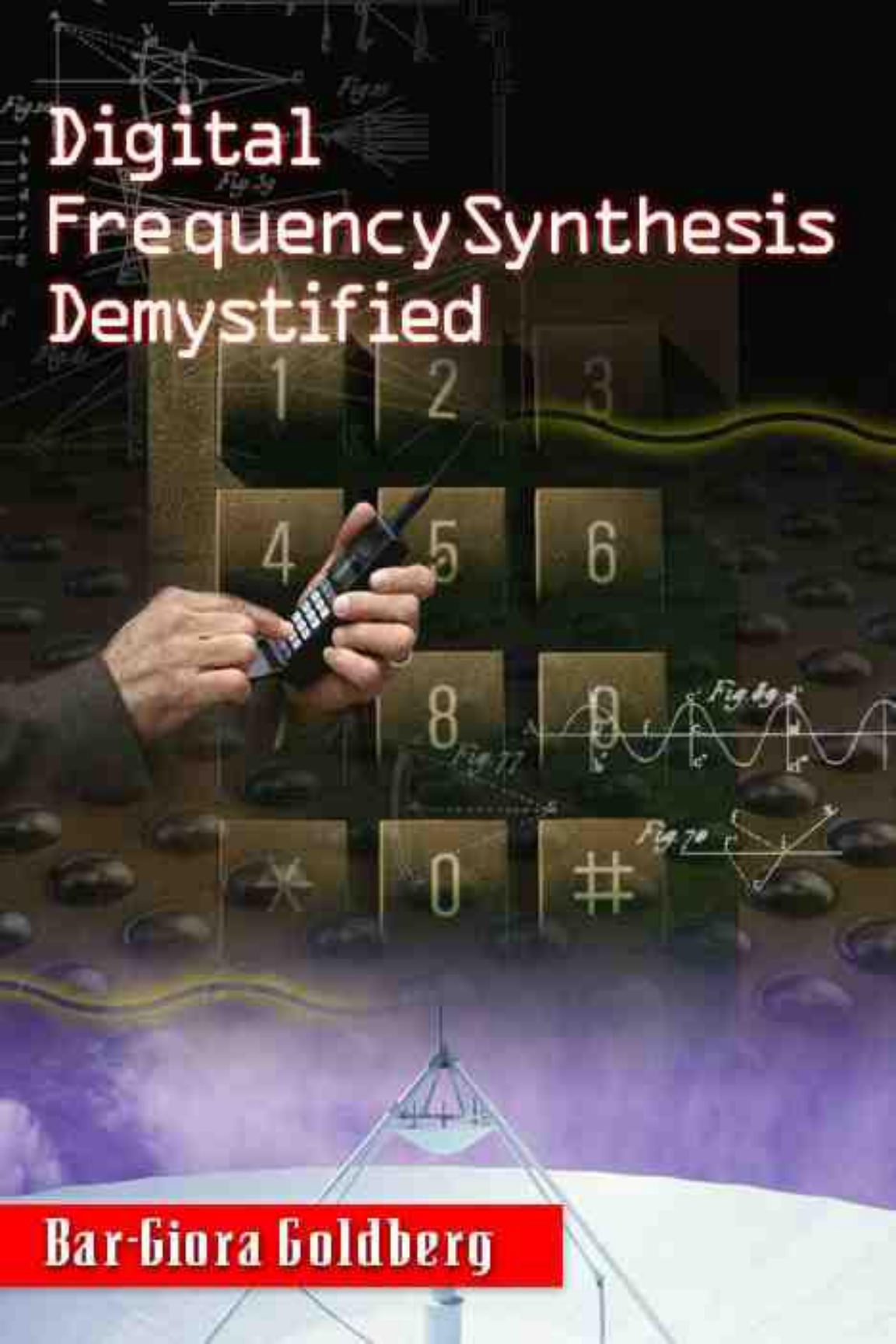

Digital
Frequency Synthesis
Demystified
Bar-Giora Goldberg

Digital Frequency Synthesis Demystified

DDS and Fractional-N PLLs

Bar-Giora Goldberg

a volume in the Demystified series

LLH Technology Publishing

Eagle Rock, VA
www.LLH-Publishing.com

Library of Congress Cataloging-in-Publication Data

Goldberg, Bar-Giora.
Digital frequency synthesis demystified / Bar-Giora Goldberg.
p. cm.
"A volume in the Demystified series."
Includes bibliographical reference and index.
ISBN 1-878707-47-7 (pbk. : alk. paper)
1. Frequency synthesizers--Design and construction. 2. Phase -locked loops. 3. Digital electronics. I. Title.
TK7872.F73G55 1999
621.3815 486--dc21 99-34856
CIP

Printed in the United States of America
10 9 8 7 6 5 4 3 2 1

Cover design: Sergio Villareal
Developmental Editing: Carol Lewis
Production: Kelly Johnson

LLH
Technology Publishing

Eagle Rock, VA
www.LLH-Publishing.com

To Moshę, Shoshana, Pnina, Amit, and Dror

This is a blank page.

Contents

ABOUT THIS BOOK AND CDROM

This book—the latest volume in our popular *Demystified* series of technical references—is an updated version of an earlier text, *Digital Techniques in Frequency Synthesis*, published by McGraw-Hill. In addition to the updated text, a CDROM has been added that contains an assortment of design tools, including an informative new reference in pdf format from Analog Devices called *A Technical Tutorial on Digital Signal Synthesis*. The CDROM also includes a fully searchable pdf file of the entire book contents. For a full list of the CDROM contents, see Chapter 10.

ABOUT THE AUTHOR

Bar-Giora Goldberg received his education at the Technion-Israel Institute of Technology in Haifa, and he worked there for 10 years in communication and spread-spectrum systems. In 1984 he cofounded Sciteq Electronics, a world leader in the field of digital frequency synthesis, and he now heads Vitacomm, a company dedicated to frequency and time engineering products for wireless and high-speed telecommunications. Giora has written extensively, has been awarded several patents, and has been a major contributor to the introduction and development of DDS and fractional-N technologies. He is also associated with Besser Associates in the US and Continuous Education International (CEI) in Europe, companies dedicated to continuous education via seminars and on-line courses.

This is a blank page.

Preface

As in the first edition of this book, my purpose is to present to the designer a comprehensive review of digital techniques in modern frequency synthesis design. The text specifically addresses practical designers, and an attempt has been made to approach the subject heuristically, by using intuitive explanations and including many design examples.

Not long ago, frequency synthesis was considered a novelty. It was used in the more complex and demanding applications. Today, frequency synthesis is so ubiquitous that it is not even possible to say that its use is growing. Frequency synthesis is now so natural that every radio design uses only synthesized signals for generation and control. This is partly because the spectrum is so precious a commodity, and its use is controlled tightly by government and industry. Other reasons include the increase in complexity of modulation, the phenomenal increase in use, and the increase in convenience. No more dialing and fine-tuning; just push the button, and the channel is locked in with an accuracy that does not require correction.

Frequency synthesis therefore has entered the age where it is used in applications such as military radios, satellite communications terminals, and radars as well as in CB radios and hi-fi consumer electronics. It is not possible to consider the huge acceleration of cellular telephony and the still emerging markets of wireless and personal communications services (PCS) without the use of frequency synthesis.

Frequency synthesis is a fascinating technological discipline, as it includes both analog and digital technologies. To design a synthesizer one has to apply a great variety of disciplines such as oscillators, voltage-controlled oscillators, amplifiers, filters, phase detectors, logic, and low-noise dc amplification and filtering. This book, however, is not intended to be a general introduction to frequency synthesis; rather, it tries to focus on a segment of the technology.

There has obviously been a trend to "go digital" in the last 15 to 20 years. Although they are usually more complicated than analog, digital technologies offer excellent repeatability, much better accuracy, improved performance, and repeatability in production. This trend has not been ignored by frequency generation technologies. There have been major advances in two key technologies. The first is known as direct digital synthesis (DDS), a technology that generates the signal digitally and converts it via a digital-to-analog converter (DAC) to a sine wave. This technology is almost purely within what is known today as digital signal processing (DSP), but it has been in development for over 15 years within the domain of radio-frequency (RF) electronics engineering. The second is the digitalization of phase-locked loop (PLL) technology, the one that is the most popular and probably covers 98 percent of frequency synthesis and its evolution to what is referred to today as *fractional-N* synthesis.

Even very recent texts on PLL frequency synthesis have defined the technique as "generation of frequencies which are exact multiples of a reference." This definition is not accurate anymore. The more advanced synthesizers generate frequencies that are related to the reference but are not always exact multiples. In fact, the very principle of fractional-*N* PLL synthesis requires that the ratio of output frequency to the reference be a rational *fraction*. It so happens that these PLL technologies are also closely related to DDS and as such include DSP, a discipline that will find increased applicability in signal generation in the years to come.

This text was written mainly as a consequence of the rapid developments of these technologies and the lack of literature, especially the two subjects mentioned above. To the best of my knowledge, no other text exists that attempts to focus on the modernization and digitalization of signal generation.

Thus, the purpose of this book is to provide an introduction, training, and bibliographical material for designers. The text has been written specifically for designers, and many design examples have been included. Although the basics of each topic are covered, it is assumed that the reader has some understanding of frequency synthesis. There are many excellent general PLL books, and I have decided not to include too much of what has been already covered extensively before.

Frequency synthesis, and especially the digital part of it, is now going through a major evolutionary period. Modern systems require high levels of integration, low power dissipation, and low cost. Digital technologies fit this bill precisely and allow the push in the technology. Frequency synthesis has received much attention from chip manufacturers, and a great variety of PLL and direct digital synthesizer chips have been available for several years. We are now seeing a major shift from the standard PLL to fractional-*N* PLL and a major increase in the use of DDS. These technologies are used in cellular and PCS applications as well as disk drives and satellite communications terminals. What can be more exciting and faster-moving than these markets today?

The collection of files on the accompanying CDROM has been provided to help the reader analyze and manipulate methods described in the text. The software has been devised for IBM PC compatibles, but it will also be applicable for Macs.

I would like to take this opportunity to thank former and current colleagues who contributed to this book while working with me or discussing various aspects of the technical details. No one can do it alone. A text like this presents a set of ideas and techniques that evolved through the ages, and personally through many years of practice. It is therefore not possible to mention all the people who made a contribution to the maturization of the technology of accurate timekeeping, but I am indebted to them.

I would like to extend my thanks to my friends and family, especially Pnina, who encouraged and supported this very long, hard effort. I would like to specifically thank my Technion mentors, Dr. Jacob Ziv and Israel Bar-David; and Henry Eisenson, a great friend, a brother, and a partner who always helps by giving support and encouragement.

Bar-Giora Goldberg

This is a blank page.

Symbols

a	angle
AM	amplitude modulation
b	angle
c	angle
CORDIC	coordinate transformation
DAC	digital-to-analog converter
dB	decibel
dBC	dB referred to carrier
dBm	dB over 1 mW
DDFS	direct digital frequency synthesizer
DDS	direct digital synthesis
Er, er	error
f_i	input frequency
f_m	modulating frequency
F_o	output frequency
F_r, F_{ref}	reference frequency
FM	frequency modulation
F_s	sampling frequency
gcd(a, b)	greatest common divisor
$H(x)$	transfer function
int(x)	integer of x
IF	intermediate frequency
K_{d}	phase detector gain constant
K_0	VCO gain constant
L_{m} (f_m)	SSB noise density at f_m from the carrier, in dBC/Hz
LFM	linear FM
LSB	least significant bit
m	index of modulation

MSB	most significant bit
NCO	numerical control oscillator
PM, ϕM	phase modulation
RF	radio frequency
S	sum (usually at accumulator output)
SSB	single sideband
T	sample time
VCO	voltage-controlled oscillator
α	angle
β	angle
ξ	damping factor
ϕ	phase
ω	radial frequency
ω_n	natural frequency
ω_0	center frequency

Chapter

1

Introduction to Frequency Synthesis

1-1 Introduction and Definitions

This text deals with emerging modern digital techniques used to generate and modulate sine waves. These waveforms are used in almost all radio applications, communications, radar, digital communications, electronic imaging, and more. Such techniques either build the waveform from the "ground up" digitally (i.e., generate all the signal parameters such as phase, frequency, and amplitude digitally) and deal with the very fundamental nature of the waveform and its features (direct digital synthesis) or are part of the digital heart of modern *phase-locked loop* (*PLL*) synthesizers. This might seem, and is indeed, a common and known subject. Sine waves are truly natural waveforms and trigonometric functions that are well known and have been researched for a long time. Furthermore, frequency synthesis is quite a mature technology with extensive literature and comprehensive coverage in the professional meetings. Why another text on the subject? What is new besides *application-specific integrated circuit* (*ASIC*) technologies and silicon densities, geometry, and integration?

While the above statements are true, there is a continuous evolution in the technology. The generation of accurate waveforms plays a crucial role in almost all electronic equipment, from radar to home entertainment equipment, so the importance of the subject is clear. Clearly, the most important reason for the utilization of the now extremely popular PLL synthesizers in consumer elec-

tronics and other very popular applications at extremely low cost (and the popularization of frequency synthesis from consumer products all the way to complex requirements) is the advance of digital technology; integrated, high densities; and low-cost silicon single-PLL chips and ASICs. However, parallel to the advance of traditional PLL synthesis, there emerged other synthesis techniques, mainly digital in nature, *direct digital synthesis* (*DDS*) and fractional-*N* PLL synthesis. Thus, the classical PLL synthesizer is now being supplemented with a sizable element of digital technology and *digital signal processing* (*DSP*). Indeed the application of DSP techniques to frequency synthesis is still at an early stage.

The generation of sine waves by using digital methodologies requires generating the waveform from the ground up. This is fundamentally different from the PLL synthesizer, where the signal is available from an oscillator. It goes back to the very basic structure of the waveform itself and deals with its very basic characteristics rather than manipulates signals that have already been generated by an oscillator. Surprisingly, some of these very basic mathematical issues are being resolved only lately.

Unfortunately, in these specific fields, there is a lack of complete understanding of the mathematics as well as the standard implementation of working hardware. The operation of a direct digital synthesizer is far from intuitive, and its artifacts are sometimes alien to our (conservative or standard) thinking. Indeed, in this ongoing research effort, we have tried to recruit some very skilled professional mathematicians in the search for (1) effective sine *read-only memory* (*ROM*) (the transformation of φ to $\sin \varphi$) compression algorithms (indeed, the same old trigonometric functions; see Chap. 7), (2) a "minimal" amount of data necessary to represent the waveform, such as to meet a specific level of accuracy, and (3) a general formulation for the performance of DDS. We have not had much luck or enthusiasm.

We understand that this might not be the most exciting topic for mathematicians, but it has tremendous importance for electronics, radio, and radar designers. The challenge has to be met within the electronics community, and we have attempted to present a comprehensive introduction.

Although there are many excellent books on PLL synthesis (see References), mostly published in the 1980s, note that this is the

first attempt to write a comprehensive text on the subject of digital frequency synthesis, direct digital synthesis, and digital and fractional-*N* synthesis; and the number of sources is not overwhelming in this newly emerging technological discipline. We have found a paucity of literature in the field; and even though many articles have begun to appear in the last few years and the technology attracts much attention in professional meetings, comprehensive texts and bibliographies are needed. This is what this text attempts to supply. Every attempt is made to present a very comprehensive, updated bibliography.

Because of the paucity of literature, in this text we attempt to present an intuitive approach supplemented by many examples, in the hope that this book fills a current need as expressed to us by many young and beginning designers as well as others who are not familiar with the details and lack an intuitive understanding.

In this text, a *frequency synthesizer* is defined as a system that generates one or many frequencies derived from a single time base (frequency reference), in such a way that the ratio of the output to the reference frequency is a rational fraction. The frequency synthesizer output frequency preserves the long-term frequency stability (the accuracy) of the reference and operates as a device whose function is to generate frequencies that are multiples of the reference frequency (multiples by a single or many numbers). These multiples may be whole or fractions; but since only linear operations are used (in the frequency domain), these numbers can only be rational. A frequency synthesizer, as defined here, can thus generate an output frequency of, say, *X*/*Y* (where *X* and *Y* are whole numbers) times the reference frequency, but not, for example, π times the reference frequency (π is not a rational number).

Three main, conventional techniques are being used currently for sine-wave synthesizers and are common throughout the industry. The most common and most popular technique uses the phase-locked loop synthesis. PLL synthesizers can be found in the most sophisticated radar systems or the most demanding satellite communications terminals as well as in car radios and stereo systems for home entertainment. The PLL is a feedback mechanism locking its output frequency to a reference. PLL synthesizers gained popularity for their simplicity and economics.

Another synthesizer technique is known as *direct analog* (*DA*) frequency synthesis. In this technique, a group of reference frequencies is derived from the main reference; and these frequencies are mixed and filtered, added, subtracted, or divided according to the required output. However, there are no feedback mechanisms in the basic technique.

The DA frequency synthesis technique offers excellent spectral purity, especially close to the carrier, and excellent switching speed, which is a critical parameter in many designs and determines how fast the synthesizer can hop from one frequency to another.

The DA technique is usually much more complicated than PLL to execute and is therefore more expensive. DA synthesizers found applications in medical imaging and spectrometers, fast-switching antijam communications and radar, *electronic warfare* (*EW*) simulation, *automatic test equipment (ATE), radar cross-section (RCS)* measurement, and such uses where the advantages of the DA technique are a must at a premium cost.

The third technique, which is the focus of this book, is direct digital synthesis (DDS), which is a digital signal processing (DSP) discipline and uses digital circuitry and techniques to create, manipulate, and modulate a signal, digitally, and eventually convert the digital signal to its analog form by using a *digital-to-analog converter* (*DAC*).

Although the direct digital synthesizer [sometimes referred to as *numerically controlled oscillator* (*NCO*)] was invented almost 30 years ago (see Ref. 9 and Chap. 10), it started to attract attention only in the last 10 to 12 years. Due to the enormous evolution of digital technology and its tools, the technique evolved remarkably into an economical, high-performance tool and is now a major frequency synthesis method used by almost all synthesizer designers from instrument makers to applications like satellite communications, radar, medical imaging, and cellular telephony and amateur radios (most of which are anything but amateur).

Direct digital synthesizers offer fast switching speed, high resolution (the step size of the synthesizer), small size and low power, good economics, and the reliability and producibility of digital designs. In addition, since the signal is manipulated digitally, it is easy to modulate and achieve accuracies not attained by analog techniques and to conveniently interface with the computing machines that usually control the synthesizer.

Another focal point of this text is the description of fractional-*N* PLL synthesis. This technique resembles DDS in almost all aspects and operates as a DDS "inside" the PLL architecture. Please note that in many designs, more than one synthesis technique is being utilized, and the designer "hybridizes" the design so that the advantage is taken of each technique being used and its weaknesses are suppressed. So it is quite common (and applications can be expected to grow) to see combinations of PLL and DDS or DA and DDS, and from time to time all three techniques are used in one design. Thus the basic three techniques indeed complement one another and enable the up-to-date competent designer to use all as needed to optimize the design as the applications and demands increase with the system complexity.

This text has 10 chapters. Chapter 1 is a general introduction and short description of frequency synthesis techniques, Chap. 2 deals with synthesizer system analysis, and Chap. 3 addresses measurement techniques pertinent to frequency synthesis. Chapter 4 details a variety of DDS technologies and deals with the quantization effects, their artifacts, and representations in DDS. Chapter 5 discusses PLL principles and the details of fractional-*N* PLL synthesis of various complexities. Chapters 6, 7, and 8 deal in detail with the cardinal components of DDS, namely, accumulators of binary and *binary-coded decimal* (*BCD*) structure, ROM lookup tables and ROM compression algorithms, and digital-to-analog converters. Chapter 9 gives a short review of state-of-the-art reference oscillators and what we consider some remarkable instruments or products on the market that are directly related to digital frequency generation.

Chapter 10 is special, as it refers to a reprint of the original 1971 article by Tierney, Rader, and Gold that kicked off the DDS industry (Ref. 8), including some footnotes. This article is special not only because of its pioneering nature but also for the fact that it deals with all the cardinal issues of the subject. The chapter also includes a description of the programs contained on the accompanying CDROM.

1-2 Synthesizer Parameters

Like any other engineering product, a *frequency synthesizer* (*FS*) needs to meet a set of specifications. In the following, a list of the

most common specifications is provided followed by a definition and industry standard conventions. Obviously, for different applications, different specifications are more important than others, and it is up to the designer to design for efficiency and economy. While a FS for a car radio needs to be moderately accurate, extremely reliable, very small and simple, and very inexpensive, a FS used in *magnetic resonance imaging* (*MRI*) must be very accurate, must have very high spectral purity, must be able to hop from frequency to frequency very quickly, and needs different modulation capabilities. While consumer electronic products need to operate in extreme environmental conditions (one expects the car radio to operate when powered up in extreme heat or cold conditions, and the vibrations of the cars are severe), the MRI spectrometer operates in a laboratory-controlled environment with little temperature variation and almost no shock or mechanical vibrations.

Designers are therefore required to compare their specifications to the best economical and practical solution. The specifications are divided into two groups: those that are related to the topics of this book and others that are more general and are beyond our scope. All the following sections pertain to generic specifications.

1-2-1 Frequency range

This specifies the output frequency range, including the lower and higher frequencies that can be obtained from the FS. The units of frequency are hertz (Hz), or cycles per second.

1-2-2 Frequency resolution

This parameter is also referred to as the *step size,* and it specifies the minimum step size of the frequency increments. So if a FS covers 10 to 100 MHz and has a step size of 10 Hz, it is capable of generating any frequency between 10 and 100 MHz in 10-Hz steps. Some manufacturers do not generate the last frequency, and so the same specification mentioned above will generate 10.0000 to 99.99999 MHz but not 100 MHz. In many applications, the step size is not fixed. This happens when a part of the synthesizer is generated by dividing a fixed frequency by a range of numbers.

1-2-3 Output level

The output power level is usually expressed in decibels (0 dBm is 1 mW). The output power can either be fixed, say, +10 dBm, or can cover a range, say, −120 to +15 dBm. This specification will also include the output power resolution, for example, 1 dB or 0.1 dB.

1-2-4 Control and interface

This parameter specifies the control methodology and the interface to the FS. The control can be binary-coded decimal (BCD) or binary; it can be parallel or via a bus (usually an 8-bit bus) or serial; it can be transparent or latched. When the control is latched, there is a register that receives the control word and upon activation (by a latch command) loads the control word into the FS (also referred to as *double buffering*). Some FSs use positive logic and others use negative; and in many general-purpose instruments, GPIB or IEEE-488 is currently the standard interface. VXI is an emerging new interface standard for instrumentation.

Most single-chip synthesizers, especially PLL, make extensive use of the serial interface to allow small packages and highly integrated functionality.

1-2-5 Output flatness

This parameter specifies the flatness of the output power and is measured in decibels (dB). For example, the output power is specified as 10 dBm ±1 dB, where dBm means decibels over 1 milliwatt (mW).

1-2-6 Output impedance

This parameter specifies the nominal output impedance of the FS and usually is also the recommended load impedance. In most radio-frequency and microwave equipment, this is 50 ohms (Ω). In video it is usually 75 Ω and in audio equipment 600 Ω.

1-2-7 Switching speed

This parameter specifies the speed at which the FS can hop from frequency to frequency. There are many definitions for this para-

meter. In some applications the requirement is to settle to within a specific frequency ($\pm x$ Hz) from the desired new frequency (say, 50 Hz or 5 kHz from the desired frequency). Suppose that the specification is to cover 10 to 100 MHz and switch to within 1 kHz in less than 100 microseconds (μs). To measure this parameter, a counter is set to measure the new frequency (say the FS is commanded to hop between 10 and 100 MHz periodically) and is timed to start measuring only 100 μs after the command bit is activated (say, for 5 μs, because the time allowed must be short relative to the specification time). If the measurement is either 10 or 100 MHz (depending on where we command the counter to measure) within ± 1 kHz, then the specification has been met. Obviously in such a measurement a pulsed counter must be used, and its gating time must be specified, too.

A more common and more demanding specification defines the switching speed by the time it takes the output phase to settle to 0.1 rad of the final phase.

If the FS generates $A \cos(\omega_1 t + \varphi_1)$ and is controlled to a new frequency, say, $A \cos(\omega_2 t + \varphi_2)$, the signal phase will go through a transient from $\omega_1 t + \varphi_1$ to $\omega_2 t + \varphi_2$ and eventually will settle at $\omega_2 t + \varphi_2$. The standard definition of *switching speed* is the time it takes the switching transient to achieve an output phase of $\omega_2 t + \varphi_2 \pm 0.1$ rad (approximately 5.7°). Note that in most cases φ_1 and φ_2 are not a part of the measurement since their values are not a parameter in the overall system and the user does not care about their values. But from time to time stringent requirements arise where the phase is also specified relative to some given reference. See Chap. 2.

1-2-8 Phase transient

In most applications, the behavior of the phase when in a transient state is not defined, as shown in part *a* of Fig. 1-1. However, many applications need to define the transient characteristics very carefully. Two typical requirements are as follows:

1-2-8-1 Phase-continuous switching. This means that during switching the phase transition shall exhibit (almost) no transient and shall ideally look as shown in part *b* of Fig. 1-1. Such a feature

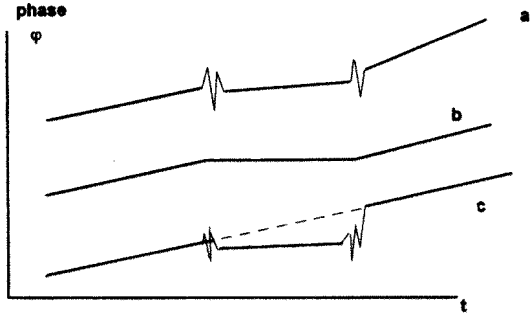


Figure 1-1 Phase switching in transition.

is important when one is attempting to generate a synthesized sweep, also known as *linear FM,* and has many applications in measurements, EW, radar, and specific modulations [e.g., *minimum shift keying* (*MSK*)]. Such a phase transient is smooth and generates very little "noise," and this is very desirable in systems and networks.

1-2-8-2 Phase memory switching. This means that if the FS runs at f_1 and then is switched to $f_2, f_3, f_4, \ldots$ and back to f_1, then it will resume the phase where it would have been if it were running continuously at f_1, as shown in part *c* of Fig. 1-1.

This requirement is simple to achieve if all the output frequencies are generated simultaneously and are switched to the specific output ($f_1, f_2, f_3, \ldots$) upon command. In such a case, every generator will continue to oscillate even when it is not used, and therefore, when it is reconnected, it will preserve its phase. However, if a single switched output is used, this requirement is sometimes quite tricky to achieve (see Ref. 12). Many applications require such a feature, e.g., coherent pulse Doppler radar imagers that frequency-hop but use coherent pulse detection (for predetection integration).

1-2-9 Harmonics

This parameter specifies the level of harmonics of the output frequency and depends on many components inside the FS. It is expressed in decibels relative to the output frequency (carrier) output power.

1-2-10 Spurious output

This specification defines the level of any discrete output frequency spectral line not related to the carrier. Most users do not consider harmonics as spurious signals. However, subharmonics, because of either multiplications or those that appear as DDS artifacts, are considered spurious signals even though they are sometimes specified separately. This parameter is expressed in decibels relative to the carrier output power. Unlike noise, spurious signals are only discrete spectral lines not related to the carrier, meaning that they exhibit periodicity.

1-2-11 Phase noise

From the purist's standpoint, there are no deterministic signals in the real world! All real signals are narrow-band noise. Every signal we generate is derived from an oscillator. Oscillators are positive feedback amplifiers with a resonance circuit in their feedback path. Since noise always exists in the circuit, upon power up this noise is amplified in the resonator band until a level of saturation is achieved. Then the oscillator passes from the transient to its steady state. Thus, the quality of the signal is mainly determined by the resonator Q. The signal that we usually refer to as a "sine wave" is actually *narrow-band noise*. The quality of the signal is determined by how much of its energy is contained close to the carrier. The center frequency is actually the average—the mean—of the noise frequency. Phase noise in a way is the standard deviation of the noise. In very high-quality signals, like crystal oscillators (Q range of 20,000–200,000), 99.99% of the signal energy can be contained within .1 Hz of the center frequency.

This parameter specifies the phase noise of the output carrier relative to an "ideal" output. The ideal output of a sine-wave generator is given by

$$A \sin (\omega_0 t + \varphi) \tag{1-1}$$

and its presentation in the frequency domain is a delta (Dirac) function at angular frequency ω_0:

$$F(\omega) = A \quad \delta (\omega - \omega_0) \tag{1-2}$$

Such a signal contains all its energy in a single frequency ω_0 and

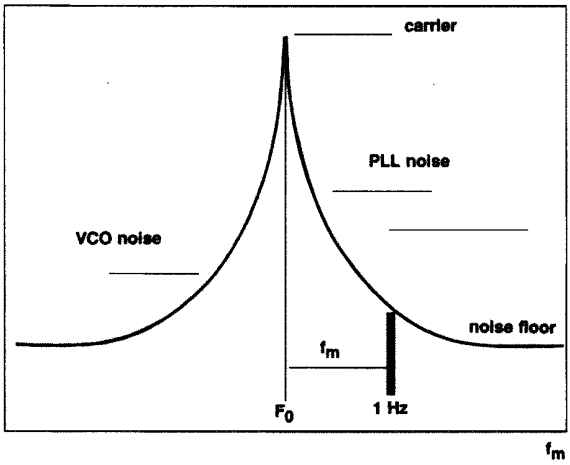


Figure 1-2 Typical phase noise plot.

has an ideal bandwidth of zero. Such a signal must be of infinite time (otherwise its spectrum will have a finite width greater than zero) and infinite power. However, the reference to a delta function is convenient for theoretical evaluations. High-quality frequency synthesizers generate signals which contain 99.99 percent of their energy in less than 1 Hz of bandwidth around the carrier. Crystal oscillators can contain 99.99 percent of their energy in less than 0.01-Hz bandwidth.

Obviously, in the real world the only signals we can generate are given by

$$A[1 + n_1(t)]\sin[\omega_0 t + n_2(t) + \varphi] \tag{1-3}$$

where $n_1(t)$ represents the amplitude instability and $n_2(t)$ represents the phase perturbations, both relative to the ideal case. These noise functions are random by nature and represent a spectrum that has to be designed to meet a specification.

In most synthesizers the amplitude noise is much lower than the phase noise and is not specified separately. However, the phase noise is a major parameter and is expressed in a few ways. The most common way is to specify the noise density in 1-Hz bandwidth at specific offset f_m from the carrier, as shown in Fig. 1-2. For the ideal signal, there is no energy at any offset from the carrier. Although this has become a de facto industry standard in defining and specifying phase noise, the measurement itself is sometimes

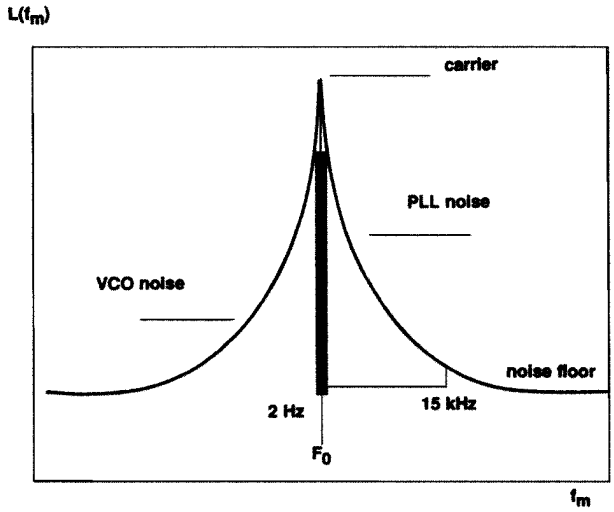


Figure 1-3 Integrated phase noise. *S/N* = signal/noise in 30 kHz (excluding 2 Hz around the carrier).

quite complicated, and the instruments necessary to make the measurement are quite expensive.

Another method of defining phase noise is to measure the integrated noise in a given bandwidth around the carrier but excluding ±1 Hz around the carrier. This is shown in Fig. 1-3. Obviously this method is related to the first one by the integration of the noise energy under the phase noise curve. However, compared to phase noise measurement, this is a simple measurement to make. The information available from such a measurement is a good indicator of the overall performance of the unit; but since this is an integrating measurement, the total noise power is measured even though its detailed spectral shape is lost. Traditionally (probably because of applications related to voice), the noise bandwidth is measured between 1 Hz and 15 kHz. So the measurement is the ratio of the total signal power to its noise from 1 to 15 kHz from the carrier (30 kHz of noise bandwidth). As an indicator, high-quality VHF/UHF synthesizers achieve a ratio of 60 to 70 dB and better.

Another method is to measure either FM noise, given in hertz root mean square or phase noise in degrees root mean square. Yet another method is to measure the phase noise in the time domain, and it is referred to as the *Alan variance*. By measuring the time

fluctuations it is possible to infer the spectrum of the signal. All these methods are related mathematically and must be consistent with one another. For detailed analysis of phase noise, see Chap. 2 and Refs. 13 and 14. Usually the FS phase noise reaches a noise floor, as shown in Fig. 1-2, and this parameter is sometimes specified, too. [A program to convert $L(f_m)$ to root-mean-square degrees is included on a disk the reader may obtain from the author (see Chap. 10).]

1-2-12 Standard reference

Since all synthesizers use a reference time base input, this specifies the reference frequency (usually 5 or 10 MHz, but there are many others), and its parameters such as stability, phase noise, spurious signals, and power level.

1-3 Auxiliary Specifications

These specifications are usually related to the execution of the specific synthesizer but are not dealt with here. Usually they include parameters such as size, power supply requirements, environmental factors, quality, and reliability.

1-4 Review of Synthesis Techniques

In this section we present a concise review of the three major synthesis techniques. But before we go into the details of these main techniques, it is worthwhile to mention what might be the simplest and crudest technique—digital frequency synthesis, namely, a programmable divider. This is not a common way of synthesizing frequencies, but it is applicable for a variety of programs. For example, a clock of 80 MHz and a divider in the range of 2000 to 4000 produce a synthesizer with 2000 frequencies, in the range of 20 to 40 kHz. The step size is not constant and actually varies (in this case) 4:1, but is 20 Hz maximum, which is good enough for many applications, especially in communications, that can make use of this simple device, which can be easily executed today by using CMOS gate array technology at extremely low power. This type of device is beyond the scope of this text.

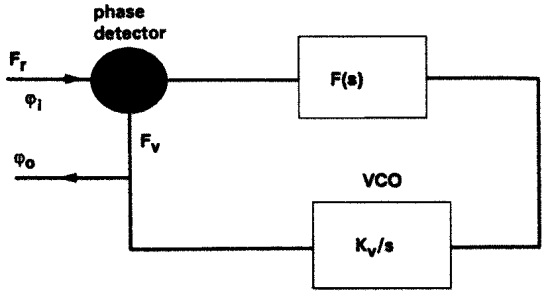


Figure 1-4 PLL block diagram.

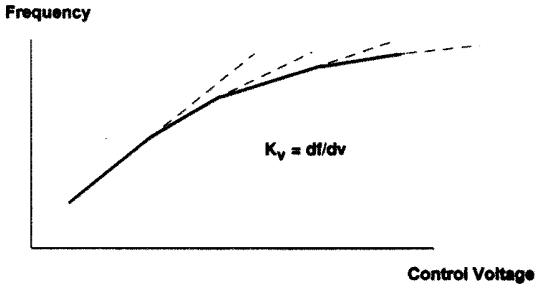


Figure 1-5 VCO control characteristics and piecewise linearization.

1-4-1 Phase-locked loop

As mentioned before, the phase-locked loop (PLL) is by far the most popular frequency synthesis technique. It is basically a nonlinear (the phase detector is a nonlinear device) feedback loop, as shown in Fig. 1-4. The PLL consists of a voltage controlled oscillator (VCO), a phase detector, a variety of dividers, and a loop filter.

