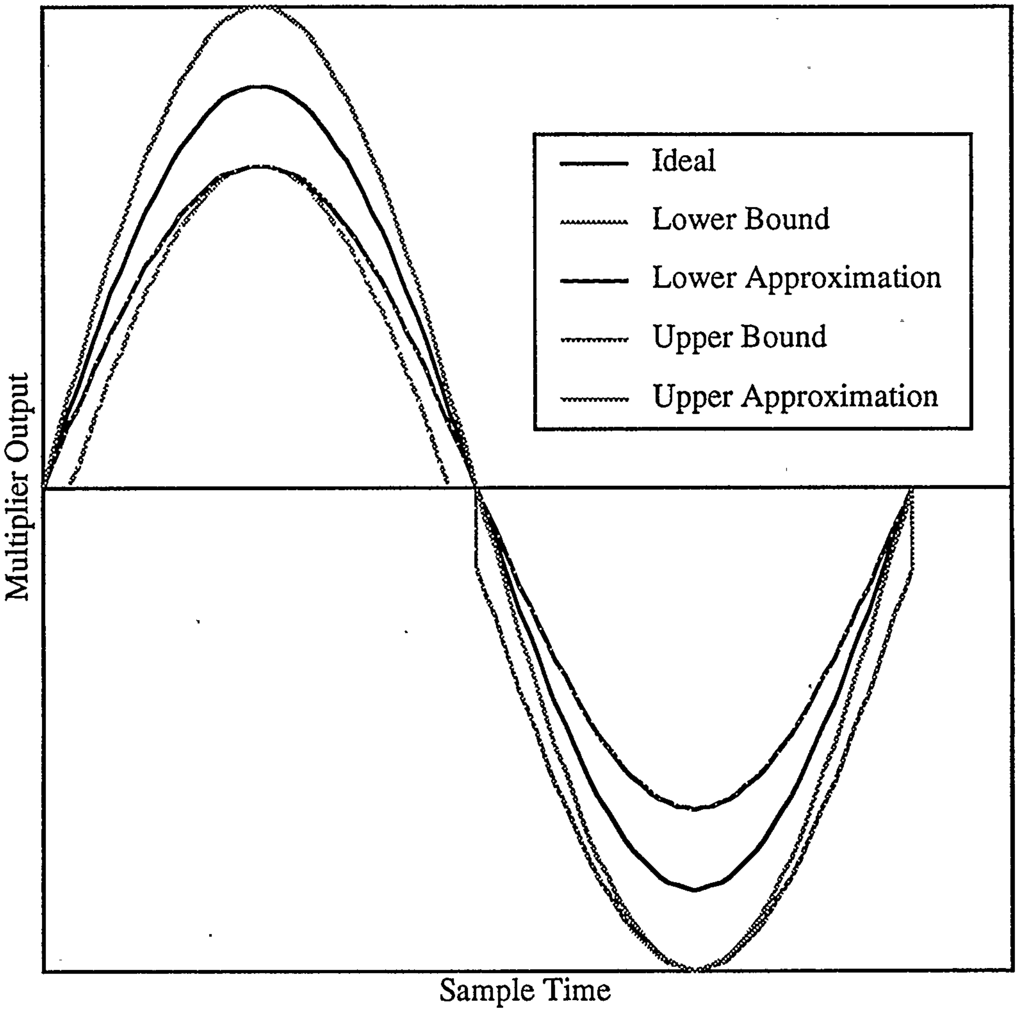

Figure 4.17 - Output of Two's Complement Truncation

If the amplitude of the ideal output is *X*, the amplitude of the lower bound will be *X-h*. Thus, the amplitude of the sinusoidal approximation to the lower bound will be given by

$$X\left(1-\frac{h}{X}\right). \tag{4.39}$$

Likewise, the amplitude of the approximation to the upper bound will be given by

$$X\left(1+\frac{h}{X}\right). \tag{4.40}$$

Using the same arguments as before, expressions for both the upper and lower frequencies of operation can be derived. The expression for the lower frequency bound, denoted by $\Omega_l T$ is given by

$$\Omega_l T = \cos^{-1}\left(\frac{2-ab\left(1-{}^{h}\!/\!_{Vb}\right)\left(1-{}^{h}\!/\!_{Ia}\right)}{2}\right) \tag{4.41}$$

while $\Omega_u T$, the upper frequency bound is given by

$$\Omega_l T = \cos^{-1}\left(\frac{2-ab\left(1+{}^{h}\!/\!_{Vb}\right)\left(1+{}^{h}\!/\!_{Ia}\right)}{2}\right). \tag{4.42}$$

4.5.5 Example

A prototype oscillator was constructed using the TMS32010 to test the proposed model. In this case, the oscillator was implemented using two's complement truncation as the quantization scheme and the coefficients were chosen such that *a=b* and *V=I*. Table 4.3 lists the ideal frequency of oscillation, the estimated minimum and maximum frequencies, as well as the actual measured frequency of oscillation. It can be seen that in all cases where *Vb>>h* and *Ia>>h*, the measured frequency of operation falls between the upper and lower frequency estimates.

Table 4.3 - Ideal, Upper, Lower and Measured Frequencies of Oscillation (Two's Complement)

V,I	Coefficients				
	0.100	0.200	0.300	0.400	0.500
4	0.0159 * * *	0.0319 * * *	0.0479 0.0080 0.0887 *0.0556*	0.0641 0.0239 0.1054 *0.0555*	0.0804 0.0399 0.1224 *0.0820*
8	0.0159 * * *	0.0319 0.0119 0.0520 *0.0294*	0.0479 0.0279 0.0682 *0.0500*	0.0641 0.0439 0.0845 *0.0625*	0.0804 0.0600 0.1012 *0.0820*
16	0.0159 0.0060 0.0259 *0.0151*	0.0319 0.0219 0.0419 *0.0334*	0.0479 0.0379 0.0580 *0.0465*	0.0641 0.0540 0.0743 *0.0625*	0.0804 0.0702 0.0907 *0.0806*
32	0.0159 0.0109 0.0209 *0.0168*	0.0319 0.0269 0.0369 *0.0325*	0.0479 0.0429 0.0530 *0.0476*	0.0641 0.0590 0.0692 *0.0625*	0.0804 0.0753 0.0856 *0.0807*
64	0.0159 0.0134 0.0184 *0.0162*	0.0319 0.0294 0.0344 *0.0321*	0.0479 0.0454 0.0504 *0.0476*	0.0641 0.0616 0.0666 *0.0639*	0.0804 0.0779 0.0830 *0.0807*
128	0.0159 0.0147 0.0172 *0.0160*	0.0319 0.0306 0.0331 *0.0320*	0.0479 0.0467 0.0492 *0.0479*	0.0641 0.0628 0.0654 *0.0639*	0.0804 0.0791 0.0817 *0.0804*
256	0.0159 0.0153 0.0165 *0.0160*	0.0319 0.0313 0.0325 *0.0320*	0.0479 0.0473 0.0486 *0.0479*	0.0641 0.0635 0.0647 *0.0640*	0.0804 0.0798 0.0811 *0.0804*
512	0.0159 0.0156 0.0162 *0.0160*	0.0319 0.0316 0.0322 *0.0320*	0.0479 0.0476 0.0482 *0.0479*	0.0641 0.0638 0.0644 *0.0641*	0.0804 0.0801 0.0808 *0.0804*
1024	0.0159 0.0158 0.0161 *0.0159*	0.0319 0.0317 0.0320 *0.0319*	0.0479 0.0478 0.0481 *0.0479*	0.0641 0.0639 0.0643 *0.0641*	0.0804 0.0803 0.0806 *0.0804*

Each table entry lists ideal, worst case upper, worst case lower and *actual* frequency in that order. An asterisk indicates that the model is not valid for this entry because Vb, Ia are not >> h.

4.5.6 Limitations and Observations

The above model has been developed as a means of predicting the performance of the oscillator when implemented with finite precision arithmetic. While the model has been shown to realistically predict the performance of the oscillator, several points should be kept in mind. First, it was assumed that the linear operation of the oscillator would dominate. That is, the sinusoidal oscillation represents the major component of the signal in the oscillator. It has been observed that if the magnitude of the oscillations is many times greater than the quantization interval, then this will indeed be the case. Secondly, the proposed model attempts to approximate the behaviour of a non-linear system with a linear system. This is a fairly loose approximation, and should not be considered a rigorous analysis. Thus, while the bound has been observed to realistically model the performance of the oscillator, it can in no way be considered a guarantee of performance. Although a rigorous guarantee of this model's correctness cannot be provided, several arguments can be made in favour of using this model. The first argument is that the phase shift through the frequency selective network does not depend on the multipliers. Thus, the multiplier values will affect only the magnitude response. The quantization effects will change the frequency of operation by changing the magnitude response of the frequency selective section, but not the phase. This allows the effects of quantization to be modeled as a simple change in the magnitude response of the frequency selective section This analysis takes advantage of the unique structure of the LDI oscillator and cannot be extended to include other structures discussed in this thesis, such as the DF2. Secondly, this model presumes that the linear mode of operation dominates the oscillators' operation. This has been seen to be the case in all prototypes constructed provided that the amplitude of oscillations is many times greater than the quantization interval. Thirdly, the model makes quantitative predictions about the behaviour of the oscillator that agree with qualitative arguments about its behaviour. Lastly, the model has been shown to agree with the performance of actual prototypes. Thus, based on these arguments, it is felt that the model presented allows approximations to the worst case performance to be computed, although these approximations cannot, and should not be considered rigorous guarantees of performance, or of the continuing oscillation of the circuit.