The VCO is a device whose output frequency depends on the input control voltage. The relation is nonlinear (a typical response is shown in Fig. 1-5) but monotonic. However, when locked, the VCO can be assumed to be linear; it is both practical and convenient for analytical purposes. Variation in the VCO control characteristics (i.e., this nonlinearity) affects the loop parameters, and loop linearization (or compensation) is used extensively. Generally, the VCO output waveform is given by

$$A_{\text{out}}[t, \omega(v)] = A(t, v) \sin[\omega(v)t + \varphi] \tag{1-4}$$

where A is the signal amplitude and ω is the angular frequency, both depending on time t, and control voltage v.

As a first approximation, we assume that A has a constant envelope (does not depend on t or v) and that ω is a linear function of v. Therefore we can write Eq. (1-4) as

$$A_{out}(t) = A \sin[(\omega_0 + K_v v)t + \varphi] \tag{1-5}$$

Here K_v is the VCO constant [rad/(V s)]. Since we assume that the frequency is linearly dependent on v and is given by

$$\omega(v) = \omega_0 + K_v v \tag{1-6}$$

as mentioned, the linearization is justified and is assumed for the purpose of simpler analysis. In reality, when the loop is locked, frequency variations are tiny, and the constant-VCO assumption is correct as a piecewise linearization of the graph in Fig. 1-5.

Since phase is the integral of the angular frequency, we can complete the approximation by writing that the VCO transfer function, given by

$$\frac{\varphi_o(s)}{V} = \frac{K_v}{s} \tag{1-7}$$

as the Laplace transfer function of the VCO output phase.

The phase detector produces an output voltage proportional to the difference in phase between its inputs and is always a nonlinear function. Typical phase detector output transfer functions are shown in Fig. 1-6. However, close to the locked position this func-

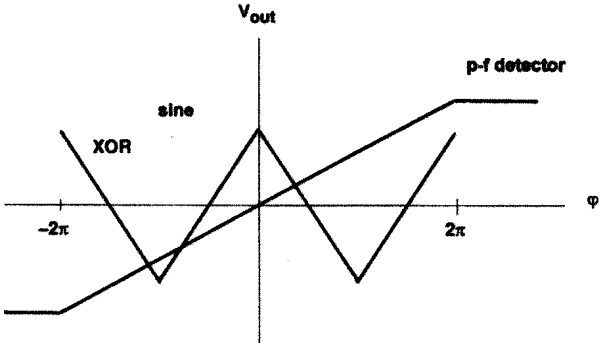


Figure 1-6 Phase detector output characteristics. p-f = phase-frequency.

tion can be assumed to be linear (this is also justified since in the locked condition most frequency synthesizers operate with a very high signal-to-noise ratio and the phase detector therefore operates mainly at a fixed-phase position). Hence

$$V_d = K_d(\varphi_i - \varphi_o) \qquad \text{V/rad} \tag{1-8}$$

where V_d is the phase detector output voltage.

Now the loop transfer functions can be described as

$$V_d = K_d[\varphi_i(s) - \varphi_o(s)] \tag{1-9}$$

Let

$$V_c(s) = V_d(s)F(s) \qquad \text{control voltage} \tag{1-10}$$

where $F(s)$ is the loop filter transfer function and V_c is the VCO control voltage. Solving these simple equations yields

$$\varphi_o(s) = \frac{\varphi_i(s)K_dK_vF(s)}{s + K_vK_dF(s)} \tag{1-11}$$

and the transfer function $H(s) = \varphi_o(s)/\varphi_i(s)$ is given by

$$H(s) = \frac{K_dK_vF(s)}{s + K_dK_vF(s)} \tag{1-12}$$

Also, following these equations will show that the *error transfer function,* defined as

$$H_e(s) = \frac{\varphi_i(s) - \varphi_o(s)}{\varphi_i(s)} \tag{1-13}$$

is given by

$$H_e(s) = \frac{s}{s + K_dK_vF(s)} \tag{1-14}$$

Since we linearized all components, given K_v and K_d, the feedback loop behavior depends mainly on $F(s)$.

Also note that the error function has high-pass characteristics, and therefore a true direct-current (dc) modulation of a PLL circuit is not possible. This function, however, also referred to as *dc frequency modulation,* is possible in other synthesis techniques.

1-4-1-1 First-order loop. A first-order loop is obtained when $F(s)$ = constant, say, A. This means that the loop filter has a fixed gain but no dependence on frequency. The gain is necessary because of the difference between the output voltage of the phase detector and the required control voltage input to the VCO. (Most phase detectors produce output voltage levels of 0 to 2 or 5 V while the VCO control might require 10, 15, and sometimes 24 or even 50 V to cover its operating range.)

In this case, the loop transfer function $\varphi_o(s)/\varphi_i(s) = H(s)$ reduces to

$$H(s) = \frac{K_v K_d A}{s + K_v K_d A} \tag{1-15}$$

For convenience we shall designate

$$K = K_v K_d A \tag{1-16}$$

and rewrite $H(s)$ for a first-order loop:

$$H(s) = \frac{K}{s + K} \tag{1-17}$$

As can be seen, this loop leaves few options to the designer since the loop parameters K_v, K_d, and A dictate the behavior of the feedback mechanism. Note that there is only one integrator in this PLL (phase is the integral of frequency, and the VCO characteristics in the Laplace domain have been described as K_v/s) and therefore only one pole in the transfer function. An intuitive approach to the loop behavior can be taken by realizing that the transfer function is that of a single-pole low-pass (RC) filter. So, for a step in the input phase, say φ_i, the output phase will be given by

$$\varphi_o(t) = \varphi_i(1 - e^{-t/K}) \tag{1-18}$$

and the phase error will be given by

$$\varphi_o - \varphi_i = \varphi_i e^{-t/K} \tag{1-19}$$

This assumes that the input phase step is fixed. This shows immediately that in such a loop, fixing K_v, K_d, and A determines immediately the dynamics of the loop and its noise *bandwidth* (BW), which is defined by

$$BW = \int_0^{\infty} |H(j\omega)|^2 \, df \tag{1-20}$$

and is given by $K/4$.

The noise BW of a PLL is an indicator of the loop bandwidth, and its calculation presents the integrated bandwidth of the loop and a measure of its speed of response.

The transfer function of a first-order loop is similar to that of an *RC* filter; and the error transfer function $e(s)$, indicating the error after a transient has settled, $e(s) = H_e(s)\varphi_i(s)$, is given by

$$e(s) = \frac{s\varphi_i(s)}{s + \frac{3}{4}K} \qquad \frac{e(s)}{\varphi_i(s)} = \frac{s}{s + \frac{3}{4}K} \tag{1-21}$$

The error function can be calculated for a phase step by using the final-value theorem, which states that steady state in the time domain can be calculated from the transfer function in the frequency domain. Accordingly, for a phase step φ_i, the final value of the error is given by

$$\int_{-\infty}^{+\infty} x(t) = \lim_{s \to 0} sX(s) \tag{1-22}$$

where $X(s)$ is the Laplace transform of $x(t)$ and is therefore in this case

$$\lim_{s=0} \frac{s\varphi_i}{s + K} = 0 \tag{1-23}$$

Thus a phase shift in the input will be tracked by the output. However, a phase ramp, or a frequency error $d\omega$, yields

$$\lim_{s=0} \frac{\frac{d\omega}{s}}{s + \frac{3}{4}K} = \frac{d\omega}{\frac{3}{4}K} \tag{1-24}$$

Thus a first-order loop when one is tracking a phase ramp (frequency change) will generate a fixed phase error, proportional to $d\omega$ and K. Obviously higher-level changes in the phase rate (parabolic and higher) cannot be tracked and create a diverging error. With only one integrator (the VCO) in the loop, this is expected.

This PLL structure is not particularly popular for FSs because of its lack of degrees of freedom in the design.

1-4-1-2 Second-order loop. This model of the PLL is the most commonly used in the FS industry. Although many designers claim that in reality there are no second-order loops (since the devices used to realize the loop filter always add poles), this represents an approximation to an analysis that is simple and yields a good theoretical approximation of the behavior of the majority of PLL designs. In this case

$$F(s) = \frac{1 + sT_2}{sT_1} \tag{1-25}$$

[note the added integrator in the network $F(s)$] and

$$H(s) = \frac{K(sT_2 + 1)/(T_1)}{s^2 + s(1 + KT_2)/(T_1) + K/(T_1)} \tag{1-26}$$

Following the common notions of control theory, we define

$$\omega_n = \sqrt{\frac{K}{T_1}} \tag{1-27}$$

and

$$\xi = \frac{\omega_n T_2}{2} \tag{1-28}$$

and the transfer function can now be represented as

$$H(s) = \frac{2\xi\omega_n s + \omega_n^2}{s^2 + 2s\omega_n\xi + \omega_n^2} \tag{1-29}$$

As in most second-order control systems, the characteristics are controlled by ω_n, also called the *loop natural frequency,* and the damping factor ξ (both designators imported from control theory). Such a loop can be controlled by $F(s)$ to arbitrary ξ and ω_n, and it can be shown (Ref. 1) that the loop BW, as defined in Eq. (1-20), is given by

$$B_L = \frac{\omega_n}{2(\xi + 1/4\xi)} \tag{1-30}$$

as shown in Fig. 1-7.

The loop filter is usually realized by either a passive network or an active integrator, as shown in Fig. 1-8. The design equations for

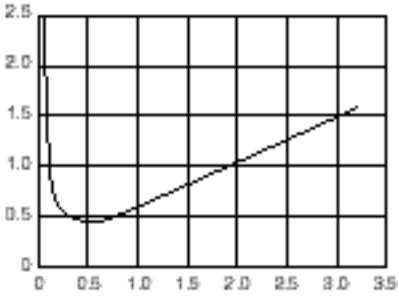


Figure 1-7 Loop bandwidth as a function of the damping factor.

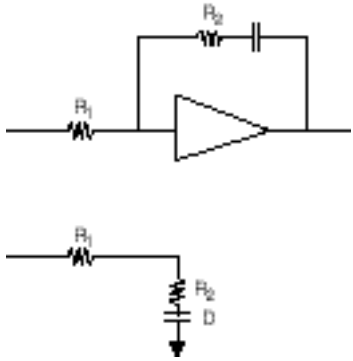


Figure 1-8 Second-order loop circuits.

the integrator network are given by

$$\omega_n = \sqrt{\frac{K}{T_1}} \tag{1-31}$$

$$\xi = \frac{\omega_n T_2}{2} \tag{1-32}$$

and for the passive network by

$$\omega_n = \sqrt{\frac{K}{T_1 + T_2}} \tag{1-33}$$

$$\xi = \frac{\omega_n (T_2 + 1/K)}{2} \tag{1-34}$$

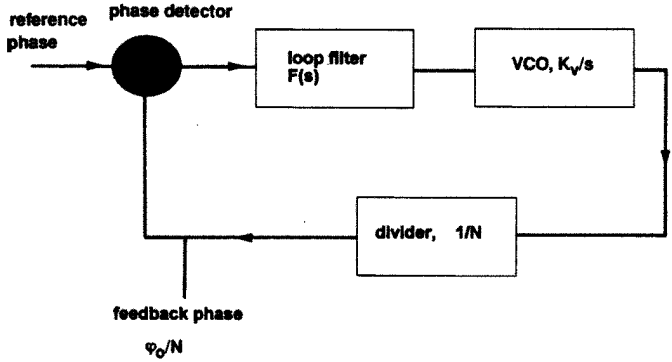


Figure 1-9 PLL for frequency synthesis.

Note that K_d is in volts per radian, K_v in rad/(s · V), K in 1/s, and ω_n in radians per second; ξ is dimensionless.

In PLL synthesizers, the output of the VCO is usually followed by a divider, as shown in Fig. 1-9. In the lock conditions, the output frequency will be given by NF_{ref}, and so by changing N, the output frequency is changed. All the equations stay the same, except the VCO constant changes from K_v to K_v/N.

The transfer function is given by

$$H(s) = \frac{KF(s)}{s + KF(s)/N} \tag{1-35}$$

For a second-order loop, it can be shown that the steady-state error for a step input and for a linear phase ramp $d\omega$ is 0 (there are two integrators in the loop), but a parabolic phase rate (linear FM) cannot be tracked and a frequency error is generated. Obviously higher-order loops are used for applications where higher-level phase changes are required, but the majority of PLL applications, especially for frequency synthesis, use second-order designs.

1-4-2 Direct analog synthesis

Unlike PLL, the direct analog (DA) technique uses arithmetic operations in the frequency domain (but no closed-loop feedback mechanisms) to convert the input reference signal to the required output frequency. The main tools for the DA technique are therefore comb generators, multipliers, mix and filtering, and division.

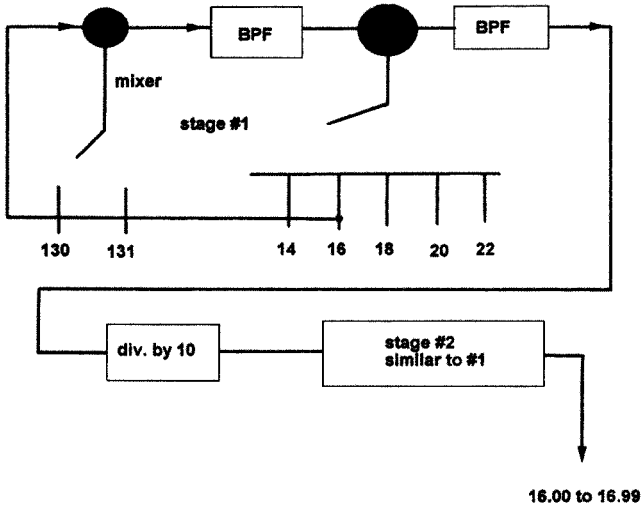


Figure 1-10 Direct analog design using multiple references. BPF = bandpass filter.

Because such operations are complex, it is desirable to design repeating building blocks, so that their production will be economical; otherwise, price and complexity are both very high.

To demonstrate the basic elements of the DA technique, we consider a tentative design of a synthesizer that covers 16.0 to 16.99 MHz of output frequency range, has 0.01-MHz (10-kHz) step size, all derived from a 10-MHz reference. This demonstration design (Fig. 1-10) requires the following reference frequencies: 14, 16, 18, 20, 22, 130, and 131 MHz. Given these reference frequencies, the generation of which is not necessarily trivial, a common block might look like that in Fig. 1-10.

Note that the output of the first stage serves as the input to the second stage (similar), and so at the output of the first stage 10 frequencies will be generated, from 16.0 to 16.9 MHz, but at the output of the second, the complete range of 16.0 to 16.99 MHz is achieved (100 frequencies). Note that by adding more similar stages, the resolution of the synthesizer can be increased to any required level. The same stage can therefore be used repeatedly without the need to generate more references.

Usually, in such designs, the reference frequencies are generated by direct analog methods rather than PLL, i.e., comb generators, filters, mixing, and dividing. As an example, one possible

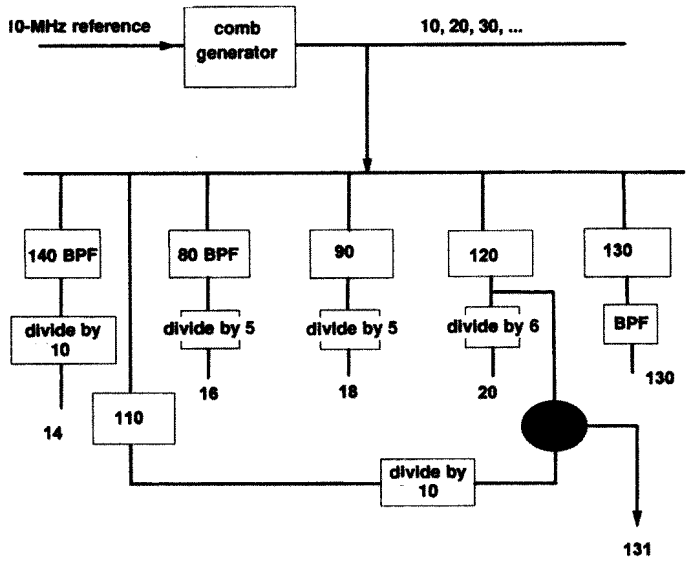


Figure 1-11 Reference generation for DA design.

method is demonstrated in Fig. 1-11. The 10-MHz comb generates 10-MHz comb lines 10, 20, 30, ..., 140 MHz. The 14 is generated by 140/10 (or 70/5), the 16 by 80/5, the 18 by 90/5, the 20 by 120/6, the 22 by 110/5, the 130 by 120 + 10 (both available), and the 131 by 120 (available) + 110 (available)/10.

This architecture usually operates in blocks of decades and is used here to demonstrate the DA principles. There can be many other variations, but this is quite a typical and efficient design. The basic element of DA is therefore mix and filter.

Note that the spectral purity depends on the spectral purity of the references (usually excellent), and the speed of the synthesizer in this case depends on the speed of the switches that switch in and out the reference frequencies and the response time of the filters.

The above design can achieve switching speeds of 3 to 10 microseconds (μs) depending on the detailed design.

Note also that if all the above operations were designed at frequencies 5 times higher, the filter's bandwidth would have increased 5 times and the speed would depend mainly on the speed of the switches. Such a design could achieve submicrosecond speed. Obviously the compromise will involve cost.

Note also that such a design does not possess the quality of phase memory, although at first it might look as if it does. Since

all the references run continuously, they all maintain phase memory. The problem occurs in the divider. We know that mix and filter preserve the phase of the references since it is basically a *single-sideband* (*SSB*) operation that can be expressed as

$$e^{j(\omega_1 t + \varphi_1)} \cdot e^{j(\omega_2 t + \varphi_2)} = e^{j[(\omega_1 + \omega_2)t + \varphi_1 + \varphi_2)]} \qquad (1\text{-}36)$$

and the phases are preserved. However, the divider (here by 10) will suffer a transient, and its output phase can be at any one of 10 possible output phase states, and so the phase preservation is lost. Therefore, there is no phase continuous switching in such a design, and this means that linear FM sweeps cannot be generated.

As a general rule for such block decade design, the output frequency is given by

$$F_{\text{out}} = F_i + \frac{F_1}{10} + \frac{F_2}{100} + \cdots = F_i + \sum_{j=1}^{n} F_j(10^{-j}) \qquad (1\text{-}37)$$

where F_i and F_j are the inputs to the block.

In the above case, if $N = 0$ were the last stage and did not contain the divide-by-10 device common to all other stages, then the output frequency range would be 160.0 to 169.99 and the number of 9s (or the resolution) depends on the number of stages being employed. This is convenient for two reasons: the frequency coverage is 10 MHz (rather than 1 MHz for the common block), and the final frequency is higher and easier to convert upward.

Early DA designs were quite complicated, had many crystal references in them, and had complex architectures. All this is gone now. All DA designs consist of repeatable blocks, which imply efficiency in production and elegance.

Another technique that is usually associated with DA is called *drift cancel* and is worth mentioning. The general idea is demonstrated in Fig. 1-12.

A comb generator is used to generate a comb spectra line, in this example 50 MHz apart, and a free-running VCO and a *bandpass filter* (*BPF*) are used to output only one line at a time.

In the example, the VCO will be tuned to 2000 MHz to output 500 MHz. If the VCO is tuned to 2050 MHz, the output will be 550 MHz; the VCO tuned to 2100 generates 400 MHz; and so on. Note that the mathematical operation here is

$$F_{\text{out}} = F_{\text{VCO}} - (F_{\text{VCO}} - F_{\text{comb line}}) = F_{\text{comb line}} \tag{1-38}$$

The VCO serves an auxiliary function and cancels itself. The VCO can be free-running since the level of accuracy necessary is only to bring the desired comb line into the BPF passband.

Although this is a simple, elegant, and economical technique, the design is usually more complicated than it shows. Excellent isolation must be maintained to eliminate leakage of undesired comb lines. In addition, the cancellation effect must be calculated relative to the phase noise of the free-running VCO. Ideally the two VCO paths should have the same delay, and the signals should cancel perfectly. In reality, this is not the case, and the cancellation effect is finite. It is usually very good close to the carrier, but degrades as we move away from the carrier. The reason is clear because the cancellation takes the form $\varphi(t) - \varphi(t + T)$, where T is the difference in the delay path of the two branches $\varphi = \omega_0 t + n(t)$. Thus the cancellation takes the form

$$\varphi_n = \omega_0 T + n(t) - n(t + T) \tag{1-39}$$

A typical cancellation profile for a 2.5-MHz BPF is given in Table 1-1. This implies the requirement of good phase noise characteristics from the VCO, since at, say, 100 kHz from the carrier, the cancellation effect will improve the VCO phase noise by only 15

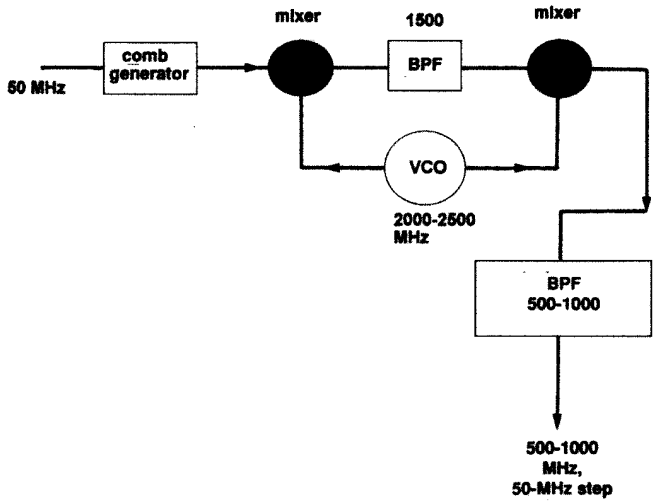


Figure 1-12 Drift-cancel loop used in DA synthesizers.

TABLE 1-1 Typical Noise Cancellation of 2.5-MHz Drift-Cancel Loop
(Delay between cancellation paths is approximately 1 μs)

Offset from carrier (Hz)	Cancellation (dB)
1×10^3	45
1×10^4	30
5×10^4	20
1×10^5	15
5×10^5	5
1×10^6	—

dB. This causes most DA designs that use this technique (and this is attractive because of simplicity and cost) to have a relatively (to PLL) high noise floor, in the order of −130 to −135 dBC/Hz, in *ultrahigh frequency* (*UHF*). (Here dBC indicates decibels referred to *carrier*.) At a premium, the VCO can be replaced by a synthesizer. DA synthesizers will not be discussed in this text in great detail.

1-4-3 Direct digital synthesis

DDS is an emerging and maturing signal generation technology. Up to 10 years ago, this technique was rather a novelty and was used in very limited applications. However, due to the enormous evolution of digital technologies (speed, integration, power, cost), digital signal processing (DSP), and data conversion devices, it is becoming increasingly popular, and its performance improves constantly.

There is a fundamental difference between DDS and DA or PLL. Although both PLL and DA techniques use digital devices, such as dividers and phase detectors, the PLL and DA techniques are fundamentally analog disciplines. The basic signal generator in both techniques is an oscillator, which is a feedback-tuned amplifier set to operate under specific conditions (controlled instability). The oscillator is manipulated to allow the generation of a range of frequencies. In DDS, the signal is generated and manipulated digitally from the "ground up," and after all the digital manipulations are completed, it is converted to an analog signal via a *digital-to-analog converter (DAC)*.

The DDS is thus a computing machine where signals are repre-

sented by numbers and should be considered a DSP discipline. There are many compelling reasons to do this.

Signal generation is, after all, the heart of every electronic device and top of the list of standard industrial reasons to "go digital." Other engineering drivers such as producibility, repeatability, reliability, and very high accuracy demand the use of digital techniques, which now infiltrate not only DSP but also signal generation. DDS adds dimensions not possible with analog designs. In a way there is some similarity between the evolution of DDS and that of digital recording and the use of the *compact disk* (*CD*). The signal has high fidelity, waveforms have been converted to numbers (an old Pythagorean dream come true), there is total control of the signal parameters at all times, and the density of digital circuitry allows a very high level of integration in small size economically.

Thus, after years of maturity, DDS is finding many applications and has been established as a fundamental and important signal generation discipline.

1-4-3-1 Basic theory. Like most DSP disciplines, the fundamental root of DDS is based on the sampling theorem, due to Shannon (see Chap. 4). The theorem states that any (stochastic, with finite energy) signal having a band-limited spectrum (i.e., the signal has no energy at frequencies above $\omega_0 = W$), such a signal can be represented by its discrete samples in time, provided that the sampling rate is at least $2F_0$, where $F_0 = \omega_0/2\pi$. A simple proof is provided in Chap. 4.

The sampling theorem shows that such a signal can be fully recovered from its samples and that, in the process, many other frequencies are being generated. These "artifacts" are also referred to as the *aliasing signals*. (See App. 4A.) We will demonstrate an intuitive explanation of what is happening in the sampled data domain.

Suppose that a viewer sits in a dark room. In front of her there are a wheel and a single bar connecting the center of the wheel to the perimeter. Suppose also that a light source is set behind her and flashes very short flashes (almost like a delta function sampling) at the rate of 10 flashes per second.

Now the wheel starts to rotate at 1 cycle per second, or 1 Hz, as seen in Fig. 1-13, say, clockwise. If the viewer is required to

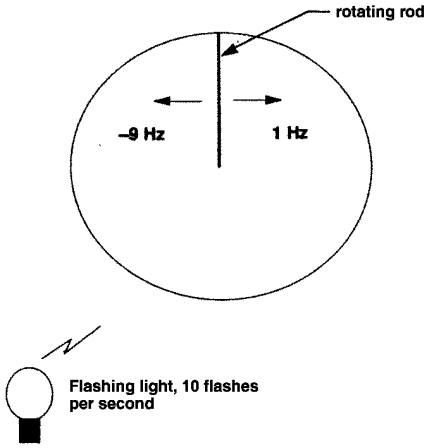


Figure 1-13 Sample data experiment. The rod rotates at 1 Hz clockwise (positive), and the light flashes at 10 Hz.

describe what is happening in front of her eyes, she will not be able to give a conclusive answer. Note that what the viewer sees (which is the reality in this case) is the bar advancing 36° clockwise at every flash. Her interpretation will be that the wheel is rotating at 1 Hz, or 11 Hz, or 21 Hz, or at any frequency that is $10n + 1$ Hz clockwise, but also at 9 Hz or 19 Hz or any frequency that is $10n - 1$ Hz counterclockwise.

This is a cardinal point in DDS theory. The same results will be observed by the viewer if the bar is rotated at $10n + 1$ or $10n - 1$, or generally n (light sampling) $\pm$ K (wheel frequency) rotations per second.

This interpretation of the experiment is, of course, the spectrum shown in the mathematical proof of the sampling theorem. The mathematical proof is enlightening by itself. We can interpret the clockwise rotations as positive frequencies and the counterclockwise rotations as negative frequencies. These negative frequencies are obviously as real as the positive ones and in DSP terminology are usually referred to as the *aliasing frequencies.*

A special interesting case arises when the rotation is exactly 5 Hz. At this point, all the positive and negative frequencies converge. This is referred to as the *Nyquist frequency* (half the sampling rate).

In DDS, the procedure of the signal sampling is reversed. And the process is therefore reversed. Suppose that, instead of sampling a sine wave, we generated its samples. (All can be calculated since the waveform is known; the sine is a perfectly known wave-

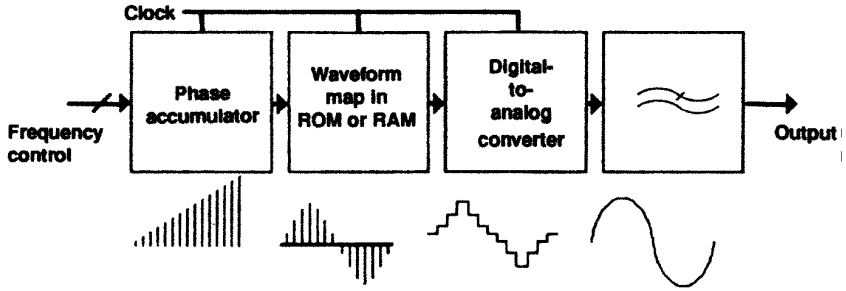


Figure 1-14 DDS block diagram and waveforms.

form, unlike a stochastic process that is random by nature.) Then according to the theorem it will be possible to reconstruct the sine wave signal perfectly.

In essence, that is what DDS is. The rest is details and techniques for efficient, economical, and fast executions.

1-4-3-2 DDS concepts. There are a great variety of DDS implementations, as will be demonstrated in Chap. 4. However, the dominant one consists of four elements: an accumulator, a sine lookup table, a DAC, and a *low-pass filter* (*LPF*). See Fig. 1-14. Remembering that the presentation of a fixed-amplitude, fixed-frequency, and fixed-phase sine wave is given by

$$A \sin(\omega t + \varphi) \tag{1-40}$$

we can trace the signal buildup as follows: The signal phase is a linear function, as shown in Fig. 1-14. The gradient or slope of the phase $d\varphi/dt$ is the angular frequency ω. To generate the amplitude of the output waveform, it is necessary to transform the phase $\varphi(t)$ to $\sin[\varphi(t)]$, and this is usually done by using a read-only memory (ROM), since the transformation is nonlinear and ROM (or RAM) is a convenient tool.

The output of the ROM is thus the digital representation of the sine wave signal amplitude, (digital) samples, and the DAC converts it to an analog sine wave. The LPF removes all the aliasing frequencies and causes the signal to appear smooth, as shown in Fig. 1-14. Note that since the signal is synthesized from the ground up, we know exactly the state of the machine at all times; and it is relatively easy to add phase shifting, frequency changes, and amplitude modulation, all in the digital domain and with digital accuracy.

The accumulator is a device that performs the function

$$S(n) = S(n - 1) + W \tag{1-41}$$

Such a device is a digital integrator and produces a linear output ramp whose slope (rate of change) is given by W, the input control word. This device is used to generate the phase ωt or in the sample data $W \cdot n \cdot T$, where T is the sampling time and depends on the clock at which the accumulator runs. The accumulator is operating as an indexer whose output (representing the phase) controls the input to the ROM (or lookup) sine table, and can be viewed as a complex, yet easy to engineer and control, counter.

Suppose that the accumulator size is N bits, say $N = 32$ bit binary device. It is therefore able to accumulate from 0 to $2^{32} - 1$. Obviously, above this number the accumulator will overflow and start again from 0. The rate of accumulation depends only on the clock rate $F_{ck} = 1/T$ and W. And W can be as low as 0—in this case the accumulator will not increment (equivalent to generating a dc signal)—or any arbitrary number $W < 2^N - 1$, which is the case if all the N inputs equal 1. If we equate 0 with zero phase and $2^{32} - 1$ with 2π, then we have a device that generates phase from 0 to 2π periodically (since the device operates modulo 2^{32}), as shown in Fig. 1-14. Figure 1-14 shows a typical cycle waveform, and Fig. 1-15 shows that incrementing 1 or $2^{31} - 1$ actually causes the same (but reverse phase, see Chap. 4) effect.

We have already seen that in sample data systems the aliasing is the "shadow" (or image or reflection) of the signal, and their similarity is demonstrated for $W = 1$ or $W = 2^{32} - 1$. They can be considered as clockwise and counterclockwise or positive and negative frequencies of equal magnitude.

For a demonstration of the operation of the accumulator, let us

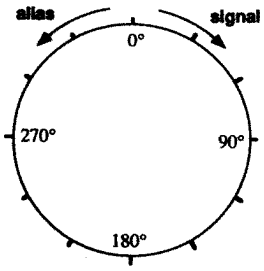


Figure 1-15 Signal and aliasing.

assume that we clock the device, an $N = 32$ bit accumulator, at $F_{ck} = 2^{32}/10$. Then if $W = 1$, it will take exactly 10 s (2^N clock ticks) to generate 0 to 2π. However, if $W = 2^{30}$, then it will take only $40/2^{32} = 10/2^{30}$ s (four clock ticks). Obviously W controls the rate of change of the accumulator, and the rate of change of the phase is the frequency ω. In the above example, for $W = 1$, the cycle is 1/10 or 0.1 Hz, while for $W = 2^{30}$ the cycle is equal to $2^{30}/10$ Hz, or $F_{ck}/4$ Hz.

Mathematically, since $\omega = d\varphi/dt$, we can rewrite

$$F_{out} = 2\pi \frac{d\varphi}{dt} = \frac{W}{2^N/T} = \frac{F_{ck}W}{2^N} \tag{1-42}$$

where F_{out} is the output frequency and T is the clock time. For this example, the cycle frequency will be given by

$$F_{out} = \frac{F_{ck}W}{2^{32}}$$

or

$$F_{out} = \frac{F_{ck}W}{\text{ACM}} \tag{1-43}$$

where ACM is the maximum number of states of the accumulator. Equation (1-43) is a generic equation for DDS.

Note that for all practical purposes, $W < 2^{N-1}$, because that is the Nyquist frequency and the point of conversion between the frequency and its "shadow" or aliasing.

Not all accumulators are binary, and we will mention others later (see Chap. 4).

The phase information is connected to the ROM which converts ωt to $\sin \omega t$ or φ to $\sin \varphi$. Since the accumulator is usually large and the memory size is limited, only part of the accumulator output bits is connected to the ROM. For example, if the 14 *most significant bits* (*MSBs*) of the accumulator are connected to the ROM and we require 12-bit output from the ROM to drive a 12-bit DAC, then the size of the required memory is $2^{14} \times 12$, which is equivalent to approximately 192,000 bits of memory and is already quite a large ROM. More on the ROM and ways to compress its size will be discussed later (Chap. 7).

We have introduced a level of truncation since not all the accu-

mulator bits are connected to the ROM. The error will be evaluated in Chap. 4.

The digital output bits of the ROM, which converted the phase information to amplitude, are now connected to the DAC and low-pass filter that generate the analog sine wave. The LPF rejects all aliasing frequencies and is therefore theoretically limited to the Nyquist frequency. It is sometimes referred to as the *antialiasing filter.*

1-4-3-3 Modulation. Since we have total control of the signal parameters, it is easy to include modulation ports to add frequency, phase, and amplitude modulation, all in the digital domain, as shown in Fig. 1-16.

Frequency is changed by changing the value of *W,* which is the basic function of the direct digital synthesizer. Because of the nature of DDS, phase continuous switching is simple to achieve. Note also that dc FM is possible in DDS. Sometimes when dc FM is required from a PLL system, it is done by locking the PLL to a direct digital synthesizer, and it is FM-modulated. The phase is changed by inserting an adder between the accumulator and the ROM, and this function is shown in Fig. 1-17. Adding (or subtracting) a number causes a phase shift. And amplitude is changed by inserting a multiplier between the ROM and the DAC.

If all modulations are employed, the output waveform will be given not as

$$A \sin (\omega t + \varphi) \tag{1-44}$$

but rather as

$$A(a) \sin [\omega(W)t + \varphi(b)] \tag{1-45}$$

Thus total digital control of the signal parameters has been achieved; see Fig. 1-16. Such devices that employ all digital modulations [and more, such as built-in linear FM (chirp) functions and standard functions like ramp and $\sin x/x$] are available in the market now.

1-4-3-4 DDS parameters. The following describes the design or outcome parameters of a typical DDS.

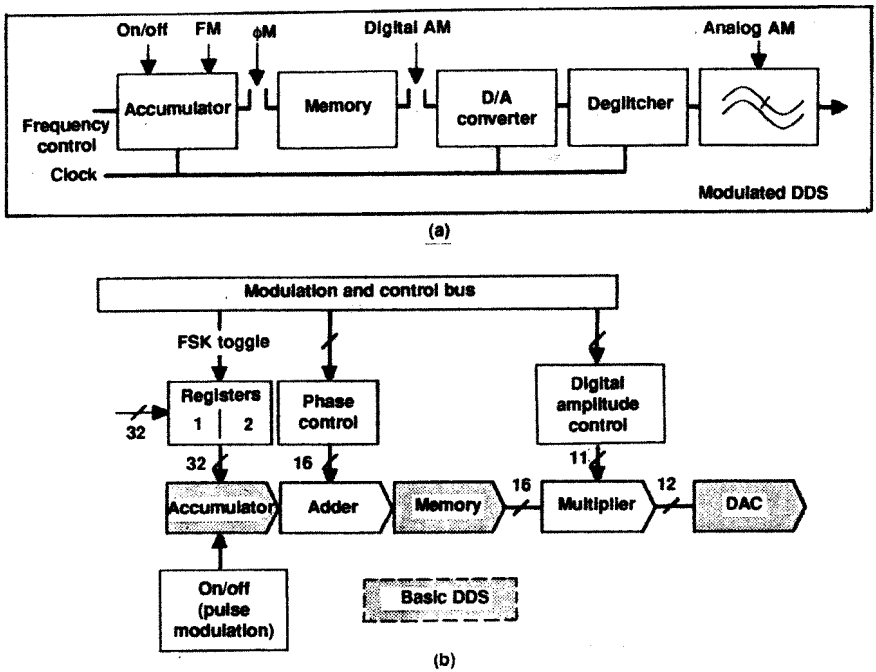


Figure 1-16 (*a*) Digital modulation in DDS. (*b*) DDS modulation implementations.

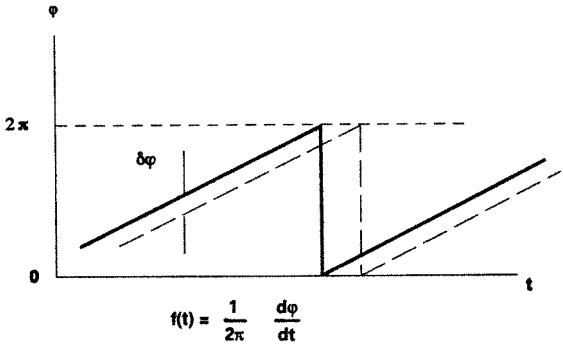


Figure 1-17 Phase control in DDS.

Output frequency. The output frequency of a DDS is theoretically limited to the Nyquist frequency. This means that at least two samples of the sine wave must be available within a cycle. Therefore theoretically the output frequency is limited by $F_{ck}/2$. Practical limitations in the realization of the LPF set the maximum output frequency at about $0.45F_{ck}$, or 45 percent of the clock frequency.

It is important to note that the aliasing frequencies are always generated. So when a direct digital synthesizer is controlled to generate, say $0.1F_{ck}$, the aliasing is generated at $0.9F_{ck}$; however, if it is controlled to $0.9F_{ck}$, it will also generate $0.1F_{ck}$! Close to the Nyquist frequency, these frequencies that cannot be filtered are spurious signals (there is a limit to the sharpness of the filter). In few applications (this represents a rarity in DDS applications), the aliasing frequencies are controlled (to generate the in-band signal) rather than the fundamental output (see Chap. 4).

Resolution (step size). The resolution or the smallest frequency increment or step size is given by F_{ck}/ACM, where ACM is the maximum number of states the accumulator produces. In the common case of a binary accumulator with, say, 32 input control bits (very typical in direct digital synthesizer chips), the resolution is given by $F_{ck}/2^{32}$, or more generally $F_{ck}/2^N$ for an N-bit accumulator.

Even at a clock rate of 100 MHz, for $N = 32$ the resolution is excellent, approximately 0.025 Hz! Therefore, the size of the accumulator controls the resolution, and increasing the accumulator size is quite simple and adds little to the cost and complexity of the design. Compared to the cost and complexity of resolution in PLL or DA techniques, this is a remarkable advantage.

Speed. The switching speed of a direct digital synthesizer is a combination of the time it takes the digital signals to propagate through the logic array and the settling time of the LPF. In most high-speed DDS implementations, pipeline in the logic is common and causes a delay. However, in a direct digital synthesizer clocked at 100 MHz, even if the pipeline (see Chap. 6) takes 25 clocks, if the LPF bandwidth is 40 MHz, the pipeline will cause a 250-ns (25×10 ns) delay and the LPF another 50 to 60 ns, so the total switching speed will be approximately 300 ns—still excellent and almost unmatched (or very costly) by the other synthesis techniques.

The pipeline delay is fixed and can be accounted for by the system in use since it is just a fixed delay and not a switching transient. Another issue is the time it takes to propagate through the accumulator itself. Suppose that in a 32-bit accumulator 14 bits are connected to the ROM. If the LSB is changed, it takes this bit $2^{32 - 14}$ clock ticks to propagate up the accumulator before it has an effect on the output! These peculiarities do not pose a real

problem, but have to be understood. If we continue with this example, it will take 2^{18} clocks, say, at 100 MHz, to impress the effect of the LSB on the output. This calculates to approximately 2.5 ms, but the frequency change of the LSB has been calculated to be 0.025 Hz. So what is the practical effect of a 2.5-ms delay in a signal of 0.025-Hz frequency? As this demonstrates, the diagonal relation between the value of the frequency change and the delay it causes actually creates no practical problem. Nonetheless, this question is asked often and needs to be well understood.

Phase continuity. Because of the structure of the direct digital synthesizer when frequencies are switched, it can be done in a smooth continuous manner. In Fig. 1-1, we can see that if it is timed correctly, the change in frequency is smooth and can be used if a sweep or specific modulation (like minimum shift keying) needs to be generated. In fact, DDS is the only accurate practical implementation of modulations like FSK, MSK, GMSK (popular for cellular applications) that require smooth phase transition.

Quantization. We will examine the quantization effects of DDS designs in Chap. 4. However, it is important to note that since a finite word length is used, digital quantization errors cause the generation of periodic quantization analog errors in the signal's amplitude, i.e., spurious signals. Today, spurious signals are the main disadvantage of DDS techniques, and most existing designs do not achieve spurious signal rejection better than 60 to 65 dB.

1-5 Comparative Analysis

It is worthwhile to highlight the different advantages and disadvantages of the synthesis techniques mentioned above.

Very wideband, the PLL technique operates from audio frequencies up to millimeter waves and depends on the frequency of the oscillators. PLL synthesizers are relatively simple; moderate to good switching speeds can be achieved; and they are low-cost and easy to apply in most analog modulations. Single-chip PLLs provide a high level of integration and low cost. However, resolution is complicated to achieve, good-quality oscillators are quite bulky, and digital modulation is complicated to apply with sufficient accuracy.

DA techniques are very wideband; via multiplications the signal can be generated up to about 100 GHz, and it will probably go further as microwave and millimeter wave technology evolves. Very high switching speeds are achieved, and the spectral purity is excellent, especially close to the carrier. However, DA synthesis is quite bulky, requires much hardware, and is expensive (sometimes very expensive); and digital or analog modulations are complicated to apply.

Direct digital synthesizers have limited bandwidth, approximately 400 MHz at this time. They are very simple and compact, and resolution comes almost free. They have a very high switching speed, phase-continuous switching, and digital producibility. However, as mentioned, the bandwidth is still limited (but improving), and spurious response is affected by quantization and DAC performance.

It is interesting to note that the three synthesis techniques do complement one another, and this is the main reason that more and more combination designs are emerging, mainly hybrids of PLL and DDS, to achieve wide bandwidth and good resolution. DA synthesis and DDS are integrated to achieve the speed, resolution, and capability of digital modulation (see the PTS, Schomandl, and Comstron products described in Chap. 9).

It is also accurate to say that both PLL and DA synthesis have achieved a high level of maturity while DDS is a technology still emerging. The PLL synthesizers still get the greatest attention from the industry since they cater to the majority of the market and are the most useful synthesis technique. As DA instruments improved in the last decades, their principles of operation did not change much, and the improvements were mainly due to evolution of microwave components.

In PLL synthesis there has been a major improvement with the introduction of the fractional-*N* technique, which is described in detail in Chap. 5. Fractional-*N* synthesis has great similarities to DDS where it is a part of the phase-locked circuit.

DDS has emerged in the last decade from a limited-use novelty into a major technology and popular synthesis technique. Part of the evolution is due to the improvement in digital technology, the introduction of low-cost integrated circuits and the evolution of data conversion devices, especially DAC technology. The relatively recent introduction of a high-speed, high-performance DAC and

DDS architecture onto a single low-cost CMOS chip enables this technology to reach a much wider range of applications.

Substantial improvements have been introduced by developing a higher level of integration of complete DDS on a chip, now including the DAC; decimal rather than binary architectures; ROM compression algorithms to shrink the size of the necessary ROM; and in the theoretical understanding and analysis tools.

1-6 Conclusion

The basic elements of the three major frequency synthesis techniques have been presented. As the requirements of communication systems, radar, and medical and industrial equipment become more stringent, frequency synthesis must evolve and comply with these requirements.

There have been major improvements in FS disciplines, mainly in VCOs and their designs and *computer-aided design* (*CAD*) tools for simulation and design, and in a variety of digital techniques, speeds, densities, integration, and functionality. At this time, digital frequency technology attracts high-level attention from system designers and industry leaders, for cellular and personal communications, radio and radar manufacturers, and government R&D programs.

The three FS techniques do compete, but mainly complement one another; and most modern designs utilize combinations of technologies to achieve high performance, compactness, and economy.

There is now a proliferation of digital and DSP disciplines in the FS market, and more can be expected. The designer must be familiar with analog, digital, and radio-frequency and microwave technologies to fully exploit their respective advantages.

Frequency synthesis is emerging as an exciting and dynamic field, with evolving challenges. New ground is broken often, and much attention is given to the emerging digital techniques in most technical meetings. The FS is now an integral part of any complex system design since the enhanced capabilities of the FS offer much more than just the generation of sine waves, for modulation capabilities and complex waveforms can be designed as part of the FS.

References

1. Floyd M. Gardner, *Phaselock Techniques,* Wiley, New York, 1980.
2. Jerzy Gorski-Popiel, *Frequency Synthesis Techniques and Applications,* IEEE Press, New York, 1975.
3. Vadim Manassewitsch, *Frequency Synthesizers: Theory and Design,* Wiley, New York, 1983.
4. Ulrich L. Rohde, *Digital PLL Frequency Synthesizers: Theory and Design,* Prentice-Hall, Englewood Cliffs, NJ, 1983.
5. R. C. Stirling, *Microwave Frequency Synthesis,* Prentice-Hall, Englewood Cliffs, NJ, 1987.
6. Roland E. Best, *Phase Locked Loops: Theory, Design, and Applications,* McGraw-Hill, New York, 1984.
7. Andrew J. Viterbi, *Principles of Coherent Communications,* McGraw-Hill, New York, 1967.
8. J. Tierney, C. Rader, and B. Gold, "A Digital Frequency Synthesizer," *IEEE Transactions on Audio and Electroacoustics,* March 1971, pp. 48–57.
9. William Egan, *FS by Phase Lock,* Robert Krieger Publishing, Malabar, Fl, 1990.
10. Ulrich Rhode, *Digital Frequency Synthesizers,* Prentice-Hall, Englewood Cliffs, NJ, 1981.
11. Sciteq Electronics, DDS-1, application note, 1990.
12. B. G. Goldberg, Device for fixing the phase of frequency synthesizer output, U.S. patent 4868510.
13. Jacques Rutman, "Oscillator Specifications: A Review of Classical and New Ideas," *Proceedings of the 31st Symposium of Frequency Control,* 1977.
14. W. C. Lindsey and C. M. Chie, "Specifications and Measurements of Oscillator Phase Noise Stability," *Proceedings of the 31st Symposium of Frequency Control,* 1971.

Chapter

2

Frequency Synthesizer System Analysis

In this chapter we present a short survey of the basic synthesizer system design parameters and analysis, common to all *frequency synthesizers* (*FSs*).

The common design goals are always to achieve the best possible performance (of those parameters important to the user) at the lowest possible cost, the smallest size and weight, and at the lowest power dissipation. Of the various parameters imposed on a design, some of the more important are (1) phase noise, (2) spurious signal level, (3) frequency range and step size, (4) switching time, and (5) size, cost, and power dissipation. Power dissipation has become a cardinal requirement for handheld equipment, and lower power dissipation always helps reliability. However, engineering is the art of compromise, and parameters must be traded off to effect an acceptable solution.

We will discuss the various demanding requirements and their effects on the design, and we will review the various operations pertinent to frequency synthesis, such as multiplication, division, mixing, phase noise, and frequency planning.

2-1 Multiplying and Dividing

As mentioned in Chap. 1, a FS performs operations of multiplication or division on the reference input frequency to generate its required outputs. This is done via true multipliers, mostly in *radio*

frequency (*RF*) and microwave circuits, digitally in DDS applications, PLL, and analog or digital dividers. We will examine the effect of these devices on the output spectra.

Consider the case of a sine wave $\sin \omega_1 t$ corrupted by a small spurious signal er $\sin \omega_2 t$. If this composite signal is multiplied by N, with the purpose to generate $\sin N\omega_1 t$, the result is

$$(e^{j(\omega_1 t)} + \text{er } e^{\omega_2 t})^N \qquad \text{(we assume er << 1)} \tag{2-1}$$

The result can be expressed by the standard polynomial

$$e^{jN(\omega_1 t)} + N \text{ er } e^{j[\omega_1 t(N-1)]}\, e^{j(\omega_2 t)} + \text{higher-order terms} \tag{2-2}$$

[these terms will become more complex to analyze if we use $(\sin j\omega_1 t + \text{er} \sin j\omega_2 t)^N$, but the net results are the same.] Since we assumed er << 1, the other higher-order terms are negligible. [If $N = 2$ and er = 0.001 (-60 = dB spurious signal), then the third term will be approximately $N \text{ er}^2 = -114$ dB. However, for high-level spurious signals, higher-order spurious signals might show up and have to be considered.]

The first term is the desired one and produces the required output frequency, now $\omega_1 N$. But the second term is an interference, a spurious signal which is generated at a frequency of $\pm\omega_2$ away from the desired output and with a voltage level of er$N$. If we define the ratio of the original signal to the spurious signal as 1/er, then multiplication results in a degradation of N. Or in decibels, the degradation is equal to $-20 \log N$.

As a demonstration, if the original signal starts with a spurious level of -80 dB, then multiplying by 2 will degrade the result to -74 dB and multiplying by 10 will degrade to -60 dB. Note that the first and major spurious signal shows at the *same* offset from the carrier as in the original signal. The first polynomial term does not show at $\pm N(\omega_2 - \omega_1)$ from the carrier, but at $\pm(\omega_2 - \omega_1)$ from the carrier!

Since this is a common error, let's use some numbers to demonstrate the effect. If the original signal (say, at 120 MHz) has a spurious signal at $\pm$1-kHz offset, after multiplication by 2, the spurious signal will show (at 240 MHz) again at an offset of $\pm$1 kHz. Other spurious signals will start to show according to the equation since higher terms might have enough energy. If the

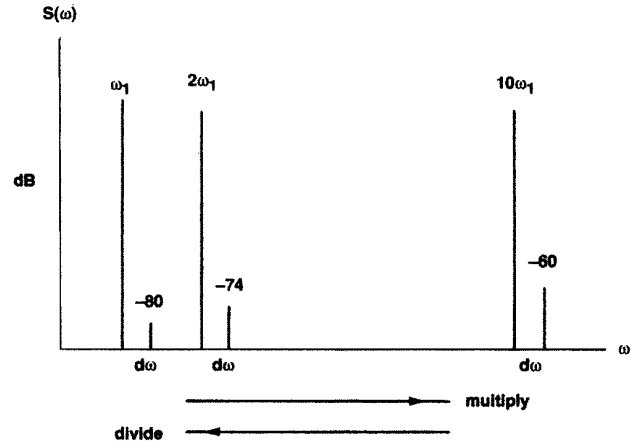


Figure 2-1 Spurious signals in decibels after multiplying or dividing. The distance from the carrier does not change.

product of N^2 and er^2 is sufficient (and this is a relative number), this additional spurious signal will show at $\pm 2(\omega_2 - \omega_1)$ from the carrier, and so on.

The reciprocal effect occurs upon division. If in the multiplier Eq. (2-2) we put $1/N$ instead of N, then the spur of the output is improved by $20 \log N$ dB. This is used in the FS again and again to improve performance. See Fig. 2-1.

For example, if the design in Fig. 1-10 were repeated 6 times, so that the range of 16. to 16.999999 MHz can be achieved, and if the requirement were for a spurious response of -70 dBC (this means that any spurious signal must be at least 70 dB below the carrier), then clearly the (similar and common) stages need to have a spurious response of only -50 dBC before the division by 10 since the divider's operation improves the output spurious signal by

$$20 \log 10 = 20 \text{ dB} \tag{2-3}$$

High-quality synthesizers employ multiple loops and dividers as a means to reduce spurious (and phase noise) signal levels. This can be achieved because the inside loops can be relatively poor in performance but will be improved by division by a high number (see Chap. 5).

2-2 Phase Noise

The phase noise of a synthesizer is a major parameter and an excellent barometer of its quality. Phase noise theory will be summarized briefly here.

If a signal had an ideal spectrum, all its energy would be concentrated in a single frequency. Such a signal does not exist, and all signals have additional noise. This noise causes the signal spectra to have a distribution, and it can be expressed as frequency or phase jitter.

There are actually two types of phase noise, long-term and short-term. The long-term phase noise has to do with the frequency reference stability, is a slowly varying process, and depends on the quality of the resonator (usually a quartz crystal), aging effects, temperature, mechanical integrity, pressure, gravity, and power supply stability. The long-term stability is usually expressed in parts per million (ppm) and is usually specified per day, over a temperature range, and per year. Long-term stability for atomic standards can achieve numbers like 1 part in 10^{13} per year in Cesium standards (see Chap. 9).

The more interesting and more complex type is the short-term stability of the signal spectrum. Some of this noise is deterministic and is caused by the power supply line (50, 60, or 400 Hz depending on the application) or vibration in the case when they are predictable. However, some of the noise is random, stochastic by nature.

A real-world signal, assuming fixed amplitude A, is given by

$$A[1 + n_1(t)]\sin[\omega_t + n_2(t)] \tag{2-4}$$

where $n_1(t)$ and $n_2(t)$ represent the fluctuations of the amplitude and phase, respectively.

The most common way to present the noise spectra graphically is in terms of $L(f_m)$, which is defined as the single-sideband (SSB) noise power and is given by the power of the signal in 1-Hz bandwidth at f_m Hz away from the carrier, divided by the total signal power. A typical spectrum is shown in Fig. 1-2.

The issue of noise in oscillators has been a theoretical research topic for many years. There are two interesting aspects of this subject. The first has to do with the mechanism of the start and

buildup of oscillations from the time of power-on of the oscillator. The second is related to the noise and its sources that are generated in the oscillator when it reaches steady state. The questions of oscillation buildup can now be modeled well and quite accurately, by using a variety of SPICE CAD software. This mechanism is beyond the scope of this book and will not be dealt with here.

The analysis of noise in oscillators is instrumental to the understanding of phase noise measurement techniques. We will therefore briefly present the basic theory of FM approximation to the phase noise calculation, and we will derive the terms and create the understanding of phase noise mechanisms and their application to measurement techniques.

Viewing Eq. (2-4), we see clearly that both AM noise and PM noise can be translated to phase noise. To gain an insight into the phenomenon, we review briefly basic modulation theory, as the noise sources modulate the "ideal" signal as shown in Eq. (2-4). We assume that the signal being investigated is synthesized and therefore possesses a "decent" level of phase noise, or $|n| << 1$. We will see soon that if this assumption does not hold (as in the case of either free-running oscillators or synthesizers with very large division ratios in the loop), the approximations can generate large errors.

A carrier being amplitude-modulated by a sine wave is given by

$$V(t) = V \sin \omega_0 t(1 + m \sin \omega_m t) \tag{2-5}$$

where ω_0 = carrier angular frequency
V = signal peak voltage
m = modulation index
ω_m = modulating frequency

Therefore, this can be rewritten as

$$V(t) = V \sin \omega_0 t + V_m [\cos(\omega_0 + \omega_m)t - \cos(\omega_0 - \omega_m)t] \tag{2-6}$$

The spectrum consists of three lines—the carrier and two sidebands, upper and lower.

For the case of *phase modulation (PM)*,

$$V(t) = \sin(\omega_0 t + m \sin \omega_m t) \tag{2-7}$$

Now, remember that

$$\cos(x \sin y) = J_0(x) + 2 \sum_{k=2i} J_k(x) \cos ky \qquad i = 1, 2, \ldots \tag{2-8}$$

$$\sin(x \sin y) = 2 \sum_{k=2i-1} J_k(x) \sin ky \qquad i = 1, 2, \ldots \tag{2-9}$$

Here J_0, J_1, J_2,... are Bessel functions of the first kind, with orders 0, 1, 2, …; see Fig. 2-2.

Now, since we assumed $m << 1$, and since

$$J_n(x) = \left(\frac{x}{2}\right)^n \left[\frac{1}{n!} - \left(\frac{x}{2}\right)^2 \frac{1}{1!(n+1)!} + \left(\frac{x}{2}\right)^4 \frac{1}{2!(n+2)!} - \cdots\right] \tag{2-10}$$

then clearly,

$$J_0(m) = 1 \tag{2-11}$$

$$J_1(m) = \frac{m}{2} \tag{2-12}$$

$$J_n(m) = 0 \qquad \text{for } n \geq 2 \tag{2-13}$$

Therefore, we can approximate Eq. (2-7) as

$$V(t) = \sin \omega_0 t + \frac{m}{2} [\sin(\omega_0 + \omega_m)t - \sin(\omega_0 - \omega_m)t] \tag{2-14}$$

Note that both AM and PM can be converted to phase noise and their relative power spectrum is identical (for the same index of modulation); however, the relative phases of the sidebands are different.

From Eq. (2-5) we can easily calculate the value of the phase causing modulation, our model for jitter. The jitter caused by $m\sin(\omega_m t)$ has the rms value of:

$$E\left(m^2 \cdot \sin^2 \omega_m t\right) = \frac{m^2}{2}$$

Note that Eq. (2-13) shows that in the case where $m<<1$, the total power of the sidebands relative to the carrier is $2 \cdot m^2/4 = m^2/2$. We can therefore conclude that for $m<<1$ (we

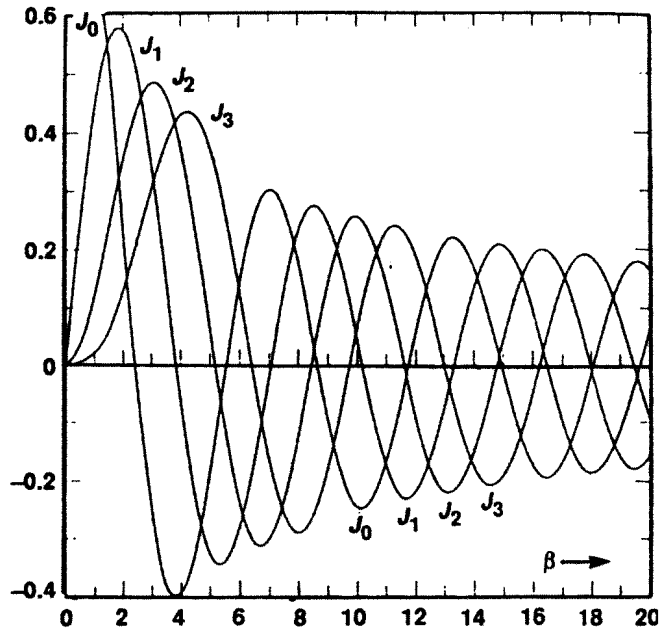


Figure 2-2 Bessel functions.

always assume that the noise is low, and in synthesizers this is always a relevant assumption), phase jitter[2] in radians is equal to the total sidebands energy (relative to the signal's total power)!

Therefore, for φ_j given in radians,

$$\varphi_j^{\,2} = \sum P_i \tag{2-15}$$

where P_i is the discrete sidebands power. Without any loss of generalization, we can infer that in the case of random noise, where the phase is not just a single sinewave (modulating the signal phase), but rather distributed noise, the same is correct, and we can replace the discrete representation with a continuous one.

Let $L(f_m)$ be the *single sideband* phase noise of the signal. Then the phase jitter φ_j, is given by:

$$\varphi_j^{\,2} = 2\int L(f_m)df_m \qquad \text{radians}^2 \tag{2-16}$$

FM noise is given by:

$$\delta F^2 = 2\int L(f_m)\cdot f_m^{\,2}df_m \qquad \text{Hz}^2\ \text{rms} \tag{2-17}$$

We multiply by two because $L(f_m)$ has been defined as single sideband and the noise is symmetrical around the carrier.

This equation also brings to light the concept of the signal being narrow-band noise. The integral of Eq. (2-16) from $-\infty$ to $+\infty$ is the relative total power of the signal! In a way, $L(f_m)$ can be considered the probability distribution function of the signal's spectrum. Therefore, when performing Eqs. (2-16) or (2-17), we must define the frequency borders we wish to cover. At very low offsets, $L(f_m)$ represents long-term stability, drift caused by aging, temperature changes and very low rate changes. As we move away from the carrier, $L(f_m)$ represents the faster changes in the phase—what we usually refer to as phase noise.

Sometimes we have no interest in the noise below, say, 20 Hz; in many audio systems this noise cannot be heard. In other applications, specially Satcom terminals where the receiving modem can track phase perturbation up to 100 Hz, phase noise is defined only above 100 Hz. Therefore, every application will define its own specification of the frequency range for this integral.

From phase noise, we can easily calculate time jitter, given by:

$$T_j = \frac{\varphi_j}{\omega_o} = \frac{\varphi_j}{2\pi f_o} \qquad \text{where } \omega_o \text{ is the center frequency} \qquad (2\text{-}18)$$

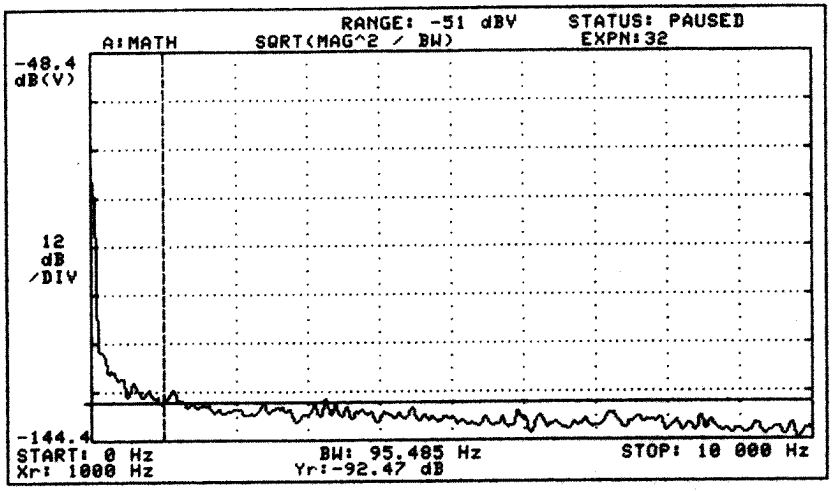
Note that $L(f_m)$ is given in dBC/Hz, the relative density of the total carrier power, and has dimensions of 1/Hz. See the PHAZNOIZ program on the disk for CAD conversion. In degrees,

$$d\phi = \varphi_j = \frac{180}{\pi} 2\int L(f_m) df_m \qquad \text{degrees} \qquad (2\text{-}19)$$

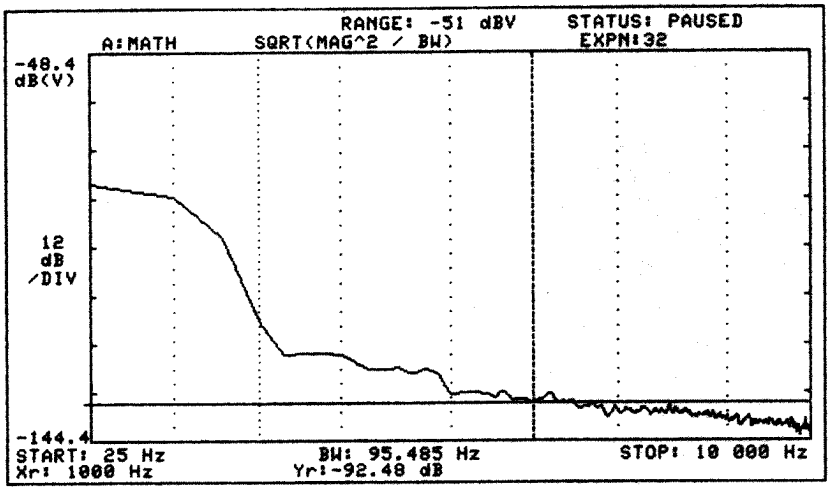
Remember that $L(f_m)$ describes the signal's total spectral distribution and when integrating from $-\infty$ to ∞, we integrate the signal's relative total power, thus:

$$2\int L(f_m) df_m = 1 \qquad (2\text{-}20)$$

A single spurious signal that is –60 dB below the carrier will therefore cause a phase jitter given by –30 dB radian or .001 radian rms. Similar jitter will be caused by a noise spectrum that is flat from 6 kHz to 11 kHz with a level of $L(f_m)$= –100 dBC/Hz; this noise will have a total power of $2 \cdot 5000 \cdot 10^{-10} = 10^{-6}$, which is –60 dB. Since the total power is the same, the overall jitter will be the same.



(a)



(b)

Figure 2-3 Phase noise with linear and log offset frequency.

For example, in the graph shown in Fig. 2-3*a,* the noise contribution from 10 Hz to 1-kHz offset from the carrier is given by

$$d\phi_{\text{rms}} \approx \frac{180}{\pi}\sqrt{2 \cdot (1000 - 10)10^{-9}} = 0.08^{\circ}\,\text{rms} \qquad (2\text{-}21)$$

Sometimes this measurement is taken directly from a spectrum analyzer reading; then few corrections have to be taken into account. One correction is that the analyzer makes the measurement with a resolution bandwidth that is different from 1 Hz (e.g., a resolution bandwidth of 100 Hz is 20 dB Hz). Therefore the measurement should be taken (in the plot, at 1 kHz from the carrier) as

$$L(f_m) = -(\text{peak} - P_{\text{measure}} + \text{res})$$
$$= -(75 + 20) = -95 \text{ dBC/Hz} \qquad (2\text{-}22)$$

where res is resolution. Other correction factors arise from the fact that the filter noise bandwidth is not exactly the resolution number (i.e., a 100-Hz resolution bandwidth does not indicate exactly 100-Hz noise bandwidth), and the manufacturer provides the correction factor. Also, the detector in use (in the analyzer) behaves differently toward a CW signal and toward noise, and a correction factor is provided, too, by most manufacturers.

Many spectrum analyzers perform the whole calculation and include the correction factors, to provide a measurement in decibels referred to the carrier per hertz (dBC/Hz). Another way of getting such accurate measurements is to use a *fast Fourier transform* (*FFT*) analyzer. Thus the signal is mixed with a reference that is known to be better. First, their frequencies are offset by, say, 1 kHz, and this allows the analyzer to measure the signal power. Next the frequencies are matched, and the signals are brought to quadrature phase. Thus the main signal nulls (cancels) and the noise alone is measured. This allows great dynamic range measurement, because of the nulling effect. Most FFT analyzers have a mathematics function that calculates the ratio of the signal to the noise in 1-Hz bandwidth.

This is usually a lengthy and complex measurement. When automated, it is also very expensive. See Fig. 2-3*a* and *b* for an L-band synthesizer, linear and log frequency of the same oscillator. A program to convert $L(f_m)$ to degrees rms is on the accompanying CD-ROM (see Chap. 10).

Before we mention the mechanisms that contribute to oscillator phase noise, we discuss noise mechanisms in the PLL circuit.

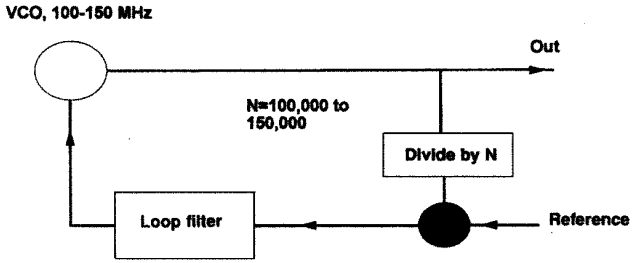


Figure 2-4 Single-loop PLL block diagram.

2-3 Spurious and Phase Noise in PLL

Note that the effect of multiplying or dividing on the phase noise performance is similar to that of the spurious noise analysis. So multiplying by N yields a degradation of 20 log N, and division by N improves performance by 20 log N.

This immediately demonstrates one of the weaknesses of single-loop PLL designs. A simple example is shown in Fig. 2-4. The phase lock basically generates an output frequency NF_r, where N is the PLL divider and F_r is the reference frequency. A PLL is therefore in effect a multiplier, and it suffers the 20 log N degradations.

Even if the reference 1-kHz signal is extremely clean and the phase detector has very low noise, or spurious signals, say, at −150 dBC/Hz, then the PLL acts as a multiplier of 100,000 to 150,000 or 100 to 104 dB, so the output (at least close to the carrier, where the loop is locked) will be limited to $-150 - (-104) = -46$ dBC/Hz, which is quite poor and will not be good enough for many applications. Therefore, as a rule and specifically in PLL circuits, the goal is to reduce the multiplication (i.e., the division ratio in the loop) effects. This translates also to a general requirement to use the highest possible reference frequency.

When PLL is used with multiple loops, this effect is attenuated by dividing the signals, as mentioned above. The result is an improvement in phase noise and better step size, at the cost of added complexity.

However, it is important to remember that the PLL acts as a bandpass filter (BPF), and a typical end effect of the behavior of a PLL circuit is shown in Fig. 2-5*a* and *b*. Beyond the loop bandwidth, the loop filter "cleans" and attenuates both noise and spurious sig-

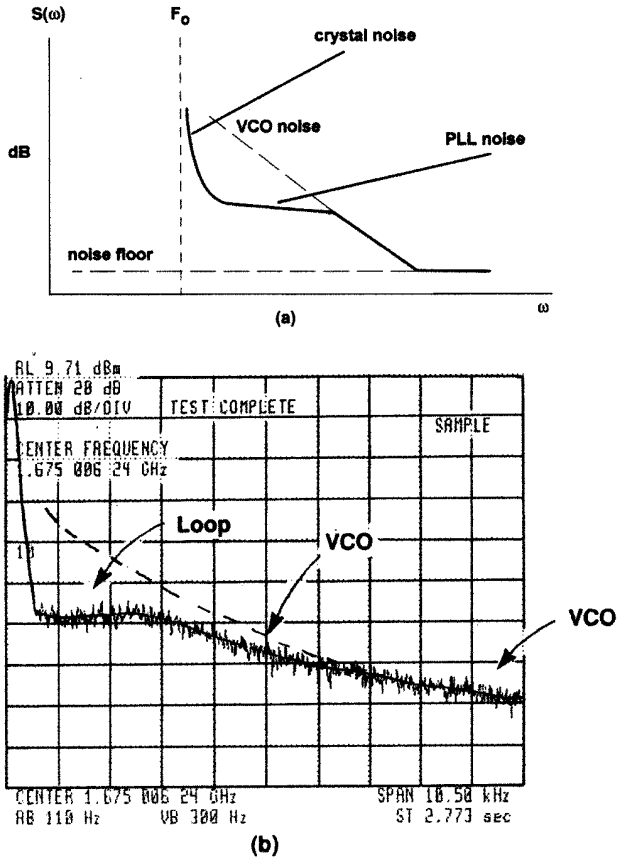


Figure 2-5 (*a*) PLL phase noise contributors; (*b*) phase noise components of PLL synthesizers.

nals. Therefore, outside the loop bandwidth the PLL will perform according to the noise of the VCO. This effect is used extensively in PLL designs, hence the importance of a good phase noise VCO. The same applies somewhat to direct analog designs since a BPF at the output cleans it from both spurious and phase noise.

From time to time, some manufacturers show an output frequency spectrum that is better than the input reference +20 log *N*. If the term used is *residual phase noise,* it means the added noise beyond 20 log *N*. When the term is *absolute phase noise* and the reference signals are high-quality crystal oscillators, it is a complicated and costly task, and it is wise to ask for an explanation. There are two possible explanations:

1. The reference has better performance than advertised. Many responsible manufacturers have margins in their specifications.
2. The reference has been improved and cleaned.

However, it is not possible to improve on $20 \log N$ by just multiplying or using a phase lock.

There are basically two methods to improve the reference signal phase noise beyond $20 \log N$. One is by locking onto the frequency of another oscillator whose noise performance is better than the multiplied reference. The improvement will be achieved outside the loop bandwidth of this locking mechanism. Another way is by applying narrow-band filters. This is a technique used especially for very high-quality synthesizers (see Sec. 9-1-2).

Now, we return to noise mechanisms in oscillators.