Chapter 5

Implementation of A Digital Sinusoidal Oscillator

In this chapter, the design of a high precision audio frequency digital sinusoidal oscillator suitable for use in a laboratory or test setting is discussed. To this end, it is required that the oscillator produce sinusoids up to 50 kHz selectable in at least 0.5 Hz increments, that the oscillator have an output word length of at least 16 bits, that the implementation use no more than a single chip for the digital section, and that an easy to use interface be included.

In the previous chapter, two different methods of approximating discrete time sinusoids were presented: pre-computed lookup tables and second order structures with poles on the unit circle. Because these structures are so varied, it is not a simple matter of using the "best" one to implement a sinusoidal oscillator because each will have different strengths and weaknesses. Of the second order structures presented, the wave digital oscillator and the normal form oscillator can be eliminated immediately because they were shown to be unfeasible when finite precision arithmetic is used. For the remaining structures, the strengths and weaknesses of each must be considered relative to the oscillator requirements.

Facilities for fabricating the oscillator were made available by the Alberta Microelectronic Centre. The Actel A1280 field programmable gate array (FPGA) was provided, along with computer and programming services. The A1280 is a one time field programmable logic device consisting of 608 combinational modules and 624 sequential modules which can be combined in an arbitrary fashion [38]. The combinational modules are used to implement combinational logic functions while the sequential modules are used to implement flip flops of various types. These can be combined to achieve a total of 998 flip flops. A maximum of 140 I/O pins are available and the manufacturer claims speeds of up to 33 MHz for a 16 bit accumulator. It is based on these device characteristics that the decisions regarding implementation of the oscillator will be made.

5.1 Selecting a Structure

In the previous section, three different structures were found to be feasible for implementing digital sinusoidal oscillators. These are the ROM based lookup table method, the DF2 oscillator, and the LDI oscillator. Each of these will be examined with respect to the requirements stated above and the available hardware.

5.1.1 Frequency Range

The first requirement that needs to be met is that the oscillator must be capable of generating audio frequency sinusoids. With the ROM based oscillator, the limiting factors on the speed of operation are the speed of the accumulator and the speed of the ROM. With the Actel A1280, the maximum speed of the device is given by the manufacturer as 33 MHz for a 16 bit accumulator. This fast enough to construct both an accumulator and a ROM which allow the oscillator to operate in the audio frequency range. In general, the parallel structure of the ROM based oscillator lends itself to high speed operation, as is the case here. Because the ROM based oscillator uses parallel arithmetic, it can generate a single output per clock cycle. This allows sample frequencies in the tens of megahertz, well above the states requirements.

The DF2 and LDI structures must be approached somewhat differently. Because these oscillators require more complex arithmetic than the ROM based oscillator, special consideration must be given to the size of the arithmetic elements. In particular, parallel multipliers are known to be simply too large to fit on a chip such as the A1280 [25]. Bit serial arithmetic, in which computations are performed one binary digit at a time, on the other hand, yield structures which are compact enough to fit on an A1280 [25, 26, 39]. It is also known that bit serial systems can easily be constructed which operate in the audio range [25, 39]. Further, a well characterized design system, allowing automated synthesis of digital hardware from a signal flowgraph exists [40]. With these facts in mind, it is decided that the DF2 and the LDI oscillators will be implemented using bit serial arithmetic if chosen. This will allow these systems to operate in the audio frequency range while fitting on a single A1280 part.

5.1.2 Frequency Resolution

All three structures, the ROM based oscillator, the DF2 oscillator and the LDI oscillator can operate in the audio frequency range. The next requirement to be considered is the frequency resolution of 0.5 Hz. Before this requirement can be examined in detail, it is necessary to select a sample frequency. In theory, it is possible to recover a band limited continuous signal from a discrete time signal up to the Nyquist rate or the half sample frequency [41]. Thus, it is theoretically possible to construct a digital sinusoidal oscillator with a sample frequency of 100 kHz and exactly recover continuous sinusoids of 50 kHz using an ideal lowpass reconstruction filter. Unfortunately, ideal lowpass filters cannot be constructed. A practical analog filter can be used for reconstruction however if the sample frequency is greater than the Nyquist rate [12]. With this in mind, the sample frequency was specified as being three times the Nyquist rate, a typical value for most practical systems [2]. This means that a 50 kHz sinusoid will be generated with six samples per cycle of the sinusoid and that the sample frequency will be 300 kHz.

According to the relations derived in the Chapter 4, the ROM based oscillator requires a 20 bit phase accumulator. This could be constructed on the A1280 part, and could be made to operate at 300 kHz. Thus, achieving a frequency resolution of 0.5 Hz is not a problem using the ROM based structure.

The frequency resolution of the DF2 oscillator is determined by the coefficient word length of the multiplier. The oscillator was shown to be most sensitive to the coefficient at low frequencies. Using the relations derived in Chapter 4, a frequency resolution of at least 0.5 Hz at a sample frequency of 300 kHz over the entire frequency range requires 34 fractional bits for the coefficient.

The frequency resolution of the LDI oscillator is also determined by the number of fractional bits in the coefficient. It is assumed that both coefficients in the oscillator will be equal. This is beneficial from the standpoint of signal scaling as the magnitude of the signals in both the forward and backward integrators will be the same. As well, there will be an almost linear relationship between coefficient value and output frequency because for small values of $\Omega_0 T$, $\sin(\Omega_0 T) \approx \Omega_0 T$. To achieve a frequency resolution of 0.5 Hz with a sample frequency of 300 kHz, it is necessary to have at least 17 fractional coefficient bits. This is a feasible size of multiplier, and much smaller than the 34 bit multiplier required by the DF2 oscillator.

5.1.3 Implementation Size

The requirement that the oscillator output be 16 bits wide dictates the size of the ROM for the ROM based oscillator. If the entire 20 bits of the phase accumulator are used as the index into the ROM and the output is 16 bits wide, then 16 megabits of ROM are required. Memory elements on gate array style chips require at least one flip flop per bit of storage. The A1280 has an absolute maximum of 998 flip flops, far short of the 16 million required. If the number of points in the sinusoid stored in the ROM is reduced, the ROM based oscillator be made to fit on the A1280. Assuming that all 998 flip flops are available for storing the sine wave, only 62 data points can be stored. If only one quarter of the sinusoid is stored on the chip, and the entire wave computed from these values as described in the previous chapter, it is only possible to have 248 sample per sine wave period; too few to be acceptable at lower frequencies. Note that this estimate assumes that the entire A1280 is devoted to the look up table. This estimate neglects the fact that extra circuitry is required for the phase accumulator and the user interface. The size problem could be circumvented if external ROM chips were used, but this was deemed unacceptable as the final oscillator is to occupy only a single chip.

Both the DF2 and the LDI oscillators have wordlengths greater than 16 bits. This is because the wordlength of the bit serial systems must be at least as long as the latency, or number of bit times required to compute an output, of the multiplier. This corresponds to a minimum 52 bit system word length for the DF2 oscillator and a minimum 28 bit wordlength for the LDI oscillator. A 16 bit output can be derived from the serial word, simply by converting the 16 most significant bits to parallel format and discarding the least significant bits.

5.1.4 The Choice

The ROM based oscillator can be removed from further consideration because it cannot fit on a single A1280 chip. Although it is capable of operating at much higher speeds than the bit serial oscillators, this is not a requirement for this application. All that remains is to decide between the DF2 and LDI structures. It has been shown that both can be constructed to meet the specifications listed at the beginning of the chapter. There are several additional factors to consider. First, the DF2 oscillator is known to have stability problems [30, 32]. While these problems can be corrected with methods which force the stability, these methods require external hardware to implement. As well, some require trial and error searches of the state space, something not feasible for a system with a 52 bit word length. On the other hand, the LDI oscillator prototypes described in Chapter 4 did not exhibit stability problems. The model developed in the previous chapter predicts that the frequency of oscillation will be stable within 23μ Hz for a sample frequency of 300 kHz. This corresponds to an accuracy of 73.6×10^{-6} parts per million. In actual practice, it is not likely that the oscillator will operate with the degree of accuracy. The chip will be driven by a crystal master clock, and the accuracy of this master clock determines the accuracy of the oscillator. Typical crystal oscillators used in digital systems have accuracy's in the order of ten's of parts per million [42, 43] causing the frequency variation to be greater than the predicted 23 μHz. As well, the LDI oscillator requires a smaller coefficient word length than the DF2, and hence a shorter signal word length, reducing the circuit size. A further advantage of the LDI structure is that it automatically generates quadrature signals, that is both sine and cosine terms. It is based on these arguments that the LDI oscillator structure was chosen to implement the oscillator as specified at the beginning of the chapter.

5.2 Implementing the LDI Oscillator

Now that the LDI oscillator structure has been selected, it is necessary to determine the most efficient means of implementing it. As stated before, bit serial arithmetic will be used in the implementation. The principle of bit serial arithmetic is to perform arithmetic operations one bit at a time, producing a result one bit at a time [26]. Thus, a single wire can carry an entire signal, regardless of wordlength. There are several ways in which the LDI oscillator can be implemented if bit serial arithmetic is used. These are:

- two multipliers with no multiplexing
- one multiplier with no multiplexing
- one multiplier multiplexed two ways
- one multiplier implementing a difference equation.

Each of these structures will be evaluated.

5.2.1 Two Multipliers - No Multiplexing

The LDI oscillator can be implemented with two multipliers simply by replacing the components in Figure 4.9 with their corresponding bit serial equivalents as shown in Figure 5.1. This figure shows the main components of the circuit, excluding control signals. Since the A1280 does not have a global reset to each flip flop, some multiplexors must be included to break the feedback loops and allow initial conditions to be set. In the above diagram, *m* represents the latency of the multiplier. In the previous section it was found that a minimum of 17 fractional bits were required for the desired frequency resolution.

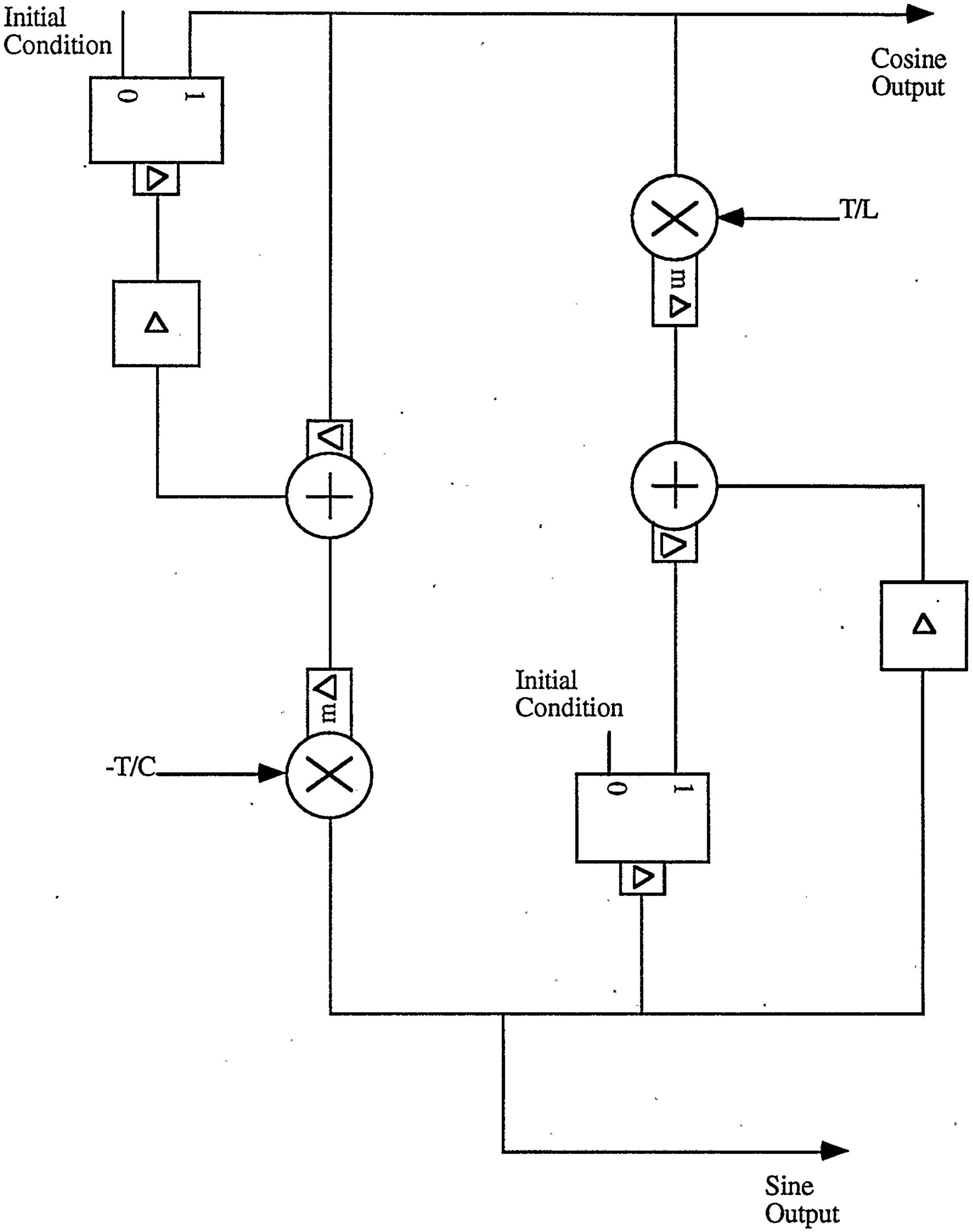

Figure 5.1 - Two Multiplier Implementation

With the addition of a sign bit, a total of 18 bits of coefficient word length is required. The multipliers used have a latency equal to 28 bit clock cycles. The total latency around the main loop of the oscillator must be equal to one word, therefore one word must be no shorter than 59 bits. While it is feasible to construct the oscillator this way, the word length is much longer than it needs to be.

5.2.2 One Multiplier - No Multiplexing

A straightforward solution to the problem of excessive word length is to simply remove one multiplier from the circuit shown in Figure 5.1. This leads to the circuit shown in Figure 5.2. It can be verified that this circuit, which is equivalent to setting the T/L coefficient to one, will still function as desired, provided $0<T/C<4$.

The minimum word length for this structure is equal to 31 bits. This is certainly more compact than the previous structure. Unfortunately, this leads to a problem of signal scaling. The signal in the backward integrator will always be greater in amplitude than that in the forward integrator. This will require either that the word length be increased to prevent overflow or that the dynamic range of the cosine signal be reduced. Neither of these options are particularly desirable. Further, the differences in signal magnitude will change as the multiplier coefficient changes which is highly undesirable.

5.2.3 One Multiplier Multiplexed

The advantages of the one multiplier solution, namely short wordlength and low hardware requirements, can be combined with the desirable signal scaling characteristics of the two multiplier solution by multiplexing a single multiplier.

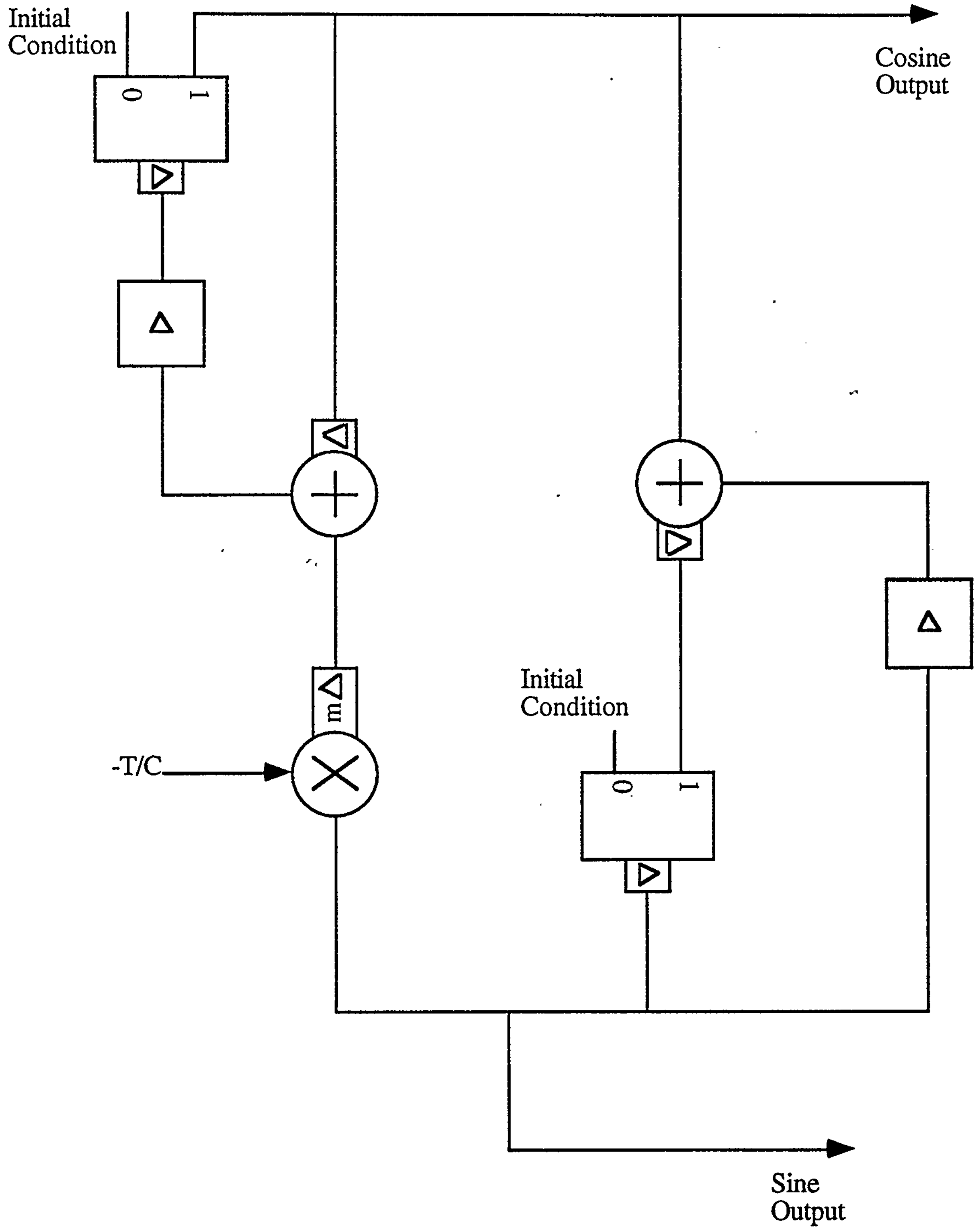

Figure 5.2 - One Multiplier Implementation

One such structure that can be used is shown in Figure 5.3. This structure was developed by Peter Graumann and L. E. Turner. The multiplexors in the system allow the initial conditions to be set explicitly, and also allow the multiplier to be shared between the two integrators in the circuit. Typically, multiplexing a multiplier will reduce the throughput of a bit serial system [39]. For this particular multiplexing scheme, the wordlength is 34 bits. Thus, one output will be generated every 68 bit times, a slightly longer time than is required by the two multiplier solution. The amount of hardware required is comparable to that of the one multiplier solution. Thus, this system retains the signal scaling advantages of the two multiplier solution while having the compact size of the one multiplier solution. The drawback is that the speed of operation is reduced.