2-4 Phase Noise Mechanism

Noise is generated in all parts and circuits, and frequency synthesizers are no exception. Usually the amplitude noise is very low and in many cases is not even mentioned or specified. However, phase or frequency noise is an important parameter, and we will briefly discuss its sources and various models developed to calculate its effects.

We will follow the Hewlett-Packard models of Ref. 7; see Sec. 2-4-2.

2-4-1 Noise in dividers

The plots of a variety of dividers, mainly TTL and ECL, are shown in Fig. 2-6. The low-frequency noise is generated by the flicker noise of the devices; otherwise, the noise depends on the device's noise figure. GaAs devices generate higher noise floors than silicon devices.

2-4-2 Noise in oscillators

According to the equations developed in the reference, the phase noise is given approximately by

$$L(f_m) = 0.5 \left(\frac{\omega_o}{\omega_m \cdot 2Q_L} \right)^2 \cdot \frac{kTF}{P_{sav}} \left(1 + \frac{f_c}{f_m} \right) \tag{2-23}$$

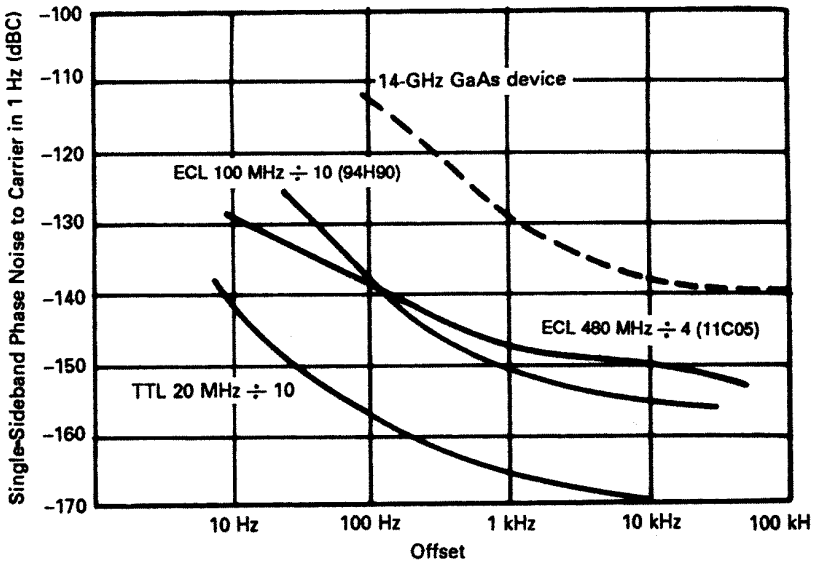


Figure 2-6 Phase noise in dividers. Solid lines, Si; dashed line, GaAs.

The oscillator phase noise depends on the device's noise figure F and the power in the active device P_{sav}, but mainly on the loaded Q of the resonator, Q_{load}, and the oscillator constant K_v, as the noise in the analog phase detector circuitry (in a PLL circuit) modulates the oscillator. This has driven many designs to reduce this sensitivity by switching resonance elements to the VCO tank circuits so that the VCO K_v is kept to the lower possible value. See Fig. 2-7*a, b,* and *c.* Some typical oscillators' performances are shown in Fig. 2-8.

2-4-3 Noise in phase detectors

Active phase detectors, using the circuits indicated in Chap. 5, generate phase noise performance of −140 to −160 dBC/Hz. We are not familiar with phase frequency detectors with substantially better noise characteristics. Such devices use sampling techniques to reduce the noise from the noise sources while sampling the phase detector output at the right time to extract the phase information. But passive phase detectors, usually balanced mixers, have excellent noise performance near −165 dBC/Hz.

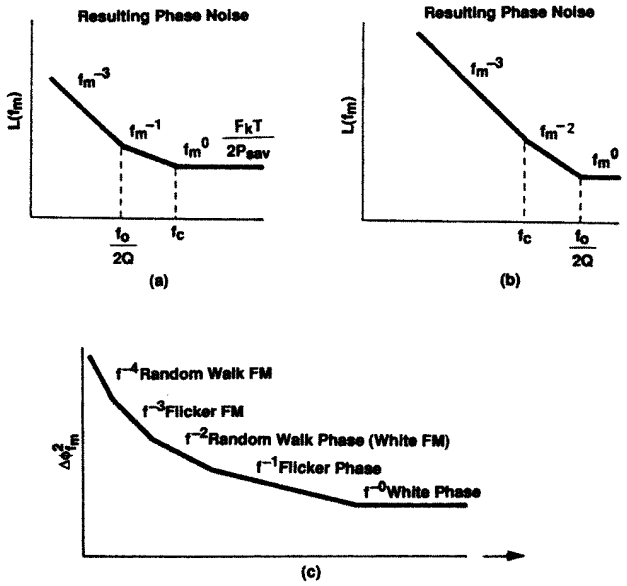


Figure 2-7 Phase noise model and sources (courtesy HP).

2-5 Mixing and Filtering

Mix-and-filter is a common functional block in FS design. Mixing two signals has the effect of adding and subtracting their relative frequencies, and the filtering passes the desired products while rejecting undesired ones.

An ideal mixer is a multiplier whose function is given mathematically by

$$\sin \omega_1 t \sin \omega_2 t = 0.5[\cos(\omega_1 - \omega_2)t - \cos(\omega_1 + \omega_2)t] \quad (2\text{-}24)$$

By multiplying the signals, we generated what are called an *upper* and *lower sidebands,* $\omega_1 + \omega_2$ and $\omega_1 - \omega_2$, respectively.

Note that a mixer is not an ideal multiplier and exhibits second, third, and higher effects.

A typical mixer response is shown in Fig. 2-9.

Any nonlinear device can be used to perform the mixing operation, and a large variety of diode bridge mixers (the classical double-balanced mixer) as well as active devices using transistors or *field-effect transistors* (*FETs*) in their nonlinear regions perform this function, e.g., the Signetics part number NE602 and such.

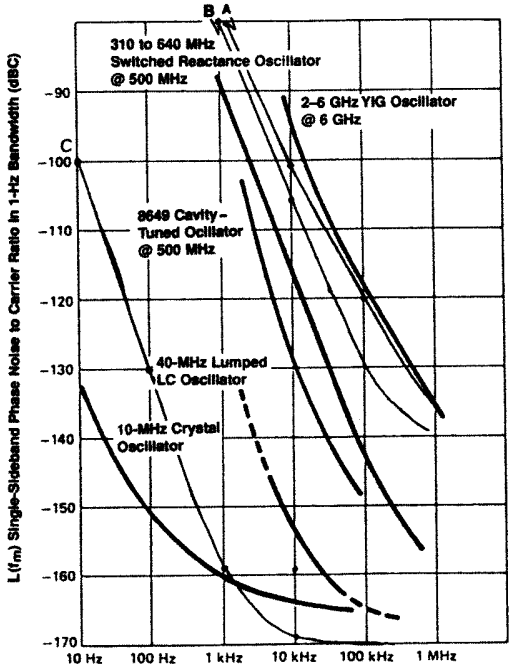


Figure 2-8 Free-running oscillators' phase noise. *A* is a negative-resistance, microstrip resonator; *B* is the same with dielectric resonator; and *C* is a 100-MHz crystal.

RF order harmonic	RF cal	0	1	2	3	4	5	6	7	8	9	10
0	>94	-	36	51	42	61	42	60	58	62	54	72
1	>94	23	-	39	12	46	22	51	35	56	48	72
2	74	68	54	71	55	73	55	74	55	64	50	77
3	81	47	41	49	40	53	42	55	42	56	44	64
4	>94	73	75	76	71	77	67	76	66	76	62	75
5	>94	70	66	67	52	71	49	66	49	71	52	65
6	>94	84	77	>86	>86	>86	81	>86	82	84	82	83
7	>94	83	65	79	65	78	75	76	60	77	55	73
8	>94	86	85	>88	81	>86	>86	>86	>86	>86	84	>86
9	93	87	84	86	80	82	73	85	75	86	84	78
10	>93	>87	>87	86	>87	>88	>88	>86	>86	>86	>86	>86

Harmonic LO order

Figure 2-9 Typical mixer intermodulation products.

It is beyond the scope of this text to go into details of spurious signal control in mix and filter circuits. Some solid analysis is available in Refs. 1, 6, and 11.

2-6 Frequency Planning

This is a cardinal parameter in frequency synthesis. When a single-loop design is employed, there is no frequency planning. However, for more complicated designs, where more than one loop is used, and when there are mixing operations, careful frequency planning is very important. The calculation of mixing products, the assignment of proper signal levels to the mixers, and the position of the spurious signals and the ability to filter them out are the real challenges in a design. As a rule of thumb, the higher the ratio of the mixing frequency to the product, the easier it is to filter and control. Many software packages are available for spurious signal analysis, since the mathematics is that of a simple spreadsheet. The operations are NF1-MF2 and check if the product falls in the range. For example, mix 30 to 50 MHz with 30 MHz to generate 0 to 20 MHz. Following the equation NF1-MF2 shows the following results:

M		0	1	2	3	4	5	6	7	8	9
N	0	0	0	0	0	0	0	0	0	0	0
	1	0	−	0	0	0	0	0	0	0	0
	2	0	x	x	0	0	0	0	0	0	0
	3	0	0	x	x	0	0	0	0	0	0
	4	0	0	x	x	x	0	0	0	0	0
	5	0	0	0	x	x	x	0	0	0	0
	etc.										

(Here − is the main output, 0 is no spur, and x is a potential spur.) This is a tough frequency plan to execute (HP 3325A/B). Mixing 120 MHz with 100 MHz would be much easier, and the number of spurious signals would be much lower. Remember that spurious signals generated from high M and N are low in level (see Fig. 2-9).

Grounding and shielding, mechanical construction, and power supply filtering are all very important, but are beyond the scope of this text.

The design of a synthesizer requires many considerations and much experience. The increased requirement for smaller size, lower power, and better performance is challenging and requires a high level of circuit integration. There are applications that can use the "cookbook" approach, but these are more demanding and require expertise and careful design.

References

1. Vadim Manassewitsch, *Frequency Synthesizers: Theory and design,*Wiley, New York, 1978.
2. Hewlett-Packard, HP3325A/B manual.
3. Hewlett-Packard, HP8662A manual.
4. W. P. Robins, *Phase Noise in Signal Sources,* Peter Peregrinus, London, 1982.
5. D. Scherer et al., "Low Noise RF Signal Generator Ddesign," *HP Journal,* February 1981.
6. Ulrich Rhode, *Digital Frequency Synthesizers,* Prentice-Hall, Englewood Cliffs, NJ, 1981.
7. Dieter Scherer, Hewlett-Packard, "Design Principles and Test Methods for Low Phase Noise RF and Microwave Sources," *Hewlett-Packard Journal,* October 1978.
8. Hewlett-Packard, phase noise measurement techniques, application note, May 1982.
9. B. G. Goldberg, Device for fixing the phase of frequency synthesizer output, U.S. patent 4868510, September 1989.
10. F. L. Walls et al., *Accuracy Model for Phase Noise Measurements,*National Institute of Standards, Boulder, CO.
11. William F. Egan, *Frequency Synthesis by Phase Lock,*Robert E. Krieger Publishing, Malabar, FL, 1990.

Chapter

3

Measurement Techniques

Frequency synthesizers become more complex as the requirements for performance and size become more demanding. Parallel to the development of synthesizer technologies, there are growing demands to test their parameters. In most manufacturing environments, these tests have to be computerized and automated.

Some of the parameters that need to be measured for compliance with specific requirements are very standard and simple to measure. These include parameters such as power consumption, output power, output power flatness, voltage standing wave ratio (VSWR), and such.

However, a variety of parameters are unique to signal sources and especially to frequency synthesizers, and the methods of their measurement are sometimes not trivial.

In this chapter we illustrate a variety of techniques used to measure such parameters as phase noise, switching speed, phase transients, and others that are unique to the synthesizers. We use the notation common in the industry to define these parameters.

3-1 Switching Speed

In a frequency synthesizer that generates more than one frequency, the *switching speed* is defined as the maximum time required to switch and settle from any frequency to any frequency within its operating range. Since in most cases the speed depends on the relative distance between the frequencies, some manufac-

turers specify speed per given frequency excursion, for example, 5 μs for a 1-MHz step, 50 μs for a 10-MHz step, 400 μs for a 100-MHz step, and so on. In other applications, the requirement is for a specific speed for any step or frequency excursion.

There are mainly two definitions of the term *switch and settle,* used in the majority of synthesizers on the market. The first one requires that the synthesizer reach a new frequency that is at a defined frequency deviation tolerance from the target, e.g., to be within 5 kHz of the required frequency.

Suppose that we hop a synthesizer between 120 and 130 MHz and toggle between these two frequencies periodically. Suppose also that the requirement for settling speed is to be within ± 5 kHz within 100 μs. A setup for this measurement is shown in Fig. 3-1. The hop command is controlling the synthesizer and is also delayed 100 μs and gates the pulse counter for 5 μs. Since the measurement is made 100 μs after the hop command, it is required that the counter measurement be 120 ±0.005 MHz or 130 ±0.005 MHz depending on the one we measure to. This is a simple test configuration, but it requires a pulse counter.

The second definition of switching speed is for the new frequency to settle to within 0.1 rad (5.7°) of the phase. This measurement is usually performed as shown in Fig. 3-2. The output of the switched synthesizer is compared (in phase—this is usually done by using a mixer) to a continuously running second synthesizer which runs at the frequency to which we measure the speed. Since the unit under test and the reference run on the same reference, the output from the mixer will suffer some transient (which is the switching transient) and then settle to a constant

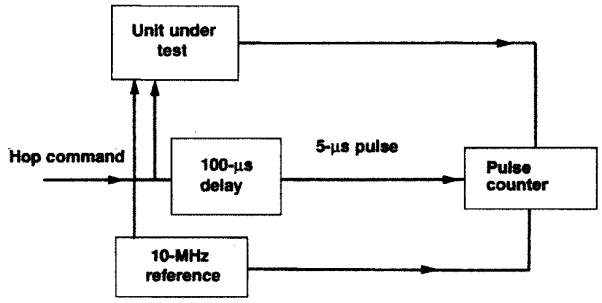


Figure 3-1 Switching time test using a counter.

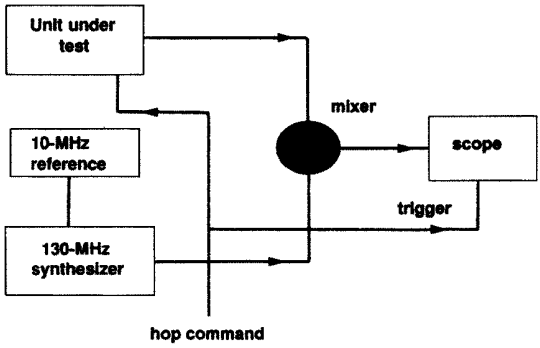


Figure 3-2 Switching time setup for phase settling time.

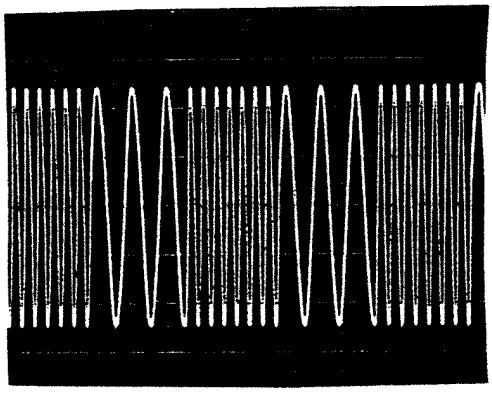


Figure 3-3 Transient phase switching.

phase, which is the phase difference between the two signals. The requirement is to settle to within 0.1 rad of the phase, as shown in Fig. 3-3. (See also Fig. 3-5.)

Since the mixer output is given by $\sin \varphi$, the measurement should be to phase 0°, given that the sine function is quite flat when it approaches 0°, as demonstrated in Fig. 3-4. This basically requires that the phase between the reference and the unit under test be 90°; otherwise, a false measurement is read.

In the case of direct digital synthesis, this measurement can be greatly simplified since the direct digital synthesizer can be switched from dc to F_1, and one can measure the speed necessary to settle to direct current without the need to mix with another

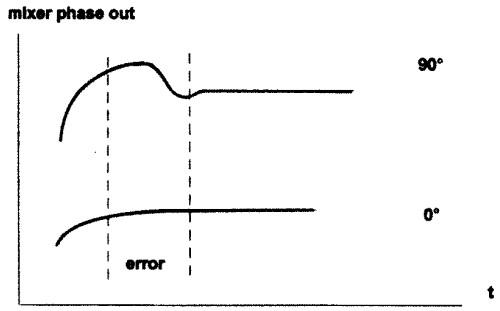


Figure 3-4 Phase transient measurement.

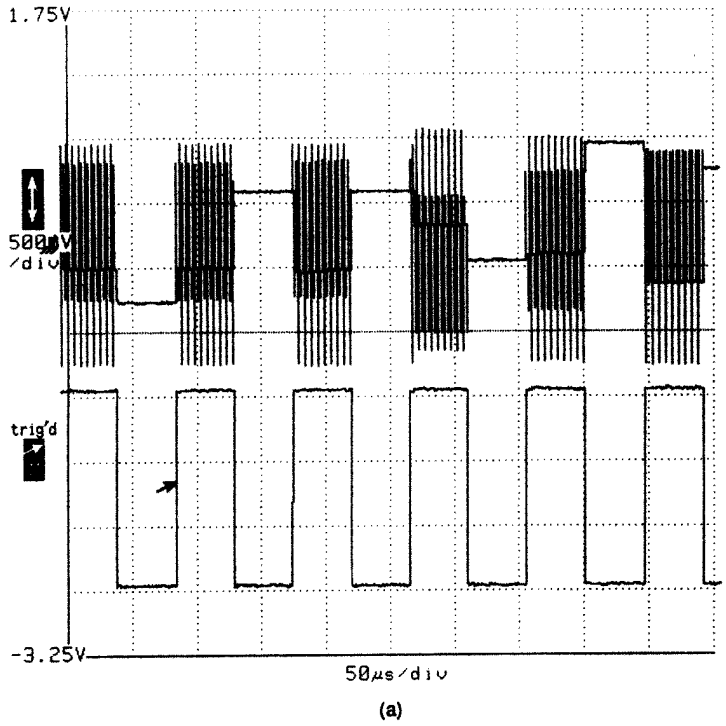


Figure 3-5 Phase settling to (*a*) random phases and (*b*) a known phase. (*c*) No transient switching. (*d*) No transient switching, MSK modulation.

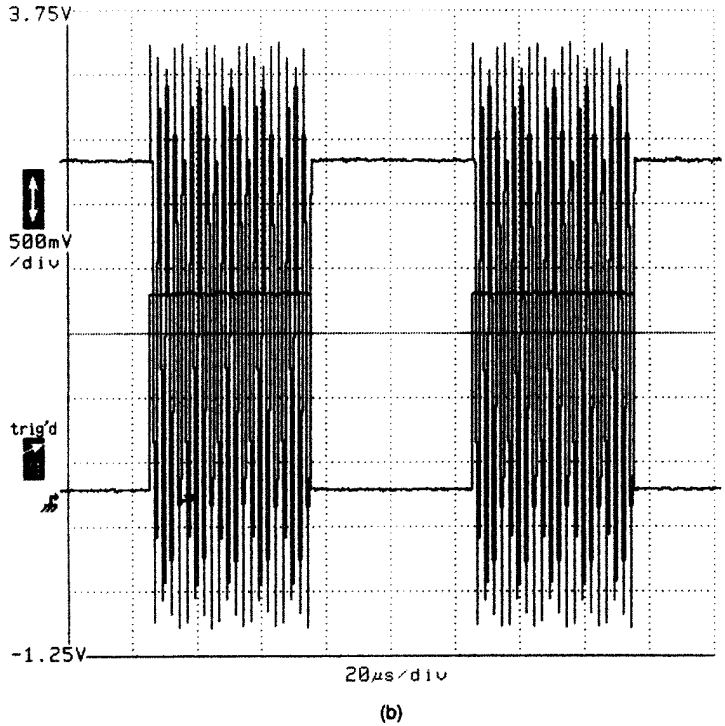


Figure 3-5 (*Continued*)

synthesizer. A plot of such measurements is shown in Fig. 3-5. Note that since the hop control is not necessarily synchronized with the reference, the direct digital synthesizer settles to different dc levels, depending on its phase. A synchronized hop is shown in Fig. 3-5*b* and a nonsynchronized one in Fig. 3-5*a*.

3-2 Phase Noise

Phase noise is measured in a variety of ways, although the three most common are integrated phase noise, FM noise, and phase noise density, which is the most detailed of the three. The first two are techniques that measure integrated noise and therefore do not provide information about the noise spectra. The third method describes the spectrum of the noise. It is therefore possible to derive the first two measurements from the third, but not vice

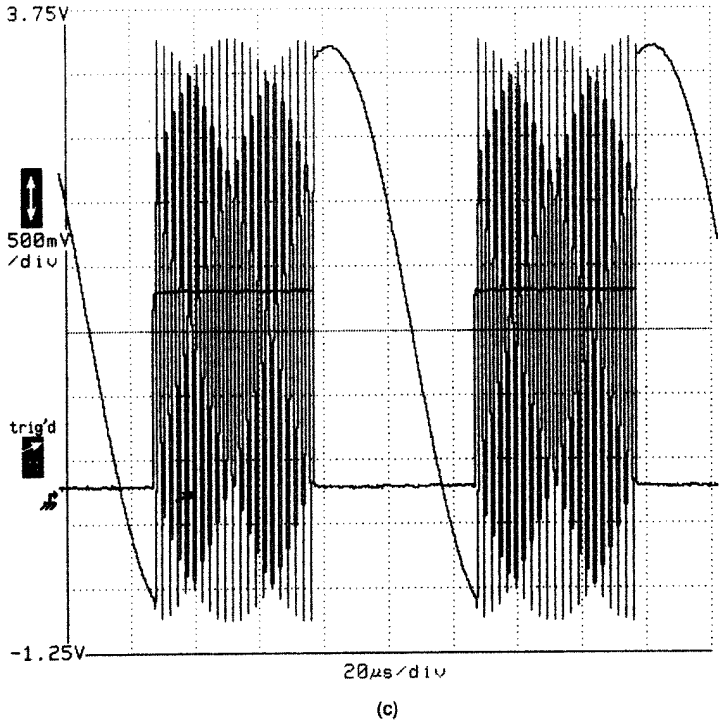


Figure 3-5 (*Continued*)

versa. [A program to convert from the noise spectrum to degrees of noise rms or time jitter is included on a disk the reader may obtain from the author (see Chap. 10).]

3-2-1 FM noise

This parameter is relatively simple to measure and is used mainly by instrument makers or other applications where the exact noise spectrum need not be described. There are also many applications, especially for FM systems (and FM is still a very popular modulation in applications like mobile communications, cellular telephony, and telemetry), where this parameter is the one that is important to the designer.

The synthesizer is connected to a calibrated discriminator, and the root-mean-square (rms) voltage at the output of the discriminator is measured by a true rms voltmeter. Since the discriminator is calibrated and its constant given by *Dd* is in hertz per volt,

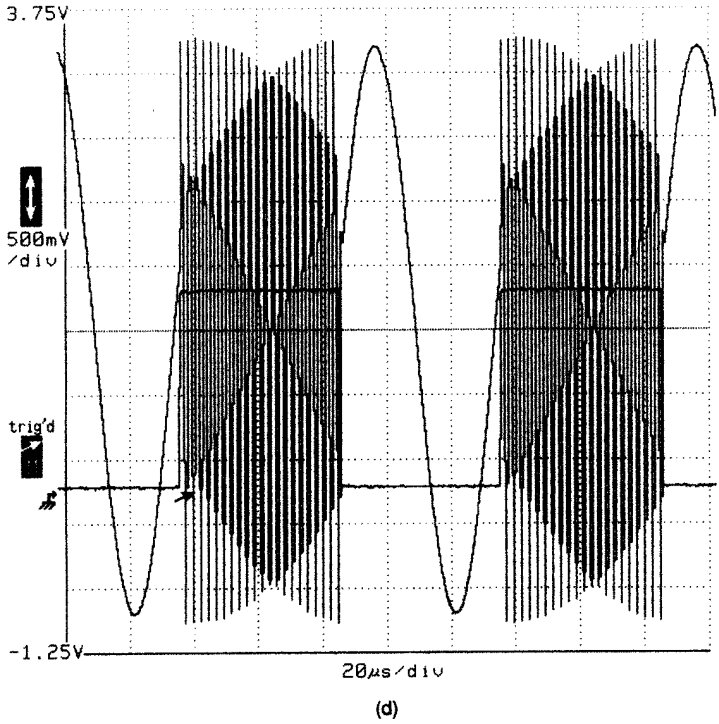


Figure 3-5 (*Continued*)

the conversion from rms volts to rms hertz is direct, as shown in Fig. 3-6. The result is given then by rms hertz or hertz and is common in standard laboratory instruments. For a typical instrument such as Fluke model 6060B or HP 8656B, the FM noise is approximately 5 to 10 Hz rms.

In order to get a measurement, a high-sensitivity discriminator is required (so that a measurement of a few hertz rms can be taken). If, e.g., a discriminator of $K_{\text{dis}} = 1$ V/MHz is used, the measurement will be almost impossible because the rms voltage will be on the order of microvolts. This leads to the use of either crystal discriminators or cavity discriminators, both using very high-Q devices for high sensitivity but narrow bandwidth.

3-2-2 Delay line discriminator

One application of a discriminator uses a delay line and a mixer, as shown in Fig. 3-7. One path to the input of the discriminator is

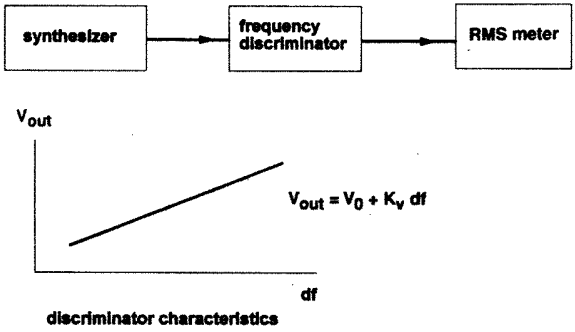


Figure 3-6 FM noise measurement using a discriminator.

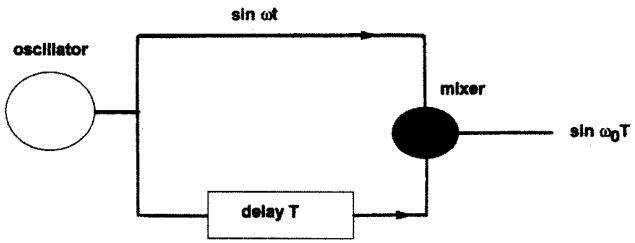


Figure 3-7 Delay line discriminator.

given by the input signal sin $\omega_0 t$. The other input is delayed by T and is given by sin $\omega_0(t + T)$. Since the mixer performs the operation of subtracting the phase of the inputs, its output is given by $\omega_0 T$. If the signal is clean, this will yield some dc output. However, some noise fluctuations of the signal cause some frequency fluctuations, dF, which translate to $d\varphi = 2\pi T\, dF$.

The voltage output depends on the mixed conversion constant K_m, where $V_{\text{rms}} = K_m\, d\varphi_{\text{rms}}$. (Note that we are assuming that the phase between the two inputs is *approximately* 90° where its sensitivity is maximum and that the noise is low so that the conversion constant can be assumed to be a constant.)

We can now rewrite the equation as $dV_{\text{rms}} = 2\pi T\, K_m\, dF_{\text{rms}} = K_1\, dF_{\text{rms}}$. It is obvious that the sensitivity of the measurement depends on T. As T approaches zero, the sensitivity goes down to zero. The noise spectrum is then given by $S\, dF(f) = dF_{\text{rms}}(f)^2/W$ where W is the noise bandwidth and therefore is given by

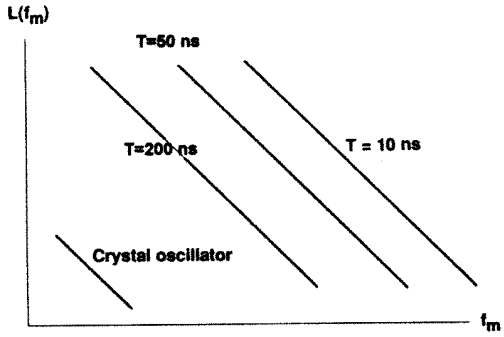


Figure 3-8 Delay line sensitivity and limitations. For $T = 500$ μs, floor at 1-kHz offset is −100 dBC/Hz.

$dV_{\text{rms}}(f)^2/K_1^2$ per hertz. Typical sensitivities of different delay line discriminators are shown in Fig. 3-8.

As can be shown, for very clean and stable sources, this method is not very practical since either it does not have the sensitivity required or a very large delay line is required. However, the equipment necessary is cheap and available in every RF lab. The delay line can be a long coaxial cable.

However, for moderate performance synthesizers (used in wireless applications) and specifically for free running VCOs, this is an ideal technique. Few test instruments are available in the market (*see RDL*), that are effective, fast and economical. Their limits can be calculated from the exact noise equation:

$$dV_{rms} = K_\phi T \sin(x)/x$$

The phase noise limit of such systems, using T ≈ 500 ns, is shown in the following table.

Offset (Hz)	$L(f_m)$ -dBC/Hz
1	---
10	46
100	75
1k	105
10k	
100k	
1M	
10M	

With phase noise density using two sources, comparison of phase and FFT can achieve the results shown in the following table.

Offset (Hz)	$L(f_m)$ -dBC/Hz
1	100
10	135
100	145
1k	160
10k	170
100k	170
1M	172
10M	172

3-2-3 Integrated phase noise

This method measures the integrated phase noise over a specific bandwidth, excluding the carrier. The standard numbers used when this technique is applied are usually 15-kHz noise bandwidth excluding 1 Hz from the carrier. The measurement is performed as follows.

The signal is mixed with a clean reference, and its frequency is offset, by, say, 1 kHz. The output power is then measured. This calibrates the phase detector constant and allows one to measure the signal power. Then the two signals are brought to the same frequency, but in quadrature phase (90°). The mixer output passes through a high-pass filter of 1 Hz and a low-pass filter of 15 kHz. The ratio of the signal to the noise is the measured performance.

3-2-4 Noise density

This phase noise measurement is the most comprehensive, and it calculates the phase noise spectrum. See Chap. 2. The relation between $L(f_m)$ and the phase or frequency noise has been shown in Eqs. (2-19) and (2-20).

3-3 Phase Continuity

Many applications require that the phase transition be smooth during switching from frequency to frequency. This is possible to do in a direct digital synthesizer or fractional-*N* PLL synthesizers

(which basically employ a direct digital synthesizer in their loop). Applications are for the generation of linear FM (chirp signals), minimum shift keying (MSK) modulation, and frequency hopping that requires low spurious signals during hopping (in network applications).

Theoretically, a direct digital synthesizer produces such a result because a change in frequency changes only the slope of the accumulator (see Fig. 1-1). However, in practice, there are limitations to the level of this smoothness.

There are two main reasons for the limitation. The first is that the change in frequency is a step function and happens within 1 clock. Thus the output from the DAC has a smooth transition, but the DAC output is followed by a low-pass filter. The low-pass filter, however, has its own "inertia" (finite bandwidth) and will generate some transient if the step size is large. Therefore, as long as the step size in frequency is low relative to the filter bandwidth, the transition will be totally smooth. However, if the step size is substantial relative to the filter bandwidth, i.e., the step is close to the total bandwidth of the direct digital synthesizer, then a transient will show; see Figs. 4-55 and 4-56.

The second is due to the accumulator structure. In the majority of direct digital synthesizer hardware, the accumulator uses pipelining (see Chap. 6). If the front end is not compensated, as is the common case (see Chap. 6 for details), then it is necessary to wait for the pipe to fill before a new frequency can be updated. If a new frequency is switched before the pipe is filled, a major transient will occur.

3-4 Spurious Signals (Especially DDS)

The measurement of spurious signals is done directly on a spectrum analyzer and is a very common and simple test. However, when this measurement is done on DDS signals, caution needs to be exercised. This must be emphasized, because designers spend a lot of time looking for problems that are just not there. The reason is as follows:

A direct digital synthesizer, especially when measured without the low-pass filter (LPF), generates a multitude of frequencies—the main output, the aliasing and their harmonics, and the clock leakage. Not only are there many signals connected to the

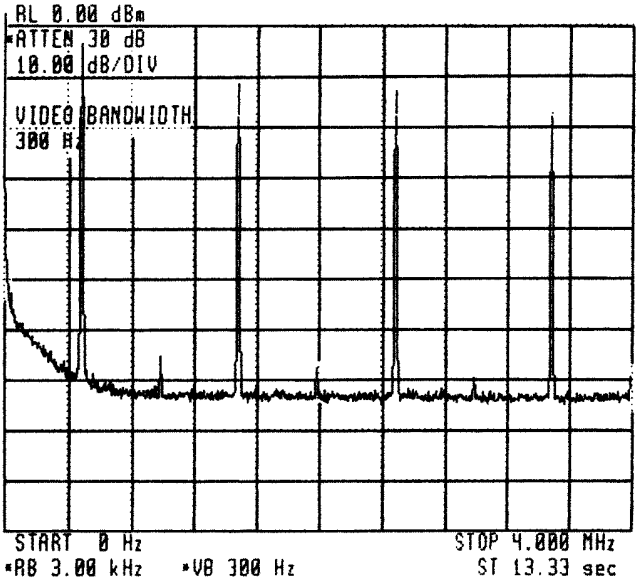


Figure 3-9 DDS output spectrum, no LPF.

front end of the spectrum analyzer, but also their relative difference in power is low. For example, for a 20-MHz clocked direct digital synthesizer with a main output of 2 MHz, the aliasing will show at 18 MHz, and the level difference between them will be theoretically 20 $\log[(\sin x)/x]$, where $x = \pi$ 18/20, yielding approximately 20 dB. For a main output of 4 MHz, the aliasing will be at 16 MHz, with a ratio of 15 dB. There are, of course, many other signals, see Fig. 3-9.

If the proper attenuation is not applied, the spectrum analyzer will generate its own intermodulation spurious signal, as shown in Fig. 3-10. These are test artifacts. The use of a LPF helps this problem since the main signal is a dominant one. However, because most DDS circuits generate many signals, even a measurement with a LPF needs to be carefully designed.

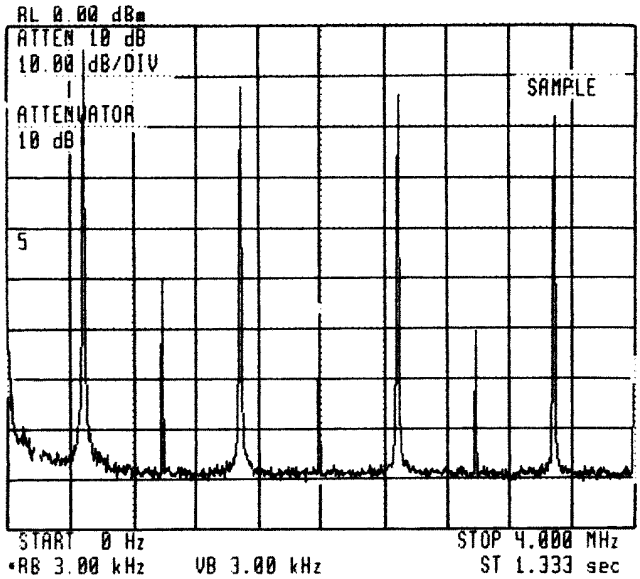


Figure 3-10 This is the same as Fig. 3-9 but with the wrong spectrum analyzer attenuation setting.

3-5 Phase Memory

The requirements for the phase behavior are shown in Fig. 1-1. The measurement is very simple. The synthesizer hops between $F_1, F_2, F_3, \ldots$, and its phase is compared to a fixed reference, say, at F_1. When the synthesizer hops to F_1, the phase must be the same, as in the previous hop.

3-6 Step Size

The output frequency and the step size of a synthesizer are measured directly on a counter. Since the frequency of DDS (the common binary implementation) generates "nonround" numbers, this has to be anticipated. For example, in a binary direct digital synthesizer clocked at 10 MHz, the most significant bit (MSB) generates 2.5 MHz; the next bit generates 1.25 MHz; but the 12th bit generates $10^7/2^{12} = 2441.40625$ Hz, and the 24th bit generates a frequency of 0.596046… Hz.

3-7 Linear FM

One of the more complex tasks is to measure the linearity of a linear FM direct digital synthesizer. Because of the inherent accuracy derived from the synthesizing of the signal at each step, the classical methods used to measure these parameters are not sufficient. The practical limitation of such a waveform is actually limited by two parameters:

1. The system has frequency quantization. Because this is a direct digital synthesizer, it can only approximate a VCO, so there is a quantization effect. For example, the Sciteq Electronics P/N DCP-1 uses a 24-bit accumulator with 500-MHz clock. The frequency resolution (quantization) is never below $500 \times 10^6/2^{24} = 29.8$ Hz.
2. The low-pass filter that follows the direct digital synthesizer has a group delay characteristic, and since it is a linear device (usually an *LC* network), there is a limit to the accuracy that an equalizer can compensate for this error. When used in critical applications, the equalizer becomes very complex, especially close to the high frequency where the group delay of the filter is very sharp. This measurement is very complicated, and the instruments on the market might have less accuracy than the generators! Our conversations with Hewlett-Packard and Racal Dana have not been conclusive as to the source of the error we measure; it might be an instrument limitation.

There are two instruments on the market that we found useful for this measurement: the Racal Dana 2351 time-frequency analyzer, built around a very high-speed, accurate counter, and the Hewlett-Packard model HP8981.

3-8 Conclusion

Measurements are complicated and demanding, as the requirements of frequency synthesizers become stricter. Automation and care must be exercised to ensure coverage and relevant results. A lot of sophisticated (and very expensive) equipment is available. However, it is also possible to design lower-cost sets, by using baseline equipment and one's own software and homemade auxiliary hardware. Parameters such as frequency accuracy, spurious

signals, amplitude flatness, and phase noise can be measured directly on a spectrum analysis. For more demanding phase noise measurements, a clean source and an FFT analyzer will do the job. A few companies now offer equipment for automated phase noise measurement, among them Hewlett-Packard, RDL, and Comstron. These are accurate setups, and they require very complex software programs for calibration and calculations. Software can also calculate from the phase noise plot parameters such as FM noise or integrated phase error (in degrees or radians). See the References for various instrumentations.

References

1. Racal Dana model 2351: pulse counter, computing counter.
2. Hewlett-Packard 5361B computing counter.

Synthesizers used to measure phase noise

3. HP 8662A phase-locked loop.
4. Comstron FS-2000, 5000 (DA).
5. PTS 300, 500, 1000 (DA).

FFT and digital spectrum analysis

6. HP 3563A.
7. Tektronics model 3052.
8. Standard Research model SR-760.

Direct digital synthesis (DDS), arbitrary signal generator (ARB)

9. HP 8770A.
10. FASS HP 8791.
11. Stanford Research model DS345.

Spectrum analysis, radio frequency

12. HP 70000 series; also model HP 856x.

Phase noise measurement systems

13. HP 3048A.
14. RDL: phase noise measurement system using delay line technique.

Chapter

4

DDS General Architecture

In the last few years, *direct digital synthesis* (*DDS*) technology has captured the attention of synthesizer designers and enjoys an unusual popularity. Various designs and architectures are used to implement DDS, and the applications vary according to the requirements. However, the basic principles of the commonly used DDS technology have been outlined in an article published in 1971 by Tierney et al. (see Ref. 1 and Chap. 10). This chapter briefly describes these various architectures and discusses their advantages and relative weaknesses.

It must be emphasized at the outset that most designers would prefer an all-digital design, since digital designs have the advantage of accuracy, repeatability, producibility, and such. In practice, most DDS designs are not "all digital." They use one analog part—the *digital-to-analog converter* (*DAC*)—and this part is the limiting factor in DDS performance applications to date. Although most DDS designs currently employed use a DAC, it is felt that if an all-digital design were achieved, the technology would become even more attractive since performance would not depend on any analog part and therefore would be totally predictable without the tolerances inherent in analog circuits.

There has not been much published on the subject of all-digital DDS, and one can hope that the scientific community will expend more effort to analyze the theoretical limits of such an approach.

Some theoretical work being done currently will be mentioned later in the chapter.

However, for some low-frequency applications, such as industrial control and audio (especially for CD applications), some technological and theoretical work has been demonstrated. Also devices are available that use delta-sigma modulation and extensive digital signal processing (DSP) operation for noise reduction and spectral noise shaping. Many CD players use single-bit DACs and generate audio, not necessarily sine waves.

This survey will present a few classical approaches to all-digital DDS and then will proceed to the more commonly used design that uses DACs, referred to here as *standard DDS*. Although some theoretical work has been done recently on error generation in the digital part of DDS designs, and a good level of understanding has been gained as to the nature of quantization errors, for all practical purposes most of the imperfections are generated in the DAC, and the performance of DDS depends today mainly on the performance (and artifacts) of this (single) analog part—its accuracy, dynamic range, intermodulation products, linearity, and dynamic characteristics. For DDS applications, there are no useful DAC models yet for analysis. DACs have to be evaluated in the circuits, and some products are better than others without our being able to point to the exact reasons. Speed, accuracy, symmetry, and balance are important, but these parameters do not tell the whole story.

Since DDS is truly a DSP discipline, we have chosen to start with a short description of digitizing techniques and then proceed to synthesis. It is imperative to understand well the digitizing process and its penalties. Synthesizing digitally is just the inverse operation to digitizing.

Imagine sampling and digitizing a sine-waveform and then converting it back to its analog form. In DDS, the digital samples are generated (because the waveform is known) according to a specific method that yields exactly the same results as sampling a sine wave. Since DDS deals with sampled data, it creates many nonintuitive "artifacts," and some are demonstrated here, in the time and frequency domains, especially in App. 4B.

At the center of DSP stands the sampling theorem (see App. 4C). The theorem states that any waveform whose power spectrum is zero beyond the frequency range $-B < f < B$ can be represented by its samples, providing that the sampling interval T is equal to or

smaller than 1/2*B*. The theorem also shows that after sampling, when the signal is reconstructed, the reconstructed spectrum is periodic; in addition to the original signal spectrum, the process generates "copies" of this spectrum around the sampling frequency and all its multiples. To eliminate these "spurious" shadows, the theorem requires the use of a perfect low-pass filter (one that is also noncausal). In reality, sharp (very fast cutoff rate) filters are used. These types of "signal copies," or *aliasing signals,* as they are referred to in DSP, will be produced in DDS, too, and will be discussed throughout this chapter.

4-1 Digital Modulators and Signal Reconstruction

As mentioned, it is rather useful to start the analysis with tools that digitize analog signals rather than those that synthesize them, since a level of insight and intuition can be gained. The most common technique used to digitize analog signals is *pulse code modulation* (*PCM*). It consists of a sample-and-hold device which samples the analog signal, followed by a digitizer which encodes the signal into its digital representation. This is, of course, the way music or video signals or any other sampled data are stored in compact disks (CDs) or digital recorders. The procedure is shown in Fig. 4-1.

This is probably the most straightforward digitizing technique, and it is the exact inverse operation of the standard DDS. In a direct digital synthesizer, since the signal is known (sine wave), the samples are generated digitally and are identical to those received from sampling a sine wave and then converted via the DAC to their analog forms.

However, a point to remember is this. If the digitized signal shown in Fig. 4-1 were connected to a digital-to-analog converter,

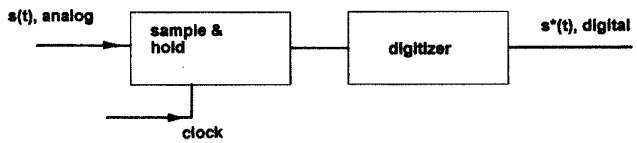


Figure 4-1 Pulse code modulation.

the analog signal at its output would be reconstructed. However, because of the quantization error (caused by the use of a finite number of bits), the quantized reconstructed signal is only an approximation of the original sine wave and therefore introduces always an error. Thus the quantized signal always represents only an approximation of the "real" analog signal.

PCM is very costly in *bandwidth* (*BW*). For example, the standard in telephony is a 3.3-kHz analog voice signal usually sampled at 8 kHz (the sampling theorem requires that the sampling rate be more than twice the signal BW). Then each sample is represented by 8 bits. [This is the standard in telephony, although the signal is also equalized (companded) according to a format, either American or European, to extend its dynamic range.] As a consequence, a 3.3-kHz analog signal is now represented by a digital signal whose rate is equivalent to 64 kbits/s (8000 samples/s $\times$ 8 bits).

In CD applications, the sampling rate is approximately 44 kHz, and the signal is represented by 16 bits for an equivalent rate of approximately 700 kbits/s (per channel).

PCM is very instrumental in understanding the operation of standard DDS. If it were possible to reverse the sequence, generate digitally the sampled signals as they come out from the sample-and-hold device and digitizer, and connect them to a DAC, then the reverse operation would occur and an analog sine wave would be produced from the generated digital samples. In the case of PCM for voice or any other data, the sampled signal is a stochastic process, and therefore this process is not applicable (see Sec. 4-13). However, a sine wave is a totally deterministic process, and its digital samples can be calculated accurately; and this is the basis of DDS principles.

The methods and practices used to generate a signal in this way are referred to here as *standard DDS* and are discussed in detail in Sec. 4-4. For many designers, the fact that a sine wave can be reconstructed accurately by using only 2.5 samples per cycle is hard to comprehend. Maybe so. As was mentioned before, sampling can be sometimes tricky and nonintuitive; but this is what the theorem proves, and this is exactly what DDS spectra show, so one has to resign oneself to these nonintuitive phenomena.

The two basic items needed to generate the digital sine waveform will therefore be a running index mechanism and a sine lookup table. The running index generator, say, a counter, will

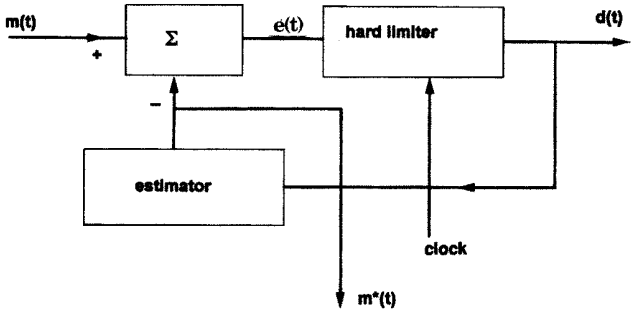


Figure 4-2 Delta modulator block diagram.

then address the lookup table, which will produce the digital sine amplitude samples sequentially. A detailed description of this mechanism can be found later in this chapter.

Another approach to digitizing an analog signal that has gained popularity, mainly for narrow band and voice signals, is delta and adaptive *delta modulation,* shown in Fig. 4-2. This method represents the analog signal by a single-bit stream, in contrast to the representation by PCM of digital words (multiple bits). We cite this method because later we mention single-bit DDS implementation.

The delta modulator is a feedback system which uses an estimator in the feedback path, based on a 1-bit error signal. The estimator determines the complexity and quality of the modulator, and obviously it is used to reconstruct the analog signal in the receiving side (Fig. 4-2).

The signal path is as follows: An analog signal $m(t)$ is applied, and since the output of the estimator is 0 at t_0, an error $e(t)$ is developed. The error is digitized in the hard limiter whose transfer function is given by

$$d(t) = \begin{cases} 1 & \text{if } e(t) > 0 \\ -1 & \text{if } e(t) < 0 \end{cases} \tag{4-1}$$

The output of the digitizer is clocked at a fixed rate, and the digital signal $d(t)$ represents the analog signal $m(t)$. Also $d(t)$ is used as an input to the estimator, which generates a signal $\hat{m}(t)$. The purpose of the estimator is to minimize the error $m(t) - \hat{m}(t)$.

In the case where the estimator is just a simple integrator, say, an *RC* circuit, a tentative output of the modulator is shown in Fig. 4-3.

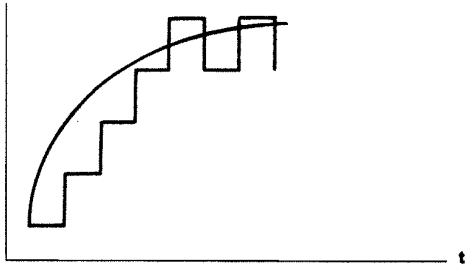


Figure 4-3 Delta modulator waveforms.

Intuitively, it is quite clear that as the clock rate increases, so increases the accuracy of the estimator, as the rate of updating the error improves.

A few of the theoretical results are presented below.

The single-integrator delta modulator signal-to-noise ratio is

$$\frac{S}{N} = \frac{0.2F_{ck}^{3/2}}{F_m F_o^{1/2}}$$

For the double integrator,

$$\frac{S}{N} = \frac{0.26F_{ck}^{5/2}}{F_m F_o^{3/2}}$$

where F_{ck} = clock frequency
F_0 = signal bandwidth
F_m = frequency applied, $0 < F_m < F_0$

More complex delta modulators use complex estimators, build memory and adaptive processing into this function, and achieve excellent results at and below 32-kHz clock rates [also known as *continuously variable-slope delta modulators* (*CVSDs*), which change the increment according to a specific algorithm; see Ref. 32]. This is possible partly because of the voice's special spectrum as well as the sophistication of the estimator.

However, in voice systems, the quality is not a firm specification, and "toll quality" in telephony is not defined in mathematical terms. In frequency synthesis, such a luxury is not available, and performance characteristics such as spurious signals are measured on accurate instruments and have to meet strict specifications.

The delta modulator was mentioned mainly to demonstrate the possibility of representing an analog signal by a single bit digitally. Since the procedure requires only a two-state device, actually a single-bit data conversion device (hardly a DAC) is necessary for the presentation. This so-called single-bit DAC is the ideal digital-to-analog converter since it requires only two accurate analog states, 0 and 1, or −1 and 1.

Such direct digital synthesizers, if they could be constructed (without the use of a DAC), hold great interest, assuming a way could be found to execute them with spectral efficiency, i.e., achieve a reasonable ratio of signal frequency to clock. So far, such designs require ratios of 1:1000 and more. (CDs measure performance at 1 kHz and sample at 12 MHz.)

If a single-bit DAC is used, it will cost in bandwidth, and the ratio between the clock and the available bandwidth will have to be high to achieve reasonable performance. Some progress made in this field will be mentioned later in the chapter.

What makes DDS work easier compared to, say, voice coding is that the synthesizer designer knows exactly the waveform that needs to be generated, e.g., a sine wave.

A general block diagram of a single-bit direct digital synthesizer is shown in Fig. 4-4, and its operation is demonstrated here. The accumulator (or counter) is used as an index generator and as usual (in DDS) determines the output frequency. The reason that an accumulator is used in all DDS circuits is that a counter can only increment by 1, whereas an accumulator can increment by any number. The accumulator (a digital integrator) serves therefore as a complex counter (or an indexer), and the rate at which the

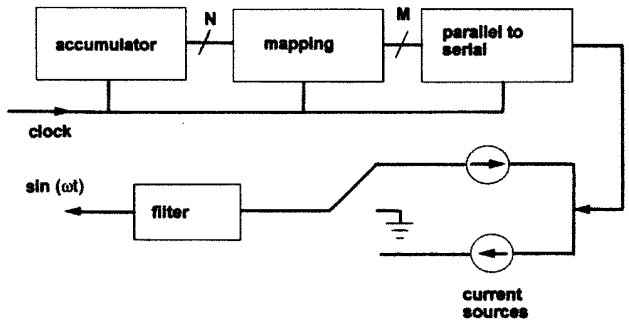


Figure 4-4 Single-bit DDS.

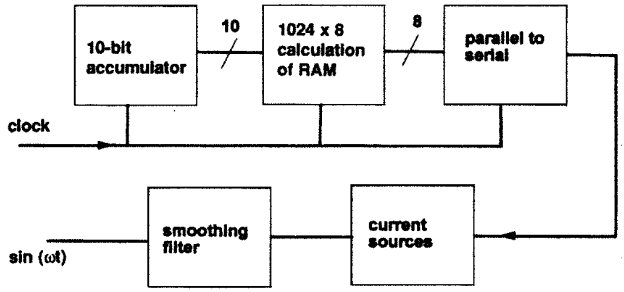


Figure 4-5 Single-bit sine approximation.

accumulator overflows determines the output frequency. The N most significant bits from the accumulator are connected to the mapping circuit, which converts the state of the accumulator to a digital representation of the analog signal. At this point, the digital representation of the sine is available and is exactly what would be received from sampling a sine wave. Then the data are provided as a single-bit stream via the parallel-to-serial device, which converts this signal to a serial bit stream that is controlling a plus-or-minus current source, which is "smoothed" to generate the sine wave. Such crude implementation is demonstrated in Fig. 4-5.

In this case, the digital representation of the sine wave is generated by the lookup table (ROM). But instead of using an 8-bit DAC, the output of the lookup table is converted to a serial 2^N (here 256 bits) signal using the correct weighting. This means that the *most significant bit* (*MSB*) has a weight of 128 bits, the MSB -1 bit has a weight of 64 bits, ..., and the least significant bit (*LSB*) has a weight of 1 bit.

Thus, the parallel-to-serial converter will slow the operation by 255 bits since each sample is represented by 255 serial bits. Or, in other words, if the parallel-to-serial device outputs the signal at a rate of 25.5 MHz, then the maximum output frequency will be less than 50 kHz, since the output of the ROM represents a rate of 100 kHz and must sample the signal at least twice per cycle.

It is obvious from this simple demonstration that the signal BW and signal quality depend on the ratio between the clock and the output frequency range. As this ratio increases, more bits can be represented by the ROM and better signal quality will be achieved, but the available output bandwidth will be lower.

This basic technique—with an additional digital signal processor to improve the efficiency of the encoder and the use of wave shaping on the spectrum so as to shape its noise outside the useful band (>15 kHz for voice) so that it can be filtered (this is also referred to as *oversampling*)—is in current use for digital audio systems. Many CD players use some forms of delta modulation and/or delta-sigma (inverse) modulation without the use of a DAC, but at a great penalty in clock rates (see Ref. 26). The ratio of the desired output spectrum in CD players (15 to 18 kHz) to the clock rate is 500 to 1600 to achieve a dynamic range of 80 to 100 dB (usually specified at 1 kHz), and the system clock is on the order of 12 to 20 MHz.

Delta-sigma modulators encode only the difference signal

$$m(t + T) - m(t) \tag{4-2}$$

as is done also in differential PCM applications and so save the information rate, i.e., spectrum, and allow better performance for a given clock speed. For more details, see Section 5-4.

These circuits take also advantage of the spectrum of voice and/or music where most of the energy is concentrated below 3 to 4 kHz and is not evenly distributed across the whole 15 kHz. As a result, compression algorithms have been developed to optimize performance for voice signals.

At these clocks, CMOS circuits are quite low-cost, achieve high levels of integration, and consume low power. Many manufacturers find it much more attractive to use oversampling than a full DAC (see Ref. 26). As already mentioned, digital designs are desired; as in an all-digital design, the repeatability and producibility are perfect, without the tolerances inherent in analog circuits, which age, change with temperature, etc.

This technique is thus quite useful when the output frequency is limited. Even in the case of sine-wave synthesis where our knowledge of the signal to be generated is so complete, where very complex DSP operations can be applied, and where the two-state current source (our single-bit DAC) can be easily converted to a three-state device (+1, −1, 0), we know of no design that demonstrated good spectral purity (spurious signal better than 60 to 70 dB below the carrier) with a clock-to-output frequency ratio of less than a few hundred to a few thousand. Digitizing and reconstruction help you to develop intuition as to the way digital frequency

synthesis works. The above examples serve as an exercise only, since this is not the focal point of this text.

As mentioned above, there has been significant progress in the application of low-frequency single-bit direct digital synthesizers, and we will come back to this issue later, after the elements of standard DDS have been covered in Sec. 4-12.

4-2 Pulse Output DDS of the First Order

The pulse output direct digital synthesizer of the first order is probably the simplest and crudest DDS type. It uses only an accumulator and uses the MSB or carry output signals as its output. The accumulator shown in Fig. 4-6 follows the equation

$$S(n) = S(n - 1) + W \tag{4-3}$$

where $S(n)$ is the accumulator output at clock tick n and W is the input control.

If we assume that the carry output is used as the output signal, and given that the accumulator performs a modulo 2^N arithmetic (for an N-bit binary accumulator), the average output overflow time is given by $T(2^N)/W$, where T is the clock period. Thus the average output frequency is given by

$$W_{av} = \frac{W}{T \cdot 2^N} = \frac{WF_{ck}}{2^N} \tag{4-4}$$

where F_{ck} is the clock frequency. A typical waveform is shown in Fig. 4-7 for $N = 4$ and $W = 4$ and $W = 5$. As long as W divides 2^N, the output is periodic and smooth, but all other cases create jitter. This is quite clear. Since the output can change its state only at

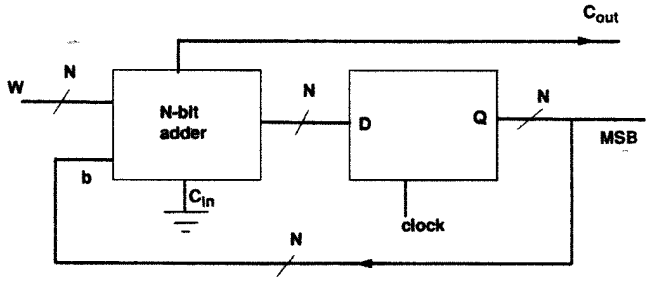


Figure 4-6 Single-bit DDS of the first order.

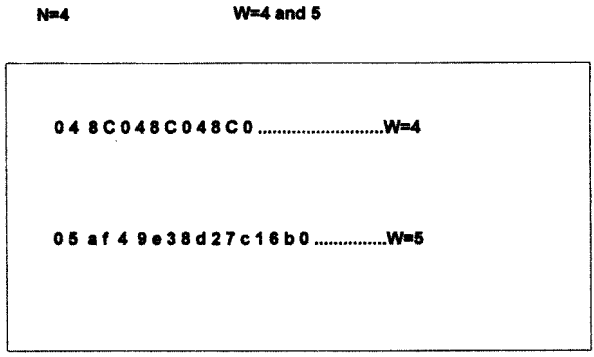


Figure 4-7 Single-bit DDS waveform.

the rate of the clock, if the desired output frequency W is not a factor (a divider) of 2^N, then a phase error is created between the ideal output frequency and the actual output.

This phase error will increase (or decrease) until it reaches a full clock period, at which time it returns to zero and starts to build itself again. This new period creates a new periodicity cycle, different from the desired output as a source for undesired, spurious signals.

Another way of viewing this phenomenon is to observe that the actual output is given by the ideal output (in this case $F_{ck}/4$ and $F_{ck}/5$) sampled and held at the clock rate, as shown in Fig. 4-7. It is clear that the *rate of error,* or the *error's periodicity,* is given by

$$T \cdot \gcd(W, 2^N) \tag{4-5}$$

where gcd(a,b) is the *greatest common divider* of a and b. In the case when $\gcd(W, 2^N) = 2^N$, there is no error, the accumulator output represents a perfect divider (the division ratio is given by $2^N/W$ or, here, 4), and the output frequency (average and instantaneous) is exactly proportional to W, with no error being developed.

However, for the case of $N = 4$ and $W = 5$, the error periodicity is

$$\gcd(16, 5) \cdot T = 80T = \frac{80}{16} = 5 \text{ cycles} \tag{4-6}$$

but the instanteneous periodicity changes!

Since only $W = 2^M$ ($M = 0$ to $N - 1$) divides 2^N, the majority of the control frequencies create additional periodicities that even-

tually manifest as spurious signals. For example, for $N = 20$, only 20 (including $W = 0$) out of $2^{N-1} = 524{,}288$ possible inputs create a nonspurious signal. Thus for all practical purposes, all output frequencies generate spurious signals.

The term *error* is used here to indicate a difference between the average frequency and its instantaneous value.

A spectral evaluation of the output can be calculated for any N and W, and the additional spectral lines will have a periodicity as calculated in Eq. (4-5).

There is an equivalent way of generating the same type of output, one that is used extensively in PLL synthesizers, by a dual modulus divider. This is shown in Fig. 4-8. See also Ref. 27. In this implementation, the divider divides by M normally (i.e., when the carry output is at state 0), but will divide by $M + 1$ (or $M - 1$) when the carry output of the accumulator is high (carry present). It can be easily shown that the average output frequency is

$$\frac{F_{ck}}{M + W/2^N} = \frac{F_{ck}}{M(1 + W/M2^N)} \tag{4-7a}$$

(The carry output frequency is $W/2^N$.) Note that for $W/2^N << 1$, the Taylor approximation of Eq. (4-7a) is

$$F_{av} = \frac{F_{ck}}{M}\left(1 - \frac{W}{M2^N}\right) \tag{4-7b}$$

We mention this fact to highlight similarities between DDS and fractional-N PLL; see Chap. 5. In both cases the jitter level

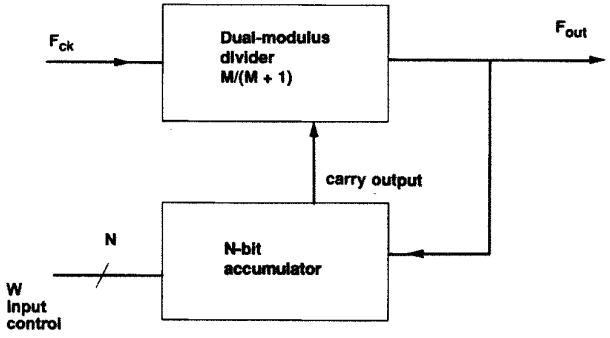


Figure 4-8 Single-bit DDS similar to structure used in fractional-N PLL.

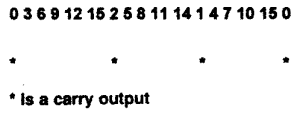


Figure 4-9 Average frequency for 4-bit accumulator with $W = 3$.

increases/decreases linearly with $W/2^N$. For low W (relative to 2^N), jitter appears quite rarely, and its energy is low relative to the desired output frequency, which is not the case for large W.

Before we analyze the numbers in detail, it should be clear that for low enough $W/2^N$ ratios, it is possible to get reasonable performance at a large sacrifice of useful bandwidth. This is quite similar to the single-bit representation of Sec. 4-2. In this sense, for specific low-speed applications, this is a practical synthesis technique.

As an example, we pick $N = 20$ and $W = 2^{12} + 1 = 4097$. The average output frequency is given by $F_{ck} \cdot 4097/2^{20}$. This is shown in Fig. 4-10. Since it takes the 2^{12} weight (the 12th bit) 256 clock ticks (2^8) to propagate to the carry output, the output frequency is $F_{ck}/2^8$ 255 times at equal intervals. Then one carry output will be shown at an irregular interval, shorter than the others by 1 clock tick (see Fig. 4-10 and App. 4B). Thus the phase does not accumulate continuously as in an ideal signal but skips 360° (of clock) at once. This discontinuity in the phase behavior generates a spurious signal. Since this happens once per 256 carry outputs, a spurious signal of about 20 log 256 = 48 dB can be expected. This is very similar to dividing by 256 or alternating the error by 256.

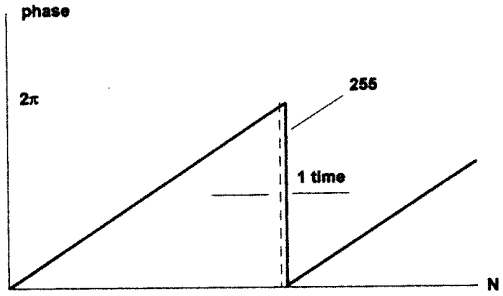


Figure 4-10 Phase transient of single-bit DDS.

However, if the control is $W = 2^{18} + 2^{16}$, the spectral picture will change dramatically. In this case, the phase discontinuity occurs every 4 cycles, and the time-domain picture as well as the spectral picture is very distorted. The case of $W = 2^{18} + 2^{8}$ might seem intuitively to yield reasonably good spurious signals since the ratio of 2^{18} to 2^{8} is large, but this is not true. The reason is that the output signal can be viewed as a product of two functions: one results from the 2^{18} command only and yields a perfect square wave of frequency $F_{ck}/4$, and the other results from the 2^{8} command, which is 1 for 1023 cycles and 0 for 1 cycle. This signal is saturated with harmonics, and since the total output spectrum is the convolution of these two functions, a high level of spurious signals occurs. There is no escape. For "good" performance, W must be very small relative to 2^{N}.

We can summarize the usefulness of this (single-bit) type of DDS as follows:

1. Reasonable and useful performance is achieved only if low bandwidth is being utilized, and the utilization is mainly in such applications as clock generators (where the average is important but not the instantaneous frequency) or locking very narrowband PLL circuits [in many cases using *voltage-controlled crystal oscillators* (*VCXOs*)] to the output signal to clean its spurious signal response. Such PLL sources have very narrow bandwidth and act as very narrowband tracking filters, thus filtering out the spurious signals.

2. Another application is possible when the accumulator is small and the clock frequency high. Then the spurious signals (which are far from the carrier) can be cleaned by a PLL or some other mechanism, as they show at a comfortable distance from the main signal. If, e.g., the clock is 640 MHz and the accumulator has 8 bits only, then the nearest spurious signal will show at

$$\frac{640}{2^{8}} = \frac{640}{256} = 2.5 \text{ MHz} \tag{4-8}$$

Such a distant spurious signal can be filtered out by a PLL circuit or some other filtering methods.

3. This type of DDS is useful for applications where instantaneous frequency is not important and only averages count, as in the case of time keeping (i.e., the number of pulses in a given time must

be accurate, but their "orderliness" is not important). However, note that by using these methods, we voluntarily wasted information available to us, since we used the timing information coming from the carry output or MSB only, but we ignored all the information available in the other accumulator output bits. We then expect that the loss of information causes a very coarse level of quantization (in this case 1 bit only), and something in the system has to give. That is, we pay in bandwidth or in spurious signal performance.

As we will see later, the common DDS and fractional-*N* PLL designs improve their performance by utilizing many more bits and reducing the quantization level substantially.

4-3 Pulse Output DDS of the Second Order

As mentioned before, the information contained in the other accumulator output bits is the error information between the carry output and the ideal signal output phase to the accuracy of the number of bits used. We refer to this number of bits as the *arithmetic being employed.* As was demonstrated before, the phase is accumulated gradually in the output of the accumulator, but shows on the output (carry or MBS) only when an overflow occurs. (See Fig. 4-11.)

Following operation, as every overflow occurs, the contents of the data in the accumulator's other bits represent the phase error between the ideal output and the carry output; it signifies where we are on the phase chart. It is therefore possible to pass the carry output pulse via a time-delay element that will be controlled by the contents of the accumulator output bits; the implementation is shown in Fig. 4-12.

Note again that only when W is a divider (factor) of 2^N does the carry output (or the MSB) change sign exactly when all the other output bits of the accumulator are 0. However, for any other W, there is a residual phase that indicates the instantaneous phase error between the ideal signal and the DDS output, also shown in Fig. 4-13. This error can therefore be compensated. The accuracy of the correction depends on the number of bits used to represent the error, denoted by Er, and the accuracy of the time-delay generator.

The performance that we can expect is therefore proportional to

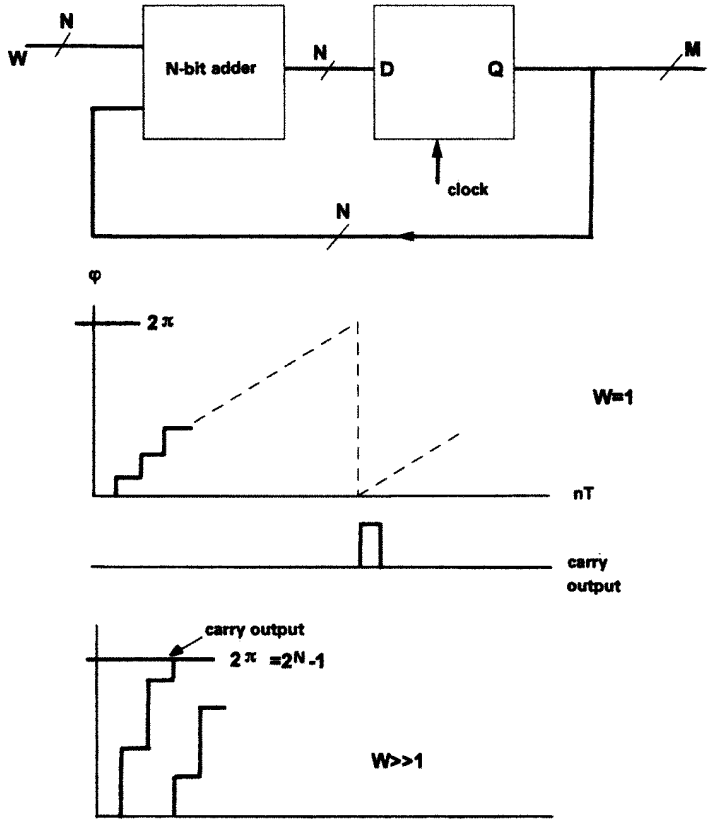


Figure 4-11 Phase built up in accumulator.

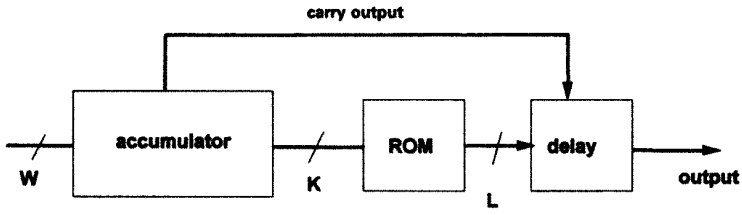


Figure 4-12 Pulse direct digital synthesizer of the second order.

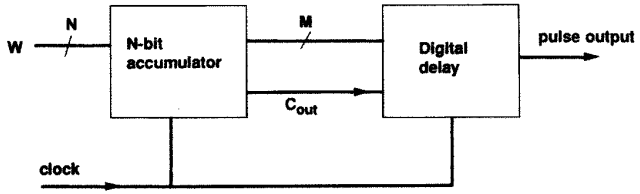


Figure 4-13 Single-bit DDS of the second order.

that of *N*-bit DDS, as analyzed later in this chapter. As for usefulness, not many manufacturers produce time-delay generators, and usually this is an analog device with limited accuracy and tolerances that depend on temperature and other conditions typical of analog designs. Lately, however, a few manufacturers, and especially Analog Devices, have introduced digital delay control elements, and this is the basis for the new design of a single-bit direct digital synthesizer to be mentioned here. See Sec. 4-12.

In addition, it is, of course, possible to generate extremely accurate time delay by presetting a counter or an accumulator. (This can be done as follows: Use an *M*-bit counter, and when the carry output flags, preset it to the address contained in the *M* output bits. Then clock it until the counter hits its terminal state, say, 0.) The drawback again is that it will require a very high-speed (compared to the signal generated) counting mechanism.

Suppose we wish to represent the accumulator error Er by 8 bits. That will require the clock rate going into the time-delay counter to be 256 times the output rate. And again, if this clock rate were chosen to be, say, 25.6 MHz, the maximum output frequency would be less than 100 kHz. This type of synthesizer will generate a square wave or a pulse, not a sine wave.

Because of the clock-to-signal frequency ratio and the limited availability of time-delay generators, this technique had very limited application. But this might change in the near future. A full evaluation of working hardware is not available yet, and although theoretically this "should" work, it will be necessary to evaluate the performance of existing hardware. See also Sec. 4-12.

4-4 Standard DDS

The standard direct digital synthesizer is a digital-and-analog mixed signal device and generates a sine wave at its output. (The

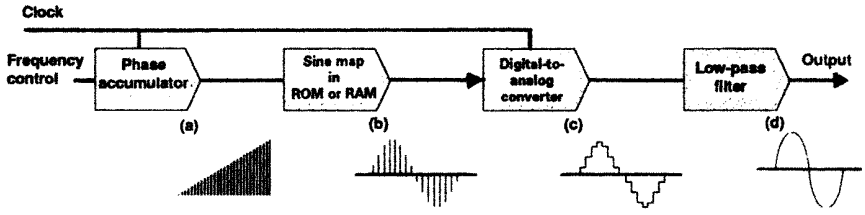


Figure 4-14 DDS block diagram.