5.2.4 One Multiplier Difference Equation

It is possible to simplify the implementation of the oscillator by using the unique structure of the LDI oscillator. The difference equation for the LDI oscillator is written as

$$\begin{aligned} i_{n+1} &= i_n + \frac{T}{L} v_n \\ v_{n+1} &= v_n - \frac{T}{C} i_n - \frac{T^2}{LC} v_n \end{aligned} \tag{5.1}$$

but can also be rearranged as

$$\begin{aligned} i_{n+1} &= i_n + \frac{T}{L} v_n \\ v_{n+1} &= v_n - \frac{T}{C} i_{n+1} \end{aligned} \tag{5.2}$$

This implies that i_{n+1} and v_{n+1} can be computed sequentially. A simple structure to implement this is shown in Figure 5.4.

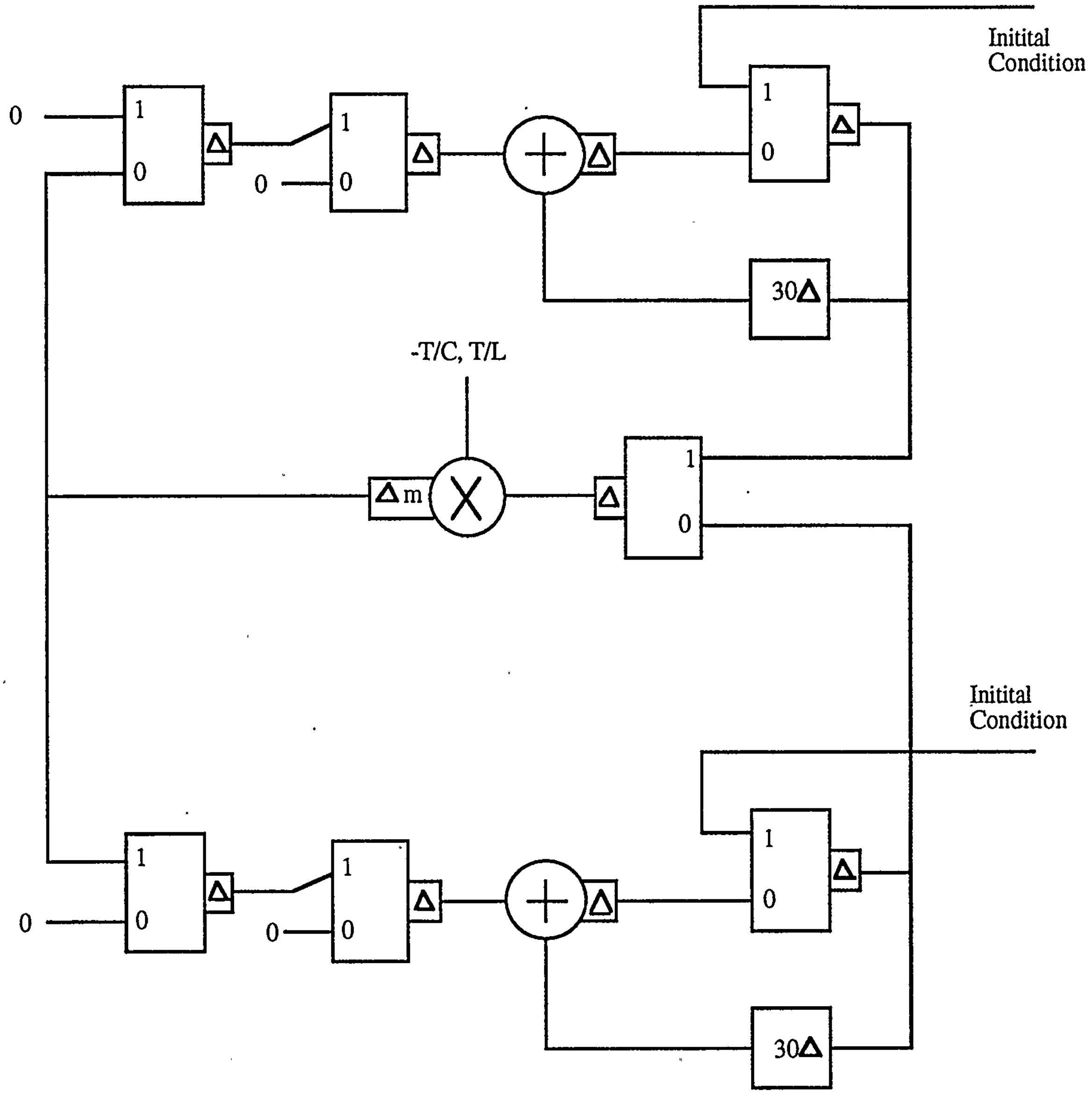

Figure 5.3 - Multiplexed Implementation

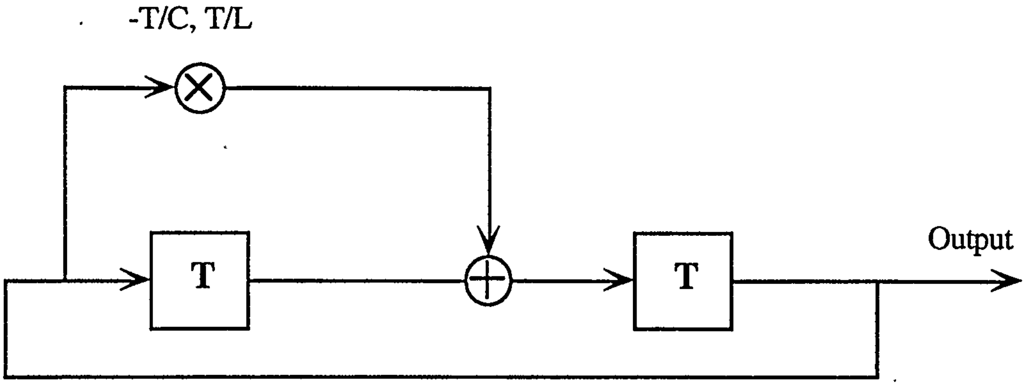

Figure 5.4 - Difference Equation Structure

With the above structure, the multiplier coefficient must alternate between $-T/C$ and T/L on alternate word times. The output will be equal to v and i on alternate word times as well. The above structure is shown assuming that the multiplier has no latency or delay. In fact, the bit serial multiplier has a latency associated with it. The structure shown in Figure 5.5 accounts for this and allows a bit serial implementation to be constructed.

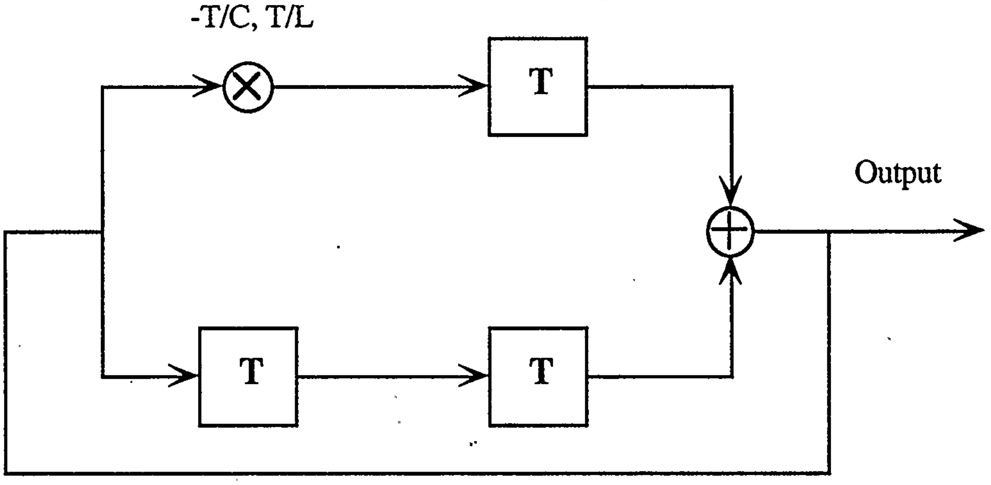

Figure 5.5 - Modified Difference Equation Structure

This structure can easily be translated into a bit serial implementation as shown in Figure 5.6.

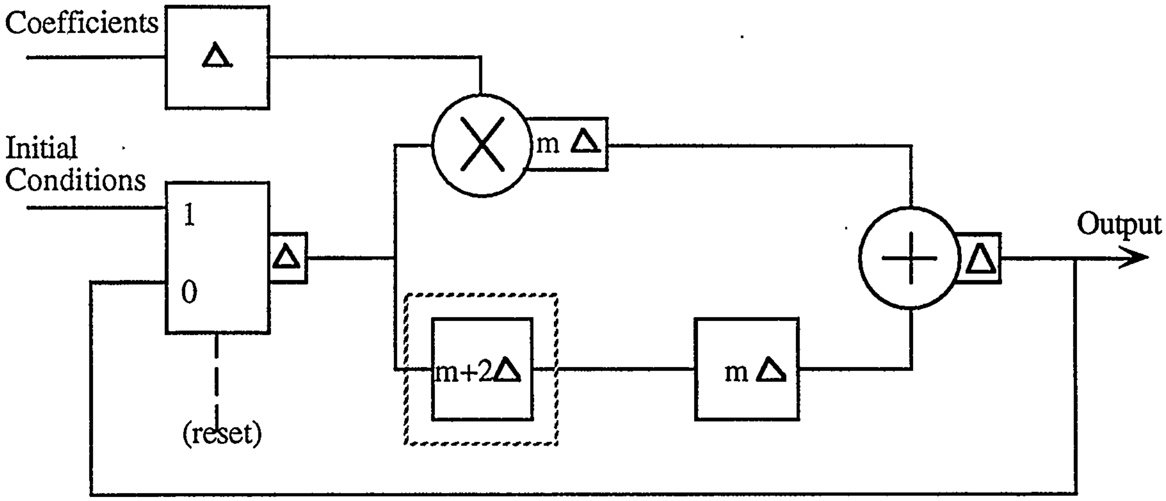

Figure 5.6 - Bit Serial Difference Equation Implementation

The minimum wordlength of this system is determined by the latency of the feedback loop through the multiplier as before, and is equal to 30 bits. Because two complete word times are required to generate an output, a total of 60 bit times are required for each output. Note that the delay element inside the dotted box is included in the multiplier unit and does not need to be constructed separately.

Figure 5.7 summarizes the size and speeds of the various bit serial implementations. The x axis lists the number of bit times required to generate a single word or sample at the output while the y axis lists the estimated size in Actel logic modules. Note that the size estimate does not include control circuitry, coefficient storage or input and output circuits. It is reasonable to assume that these sections will be similar for each implementation. A 20 MHZ clock was chosen as the maximum acceptable bit rate. This speed can be achieved in a straightforward fashion with a gate array such as the A1280. While higher speeds are possible given sufficient fine tuning of the circuit, it was decided to avoid this if possible. If a 300 kHz sample rate is to be achieved, then a single sample can take no longer than 66 bit times to be generated. Thus, any acceptable solution must be to the left of the dotted line at 66 bit times, as shown in Figure 5.7.

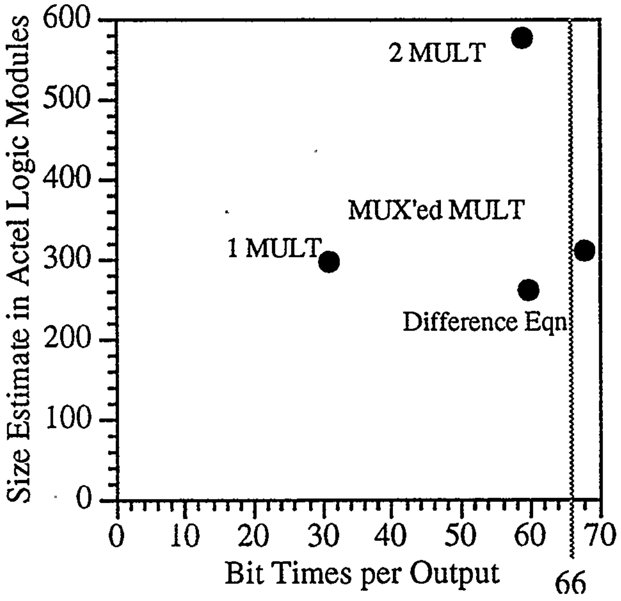

Figure 5.7 Size and Speed Comparison

It can be seen that the multiplexed one multiplier solution, which requires 68 bit times to generate each sample, exceeds this limit. While it may be possible to fine tune this circuit to achieve the desired throughput, this is unnecessary as alternative solutions exist. The fastest implementation is the one multiplier solution. This solution was deemed to be unacceptable, however, because of the variations in signal amplitude throughout the circuit that are inherent in this design. The two remaining choices are the difference equation implementation and the two multiplier solution. It can be seen that the difference equation solution offers almost the same speed as the two multiplier solution, yet requires less than half as much hardware. Clearly, this is the optimal choice for implementation because it offers acceptable speed with the most compact hardware. The difference equation solution was used in the final implementation on the A1280 device.

5.3 Implementation Details

In addition to the bit serial circuit shown in Figure 5.6, several other components were required for a complete implementation. Some means of generating and storing the oscillator coefficients is required, as is a user interface.

In order to simplify coefficient storage, it was decided that the coefficients should be fixed such that $-T/C = T/L$. There are several advantages to doing so. First, this ensures that the amplitude of both the sine and cosine waveforms will be equal, regardless of the coefficient values. Secondly, only a single coefficient needs to be stored. The other coefficient can be simply and efficiently generated by computing the negative of the stored coefficient. This reduces the size of the coefficient unit.

The coefficients of the oscillator determine the frequency of oscillation and the structure of the coefficient unit is dictated by the user interface. Therefore, the user interface must be specified before the coefficient unit can be completely design.

It was decided that the user interface would consist of an input for setting the frequency range, increment and decrement push buttons and fine increment and decrement push buttons. The frequency range control is used to select the initial operating frequency from a limited number of choices. The increment and decrement as well as the fine increment and decrement controls are used to adjust the frequency of oscillation. In the final implementation, the frequency range control was a 5 bit bus, allowing 32 different frequency ranges to be selected. It was decided to divide the frequency ranges into 32 coarse increment and decrement divisions. This allows frequencies to be specified to within approximately 0.1% full scale range using only the range control and the coarse increment and decrement. As well, this accuracy can be achieved with a single range selection and no more than 16 presses of the increment or decrement controls. The fine increments were defined to be the smallest amount of change possible (i.e. one least significant bit) of the coefficient.

This form of interface led to the structure for storing and generating coefficients shown in Figure 5.8. This structure is based on a recirculating loop with a length equal to the system word length to ensure synchronization. The loop is broken with a multiplexor to allow the frequency range to be set. The coefficient for the initial frequency range is simply the 5 bit range input connected to the most significant fractional bits of the coefficient word. The control for the multiplexor, as well as the oscillator reset circuitry, is connected to the 5 bit input through a state machine so that the entire oscillator is reset whenever the range setting changes. The increment and decrement controls are connected to state machines so that whenever one of these buttons is pressed, an appropriate value is added to or subtracted from the value stored in the recirculating register.

During the schematic capture and simulation phase of construction, it was found that significant amounts of space remained available on the A1280. It was decided to increase the signal word length from 30 bits to 32 bits. The maximum expected speed of operation of the circuit is 20 MHz; a 32 bit word length allowed a maximum sample rate of 312.5 kHz. Because the bit serial multipliers cannot represent coefficients greater in magnitude than one, the frequency of oscillation is limited to one sixth the sample frequency. This yields a maximum sinusoidal frequency of 52.1 kHz, slightly greater than that originally specified. The extra two bits of wordlength increase the accuracy of the oscillator without reducing the speed unacceptably.

The remainder of the extra chip space was used to construct a frequency counter. This allows the user to set the frequency of oscillation without using an external frequency counter. The counter functions by counting the number of rising edge zero crossings in a tenth of a second time period. The number of such transitions is equal to ten times the frequency in kHz. An display driver, capable of driving seven segment LCD and LED displays was included in the final chip implementation. As the frequency counter was not considered to be an essential part of the circuit, it was felt that making the counter accurate to 0.1 kHz was acceptable.

Both the sine and cosine outputs of the oscillator are brought out in parallel format. Because many I/O pins were available, it was decided to make the outputs 18, rather than 16 bits wide. This allows the use of high precision D/A's to be used in conjunction with the oscillator. In addition, a format control allowing the output to be in either two's complement or offset binary format was included to allow a broad range of D/A's to be connected to the oscillator.

The circuit described above was programmed into the Actel A1280 device. In order to simplify testing, a printed circuit board containing the oscillator and user interface, including an LCD display for the frequency counter and two D/A's with reconstruction filters was constructed.