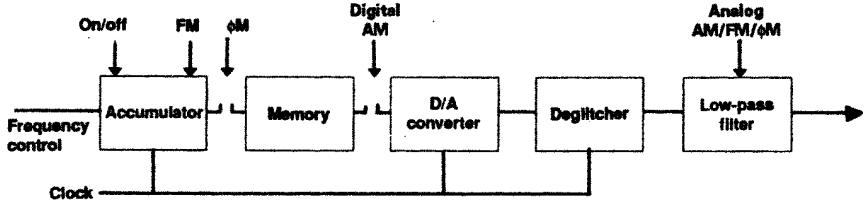


Figure 4-15 DDS block diagram and functionality.

previous devices generated square waves or timed transitions.) A general block diagram of standard DDS is shown in Figs. 4-14 and 4-15. Its output is shown in Fig. 4-16.

The output is a relatively smooth sine waveform, but because of the series of quantization processes occurring in the execution of the synthesizer, it is always only an approximation of the ideal waveform.

The input to the accumulator, the first DDS element, is the frequency control word, and it can be used for frequency control and modulation. Since W is a digital word, changing W changes the output frequency. In DDS this function is performed quickly, accurately, and without phase irregularities, hence the applicability for modulation.

The accumulator, which is a discrete digital integrator, is used as the DDS indexer, or a variable-rate counter; and the rate of its integration, controlled by the input word and the clock, generates a digital ramp. And this signal represents the phase WnT (T is the clock period, and n is the running index). And since at its output we have the digital signal that represents phase, it is rather simple to add phase control, by putting an adder (or subtractor)

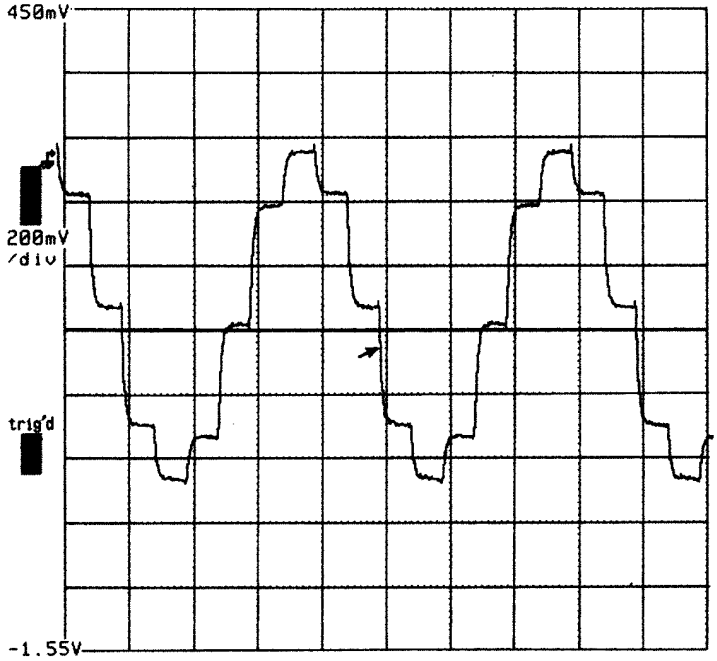


Figure 4-16 DDS output waveform before LPF.

between the accumulator and the ROM (see Fig. 4-17). The result of the extra device is shown in Fig. 4-18.

The adder (or subtractor) operates as a phase shifter and can be used for phase shifting or phase modulation. See also Sec. 6-5. The slope of the accumulator output, which is interpreted by the synthesizer as the rate of change of the phase, controls the output frequency. As the phase increment is given by

$$\delta\varphi(t) = \frac{W(2\pi)}{2^N} \tag{4-9}$$

$$\delta T = \frac{1}{F_{ck}} \qquad \text{time increment is clock cycle}$$

The output frequency is given by the phase derivative

$$F(t) = \frac{\delta\varphi}{2\pi\,\delta T} = \frac{F_{ck}W}{2^N} \tag{4-10}$$

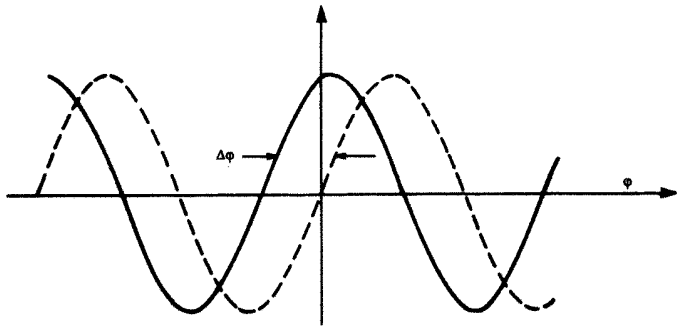


Figure 4-17 DDS phase-modulated output waveform.

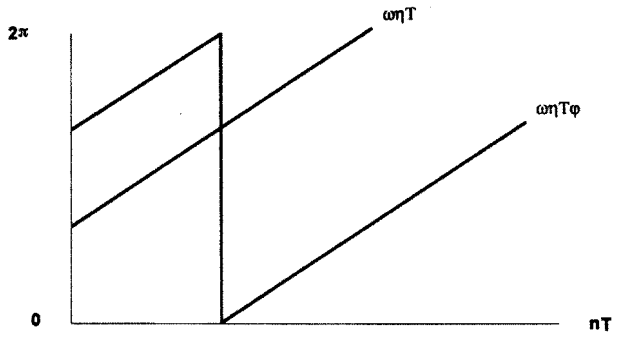


Figure 4-18 Phase modulation in DDS.

The ROM converts the phase information to its amplitude presentation, i.e., performs the transformation

$$WnT \quad \sin(WnT) \tag{4-11}$$

The ROM serves as a lookup table, converting its index (phase) input to sine amplitude samples. The transformation is, of course, nonlinear.

Note that the output of the ROM is identical to that of sampling a sine wave by using pulse code modulation (PCM), as described in Sec. 4-1. Since the output of the ROM represents the amplitude of the sine, amplitude modulation is possible if a multiplier is inserted between the ROM and the DAC. The effect of the multiplier is shown in Fig. 4-19. The output of the ROM (or the multiplier) is then connected to a DAC that transforms all the digital information (and manipulation or modulation) to an analog signal.

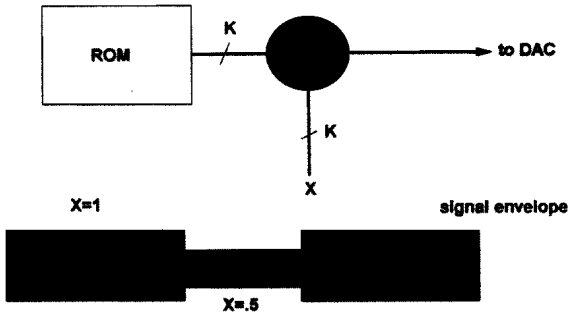


Figure 4-19 Digital AM and signal envelope in DDS.

As explained above, the output of the DAC incorporates all the frequencies and their aliases, shown in Fig. 4-59 (for a synthesizer clocked at 1 MHz and controlled to produce 0.5-MHz output). Note that the DAC is not generating the analog output as a series of delta functions (as described in the sampling theorem), but rather as delta functions followed by a sample-and-hold (S&H) device; see Fig. 4-20.

Thus the output spectrum is not given by a repeating spectrum, as the sampling theorem predicts, but rather by the computed ideal spectrum multiplied by the spectrum of the sample-and-hold device, given by

$$\frac{\sin \pi fT}{\pi fT} \tag{4-12a}$$

and the power spectrum by

$$\left(\frac{\sin \pi fT}{\pi fT}\right)^2 \tag{4-12b}$$

This calculates to approximately a 4-dB loss at the Nyquist frequency, given by $fT = 0.5$. (See Fig. 4-21.)

We therefore expect to find the aliasing frequency energy attenuated according to the ratio expressed in Eq. (4-12*a*) and no energy at $f = 1/T$, the clock frequency. The residual energy shown at the clock frequency of a real-world signal represents the (analog) clock leakage. As an integral part of the DDS (required by the sampling theorem), and in order to define the signal uniquely, a *low-pass filter* (*LPF*) is added to filter out all aliasing frequencies. Very few systems use the aliasing frequency as an output.

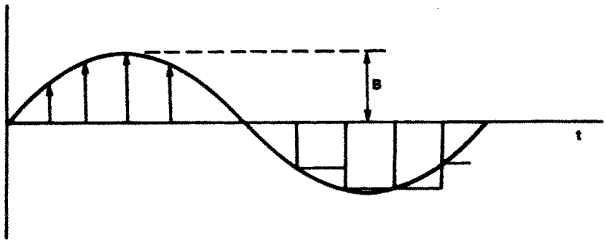


Figure 4-20 DAC versus ideal delta function DDS output.

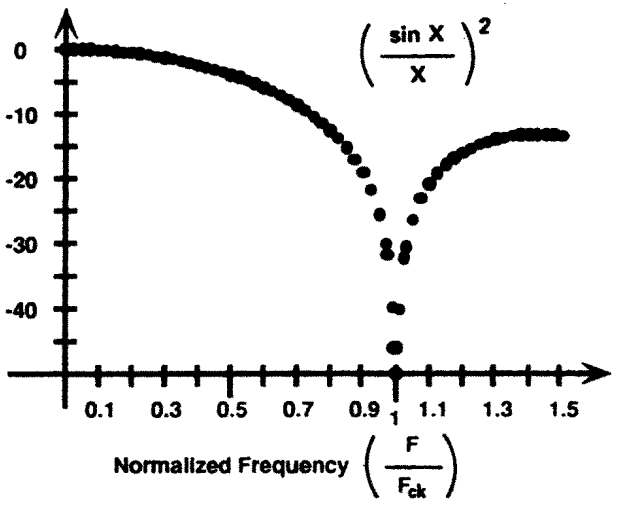


Figure 4-21 DDS output spectrum caused by DAC. For $F = F_{ck}$, output power drops to 0.

Again, note that because of the ambiguity introduced by sampling data, controlling the direct digital synthesizer to produce a frequency or its aliasing yields exactly the same output spectrum. As a demonstration, let's assume a direct digital synthesizer having a 4-bit accumulator. The output of the lowest output frequency, produced when the input control is 0001 (shown in Fig. 4-22), is equal to the output produced for an input control of 1111, except the phase is reversed by 180°. The output of control 0100 is equal to that of 1100. Obviously the sum of a controlled frequency and its aliasing is always a constant equal to the size of the accumulator. Thus W and $W^* = 2^N - W$ are a "pair" (see Chap. 6).

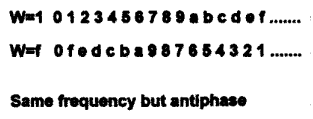


Figure 4-22 Signal and aliasing in DDS 4-bit accumulator.

Another way of looking at these twin signals is to specify one of them (say, 0001) as rotating clockwise and the other (1111) counterclockwise, or generating positive and negative frequencies. This can be tricky but also useful in some ways. The following is a simple demonstration:

Suppose that the direct digital synthesizer output needs to be upconverted but the user wishes to use both the upper and lower sidebands, as shown in Fig. 4-23.

At the output of the *upper sideband* (*USB*) filter, the output frequency increases with the output frequency of the synthesizer since

$$F_{\text{out}} = F_0 + F_{\text{DDS}} \tag{4-13}$$

But at the output of the lower sideband filter, the inverse happens since

$$F_{\text{out}} = F_0 - F_{\text{DDS}} \tag{4-14}$$

This can be inconvenient and can be confusing to the user. However, when the lower sideband is used, if the synthesizer is controlled to the same frequencies as before but the MSB is changed from 0 to 1, then aliasing will be generated and the output frequency will increase with the increasing control! The output fre-

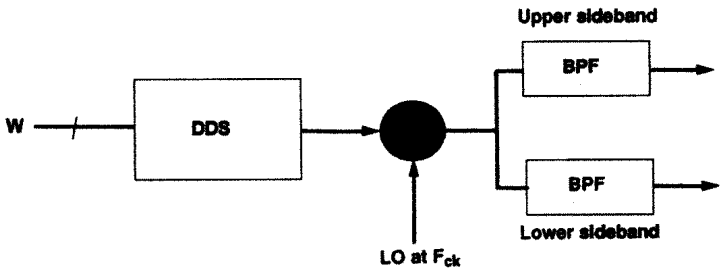


Figure 4-23 SSB generation using DDS and local oscillator (LO).

quencies for control of 0100 and 1100 are the same, and the output frequencies for 0110 and 1010 are the same. All the in between values are also reciprocal "pairs," and as one control generates increasing synthesizer output from 0100 to 0110 , the other generates the decreasing output given by the input control from 1100 to 1110 . (See Fig. 4-24.)

Here generating the aliasing signal helps ease the control mechanism. Note that the MSB manipulated here is the bit generating the Nyquist frequency, one that is usually not used in standard DDS applications.

In a binary direct digital synthesizer, if the accumulator consists of N bits, the most significant control bit generates

$$W = 2^{N-1} \qquad F_{\text{out}} = \frac{F_{\text{ck}} \cdot 2^{N-1}}{2^N} = \frac{F_{\text{ck}}}{2} \tag{4-15}$$

This is the Nyquist frequency. Therefore, W in most DDS applications controls mostly $N - 1$ bits, but not the Nth. Only in limited applications like the one mentioned above will the MSB be controlled. In this case, the aliasing signal will be in band and equal to

$$F_{\text{out}} = \frac{F_{\text{ck}}(2^N - W)}{2^N} \qquad W > 2^{N-1} \tag{4-16}$$

The output LPF filters the fundamental and rejects the aliasing. The filter complexity depends on the ratio $F_{\text{max}}/F_{\text{ck}}$. For most DDS designs requiring $0.4 < F_{\text{max}}/F_{\text{ck}} < 0.45$, seven-section elliptical (Cauer) filters are common. Since the aliasing signal is a spurious signal, it must be rejected by the spurious signal specification requirement.

Unlike the ideal (sin X)/X (impulse response; see Fig. 4-25) filter required by the sampling theorem, the filters realized in hardware are causal and introduce group delay variation that causes phase distortion. If necessary, a phase equalizer can be added. Another way is to use a digital filter that approximates the (sin X)/X function, mostly the *finite impulse response* (*FIR*) type.

The output power depends on the output frequency and is given by

$$\left(\frac{\sin \pi fT}{\pi fT}\right)^2 \tag{4-17}$$

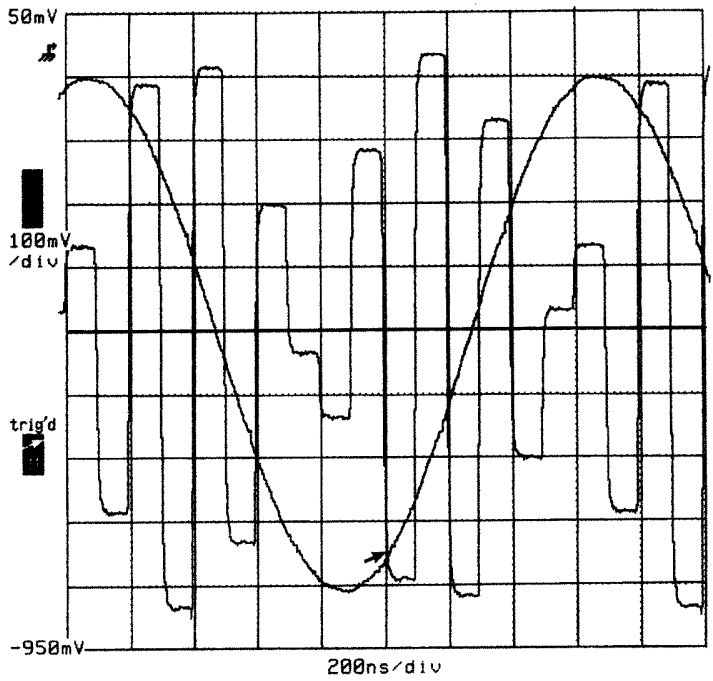


Figure 4-24 Two direct digital synthesizer–generated signals: $W = 1$ (0001) and $W = 7$ (0111).

as shown in Fig. 4-21. The filter usually adds some loss, and for flatness some amplitude equalizers can be used to improve output power flatness.

Since the LPF has a nonlinear group delay, especially close to the cutoff frequency, few DDS designs (such as linear FM chirps, where phase behavior is important) use phase equalizers. This, however, adds to the switching time. (Group delay can be equalized only by adding delay.)

Sometimes, but not too often, in hardware realization of DDS, a sample-and-hold device is inserted between the DAC and the LPF (Fig. 4-15). This device "cleans" the output spectra, and it is discussed in Sec. 4-11. Some applications of this device can help reduce amplitude variation. If the sample-and-hold device actually performs the function of passing the signal only at a portion of the clock time but grounding the output for the rest of the time, then the $(\sin x)/x$ function widens and the output spectrum flattens.

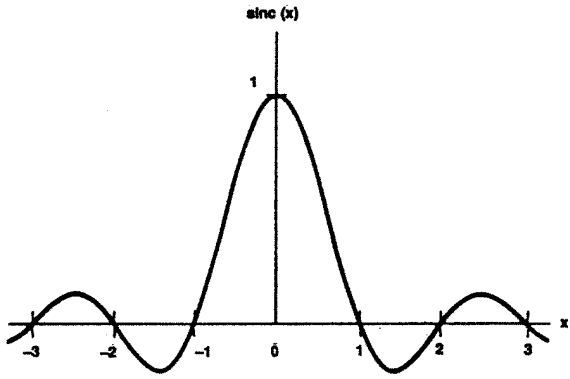


Figure 4-25 sinc $[(\sin x)/x]$ function.

We can then recap the operation of a standard direct digital synthesizer as follows: An accumulator is usually used to implement a digital linear ramp signal. This signal will range from zero to the full size of the accumulator. If the accumulator is a binary type (the most common) using N bits, then the accumulator size is 2^N. If accumulation is done in another way, say, BCD, the accumulator will again count from zero to the maximum state of the accumulator (ACM). We will refer to the zero state of the accumulator as zero phase and the maximum state as 2π or 360°. The accumulator performs its arithmetic operation modulus ACM, exactly as a sine wave is periodic with period 2π.

It is therefore possible (and necessary) to view the output of the accumulator as the sine-wave phase. Therefore, given the size of the accumulator, ACM, and clock rate, the output periodicity (frequency) and the step size can be calculated easily. The input to the accumulator controls the rate of change of the phase and by definition commands the signal's frequency.

If an accumulator is of size ACM and is clocked at F_{ck}, the lowest output frequency, which is also its step size, is given by

$$F_{res} = \frac{F_{ck}}{ACM} \tag{4-18}$$

and the general output frequency is given by $F_{ck}W/ACM$, where W is the input control word to the accumulator. For example, when the accumulator is binary, and $N = 20$ bits, $ACM = 2^{20}$ and the

resolution is $F_{ck}/2^{20} = F_{ck}/1048576$. However if ACM is exactly 1000000—and this can be achieved by using six BCD accumulator stages (or a binary accumulator and detection of state 999999, so that at this point all is reset back to zero; see Chap. 6)—then the resolution will be given by $F_{ck}/1000000$.

The accumulator is the device that determines the synthesizer's resolution, cycle length, and method of control (BCD, binary, others), and defines the signal's phase at every clock cycle. The accumulator can be set to a specific state or reset to 0, and this will determine the output sine-wave phase state. In almost all cases, only part of the accumulator output bits is used to drive the ROM. This causes a quantization error since part of the phase information available in the accumulator never shows in the input of the ROM. The accumulator therefore is the element of the direct digital synthesizer that determines the main frequency output and other periodicities causing spurious signals.

The ROM simply converts its input phase information ωt to the amplitude presentation $\sin \omega t$. The output of the ROM presents the sine-wave signal amplitude. Once again, since the output of the ROM has a finite bit length, another quantization error is introduced. The output of the ROM can never be perfect since an error already exists at its input. However, it is desirable and common to minimize the error introduced by the ROM, and this requires a relation between the word size at the input and the output of the ROM. If, from engineering considerations, we use only 8 bits at the ROM output, we will show later that it does not make sense to use 20 input bits (this will require an immense ROM whose output quantization is only 8 bits). Clearly, a reasonable relation between the number of input and output bits for the ROM must be achieved. The errors introduced by the ROM have an effect on signal quality, i.e., spurious signal levels and harmonics, but this does not affect the location of the spurious signals. These are calculated by the accumulator only.

The DAC receives the ROM output and converts the signal from its digital representation to its analog "real-world" form. Because of the sampling effects, a low-pass filter follows the DAC to smooth the signal and get rid of the sampling artifacts (aliasing signals). From time to time (but not always), a sample-and-hold device is connected between the DAC and the LPF. This device samples the

DAC output in order to clean the DAC output waveform and reduce the levels of transients that are inherent in DACs.

The direct digital synthesizer, unlike any other way of generating signals, builds the signal from the ground up, does not use a VCO or any other sine-wave source as the basic signal source (but needs a time base), and is totally a mathematical object; it is truly a numbers product. All parameters are represented by numbers; all periodicities of the signal and of the errors can be completely calculated. The only point where predictability is lost is in the DAC, the only analog element in the direct digital synthesizer. Here our ability to predict performance is only as good as our DAC model.

The direct digital synthesizer thus generates the signal by converting its quantized values. Since the quantization error is different for every state, it is possible to generate identical frequencies but with different levels of spurious signals (this is a question that often comes up). This can happen because the phase points can be different and therefore generate different quantization errors. If, e.g., we generate $F_{ck}/32$, there are many accumulator combinations that will generate this frequency. By going through the phase $P(0)$, $P(2^N/32)$, $P(2 \cdot 2^N/32)$, …, $P(31 \cdot 2^N/32)$, or $P(1)$, $P(2^N/32 + 1)$, …, $P(31 \cdot 2^N/32 + 1)$ [$P(x)$ is the phase address from the accumulator to the ROM], we go through different "constellations" that have different quantization errors.

This also explains why the spurious signals in the direct digital synthesizer often are not fixed in energy but are rather "bubbling." Some users prefer to specify these spurious signals at their average value, others prefer to use the worst-case scenario.

As mentioned, the frequencies and resolution are generated in the accumulator. We will now present decimal DDS applications, used as extensively as their binary-based counterparts.

4-4-1 Binary-coded decimal DDS

There are many applications, especially in instrumentation, where the use of a binary direct digital synthesizer causes a problem since the resolution is not a decimal number and not natural to users. In such applications, it is desirable to clock the direct digital synthesizer with a convenient reference, say, 10, 20, 40, 80, 100 MHz, and receive decimal, round step size and control.

For example, if a 24-bit binary accumulator is clocked with a 10-MHz clock, the frequency resolution is

$$\frac{10^7}{2^{24}} = 0.596046 \quad \text{Hz} \tag{4-19}$$

For such a direct digital synthesizer, a control word of, say, $W = 000123 = 291$ decimal generates the output frequency of $291 \cdot 10^7/2^{24} = 173.449$ Hz. This clearly cannot be used by an operator since the numbers need to be calculated at all times. We are decimal animals.

If a round resolution is required, it is necessary to generate a clock of 16.777216 MHz (2^{24}), locked to the external reference (usually 10 MHz), to get 1-Hz resolution, or even more complicated numbers to get 0.1-Hz resolution. Even this is not convenient because the control word is binary and needs constant calculation. One of the ways around this problem is to operate with a very large accumulator and let a microprocessor calculate the control word so close to the desired frequency that it will practically not be possible to measure the error. For example, with a 10-MHz clock, an accumulator of 48 bits generates an error of

$$\frac{10^6}{2^{48}} = 3.5 \text{ nHz} \tag{4-20}$$

It is therefore possible to generate every frequency, with, say, 1-mHz step, with a maximum error of 3.5 nHz!

Although it is very tiny, the error still exists, and the procedure requires a large accumulator (usually not a problem) and a computer to calculate the required control word. The calculation requires time and will substantially slow the synthesizer speed. Therefore, there are some applications where such an approach is practical and others that require fast switching speed and cannot afford the calculation time.

Some users then need complete BCD direct digital synthesizer implementations. The first such architecture was developed by Jackson; see Ref. 22. What Jackson suggested was to use 10 bits of binary accumulators, and when a carry function occurs, preload the accumulator with the binary value equal to 24 decimal (or 18 hexadecimal). Since 10 bits of binary accumulation accumulate to 1024 states, preloading by 24 yields an accumulator that passes

through only 1000 states. In his patent, Jackson demonstrated the operation with a clock of 8 MHz. With three such accumulators and the addition of the last stage of a 3-bit binary accumulator, the total count is

$$\text{ACM} = 1000 \cdot 1000 \cdot 1000 \cdot 8 = 8 \times 10^9 \qquad (4\text{-}21)$$

and with a clock of 8 MHz (8×10^6) the resolution is exactly 0.001 Hz ($F_{ck}/\text{ACM} = 8 \times 10^6/8 \times 10^9 = 0.001$ Hz = 1 mHz).

The control is, however, binary since the accumulator is binary. To use BCD control, some conversion mechanism from BCD to binary is needed. (This requires a substantial addition of logic.)

Different variations of this approach are possible, and this is definitely an elegant and convenient way of generating "nice" round and decimal resolution using "nice" round clocks while still staying in binary arithmetic. The use of 10-bit accumulators is convenient since 2^{10} is relatively close to 1000, but other organizations of the arithmetic functions are possible, too.

Another method for pure BCD accumulation was suggested by Goldberg (see Ref. 11). In this architecture, the whole direct digital synthesizer operates on pure BCD logic. The accumulator blocks are 4-bit BCD accumulators, allowing inputs of 0 to 9, and the whole accumulation chain is built from these building blocks, i.e., pure BCD logic throughout the whole logic chain.

The last accumulator stage (MSBs) is either BCD or binary for convenience. The input control and the interface to the ROM are therefore BCD, too. For example, the Sciteq Electronics ASIC part number SCX-50 is a BCD accumulator that accumulates to 400,000,000 states, and it includes eight BCD accumulators (32 bits, since each BCD stage has 4 bits) and a 2-bit binary accumulator at the top. Using a 40-MHz clock rate, this accumulator generates exactly 0.1-Hz resolution, using a decimal control. If a 20-MHz clock is used, the most significant bit is ignored, and the accumulator counts now to 200,000,000, generating again a 0.1-Hz step. If a 10-MHz clock is used, the 2 MSBs are ignored, and the step size is again 0.1 Hz. All operations, including control, are decimal. (STEL P/N 1176 is a BCD direct digital synthesizer chip with an 8×10^8 accumulator, and it includes the ROM lookup table, generating 0.1-Hz step when clocked at 80 MHz.)

In the last configurations (SCX-50, clocked at 10 MHz), using an accumulator of size 10^N and clocking at 10^M, the design is pure

BCD, all accumulators including the MSBs are BCD, and it counts to 10^N (100,000,000 in the specific demonstration above). The resolution is generally given by F_{ck}/ACM; it is therefore 10^{M-N} ($F_{ck} = 10^M$ and ACM $= 10^N$).

This is a relatively simple procedure to implement, by using BCD adders and registers similar to the binary construction. However, the mapping of the ROM is somewhat more complicated since the addresses are only at BCD numbers, and this causes not only complication in the ROM addressing way, but also a great waste of memory. In this respect, the Jackson method is more efficient.

For example, to map 000 to 999 (1000 accumulator states, using 12 bits; see Figs. 4-26 and 4-27), into 10 output bits, we use the following equations:

$$S(i) = 512 + \text{int}\left(511 \sin \frac{2\pi}{1000} \cdot i\right) \qquad \text{sine values} \tag{4-22}$$

$$i = \text{address 0 to 999 to ROM} \tag{4-23}$$

The address's 100s digits are multiplied by 256, the 10s by 16, and the 1s stay as they are. Therefore, the address equals:

$$J_1 = \text{int}\ \frac{i}{100} \tag{4-24}$$

$$J_2 = \text{int}\ \frac{i - 100J_1}{10} \tag{4-25}$$

$$J_3 = i - 100J_1 - 10J_2 \tag{4-26}$$

$$\text{Address} = 256J_1 + 16J_2 + J_3 \tag{4-27}$$

For example, for $i = 125$ decimal,

$$J_1 = 1 \qquad J_2 = 2 \qquad J_3 = 5 \tag{4-28}$$

and the address is 256 + 32 + 5 = 293 hexadecimal.

Note that starting in address 000 (hexadecimal), after 10 clock ticks, the accumulator produces the output 010 (hexadecimal), which is 10 (ten) in BCD but 16 (decimal weight) in hexadecimal. Since all memory devices are binary, many addresses are never accessed and therefore are wasted. As can be seen, in this application approximately 75 percent of the memory is never used,

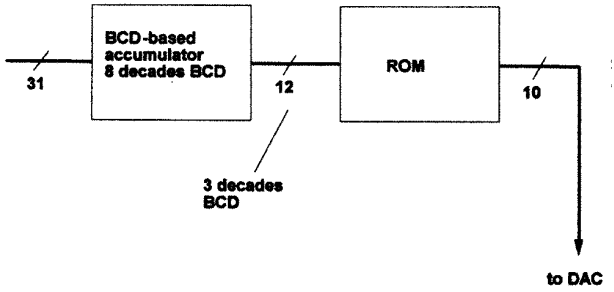


Figure 4-26 Diagram of a specific BCD direct digital synthesizer (Sciteq Electronics DDS3).

NT (time)	Phase (degrees)
0	0
1	36
2	72
3	108
4	144
5	180
6	216
7	252
8	288
9	324
10	0

Figure 4-27 Decimal DDS phase increments; $F_{out} = F_{ck}/10$.

because it is never accessed. (Here 12 input bits that generate 4096 binary addresses generate only 1000 BCD states.)

Remember, in this structure all parameters are weighted BCD. If an adder is added between the accumulator and the ROM, as will be discussed later, to perform phase modulation, the phase increments will be BCD, too. This means that the order will not be 180°, 90°, 45°, 22.5°, , as in the binary case, but 288°, 144°, 72°, 36°. The bit weight is calculated as follows: The MSB has a weight of 8 and therefore generates

$$\frac{360 \cdot 8}{10} = 288° \tag{4-29}$$

The second bit generates

$$\frac{360 \cdot 4}{10} = 144° \tag{4-30}$$

More is said about this later in the chapter.

So BCD direct digital synthesizer implementations are mainly used when decimation is important in applications; when a manual control is applied to the direct digital synthesizer (rather than a computer) and such a user cannot calculate in binary or hexadecimal; and in instrumentation. Many instrument makers (Wavetek, PTS, Sciteq, Comstron) have adapted this technique and couple it with PLL or direct digital synthesis in a variety of standard laboratory instruments and OEM parts.

For more on this topic see Chaps. 6 and 7.

4-5 Randomization

We have mentioned before that the main drawback of DDS technology is its spurious signal performance. It was already demonstrated that the spurious signals are generated as a result of the periodicity of the quantization errors. Another source of spurious signals is the intermodulation effects in the DAC by mixing signals, their harmonics, aliasing, and the clock signal. Over the last few years spurious signals have become the only drawback of DDS technology, as the other traditional weakness—limited bandwidth—has been overcome and DDS products can cover hundreds of megahertz of bandwidth.

Quantization effects, which are the source of DDS errors, have a much lower contribution to spurious signals in hardware DDS products where the DAC controls performance.

To eliminate the spurious signal periodicities, which affect both the quantization errors and the DAC artifacts (usually caused by intermodulations in the device and effects of its nonlinearity, both static and dynamic), a few methods have been devised. All these methods try to convert the spurious signal energy to evenly distributed noise by randomization and dithering. Because quantization always introduces a level of error and because this error cannot be totally eliminated, the only effect we can hope for is that discrete spectral lines (spurious signals) can be distributed and

converted to "noiselike"; therefore convert discrete spectral lines to phase noise. The randomization therefore attempts to destroy the periodicity of the quantization effect.

An optimal procedure will just do that and will not add any more noise beyond the energy in the spurious signal components. Other nonoptimum procedures add noise energy. All the methods derived so far use a randomization procedure.

4-5-1 Wheatley procedure

Wheatley (Ref. 17) has demonstrated a randomizing technique as shown in Fig. 4-28. The device consists of an accumulator and some DSP and produces a square wave (not a sine). At each overflow of the accumulator, a random number X (K vector) is added to the contents of the accumulator, $0 < X < K - 1$. At the same time, the previous value of X is subtracted. The subtraction of X is performed so that the average of $X(i) - X(i - 1)$ becomes 0, and the average output frequency does not change from the one that was commanded. Thus a random number is added on each overflow with the range $-(K - 1)/2 < X < (K - 1)/2$. (The same could be achieved by adding and subtracting alternatively.) Usually, a *pseudorandom number* (*PRN*) sequence generates X. The PRN sequences are widely used in communications, coding, and scrambling; see Ref. 8.

To analyze the result, the normal phase (time) error, caused by the discrete nature of the synthesizer (see Fig. 4-10), is designated

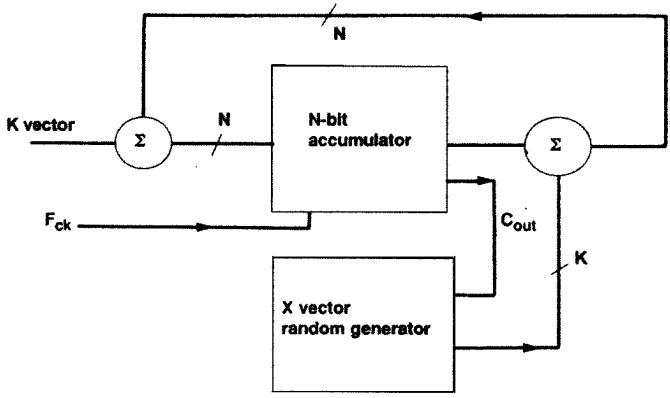


Figure 4-28 The Wheatley procedure. (1) $0 < X < K, p(x) = 1/K$; (2) add new X per new C_{out}; (3) subtract old X per new C_{out}.

as e_r, and $e_r < K$. So the probabilities of $p(e)$ and $p(T - e)$—the error will be either e_r or $T - e_r$, where T is the clock cycle time—are

$$p(e_r) = \frac{e_r}{K} \tag{4-31}$$

$$p(T - e_r) = 1 - \frac{e_r}{K} \tag{4-32}$$

The expected (average) value of the delay error τ equals

$$\tau_{av} = \tau p(e_r) + (\tau - T)p(T - e_r) = 0 \tag{4-33}$$

This is quite intuitive since the procedure of subtracting X from its previous value generates a random number whose average equals 0. This is necessary since X is always a positive number. (Obviously the average of $X_i - X_j$ is zero.)

Wheatley went further to prove that by following this procedure all the spurious signal energy is converted to phase noise. The noise spectrum power density is given by

$$\frac{N_0}{C}\bigg|_{3/4} = \frac{\pi^2 f}{3F_{ck}^2} \quad \text{dBC/Hz} \tag{4-34}$$

f = output frequency

where F_{ck} is the clock frequency. The novelty of the Wheatley procedure lies in the principle; however, the technique is not easy to implement, especially in high-speed logic, and its practical results are limited. Obviously this technique attempts to generate an all-digital direct digital synthesizer, and as a consequence, the price for reasonable performance is a very large value of clock per usable bandwidth (or F_{ck}/w), so the output frequency range is limited.

Wheatley's procedure is an extremely innovative approach to spurious signal cancellation, and studying it can bring great insight to the nature of the problem.

A similar procedure can be followed when a complete direct digital synthesizer is used and operates on the digital word controlling the DAC rather than only the carry output bit.