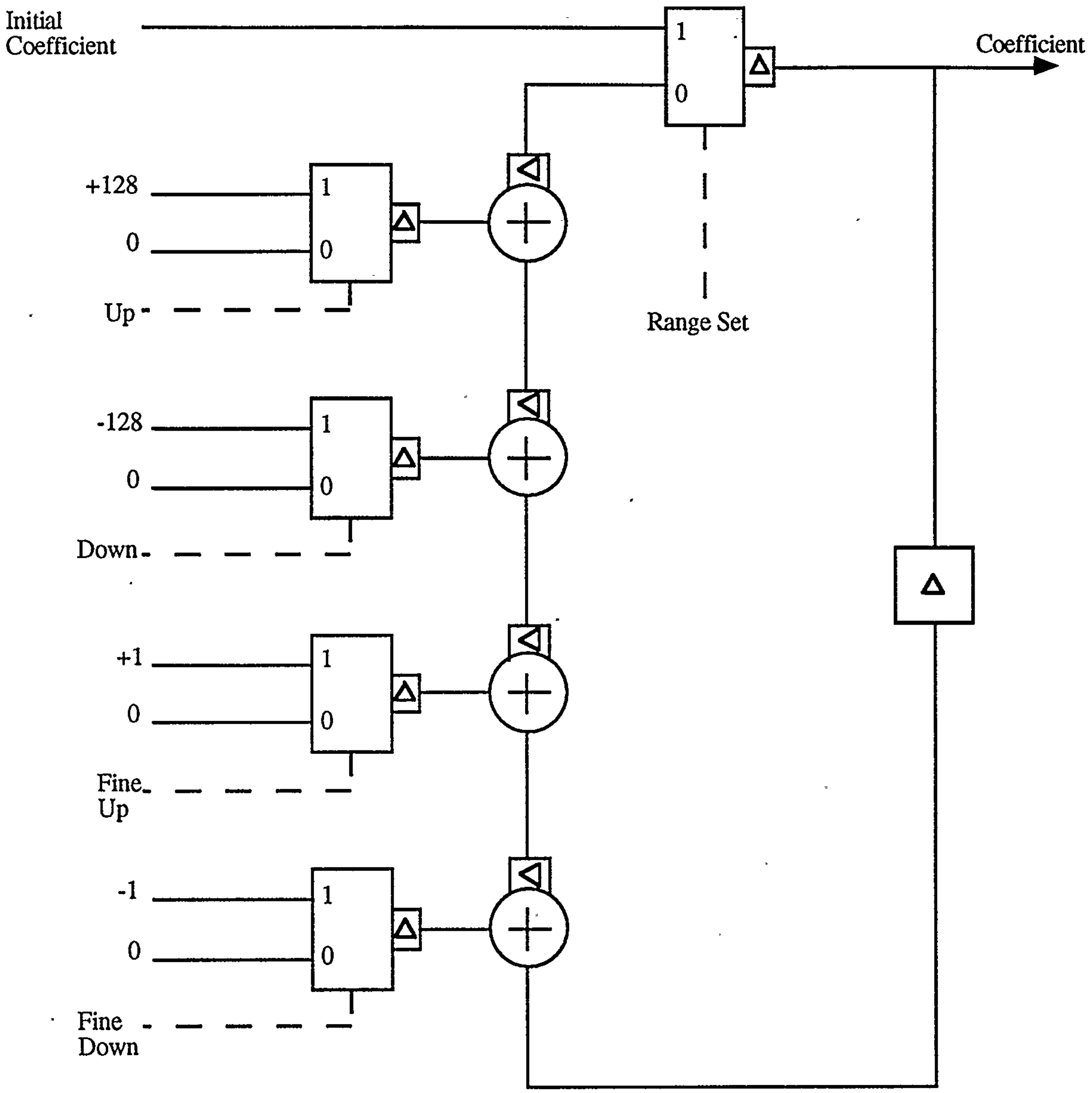

Figure 5.8 - Coefficient Storage Unit

5.4 Testing and Performance

The oscillator was tested using the Hewlett-Packard Structural Dynamics Analyzer (SDA) to measure the spectrum of the oscillator output. Because 12 bit D/A's are used, the spectral purity is expected to be no greater than 66 dB.

The output spectrum of the oscillator operating at 1.6 kHz with a bit clock rate of 20 MHz is shown in Figure 5.9. Figure 5.10 shows the output spectrum for the oscillator operating at 14.0 kHz. Other tests were conducted with the oscillator operating at frequencies of 3.1 kHz, 4.7 kHz, 6.2 kHz and 9.3 kHz. In all cases, the spectral purity of the output is 66 dB. This implies that the accuracy of the oscillator is limited by the D/A's.

A low frequency test, with the oscillator operating at 0.8 Hz was performed as well. The output spectrum for this test is given in Figure 5.11. It can be seen that the spectral purity is 66 dB at this frequency as well.

Because the SDA has a bandwidth of only 25 kHz, the full range of the oscillator could not be tested. In order to test the operation of the oscillator at high normalized frequencies, the 20 MHz bit clock was replaced with an 8 MHz bit clock. This change caused the maximum output frequency to be 20.8 kHz, which is within the SDA's bandwidth. Measurements of the output sinusoids generated with an 8 MHz bit clock also yielded 66 dB spectral purity.

The tests above indicate that the spectral purity of the oscillator is limited by the D/A's used to convert the digital signal to an analog signal. Unfortunately, more accurate D/A's were not available for more testing. It is reasonable to conclude, however, that given the 12 bit D/A's used, the accuracy of the output is limited by the D/A's and that the oscillator has at least 66 dB spectral purity over its entire operating range.

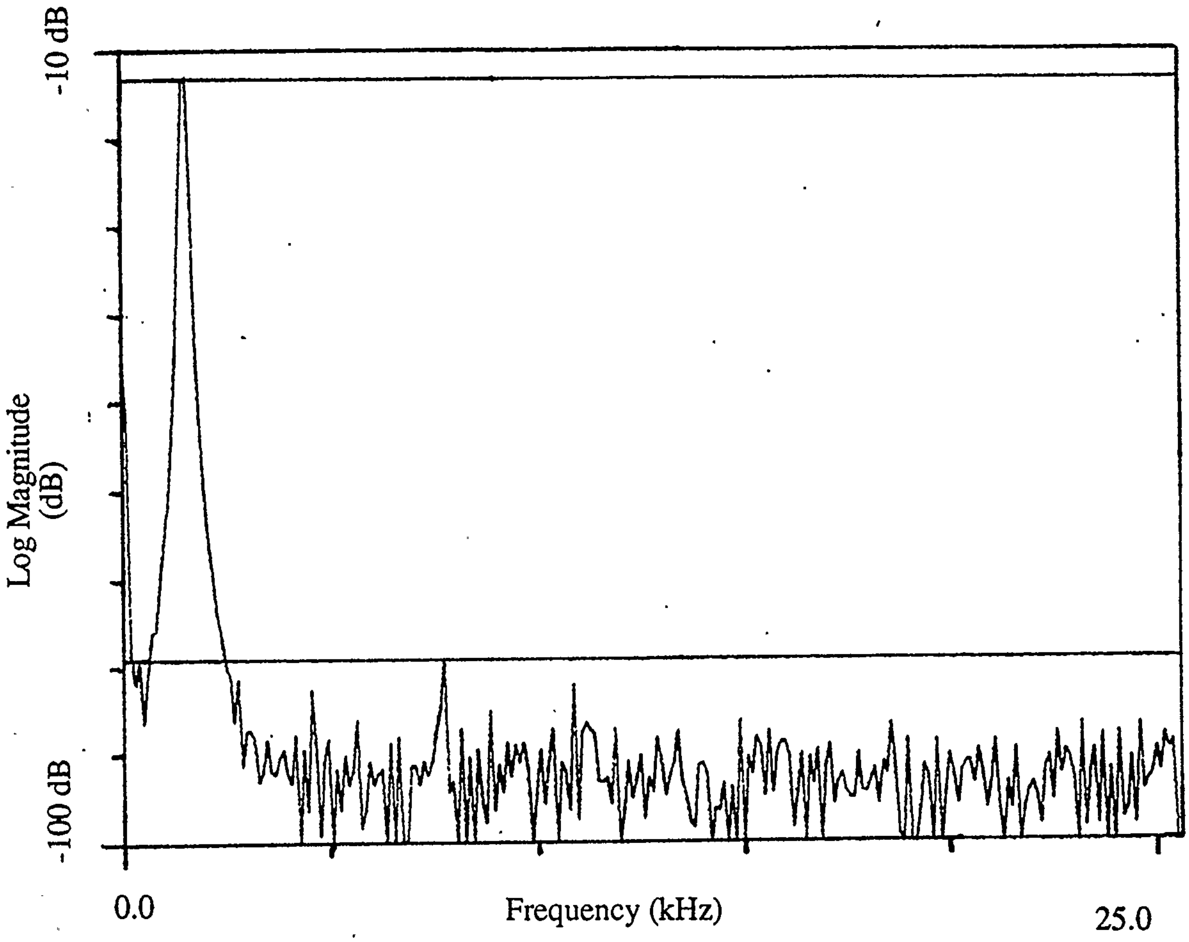

Figure 5.9 - Oscillator Spectrum (1.6 kHz)

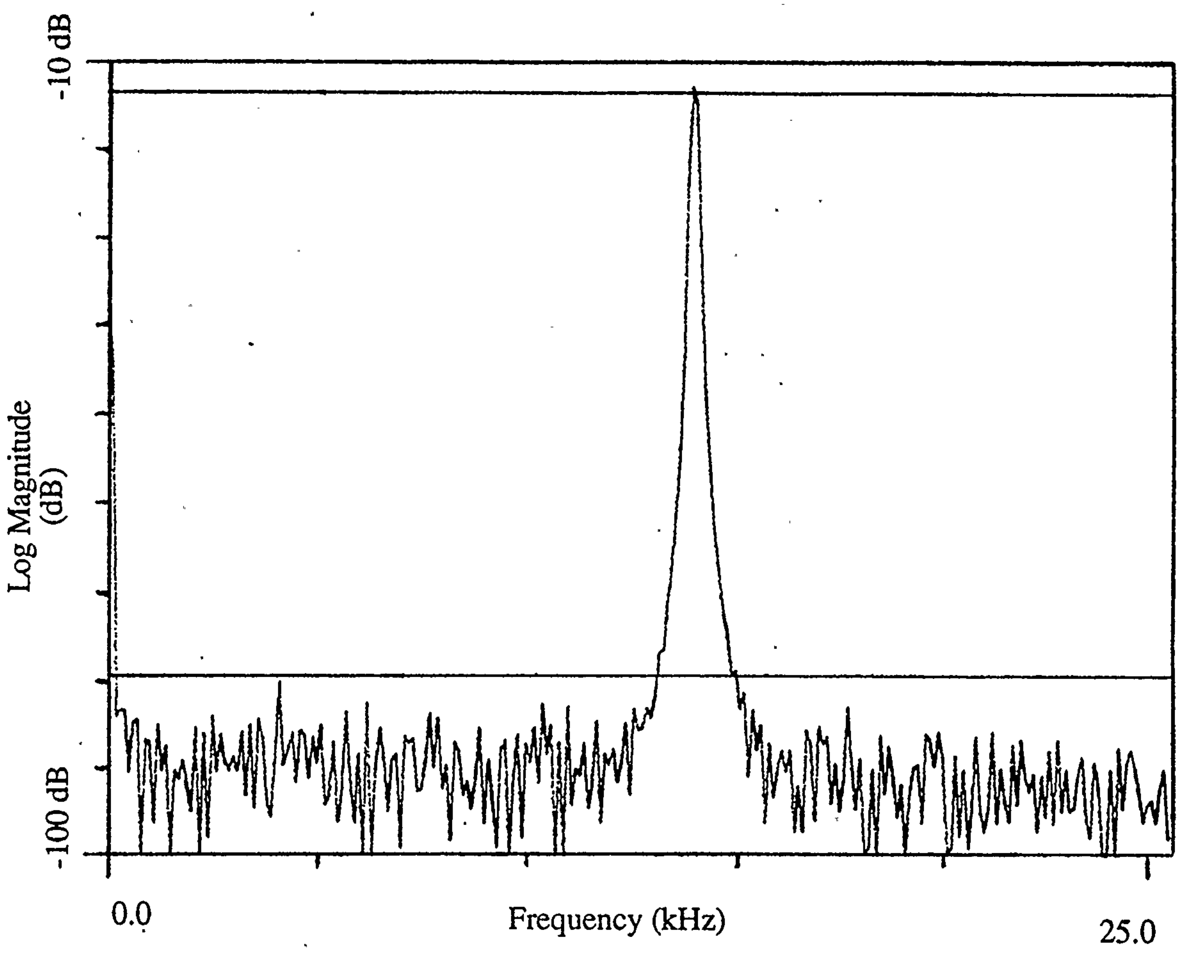

Figure 5.10 - Oscillator Spectrum (14.0 kHz)

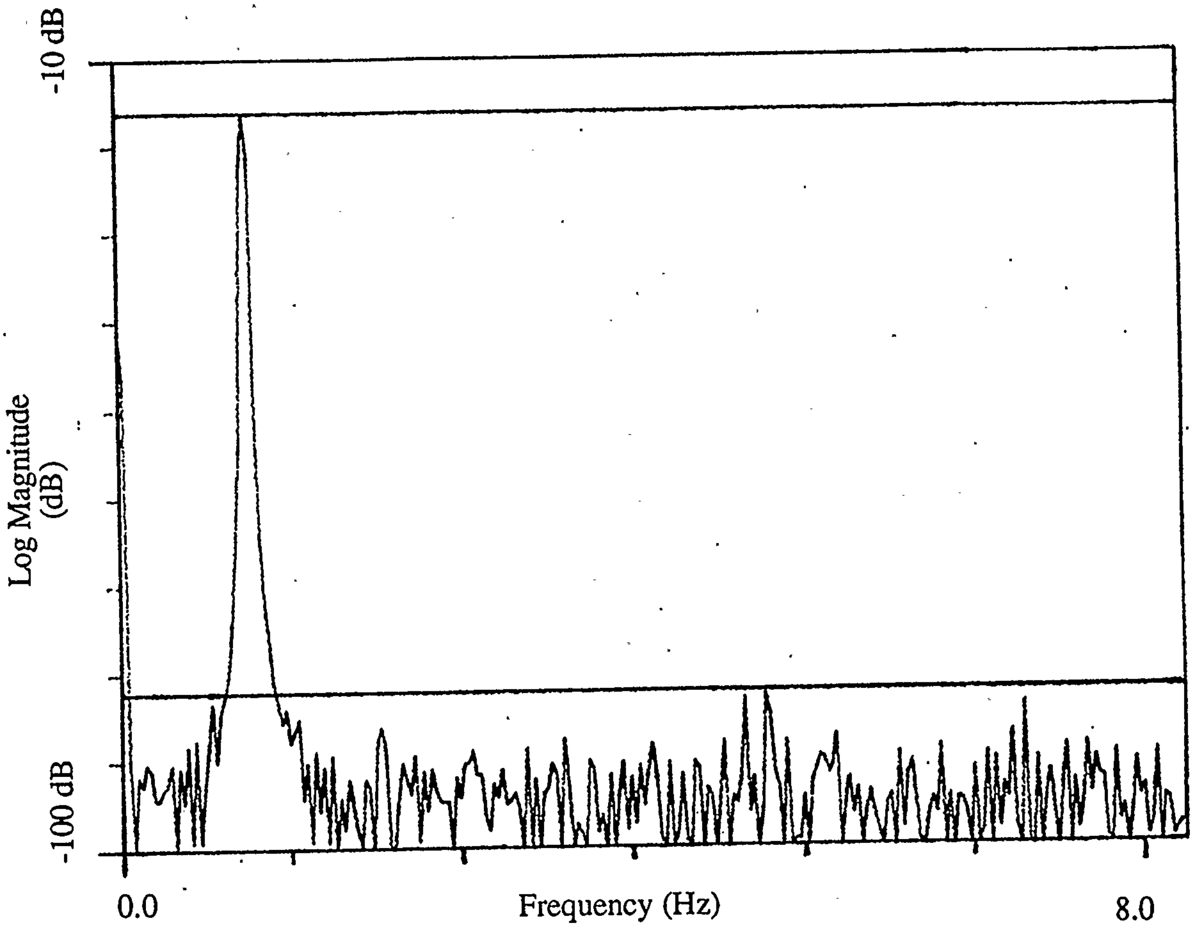

Figure 5.11 - Oscillator Spectrum (0.8 Hz)

Chapter 6

Conclusions

The goals of the research described in this thesis were to determine a general limit cycle bound, applicable to any recursive digital filter implementations, and to create a compact, highly accurate digital sinusoidal oscillator. The derivation of the General Bound to account for arbitrary filter structures meets the first goal. The second research goal was met by the implementation of a functional single chip digital sinusoidal oscillator.

6.1 Achievement of Research Goals

The first research goal considered was characterizing and eliminating limit cycles due to product quantization in digital filters. The existence of limit cycles was explained using the concept of Lyapunov functions. It was shown that although an ideal filter will always show a decrease in the value of its Lyapunov function from state to state, the signal quantization operations in a digital filter may cause the value of the Lyapunov function to increase from one state to the next. This effect can lead to limit cycles, as was demonstrated in Chapter 3. The General Bound derived in Chapter 3 provides a means of bounding the magnitude of any possible limit cycles (as well as any other effects due to quantization) for any filter structure, regardless of filter order, or quantization operation placement. It is also shown that the closed form General Bound is equivalent to that reported by Yakowitz and Parker [21] for filters with the quantization operations located at the states. As well, it was shown that the l_1 norm provides a bound that is equal to or tighter than both the iterative and closed form General Bound. Lastly, a means of determining the number of iterations required for the iterative General Bound and the l_1 norm was found.

It was also shown that the limit cycle bounds can be used to scale filter implementations so that limit cycles do not appear at the output of the filter. Because of the large signal wordlengths which are becoming common in both DSP chips and custom implementations, it is often an easy task to accomplish this scaling. Lastly, it was demonstrated that the structure of a filter can effect the non-ideal performance, even if the ideal performance is the same.

The second goal of the research was to study different methods of generating digital approximations to discrete time sinusoids. Three practical methods of generating sinusoids were examined. The first, consisting of an accumulator used as an index into a look up table is commonly used in industrial applications. While being simple and highly accurate, it tends to require large amounts of storage space for the look up table. The second method consists of a simple second order direct form filter with poles on the unit circle. Ideally, a second order system with poles on the unit circle will have an oscillatory response that does not decay over time. In practice, this oscillator has a high sensitivity to its coefficient at low frequencies, as well as known stability problems. The third method of generating sinusoids, known as the LDI oscillator, also involves a second order filter with poles on the unit circle. This oscillator is derived from an analog prototype and generates both sine and cosine terms and has low sensitivity to its coefficients at low frequencies. Several other second order filters with poles in the unit circle were considered as oscillators, but these were discarded because it was found that quantizing the coefficients would move the poles off of the unit circle. If the poles are not on the unit circle, then the magnitude of the oscillations will change over time, which is unacceptable.

In addition to studying methods of generating sinusoids, a further aspect of the research was to construct a single chip, high precision audio frequency oscillator. Because a field programmable gate array was made available for implementation, a requirement that the oscillator fit on a single device was included. The oscillator was required to be able to generate sinusoids up to 50 kHz, and the frequency must be selectable in at least 0.5 Hz increments with the outputs being at least 16 bits wide. An examination of the three structures showed that the LDI oscillator yielded the most compact implementation that met the specifications. The LDI oscillator was implemented using bit serial arithmetic and a prototype demonstration board was constructed. The prototype did not display any instability and was found to generate outputs with a measured spectral purity of 66 dB.