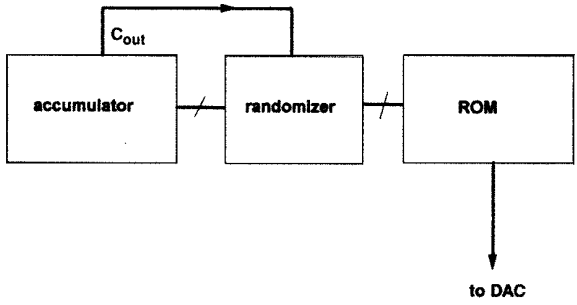


Figure 4-29 Randomizing accumulator output.

4-5-2 Randomizing sine output

This method uses a complete direct digital synthesizer including the DAC and randomizes the output of the accumulator (Ref. 24; Fig. 4-29). The difficulty here is that a DAC is employed in the circuit, and some spurious signals generated inside the DAC have nothing to do with the calculated digital quantization spurious signals, since they are generated by the DAC nonlinearities and imperfections. For example, because the DAC is an analog device with finite dynamic range, the output signal and its harmonics intermodulate with the clock signal that always leaks to the analog summing node that represents the output. Remember that the DAC generates a multitude of frequencies—the desired plus all the aliasing—and the clock is always present in its analog portion.

Our experience has been that this technique did not provide substantial improvements, since many DAC spurious signals are not affected, and they can be the dominant spurious signals.

Another attempt was demonstrated in Ref. 24. See Fig. 4-30. This figure is clearly an extension of the Wheatley procedure and again shows improvements in quantization spurious signals but not in those derived from the analog component. Figure 4-30 is a

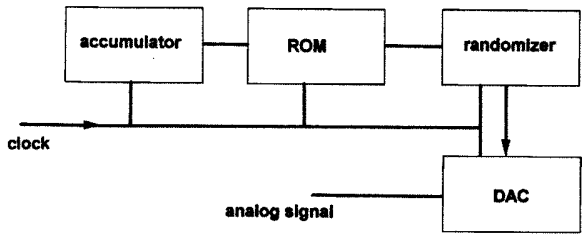


Figure 4-30 ROM output randomization.

practical simplification of the randomization procedure (see Ref. 24). Noise is added to spread the digital errors and has a marginal effect on the analog-derived spurious signals.

As was mentioned, the majority of spurious signals in standard DDS designs are analog anyhow and are generated in the DAC. However, when we come back to the issue of a single-bit direct digital synthesizer later in the chapter, the Wheatley procedure is worth remembering.

4-6 Quantization Errors

All DDS applications suffer some quantization since DDS is a DSP discipline and finite arithmetic is used. A survey will be presented to quantify these errors.

At the outset we should mention that the evolution of digital techniques and ASIC levels of integration allows the implementation of very accurate DDS. Most designs produce −75 to −80 dB of worst-case spurious signals (in the digital domain; i.e., if the output of the ROM is analyzed by a computer or a fast Fourier transform analyzer, the spurious signals will be found at these remarkable levels). And the problems and performance degradation occur only when the signal is converted to analog form in the DAC.

4-6-1 Digitized model

Some serious theoretical work has been done by Nicholas, Samueli, and Cercas on mathematical models for the evaluation of spurious signals due to quantization. A mathematical model of a direct digital synthesizer is shown in Fig. 4-31; see Refs. 3, 6, 7, and 12.

The first source of noise $P_1(n)$ is introduced because of the truncation in the input to the ROM. This takes into consideration the quantization effect in the accumulator and the fact that only part of the accumulator bits is connected to the ROM.

If the accumulator is composed of N bits and only W bits are connected to the ROM, then the accumulator can be considered as a device that generates only W integer digits and $N - W = F$ fraction bits. All the numbers that are generated in the lower F bits do not affect the output directly and manifest themselves only through their propagation to the W higher bits. This is a major

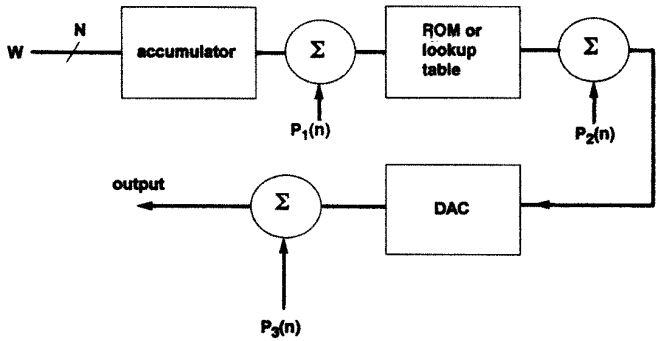


Figure 4-31 DDS quantization model.

source of error generation, since there are a great many phases that are multiples of $2\pi/2^N$ which are generated in the accumulator but for which there is no corresponding storage in the ROM. Clearly a quantization error occurs.

The periodicity of this error determines the position of the spurious signal that it will generate. Obviously, by the nature of the operation of the accumulator, the error signal is periodic (no signal can generate a periodicity lower than F_{ck}/ACM). In fact, all operations are periodic and thus deterministic.

Table 4-4 shows how to determine these periods as a function of *F, N*, and the input control word *W*. Similar results have been obtained by Cercas (see Ref. 6).

As can be seen (Fig. 4-31) for the word size *F* (the word size used to control the ROM), only frequencies for which only *F* bits are controlled do not generate phase quantization (in this arithmetic). Their number is negligible relative to the total number of frequencies. Chapter 8 deals with this issue is greater detail.

In the case where $N = 32$ (and $K = 14$), *K* bits connect the accumulator to the ROM. The total number of generated frequencies (up to the Nyquist frequency) is 2^{31}, or approximately 10^9. The number of frequencies that do not generate phase quantization (in 14-bit arithmetic) is only 2^{13}.

Although these frequencies do not generate phase quantization in the direct digital synthesizer arithmetic, there is always a phase quantization error since the real value of the phase is represented by a finite bit word and these frequencies will generate spurs, too. Only these control frequencies which generate a signal

that is a divider of ACM—a total of $N - 1$ in the case of a binary accumulator—are free of spurious signals. Since 2^N divides by only 2^G ($G \leq N$), only frequencies that control only 1 bit (i.e., control inputs of weight $W = 2^G$) are free of spurious signals. Any other frequency creates a residual phase that propagates in the accumulator and generates a new cycle that will be expressed as a spurious signal.

Even the $W = 2^G$ controlled signals are not yet perfect sine waves, and their quantization causes harmonics. However these are the only spurious signals generated.

As a demonstration, let's pick $N = 4$ and $W = 2$. In this case, after 8 clocks, the accumulator will reach the exact state 2^N (2π). Thus the output frequency will be $F_{ck}/8$, and the accumulator states will be

$$0, 2, 4, 6, 8, 10, 12, 14, 0, \ldots$$

However, if $W = 6$, then the cycle will be

$$0, 6, 12, 2, 8, 14, 4, 10, 0, \ldots$$

In this case, 3 cycles of the fundamental output frequency (of frequency $\frac{6}{16} F_{ck}$) were needed to come back to the original state, and this will create spurs at one-third (and its harmonics) of the output frequency given by $6F_{ck}/2^4$.

This can be formulated in the following way: The phase accumulator performs the operation of a digital integrator followed by a modulo 2^N operator. While the accumulator generates its main output, given by $W/2^N$ (for a normalized clock), there is another periodicity (and generally periodicities) being generated, denoted by P, the one for which the phase relation $\varphi(i) = \varphi(i + P)$ for all i holds. While the main output frequency is given by

$$\frac{2^N}{WT} \qquad T \text{ is the clock time} \tag{4-35}$$

the other periodicity will be determined by

$$\frac{2^N}{\gcd(W, 2^N)} \tag{4-36}$$

where $\gcd(a, b)$ = greatest common divisor of a and b.

Only $W = 2^G$ ($G < N$) generates a solution

$$\gcd(2^N, W) = W$$

However, for the majority of W, this is not the case, and the secondary periodicity will manifest itself as a spurious signal. In the case of $N = 4$ and $W = 6$,

$$\frac{2^4}{\gcd(16, 6)} = \frac{16}{2} = 8 \tag{4-37}$$

Therefore, the normalized main output frequency is (16/6) T = 2.6666T. The period of the secondary term is $P = 8T$. The spectrum of the digital sine, at the output of the ROM, assuming a perfect ROM, will now consist of a cardinal periodicity of 2.666$\cdots T$ (clock ticks) but also one that is of 3 signal cycles, or 8 clock ticks. The ratio between them is 8/2.666$\cdots$ = 3. Therefore, a spurious signal at the fundamental $\pm$⅓ (and its harmonics) will exist (see Fig. 4-73).

This, of course, is the consequence of dealing in the sampled (finite accuracy data) and not in the continuum domain. The ratio of 2.666$\cdots$/8 = 1/3 is an indication of the location of the spurious signal relative to the main output. For F_{ck} = 20 MHz, $N = 4$, and control of $W = 6$, the DAC generates a fundamental output of 20 · 6/16 = 7.5 MHz and spurious signals at $\pm$2.5 MHz (20/8 MHz) away from the fundamental (see the figure).

Generally, Nicholas (Ref. 3), and Cercas (Ref. 6) showed that a procedure for finding the location and the size of the spurious sig-

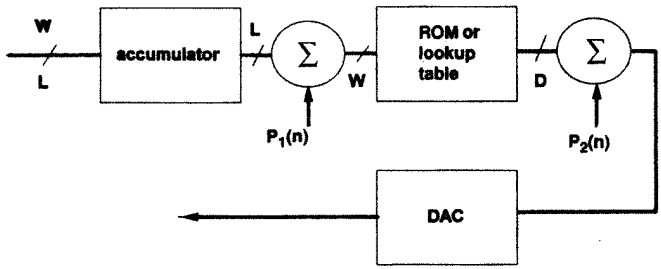


Figure 4-32 DDS noise model. L = accumulator size; W = ROM input size; D = ROM output size; W = input control, $< 2^{L-1}$; $B = L - W$.

nals caused by phase quantization to the ROM can be calculated as follows (the noise model is shown in Fig. 4-32):

$$N = \text{accumulator bit size, binary} \tag{4-38}$$

$$W = \text{number of bits connected to ROM} \tag{4-39}$$

$$W = \text{control word} \tag{4-40}$$

$$N - W = B \tag{4-41}$$

$$\begin{gathered} F_{\text{ck}} = \text{clock frequency} \\ T = \frac{1}{F_{\text{ck}}} \qquad \text{(clock tick duration)} \end{gathered} \tag{4-42}$$

The output spectral lines, of which only one is the desired signal (and therefore the rest are spurious), will show at all periodicities that are multiples of

$$\frac{T \cdot 2^N}{\gcd(W, 2^N)} \tag{4-43}$$

This is the cardinal relation for locating spurious signals.

However, Nicholas and Cercas went further to calculate the quantization errors (digital, assuming the use of a perfect DAC) caused by the truncation of the accumulator:

$$\text{Number of spurs} = \frac{2^{B-1}}{(W, 2^B)} \tag{4-44}$$

$$\text{Location: equally spaced between 0 and } \frac{F_{\text{ck}}}{2(W, 2^B)} \tag{4-45}$$

They also approximated the spurious signal level.

Note that the desired output frequency periodicity is given by

$$\frac{T \cdot 2^N}{W} \tag{4-46}$$

and is one of the frequencies included in the term in (4-43), because by definition

$$W = K(W, 2^N) \qquad K \text{ an integer} \tag{4-47}$$

The general simple approximations for amplitude modulation (AM) and *phase modulation* (*PM*) spurious signals are given by

$$\text{PM spurious signals} \approx 10 \log \frac{(\pi \cdot 2^{-W})^2}{3} \quad \text{dBC} \qquad (4\text{-}48a)$$

$$\text{AM spurious signals} \approx 10 \log \frac{2^{-2L}}{6} \quad \text{dBC} \qquad (4\text{-}48b)$$

where L is the number of DAC bits.

The ROM in most cases uses some compression algorithm which introduces errors on top of its quantization. These errors are designated $P_2(n)$ in Fig. 4-28, and $P_3(n)$ is the analog inaccuracy of the DAC including its quantization effects. This model mainly follows Ref. 3.

The locations of the spurious signals are therefore determined by only the accumulator and the control word, but their relative strength is affected by the ROM and DAC, too. Spurious performance is shown in Fig. 4-33.

The simplest evaluation for the calculation of total quantization error (one used often in DSP) starts by assuming that there are

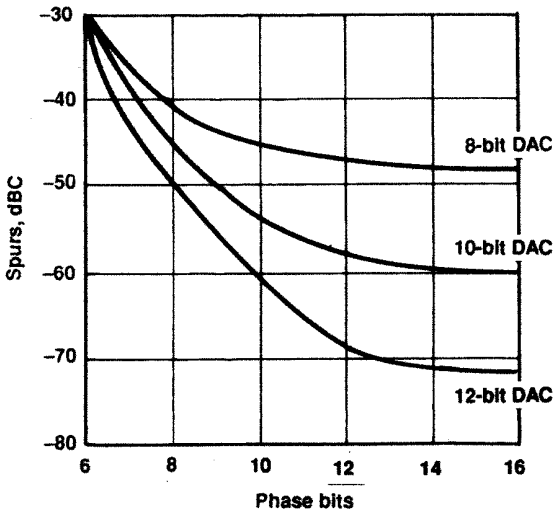


Figure 4-33 Theoretical spurious signal response (Ref. 2).

no errors introduced by any of the elements and that the only source of error lies in the quantization process, in the DAC. In this case, if the DAC is represented by D bits, the integrated signal to quantization noise is easily calculated as follows: The sine wave has a peak-to-peak amplitude of 2^D, and therefore its power is given by

$$\left(\frac{2^D}{2\sqrt{2}}\right)^2 = 2^{2D-3} \tag{4-49}$$

The quantization error is assumed to be random and equally distributed from -0.5 to 0.5. The quantization noise power is thus given by

$$\mathrm{Er} = \int_{-0.5}^{0.5} x^2 p(x)^2 \, dx \tag{4-50}$$

and $p(x)$ is the probability density function of the error and is assumed to be flat between -0.5 to 0.5. The value of the error power is therefore Er = 1/12. Thus, the integrated *signal-to-noise ratio* (*SNR*) of an ideal direct digital synthesizer would be

$$\frac{2^{D-3}}{\mathrm{Er}} = 1.5 \cdot 2^{2D} \tag{4-51}$$

$$= 6D + 1.78 \qquad \text{dB} \tag{4-52}$$

which is a fundamental ratio in DSP theory. The performance for different values of D is shown in Fig. 4-34 and in Table 4-1, and it is basically 6 dB/bit.

This noise includes, of course, all error signals at the ideal direct digital synthesizer output. There was an assumption that the noise is random and evenly distributed, and for large D this is justified, although from the pure mathematical standpoint it is only an approximation, since nothing is random in this direct digital synthesizer and all errors can be calculated.

However, a further approximation can be made. If the errors were random, then the quantization noise would manifest itself as white noise, equally distributed across the operational BW of the direct digital synthesizer. One should expect a $(\sin x)/x$ noise spectrum, but once again as an approximation we can assume that the noise is white and occupies a BW of $1/T$, where T is the direct dig-

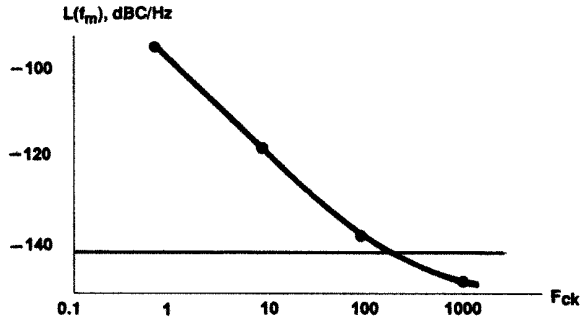


Figure 4-34 Quantization noise floor including crystal noise: $-(6D + 1.78 + 10 \log F_{ck})$ dBC/Hz; $D = 8$.

TABLE 4-1 Theoretical Signal to Quantization Noise Ratio as Function of Number of Bits *N*

N	Signal/quantization (dB)
6	37.6
7	43.6
8	49.6
9	55.6
10	61.6
12	73.6
14	85.6
16	97.6
18	109.6
20	121.6
24	145.6

ital synthesizer clock rate. (This is approximating a flat spectrum from –1/2T to 1/2T.)

The noise power density N_0 can therefore be approximated as

$$\frac{C}{N_0} = 6D + 1.78 + 10 \log \frac{1}{T} \tag{4-53}$$

The noise floor performance for a few direct digital synthesizer parameters is shown in Fig. 4-35. This approximation calculates the quantization noise density. In a real direct digital synthesizer, the noise of the clock should be added, and some typical numbers are shown in Fig. 4-34. State-of-the-art (as of mid-1999) phase noise obtained from crystal oscillators will be discussed in Chap. 9.

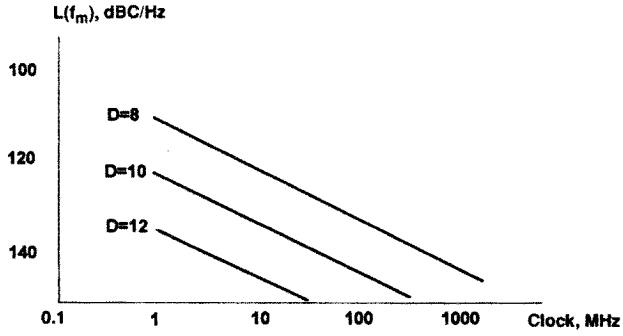


Figure 4-35 Quantization noise floor: $-(6D + 1.78 + 10 \log F_{ck})$ dBC/Hz.

The approximation above is quite crude, but it offers a general rule of thumb. Not much has been published regarding exact calculations or closed form analytical expressions for the evaluation of direct digital synthesizer phase noise performance. Most of the analytical work presented above deals with the quantization errors in the accumulator. Since ROMs use compression techniques and introduce errors, there cannot be a close-form formulation for the spurious signals that they add. So designers must resort to exhaustive fast Fourier transform (FFT) runs and hardware testing. Measurements are performed that show the approximation $-6D - 1.78 - 10 \log(1/T)$ to be within ± 2 dB.

The DAC is still a part with rather poor modeling for direct digital synthesizer applications, and no alternative to hardware testing exists yet. Many DACs generating "poor" time-domain response prove very useful for DDS applications, and others that seem to have a "nice" time response generate disappointing spurious signal performance. It is clear that parameters such as linearity, symmetry, and speed are cardinal but do not reveal the complete picture. This is an area that requires work and time for understanding and modeling.

A few interesting observations can be made. First, for practical applications, W (number of bits at ROM input) should be bigger than or equal to $D + 2$ (D = number of ROM output bits). As a rule of thumb, and in order to provide good performance and economics, generally designers use $W = D + 2$ or $W = D + 3$.

It is easy to gain an intuitive insight for this requirement by just looking at some numbers. Bear in mind that the sine is a nonlin-

ear function, and as a consequence dynamic range is lost in the mapping. The price paid is the increase in the number of input bits to the ROM. For example, for an 8-bit by 8-bit ROM, the mapping will be given by

$$\sin(n) = \text{int}\left(127 \sin \frac{n \cdot 2\pi}{256} + 0.5\right) + 128 \qquad (4\text{-}54)$$

Viewing the output of the first 64 values (the first quadrant) shows that many addresses of the DAC will never be accessed; see Table 4-2.

Obviously, there is loss of information, and arbitrary quantization has been introduced that yields error (spurious signals). However, running the same program with a 10-bit by 8-bit ROM gives the mapping as

$$\sin(n) = \text{int}\left(127 \sin \frac{n \cdot 2\pi}{1024} + 0.5\right) + 128 \qquad (4\text{-}55)$$

The result is shown in Table 4-2.

Now, every address of the DAC is accessed, and every possible output is being generated without skipping values. This eliminates deliberate and arbitrary quantization shown in Table 4-2.

It is interesting to compare the last run with a 12-bit by 8-bit ROM. Since this ROM is 4 times as dense, the values repeat those of the previous one if we look at every fourth address; however, the added accuracy is demonstrated in the "fine-tuning" in between. The cost is the ROM complexity, and since in most direct digital synthesizer applications the spurious signal performance is controlled by the DAC anyhow, the rule of thumb $W = D + 2$ is very practical.

Another interesting observation is that the digital worst-case spurious signal approaches asymptotically the value 2^{-W} at the output of the accumulator. Note that this is not the spurious signal level at the output of the ROM, since this value must depend on D. However, it puts an upper limit on the performance of the direct digital synthesizer when W is given, by assuming that the ROM is of finite accuracy. Both these numbers are in the digital domain and, as mentioned before, do not include the DAC errors.

As a practical matter (and a rule of thumb), for a direct digital synthesizer using D larger than 7 (D is the number of DAC bits),

TABLE 4-2 Digital Samples for Mapping 10 Bits into 8 and 8 Bits into 8

10 bits to 8															
128	128	129	130	131	131	132	133	134	135	135	136	137	138	138	139
140	141	141	142	143	144	145	145	146	147	148	148	149	150	151	152
152	153	154	155	155	156	157	158	158	159	160	161	161	162	163	164
164	165	166	167	167	168	169	170	170	171	172	172	173	174	175	175
176	177	178	178	179	180	180	181	182	183	183	184	185	185	186	187
187	188	189	189	190	191	191	192	193	193	194	195	195	196	197	197
198	199	199	200	201	201	202	203	203	204	204	205	206	306	207	207
208	209	209	210	210	211	212	212	213	213	214	215	215	216	216	217
217	218	218	219	219	220	221	221	222	222	223	223	224	224	225	225
226	226	227	227	228	228	229	229	230	230	230	231	231	232	232	233
233	234	234	234	235	235	236	236	236	237	237	238	238	238	239	239
240	240	240	241	241	241	242	242	242	243	243	243	244	244	244	245
245	245	245	246	246	246	247	247	247	247	248	248	248	248	249	249
249	249	249	250	250	250	250	251	251	251	251	251	251	252	252	252
252	252	252	252	253	253	253	253	253	253	253	253	254	254	254	254
254	254	254	254	254	254	254	254	254	254	254	254	254	254	254	254

8 bits to 8															
128	131	134	137	140	143	146	149	152	155	158	161	164	167	170	173
176	179	182	185	187	190	193	195	198	201	203	206	208	210	213	215
217	219	222	224	226	228	230	231	233	235	236	238	240	241	242	244
245	246	247	248	249	250	251	251	252	253	253	254	254	254	254	254
254															

the output spurious signal for careful designs, using good grounding techniques and very high-performance DACs, is given by 2^{-2D}, or $-6D$ dB (see Fig. 4-33). In spite of many claims by many manufacturers, at the modest 20-MHz clock speed, we have not seen yet any direct digital synthesizer achieving better than -60 dB using 10-bit DACs and not one that can achieve -72-dB spurious signal using 12-bit DACs!

The addition of a sample-and-hold device to "deglitch" the DAC output helps clean the wideband spurious signals but not the close-in spurious signals generated by the DAC nonlinearities (see Figs. 4-69 to 4-73). More is mentioned on this topic in Chap. 8.

4-7 Logic Speed Considerations

There is an obvious and constant requirement to increase the direct digital synthesizer speed and its performance as much as possible. A wider-range direct digital synthesizer can be divided down to improve spurious signals and still maintain workable bandwidth; and as a building block for complex synthesizers, it can cut total system complexity, size, and cost.

The speed of a direct digital synthesizer is controlled by the speed of the logic, including the arithmetic operations, the memory, and the DAC. Although accumulators can operate up to 2 GHz, using pipeline techniques (see Chap. 6) and DACs can operate up to 2 GHz in speed, the memory is usually the slow item in the chain. Traditionally, DAC technology is faster than direct digital synthesizer logic by a margin of 2 to 3 to 1. Hence multiplexing techniques can be developed to increase the logic speed, at the cost of complexity, so the full speed of the DAC can be utilized (see Ref. 10). Multiplexing techniques are also demonstrated in Chap. 8.

4-8 Modulation

In DDS all parameters are generated digitally and are therefore easy to manipulate. That means that very accurate and fast modulation is easy to implement.

A complete descriptive block diagram is shown in Fig. 4-36. The accumulator input is the frequency control port. Since the direct digital synthesizer is switching very fast and is phase-continuous, it is easy to change the frequency and generate frequency modu-

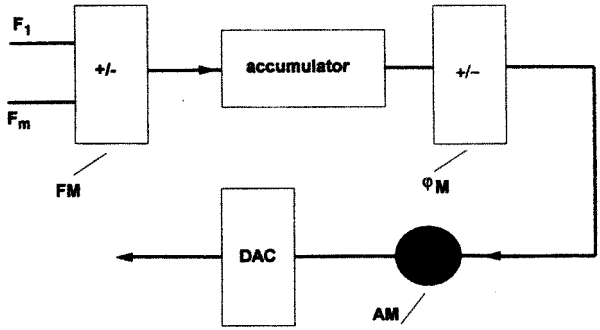


Figure 4-36 DDS modulation, amplitude, phase, and frequency.

lation (FM). For example, if the FM control has an adder, then one of the inputs to the adder F_1 can be the carrier frequency, F_2 can be a group of frequency deviations, and either FSK (frequency shift keying) or M-FSK (multiple FSK) signals can be generated easily. Note that unlike the case of FSK generated by controlling a phase-locked VCO, in this case all the generated frequencies are fully synthesized. A typical plot is shown in Fig. 4-37. (See Fig. 4-57.) The output frequency will be given by the control $F_1 + F_2$.

Another interesting application is to put an accumulator in front of the accumulator, as shown in Fig. 4-38. Since the accumulator performs an integration function, the phase output will result in a quadratic function (since we use two accumulators), and the result will be a linear FM or chirp signal whose instantaneous frequency is shown in Figs. 4-38 and 4-39. This waveform is useful for test equipment, when swept response is important to evaluate components' frequency response, and in radars and spread-spectrum communications, where chirp signals are used to spread the signal energy across a large bandwidth and reduce the

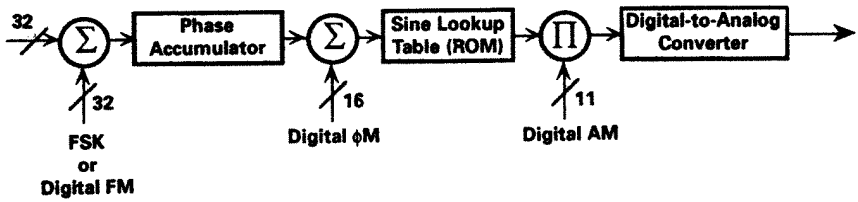


Figure 4-37 DDS digital modulation: phase, frequency, and amplitude.

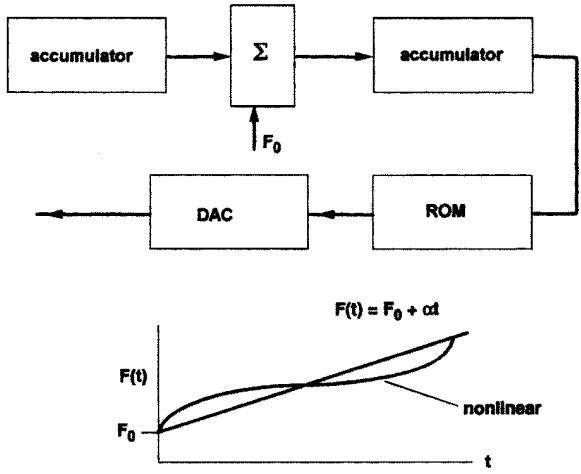


Figure 4-38 Linear and nonlinear FM in DDS.

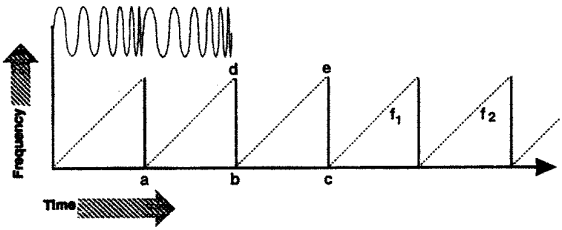


Figure 4-39 Linear FM (chirp).

spectral density, for antijamming, for covert communications, to reduce the effects of multipath and fading, and for electronic warfare. There are growing applications for this signal in synthetic aperture radar (SAR) and imaging radars, for surveillance, all-weather landing, and agricultural surveys.

Sometimes a nonlinear chirp is desirable for a variety of reasons, and this is easy to implement by replacing the first accumulator with a RAM (or any other control mechanism) that can be loaded by software and changing the signal characteristics as required (Fig. 4-40); the RAM changes control depending on the frequency output (8 MSBs).

Chirp synthesizers are described in greater detail in App. 4A; also see Secs. 4-13 and 4-14.

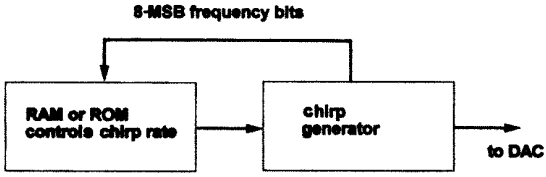


Figure 4-40 Linear and nonlinear chirp generation, using frequency output control rate change.

Note that unlike PLL circuits, in DDS applications (and only in DDS signal generation) it is possible to perform dc FM, which means that the modulating signal can have spectra all the way down to direct current, which is not possible in PLL circuits. Direct digital synthesizer FM circuits are gaining popularity in many applications, ranging from cellular telephony, to amateur radio applications (used as BFO), all the way to very complex electronic warfare systems.

Very fast modulation, synthesis accuracy, and small size give it the edge over conventional VCO applications.

Phase modulation is easy to implement. (See Fig. 4-41.) Since the output of the accumulator represents the phase of the signal, putting an adder or subtractor between the accumulator and the ROM performs phase modulation, adding or subtracting.

Since the MSB represents 180°, the MSB − 1 represents 90°, the next one 45°, etc. This will be the modulating signal phase weight. Thus an 8-bit adder provides 360/256 = 1.4° phase resolution, and a 12-bit adder provides better than 0.1° of phase resolution.

Note that the digital accuracy of the modulation is perfect and that the analog accuracy depends on the accuracy of the DAC. FSK is shown in Fig. 4-42, AM in Fig. 4-43, and SSB in Figs. 4-44 and 4-45.

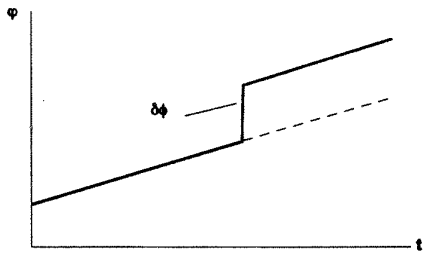


Figure 4-41 Phase modulation in DDS.

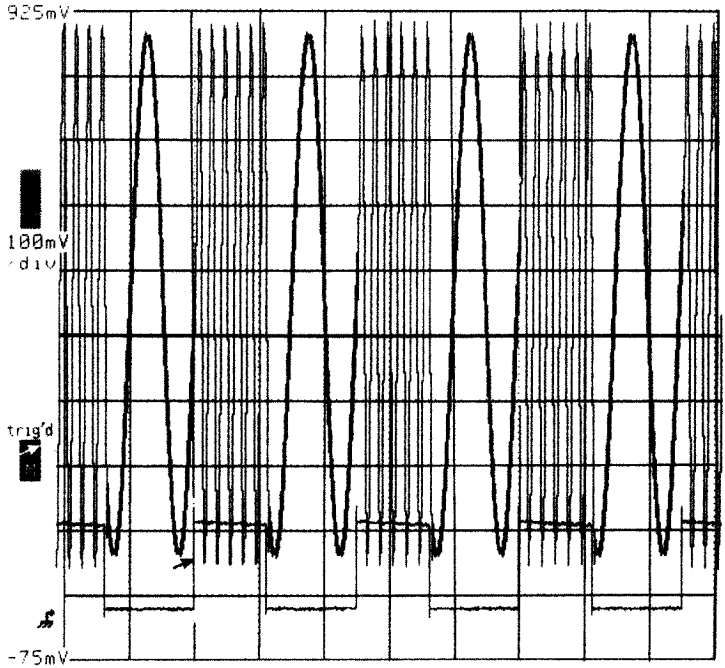


Figure 4-42 DDS1 FSK.

An obvious consequence is the fact that the quadrature sine waves sin ωt and cos ωt are very simple to implement in a direct digital synthesizer, as shown in Fig. 4-46. Such an implementation is much more accurate than any analog technique. Generation of phase shifting at great accuracies is important in almost all applications for receivers where the received signals are converted downward to baseband for digital processing and for a variety of modulations.

As already mentioned above, when BCD accumulation is used (Figs. 4-26 and 4-27), the phase weight of the bits is now different from that of the binary architecture. In the binary case, the phase weights are binary weighted with the MSB being 180°. In a BCD implementation, the MSB digit, 8, has a weight of $360 \cdot 8/10 = 288°$. The next three are 144°, 72°, and 36°. The next will be the 8 weight of the second BCD digit and will be 28.8°. Sometimes, in BCD designs, the top MSBs are binary. As mentioned above, a BCD direct digital synthesizer that runs using a 40-MHz clock can use the 2 MSBs as binary bits, since the count to 4 is required. In this case,

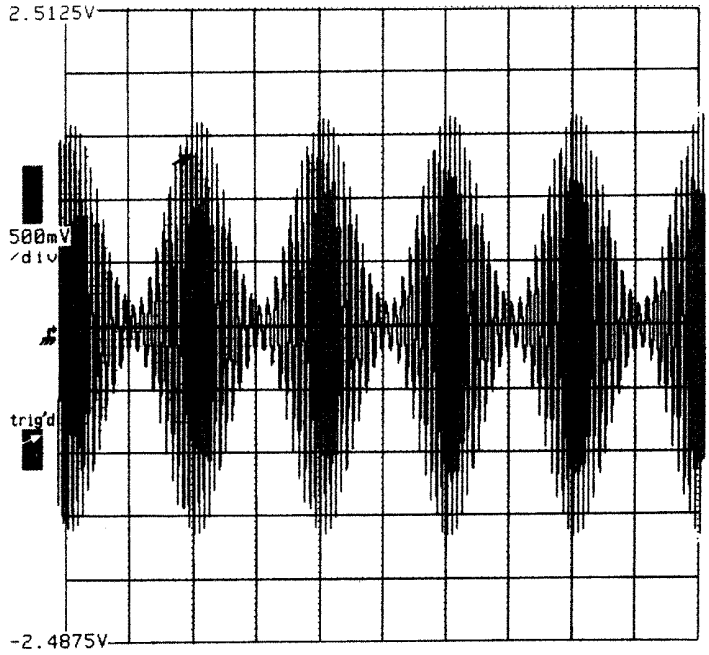


Figure 4-43 DDS1, AM 80 percent.

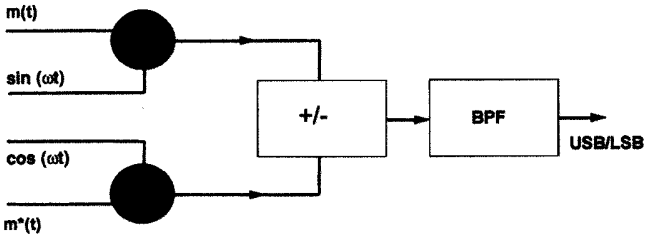


Figure 4-44 SSB modulator.

the MSBs will have a weight of 180, 90° accordingly, but the following bits, being BCD, will assume the values 72°, 36°, 18°, and 9°.

Amplitude modulation is a little more complex to achieve, but again, compared to the analog technique, it is very accurate, low-cost, and small in size (see Figs. 4-19, 4-37, and 4-43).

Since the output of the ROM represents the amplitude of the sine wave, a multiplier between the ROM and the DAC will enable amplitude modulation. Again, since all operations are performed

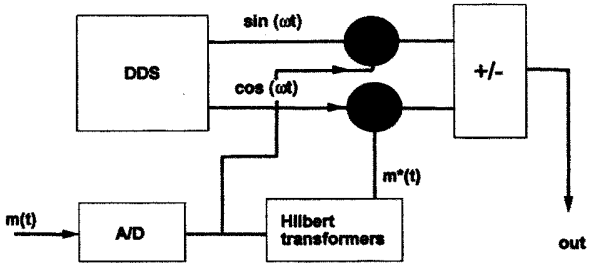


Figure 4-45 All-digital SSB modulation.