6.2 Further Research

For both the limit cycle research and the sinusoidal oscillator research, there are several areas in which further study could be made. In the case of the limit cycle research, the region of the state space in which limit cycles can occur could be searched to determine if limit cycle oscillations do in fact occur. The General Bound and the l_1 norm bound the magnitude of limit cycles, but these bounds are pessimistic estimates. The filter may not actually display limit cycles, or the limit cycles that do occur may not be as large as these bounds. Consider the case of the 5th order LDI all pole filter in Chapter 3 (Figure 3.7). If the quantization operations are located at the states, the limit cycle bounds for each of the states are 10, 16, 15, 16 and 14. The total number of possible states in the region where limit cycles can occur is 11,793,600. The number of states is low enough that it may be possible to exhaustively search for any limit cycle trajectories. Any search would be specific to a particular structure and a given set of coefficients.

Significant further research can be performed in the area of the LDI oscillator. First, a rigorous bound on the frequency stability and the spectral purity of the oscillator is highly desirable. One way of guaranteeing the absolute periodicity of the LDI oscillator is to perform an exhaustive search of its state space. For the implementation described in Chapter 5, which has 32 bit state variables, an exhaustive search would require examining $2^{32} \times 2^{32}$ or about 18.5 billion billion points. If each point were examined only once, and this operation took a single nanosecond, the entire search would require 18.5 billion seconds or about 590 years. Clearly, this is not a feasible approach.

There are other areas which could be examined, however. In the testing section, it was shown that the spectral purity of the output was limited by the 12 bit D/A's used. Further spectral testing could be performed using higher accuracy D/A's, up to the 18 bit limit imposed by the oscillators outputs. Also of interest is the possibility of using the oscillator as a modulator by varying the coefficients with time. If the coefficients are varied in time, the frequency of oscillation will also vary. The stability of the oscillator must be carefully considered however, as changing the coefficients could cause the signals to overflow. In the course of testing the oscillator, it was found that if the coefficients were varied slowly, the oscillator would continue to oscillate and that overflows would not occur. On the other hand, if the coefficients were changed rapidly, or were changed in large increments, the oscillator would quickly become unstable. Determining the magnitude and rate of change in the coefficients that can be tolerated before the oscillator becomes unstable would allow the oscillator to be used in modulator applications. Lastly, the oscillator has potential for use in analog test applications. The

LDI oscillator is a compact, controllable sinusoid generator, and with the addition of a compact D/A, could be used in on-chip test applications in analog designs.

References

[1] E.A. Robinson, "Spectral Estimation - A Historical Survey," *Proceedings of the IEEE*, vol. 70, no. 9, 885-906, September 1982.

[2] L.B. Jackson, *Digital Filters and Signal Processing.* Kluwer Academic Publishers, 1986.

[3] K. Ogata, *Modern Control Engineering.* Prentice-HAll, 1970.

[4] R.H. Middleton, G.C. Goodwin, *Digital Control and Estimation.* Prentice-Hall Inc., 1990.

[5] B.C. Kuo, *Automatic Control Systems.* Prentice-Hall Inc., 1987.

[6] J. Bellamy, *Digital Telephony.* John Wiley and Sons, 1982.

[7] L.R. Rabiner, "Terminology in Digital Signal Processing," *IEEE Transactions on Audio and Electroacoustics*, vol. AU-20, 322-337, December 1972.

[8] R.A. Gabel, R.A. Roberts, *Signals and Linear Systems.* John Wiley and Sons, Inc., 1987.

[9] B.D. Green, L.E. Turner, "New Limit Cycle Bounds for Digital Filters," *IEEE Transactions on Circuits and Systems*, vol. CAS-35, no. 4, 365-374, April 1988.

[10] A. Antoniou, *Digital Filters: Analysis and Design.* McGraw-Hill Book Company, 1979.

[11] A.B. Carlson, *Communication Systems*, 3rd. McGraw-Hill Book Co., 1986.

[12] R.A. Roberts, C.T. Mullis, *Digital Signal Processing.* Addison-Wesley Publishing Company, 1987.

[13] L.E. Turner, "Elimination of Constant Input Limit Cycles in Recursive Digital Filters Using a Generalized Minimum Norm," *IEE Proceedings*, vol. Pt. G, no. 3, 69-77, June 1983.

[14] I.W. Sandberg, J.F. Kaiser, "A Bound on Limit Cycles in Fixed Point Implementations of Digital Filters," *IEEE Trans. Audio Electro.*, vol. AU-20, no. 2, 110-112, June 1972.

[15] A. Fettweis, K. Meerkotter, "Suppression of Parasitic Oscillations in Wave Digital Filters," *IEEE Transactions on Circuits and Systems*, vol. CAS-22, no. 3, 239-246, March 1975.

[16] H.J. Butterweck, J.H.F. Ritzerfeld, M.J. Werter, "Finite Wordlength Effects in Digital Filters: A Review," *Eindhoven University Technical Report*, vol. E-205, 94, October 1988.

[17] S.R. Parker, S.F. Hess, "Limit Cycle Oscillations in Digital Filters," *IEEE Transactions on Circuit Theory*, vol. CT-18, no. 6, 687-697, Nov. 1971.

[18] C.W. Barnes, A.T. Fam, "Minimum Norm Digital Filters Which are Free From Limit Cycles," *IEEE Transactions on Circuits and Systems*, vol. CAS-24, no. 10, 569-574, October 1977.

[19] Y.Z. Zhang, T.W. Cole, Y.F. Yao, "Optimal Removal of LCO's in Digital Filters," *Proceedings 1990 International Conference on Acoustics, Speech, and Signal Processing*, vol. 3, 1297-1300, March 1990.

[20] M. Sarcinelli, P. Diniz, "Synthesis of Passive Lattice Digital Filters Which are Free of Constant Input Limit Cycles," *IEEE International Symposium on Circuits and Systems Proceedings*, vol. 3, 1664-1667, 1989.

[21] S. Yakowitz, S.R. Parker, "Computation of Bounds for Digital Filter Quantization Errors," *IEEE Transactions on Circuit Theory*, vol. CT-20, no. 4, 391-396, July 1973.

[22] E.I. Jury, *Theory and Application of the Z-Transform Method.* John Wiley and Sons, 1964.

[23] H. Anton, *Calculus with Analytic Geometry.* John Wiley and Sons, 1984.

[24] W.K. Nicholson, *Elementary Linear Algebra with Applications.* Prindle, Weber and Schmidt, 1986.

[25] B.D. Green, "A Single Chip Programmable Digital Filter," Master's thesis, University of Calgary, Calgary, Alberta, October 1987.

[26] P. Denyer, D. Renshaw, *VLSI Signal Processing: A Bit Serial Approach.* Addison-Wesley, 1985.

[27] *Digital Signal Processing*, Harris Corp., 1991.

[28] *HP 8904A Multifunction Synthesizer*, Hewlett-Packard, 1987, HP Technical Literature.

[29] J. Studders, "Phase Accumulator for Direct Digital Synthesis Resolves to Within 0.7 Hz," *Analog Dialogue*, vol. 25-1, 10-11, 1991.

[30] K. Furuno, S.K. Mitra, K. Hirano, Y. Ito, "Design of Digital Sinusoidal Oscillators with Absolute Periodicity," *IEEE Transactions on Aerospace and Electronic Systems*, vol. AES-11, no. 6, 1286-1296, November 1975.

[31] , *Standard Mathematical Tables*, 28th. CRC Press, 1981.

[32] A.I. Abu-el-Haija, "Digital Oscillator Having Low Sensitivty and Roundoff Errors," *IEEE Transactions on Aerospace and Electronic Systems*, vol. AES-22, no. 1, 23-33, January 1986.

[33] A.I. Abu-el-Haija, "A Digital Incremental Oscillator," *IEEE Transactions on Aerospace and Electronic Systems*, vol. AES-22, no. 5, 545-552, September 1986.

[34] A. Fettweiss, “Some Principles of Designing Digital Filters Imitating Classical Filter Structures,” *IEEE Transactions on Circuit Theory*, vol. CT-18, 314-316, March 1971.

[35] A. Sedlmeyer, A. Fettweiss, “Digital Filters with True Ladder Configuration,” *International Journal of Circuit Theory and Applications*, vol. 1, 5-10, March 1973.

[36] L.T. Bruton, “Low Sensitivity Digital Ladder Filters,” *IEEE Transactions on Circuits and Systems*, vol. CAS-22, no. 3, 168-176, March 1975.

[37] L. Strauss, *Wave Generation and Shaping*. McGraw-Hill Book Company, 1960.

[38] *ACT 2 Field Programmable Gate Arrays*, Actel Corporation, 955 E. Arques Ave. Sunnyvale, CA, 1991.

[39] R.K. Nagalla, “Synthesis of Bit-Serial DSP Systems,” Master’s thesis, University of Calgary, Calgary, Alberta, 1992.

[40] P.J. Graumann, *TRANS User's Manual*, University of Calgary, 1992.

[41] A.V. Oppenheim, R.W. Schafer, *Discrete Time Signal Processing*. Prentice Hall, 1989.

[42] M. Willrodt, “Quartz Crystal Oscillators,” *Bench Briefs*, vol. 13, no. 3, .

[43] *CTS Knights, Inc. Guide to Quartz Frequency Management Devices*, CTS Knights, Inc., 400E Reimann Avenue, Sandwich, Illinois.