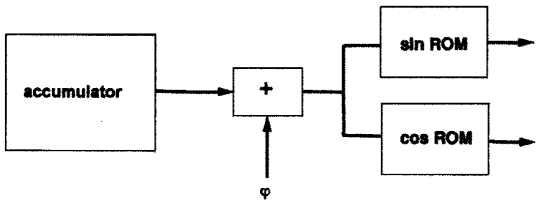


Figure 4-46 Digital quadrature modulation in DDS.

digitally, the accuracy of the AM depends only on the accuracy of the DAC. A 12-bit multiplier generates ideally 72 dB of dynamic range and for practical purposes better than 60 dB.

A time-domain plot of a typical modulator is shown in Fig. 4-43. Note that this implementation represents a first-quadrant multiplier, since all numbers are positive only. This means that AM is possible but *double sideband* (*DSB*) modulation is not, since negative numbers cannot be generated. To turn the multiplier into a four-quadrant device, it is necessary to digitally decode the two MSB digits of the multiplying signals and reverse the phase.

This operation is easier to implement when the numbers are represented in 2's complement rather than binary.

Obviously 1 bit or 6 dB of dynamic range is lost.

However, this implementation and the fact that digital multipliers of 16-bit words are fast, low-power, and low-cost present an opportunity to implement complex modulations either for digital communications or for analog ones like *single sideband* (*SSB*), digitally, with excellent results. Traditionally, SSB modulation is performed by using analog multipliers and crystal filters, as shown in Fig. 4-44.

TABLE 4-3 Spurious Level and Location in DDS

For N accumulator size:
M = number of bits to ROM
F_r = output frequency
F_{ck} = clock speed

$C_1 = F_r/\gcd(F_r, 2^{(N-M)})$
$C_2 = 2^{(N-M-1)}/\gcd(F_r, 2^{(N-M)})$

Spur frequency: $F_{spur} = F_n \cdot \gcd(F_r, 2^L) F_{ck}/2^N$

Spur magnitude:
For $K = (F_n - C_1)C_1^{(C_2-1)}/2^N$ modulo $2C_2$

Spur magnitude: $2^{-N}\csc(K/2C_2)/2C_2$

$\gcd(a,b)$ = greatest common divisor of a and b

After Cercas, Samueli, and Nicholas.

In the analog execution, the carrier leakage and the accuracy of the multipliers yield rejection of the carrier $\omega_0 t$ and the undesired sideband by no more than 25 to 30 dB. This is not sufficient in many applications such as telephony and radio. Because of the inherent accuracy of the digital implementation, 50- to 60-dB rejection can be realized, across a wide range of temperatures (Fig. 4-45).

A useful analog amplitude modulation mechanism used in direct digital synthesizer technology has analog form, by modulating the dc voltage driving the DAC reference voltage. Since most DACs are the multiplying type, i.e., the output current is proportional to the reference voltage, a level of AM can be achieved. This is done mainly to cover very fine amplitude steps (say, 0.1 dB), by driving the reference voltage by the low-speed DAC, and change its output voltage by a digital control. Very accurate results can be achieved if a range of 0 to 10 dB is used.

Another important modulation that is easy to implement in direct digital synthesizer is pulse modulation. By resetting the output digital signals to the DAC, the output is completely eliminated. In this case the on/off switch quality is such that the attenuation is infinite since the signal is just not there at all.

There are two types of pulse modulation in a direct digital synthesizer. The one, as mentioned above, resets the whole logic part. This will cause the direct digital synthesizer to be in a known state, denoted as zero phase. When the reset is released, the signal will start to generate its waveform from the same initial phase.

A second type of reset is to reset only the digital data at the input to the DAC. In this case, the whole digital part continues to run as usual, but all the input bits to the DAC are 0. In this application, when the reset is canceled, the signal will continue at a phase point where it would be if it had never stopped. The state of the phase is important in many applications such as pulse radar and electronic warfare. This is sometimes referred to as *phase memory.*

Almost all direct digital synthesizer chips or cards offered in the market today have some type of modulation capabilities, as demonstrated in Sec. 4-9.

4-9 State-of-the-Art Components and Systems

The following paragraphs present some of the advanced DDS devices on the market and describe their internal structures.

4-9-1 Very high-speed direct digital synthesizer

For applications requiring very high speeds, there are at least three sources for DDS products. The most advanced device is the SP-2002 from Plessey, now Mitel, which is a fully integrated ECL direct digital synthesizer chip. [Ed. Note: Mitel informs us that this device had been withdrawn from the market but has now been resurrected and should be available again now through sales representatives and distributors.] The chip includes a 31-bit accumulator, ROM, and DAC to produce two quadrature outputs, as

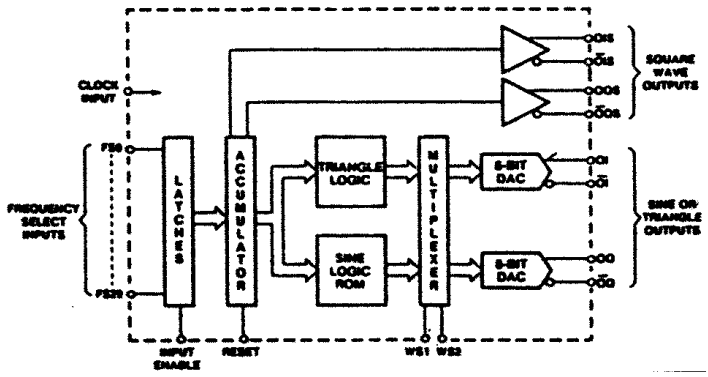


Figure 4-47 Plessey SP-2002, integrated DDS. (*Courtesy of Plessey.*)

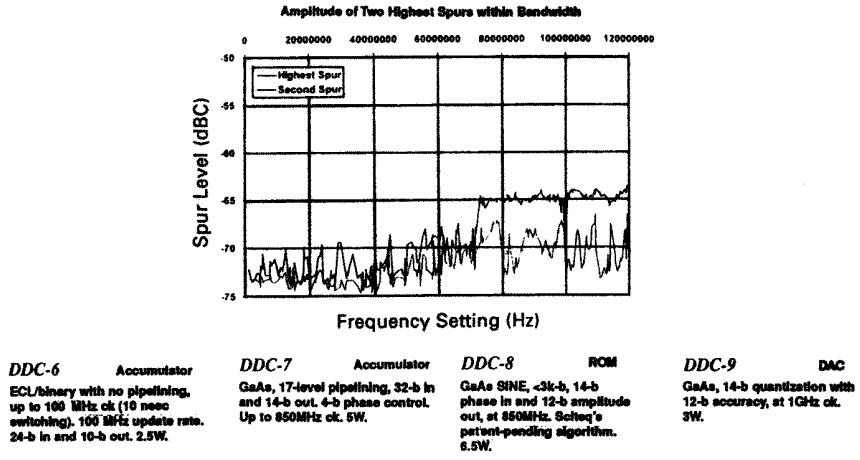


Figure 4-48 Very high-speed DDS components and performance; note spurious level as output frequency increases.

shown in Fig. 4-47. The device clocks up to 1.6 GHz, but the MSB cannot be programmed so the maximum output frequency is $F_{clk}/4$.

Sciteq Electronics offers a family of products—ADS-2, ADS-3, and ADS-6—clocked at 400, 640, and 1000 MHz, respectively. These are GaAs devices, including a 32-bit accumulator, 14 × 12-bit ROM, and a 12-bit DAC. See Fig. 4-48.

Sciteq Electronics offers a family of products—the ADS-4 and ADS-6—with clock rates varying from 400 to 1600 MHz.

All these devices operate with 8 bits of amplitude resolution (ADS-6 has 12) and offer spurious signal response of approximately −45 dB. All (with the exception of Plessey) use a 1000-MHz DAC produced by Tri-Quint, P/N 61xx.

Sciteq Electronics' ADS-6 is the only high-speed direct digital synthesizer product that offers 12 bits of amplitude resolution at close to 1000-MHz speed; see Fig. 4-50. It uses a new 12-bit DAC, P/N 60XX, produced by Tri-Quint, too. However, at this time, the DAC performance limits the direct digital synthesizer overall spurious signal response to the equivalent of 8 to 9 bits.

4-9-2 Medium-speed direct digital synthesizer

There is a great variety of medium-speed direct digital synthesizer products on the market, using CMOS and bipolar technology.

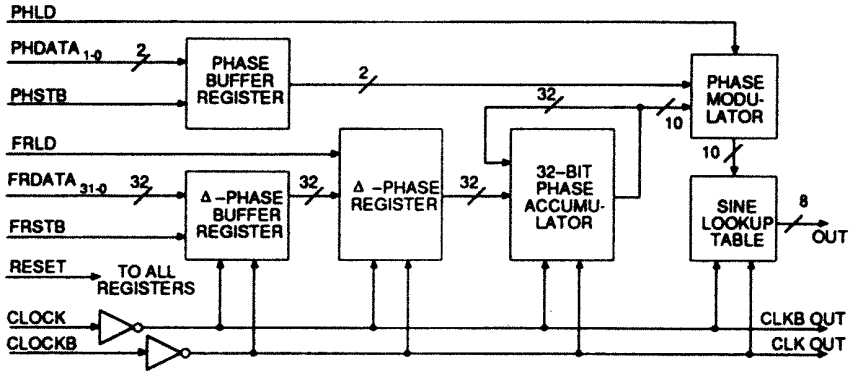


Figure 4-49 Block diagram of 1000-MHz 8-bit direct digital synthesizer STEL P/N 2173. (*Courtesy of Stanford Telecom.*)

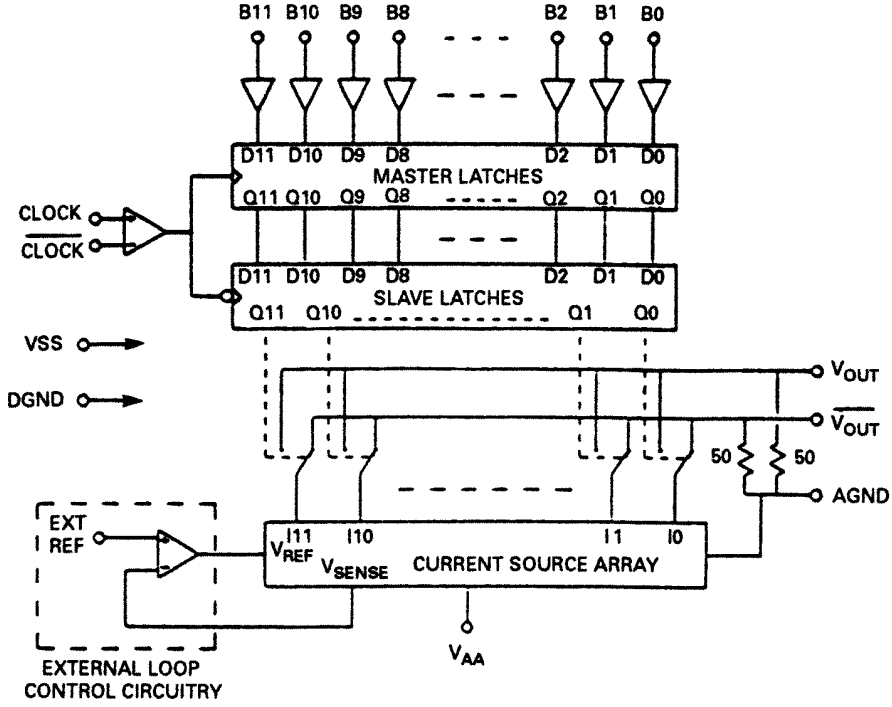


Figure 4-50 Functional block diagram of 12-bit DAC, clock speed >1000 MHz GaAs technology (TQS). (*Courtesy of Tri-Quint.*)

Most of the broader-market devices operate in the digital domain only and a DAC must be added to produce a sine wave. However, Analog Devices (www.analog.com/DDS) offers the AD98x0 line of Complete-DDS devices, which integrate a high-speed, high-performance DAC and DDS architecture onto a single CMOS chip.

AD98x0 C-DDS products include on-chip 10-bit signal DACs optimized for low output distortion. The devices have 32-bit phase accumulators, which enable output frequency tuning resolutions much finer than a PLL-based synthesizer. The AD9850 has a tunable output resolution of 0.06 Hz, with a clock frequency of 125 MHz; the AD9830 has a tuning resolution of 0.012 Hz, with a reference clock of 50 MHz. Furthermore, the output of these devices is phase-continuous during the transition to the new frequency. The highly integrated AD98x0 C-DDS devices are packaged in very small surface-mount packages and are quite cost-effective. C-DDS synthesizers dissipate much less power than earlier discrete DDS solutions. For example, the AD9850 dissipates 155 mW at 3.3 V when generating a 40-MHz signal, with a 100 MHz reference clock. Table 4-4 lists the specifications of the AD9850.

Qualcomm's (www.qualcomm.com/ProdTech/asic/synth.html) P/N Q2368 CMOS DDS can be configured as a single high-speed DDS capable of operating at 130 MHz clock speed, or as two independent DDS functions each capable of operating at 65 MHz speeds. FSK and PSK modulation modes in addition to linear frequency mode (CHIRP) are also supported.

Sciteq's P/N DDS-1 hybrid device offers sine-wave output (the DAC is included in the device) in addition to FM, phase modulation, and digital amplitude control (the multiplier is included in the device). The device operates using 16-bit logic but uses a 12-bit DAC; frequency and phase control are parallel and can be updated at the clock rate. The specifications are shown in Table 4-5. The analog spurious signal response is approximately –65 dB.

Stanford Telecommunications (www.stelhq.com) offers a family of direct digital synthesizer CMOS products, P/N 11xx. These devices are offered at 30-, 50-, 60-, and 80-MHz clock rate, with and without quadrature outputs and phase modulation, and some have dual DDS functions. The P/N 1176 is BCD direct digital synthesizer chip with 3 binary MSBs; it runs at 80 MHz and has 3 bits of phase modulation.

TABLE 4-4 AD9850 Specifications

Maximum clock frequency	125 MHz
Maximum output Nyquist-frequency bandwidth	62.5 MHz
Frequency tuning word resolution	32 bits
Phase tuning word resolution	5 bits
Supply voltage	+3.3 V or +5 V
Power dissipation @ max. operating conditions	155/380 mW
Worst-case narrowband SFDR (±50-kHz window) @ max. clock	72 dBc
Wideband SFDR (Nyquist) @ 20-MHz Aout	58 dBc
Wideband SFDR (Nyquist) @ 40-MHz Aout	54 dBc
Control interface	Parallel/serial

Courtesy of Analog Devices.

TABLE 4-5 Integrated DDS Specifications (DDS-1)

Frequency:	
Range	DC to 11 MHz (@ 25-MHz clock)—typically up to 45% of clock
Resolution	$F_{ck} \div 2^{32}$ (0.00582... Hz @ 25-MHz clock)
Control	TTL-compatible, parallel, positive-true logic with strobe
Output:	
Level	Sine wave @ ~0.8 $V_{p\text{-}p}$ into 50 Ω (complementary outputs), DC coupled (DC bias = ~ −0.4 V)
Resolution	11 bits, binary weighted (<0.006 dB steps)
Control	TTL-compatible, parallel, positive-true logic with strobe
Spectral Purity:	
Harmonics	< −40 dBC
Spurious	< −60 dBC (measured at the output of the DAC)
Phase noise	Equivalent to the noise of the DDS clock
Phase:	
Range	0° to 360°
Resolution	16 bits (<0.006° steps)
Control	TTL-compatible, parallel, positive-true shared data bus with upper 16 bits of frequency control word
Environmental:	
Operating temp	0°C to +70°C
Storage temp	−40°C to +85°C
Power Supply (max):	
Logic (V_{CC})	+5 V ± 5% @ 300 mA for 20-MHz clock
DAC (V_{EE})	−5.2 V ± 5% @ 180 mA (independent of clock frequency)
Power Supply (typical):	
Logic (V_{CC})	+5 V ± 5% @ 210 mA for 20-MHz clock
DAC (V_{EE})	−5.2 V ± 5% @ 100 mA (independent of clock frequency)
Mechanical:	
Package (chip)	84-pin ceramic PLCC equivalent package
Dimensions	1.14" × 1.14" × 0.25" with 0.050" lead spacing
Evaluation board	2.5" × 3" pcb containing IDC headers for control and SMA connectors for RFIN/OUT

Courtesy of Sciteq Electronics.

Harris (www.semi.harris.com/data/fn/fn2/fn2813/FN2813.pdf) P/N HSP 45102, 45106, and 45116 offer complete digital direct digital synthesizer chips, using 12- and 16-bit logic at a speed rate up to 50 MHz. Some include serial interface (45102), and others have an amplitude modulation option (include a complex multiplier) (P/N 45116).

4-10 Performance Evaluation

As was mentioned above, the direct digital synthesizer is a combination of digital and analog functions. The digital part is completely deterministic, and the errors, mainly quantization, can be fully calculated or simulated. However, the DAC—the analog part of the device—is too complex to model and is a main contributor to the performance degradation relative to the computed one. The following subsections list important parameters and their performance relative to the calculated values.

4-10-1 Switching speed

The switching speed of a direct digital synthesizer depends mainly on the delay of the output low-pass filter and the propagation delay of the logic. In many designs, the arithmetic calculations in the accumulator and the ROM conversion are done in one cycle of the clock. However, in many other designs, very extensive pipelining is used. As a consequence, the output suffers a delay.

The pipeline delay depends on only the number of pipeline stages and the clock speed and is therefore a predictable number. A direct digital synthesizer which uses 28 levels of pipeline in the logic and is clocked at 50 MHz suffers a input-to-output delay of $28 \cdot 20 = 560$ ns. In most cases when pipelining is used, its delay is much higher than that of the LPF.

Switching speed characteristics are shown in Fig. 4-56. The nature of the direct digital synthesizer is that it exhibits phase-continuous switching.

4-10-2 Phase noise

DDS exhibits excellent phase noise performance. Since the output spectrum is always lower than the clock, there is an effect of division, and therefore a phase noise improvement. Usually, the output phase noise tracks that of the clock, with the associated noise

floor. The noise floor can be approximated by assuming a signal-to-noise ratio of 6N (N is number of DAC bits), over a bandwidth equivalent to the clock. For a 10-bit direct digital synthesizer running at 50 MHz, the noise floor is approximated as:

$$-6N - 10 \log F_{ck} = -60 - 77 = -137 \text{ dBC/Hz} \quad (4\text{-}56)$$

4-10-3 Spurious signals

This issue has been covered above. The state of the art can be summarized, based on our best knowledge of performance available from the different manufacturers, as follows:

Clock speed (MHz)	Spurious signals (dBC)
20	−70
50	−60
100	−55
400	−50
640	−45
1000	−30

4-10-4 Phase continuity

This issue was discussed above. All direct digital synthesizer products achieve phase-continuous switching. However, for fast switching and linear FM (chirp) applications, either no pipeline or total compensation is necessary. The state of the art today is a 500-MHz clock speed (approximately 200-MHz bandwidth), with a frequency update of 2 ns, using GaAs technology—Sciteq's P/N DCP-1. (See Fig. 4-57.)

4-10-5 Resolution

This subject was discussed already; it is one of the important advantages of DDS.

4-11 Sample-and-Hold Devices

Sample-and-hold devices are complex analog parts whose function in a direct digital synthesizer is to sample the output of the DAC after it settles, so that the DAC transitions can be "cleaned." Because of the required linearity, the sample-and-hold device is complicated and usually quite expensive. (See also Chap. 8.)

A comparison of DDS performances with and without a sample-and-hold device is shown in App. 4B. As expected, the close-in spurious signals (those that are caused by quantization or intermodulation) cannot be improved, but the wideband spurious signals generated by the sharp DAC transitions are cleaned well.

A few parts have been evaluated, and they exhibit excellent results. The Acculin P/N AL1210, defined as a 12-bit 125-MHz device, exhibits 12-bit accuracy at 50-MHz clock speed and 8-bits accuracy at 100-MHz clock speed. Analog Devices' P/N AD9101, defined also as 12-bit 125-MHz device, has 12-bit accuracy at 50 MHz and 8-bit accuracy at 125-MHz clock speed. Both devices are monolithic and produce excellent results.

Micro Networks offers an excellent device, the MNHT1010, a 125-MHz device having 8-bit accuracy at 125 MHz. This device is a hybrid. Hewlett-Packard developed a 125-MHz direct digital synthesizer few years ago and reported a sample-and-hold spurious signal improvement of up to 20 dB, at 125-MHz clock speed. Another improvement gained by use of a sample-and-hold device lies in the fact that its pulse width is substantially shorter than the clock. Thus it can be used as a switch open only for the "on" time. In most cases this is approximately $T/2$. This causes the output spectrum to have the shape of $(\sin x)/x$, but now $x = \pi T/2$, which means that the spectrum is much flatter. As an example, for the case $F = 0.4F_{dk}$, the regular direct digital synthesizer loses

$$20 \log \left(\frac{\sin 0.4\pi}{0.4\pi} \right) = -4 \text{ dB} \tag{4-57}$$

while the use of a sample-and-hold switch with a hold time of $T/2$ will cause an amplitude loss of

$$20 \log \left(\frac{\sin 0.2\pi}{0.2\pi} \right) = -0.6 \text{ dB} \tag{4-58}$$

However, to gain this flatness, it is necessary to reduce the pulse width accordingly.

4-12 Single-Bit DDS Revisited

The drive toward an all-digital direct digital synthesizer will probably continue until we get useful results. This section presents

some of my thoughts regarding this issue. Research at MIT and Sciteq Electronics is now in the phase of hardware implementation, so it will not be possible to discuss the performance of working hardware; but I believe that this is one method that can work. Unfortunately I am not aware of any other publications on this issue, although some work has been done in the Massachusetts Institute of Technology laboratories; but to the best of my knowledge the material has not been published.

Let's turn back to the direct digital synthesizer implementation of Secs. 4-2 and 4-3. The problem created by this method was caused by the very high level of quantization since only the carry output bit is used as an output but the state of the accumulator is completely ignored. The additional information available at the state of the accumulator when the carry output is generated can be used to control a delay element, as mentioned in Sec. 4-3. In the last few years, the demand of both digital and high-resolution graphics created the need for very accurate digitally controlled electronic delay lines, and these devices will be used here (see Fig. 4-51).

Let's look at a simple example, for an accumulator of 4 bits controlled with an input of $W = 3$. The complete cycle looks as follows:

Accumulator output	**Carry output**	
0000 (0, or 16)	1	Cycle begins.
0011 (3)	0	
0110 (6)	0	
1001 (9)	0	
1100 (12)	0	
1111 (15)	0	
0010 (2)	1	
0101 (5)	0	
1000 (8)	0	
1011 (11)	0	
1110 (14)	0	
0001 (1)	1	
0100 (4)	0	
0111 (7)	0	
1010 (10)	0	
1101 (13)	0	

0000 (0 or 16) actually completes the cycle. Cycle begins or ends.

...

Ideally we would like to generate a transition every 16/3 = 5.3333 cycles, but this is not possible because this structure can generate a transition only at integer multiples of the clock speed.

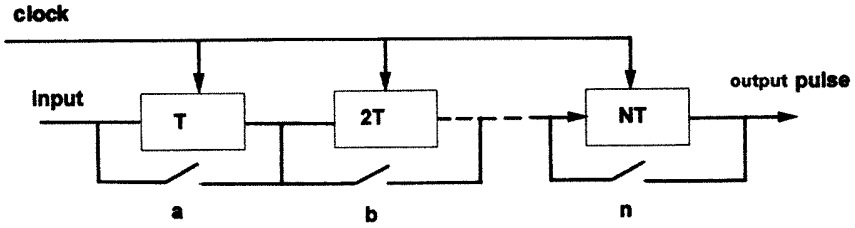


Figure 4-51 Digital delay line, consisting of binary incremented delays, must be synchronous to create digital accuracy. $T = aT + b(2T) + c(4T) + \cdots$ where $a, b, c, \ldots$ are either 0 or 1.

This is indeed the crux of the problem. Although the average frequency is correct, we generate 3 transitions in 16 clocks, the instantaneous frequency is wrong as the transitions occur after 6, 5, and 5 clocks, and therefore it generates spurious signals.

But the error can be calculated easily. After the first transition the error is $-\frac{2}{3}$ clock (we should transition after 5.333 clocks, and we transit after 6), and after the second transition it is $-\frac{1}{3}$ clock (we should transit after 10.666, and we do after 11). There is therefore a clear relation between the error and the parameters W (input control word) and D (accumulator output at the moment of carry generation). The error is exactly $-D/W$. This can be explained as follows: The accumulator increments at W per clock in a linear manner, so a "leftover" of D is exactly D/W after where it "should be."

By using a digital delay generator, the carry output is first connected to a logic circuit that calculates the ratio D/W and delays the carry signal accordingly. The calculation can be done by software or can be stored in a ROM with D and W as input parameters. Now suppose that the digital delay line is 8 bits. Let's assume therefore that the clock has been divided into 256 intervals (8 bits). Every main clock cycle has the weight of 256 subclocks. The total cycle is therefore $256 \cdot 16 = 4096$ subclocks. After the first transition, the carry output will be delayed by $256 \cdot \frac{2}{3}$, and after the second transition by $256 \cdot \frac{1}{3}$. Now the new intervals will be as follows (refer to the beginning of the cycle):

The first happens after $6 \cdot 256 - \text{int}(256 \cdot \frac{2}{3}) = 1366$, the second interval after $11 \cdot 256 - \text{int}(256 \cdot \frac{1}{3})$, the third after $16 \cdot 256 - \text{int}(0 \cdot 256/3)$, or intervals of 1366, 1365, 1365. The delay line is digital and so only integers can control its delay, thus the integer(x) function.

Now, the phase jitter has decreased 256 times or improved by 48 dB! As an example, for an accumulator clocked at 2 MHz, the subclock generated by the delay increments is 256 times faster, or equivalent to 512 MHz. The rms jitter, if the delay line was ideal, can be easily calculated and is now 1.45 ns!

We believe that this is one basic procedure to generate an all-digital direct digital synthesizer.

The negative delay need not worry the designer because there are a variety of simple procedures to convert it to positive delay. For example, delay the main clock always by 1 main clock cycle, and now delay by a positive number given by $1 - D/W$. Also note that $W < D$ always (the carry overflow can never be as large as W). Because the error is still periodic and therefore will generate a spur, although attenuated by 48 dB, it is recommended to use a randomization procedure to destroy this periodicity and achieve a direct digital synthesizer free of spurious signals. Many procedures are possible, similar to the Wheatley or others mentioned above.

The one "sticky" issue is that some delay generator (analog type, A/D 9500) uses a DAC and is a mixed analog and digital device. The effects of the analog portion have not been analyzed yet, although they will be different from those in a direct digital synthesizer DAC, since the operation here is completely different. A digital implementation is shown in Fig. 4-51, but there is no such device on the market.

Such hardware is now under investigation and test results will be published in the future.

For applications, we would point the designer to Analog Devices' P/N AD9500 and such other delay lines. These are 8-bit delay lines (256 states) with very high accuracy and resolution from 50 to 100 ps! This could be the equivalent of running the counters mentioned in Sec. 4-3 at 10 GHz!

4-13 Arbitrary Waveform Generators

If DDS generates sine waves by storing their known sampled values, it is possible to extend the utility, store any waveform desired, and clock it out accordingly. This is perfectly consistent with the principles of the sampling theorem.

If a desired waveform is available or known to us, its digital samples can be stored in memory. Similarly, it is possible to calculate a required waveform, store it in memory (usually RAM), and, when required, sequence the waveform out and filter it as required by the sampling theorem.

Such advanced direct digital synthesizer designs are now very common. They are used for testing and simulation for applications like military simulation of threat signals, testing of computer hard disks, vibration analysis, sonar and nondestructive testing, complex communication waveforms, linear FM, speech synthesis, and such. The principle of operation is again exactly the inverse of the sampling theorem. In the sine-wave implementation (DDS, indeed a subset of arbitrary waveform generators), the luxury is in the fact that only slightly more than 2 samples per cycle are required; thus the practical BW of the direct digital synthesizer is close to 50 percent of the clock rate. In arbitrary waveform generators, the waveform is arbitrary, and the ratio is usually closer to 5 to 10 percent. All arbitrary waveform generators can generate sine waves and usually have a separate BW specification for sine generation, for the reason mentioned above.

The variation between the multitude of products on the market is in the size of the memory, interface software, clock speed, modes of operation, and horizontal resolution (size and accuracy of DAC). Since arbitrary waveform generators are beyond the scope of this text, we'll just mention a few popular models, with a short summary of their specifications.

- Hewlett-Packard—up to 125 MHz, 12 bits, 8-ns update
- Wavetek—up to 50 MHz, multichannel, 12 bits
- Tektronix—up tp 50 MHz
- LeCroy—up to 400 MHz, 8 bits
- Analogic—up to 800 MHz, 8 bits

There are also many personal computer cards, and VME and VXI cards, that perform these functions.

An arbitrary waveform generator that can generate a complex signal, i.e., the signal and its quadrature, can simulate any waveform, including modulations like SSB, quadrature AM (QAM), and such. However, this requires generation of the quadrature of baseband signals (actually their Hilbert transform). Most modems use arbitrary waveform signal generation because they require com-

plex signaling, and analog implementations are complicated and require tuning in manufacturing. Arbitrary waveform generators are the extension of direct digital synthesizer technology, and they have become an important discipline in the growing family of signal generation and instrumentation.

4-14 Digital Chirp DDS

Another set of DDS applications is dedicated digital chirp generators, used in sweep oscillators for testing, for imaging radars and altimeters, and for compressive receivers and semiconductor process control.

The chirp generators generate a linear FM signal that is fully synthesized and therefore achieves linearity and accuracy not possible with regular analog techniques (VCOs). The chirp generator is similar to the regular direct digital synthesizer but includes a dual accumulator and therefore generates a quadratic phase.

Such a device, assuming the use of an N-bit accumulator, input control of W, and a clock of F_{ck}, will generate a chirp rate of $(F_{ck}W/2^N)/(1/F_{ck})$ because for every clock tick, the frequency changes by $F_{ck}W/2^N$ and that rate of change is F_{ck}. Therefore the equation for the chirp rate is

$$\text{Chirp rate} = \frac{F_{ck}^2 W}{2^N}$$

The CMOS implementation for this device is available from the STEL P/N 1180 (clocked at 60 MHz) and from Sciteq's P/N DCP-1 clocked at 500 MHz. For example, for a 500-MHz clock rate and $N = 24$, the minimum chirp rate is for $W = 1$ and is $(500{,}000{,}000)^2/2^{24} = 1.49 \times 10^{10}$ Hz/s, or 14.9 kHz/μs.

The chirp rate is perfect except that there is always a level of quantization. For example, the device described above has a minimum step size of $500 \times 10^6/2^{24} = 29.8$ Hz. Another problem is the LPF group delay, especially at the high end of the band. If this becomes important in the application, a phase equalizer needs to be added to compensate for the filter group delay.

However, the repeatability and linearity of these signals are so superior to their analog equivalents that many users gravitate toward this solution.

4-15 Conclusion

DDS technology has emerged as a viable technology in the last 10 years. From modest performance, covering very limited bandwidth, it is now popular and covers a wide range of applications up to 1000-MHz speed. ASIC designs accommodate all the logic functions at low power and low cost. Direct digital synthesizer operates either by itself or in conjunction with other synthesis technologies—PLL and DA—to improve resolution, speed, and cost.

DDS is an extension of the digital revolution into the *radio-frequency* (*RF*) domain. This is a viable technology that has seen its birth in the last 10 years but will continue to see evolution and improvement in lowering power, increasing bandwidth, and probably integration with some PLL functions.

Performance suffers from limitations in spurious signal performance, but this can be overcome by dividing. Otherwise, direct digital synthesizer exhibits excellent features, resolution, speed, phase noise, simplicity, and cost. As the trend to digitization continues to accelerate, the market and utilization of direct digital synthesizer technology will increase gradually and steadily. We expect to see more integration of DDS and PLL, because the combination offers so many advantages over PLL alone. With the improvement in DAC technology and evolution of DSP, DDS performance and utility will improve, too.

Appendix 4A DDS Applications

Because of its inherent advantages (digital producibility, small size, low power, high resolution, fast speed, smooth phase transition, low cost) and disadvantages (mainly limited bandwidth and spurious signal response), direct digital synthesis finds and expands its place in the overall synthesis market.

The following are some typical applications:

1. *Stand-alone synthesizer, phase modulator, and sweep generator.* The relative simplicity and digital accuracy allow implementations at low cost. Sweep (linear FM) speeds are possible now at up to a 500- to 600-MHz update rate. Most such applications use the direct digital synthesizer as a tone-generating instrument, as a tester for frequency response of networks, and as a signal to be used in linear FM radars, proximity fuses, and altimeters. The

applications for high-speed chirp signals are especially attractive for *synthetic aperture radars*(*SARs*). At low frequencies, the technology is already mature, and the use of microprocessor control allows flexibility in programming and utility. For wideband chirp generators, see item 8.

2. *Clock generators.* These applications are used for clocking of digital and computing systems and have the advantage of resolution, size, and fast agility.

3. *Accurate phase and amplitude modulators for modem applications.* The direct digital synthesizer is the only true MSK (minimum shift keying) modulator for modems. All the analog implementations are approximations. In other modem applications where wave shaping is a must, DDS disciplines and arbitrary waveform generators (ARBs) are natural, and they have dominated this market for a few years now.

4. *Part of a PLL synthesizer, where the direct digital synthesizer provides a limited BW but an arbitrarily fine resolution.* In these cases, the direct digital synthesizer covers anywhere from 1 kHz to 10 MHz with fine resolution and excellent phase noise and spurious signals. The PLL provides the bandwidth and high-frequency capabilities.

5. *Part of direct analog (DA) synthesizers.* It provides limited BW but high resolution and speed, thus reducing size and cost.

6. *Part of PLL where dc FM is required.* PLL cannot achieve dc FM, but if a PLL is locked to a direct digital synthesizer and the synthesizer is modulated, even at very low frequencies, the loop of the PLL can be relatively wide and still perform dc FM.

7. *Arbitrary waveform generators.* This is a very big market by itself and is not included here in great detail. The applications are for simulation of real-world signals, testing, and simulation.

8. *Linear FM synthesized signals.* These are for sweep test instrumentation as well as applications like synthetic aperture radars and process control (IC manufacturing). These applications emerge now as bandwidth expands to hundreds of megahertz. The digital design ensures linearity, repeatability, and total control with digital accuracy, including phase modulation, and replaces very complex and expensive existing analog methods. In most cases, GaAs logic is being used to achieve the update rates, but this speed and this performance have already been achieved in silicon technology.

A device clocked at 500 MHz, updating frequency every 2 ns, using two 24-bit accumulators, and yielding an approximately 30-Hz step ($500 \times 10^6/2^{24}$) has been demonstrated (end of 1992). The bandwidth is above 200 MHz, and worst-case spurious signal performance is −50 dBC. The minimum sweep rate is approximately 15 kHz/μs. The device has also a 12-bit phase modulation capability and the ability to set a start frequency.

A unique application for the phase modulator (applicable to all direct digital synthesizer designs that have phase control) is to use this function as a linearizer of the group delay of the output LPF. Since the instantaneous frequency is available, this can map (via a ROM) the group delay of the filter and digitally make corrections and adjustments to approximate a linear group delay.

Unquestionably, the controlling technology in the frequency synthesis discipline is PLL. However, we expect direct digital synthesizer to continue to penetrate some of the traditional PLL markets such as the following:

- Clock recovery in disk drives. Direct digital synthesizer is robust against mechanical shock and vibration, is more reliable, and has better and cheaper producibility.
- Digital cellular telephony, for waveform generation and shaping.
- Data modems, for high-accuracy phase modulation.
- In conjunction with RF/MW single-sideband (SSB) modulators, for the purpose of preserving phase information, for functions like phase shifters, and simulators.
- Arbitrary waveform generator at high speed and wide bandwidth.
- Merging of DDS principles with PLL; see Chap. 5. This is probably the cardinal area of growth for DDS.
- Vector modulators, incorporating digital AM functions for modems and simulators.

Appendix 4B DDS—Spectra and the Time Domain

DDS is a complicated discipline. A real understanding of its many artifacts can be gained mainly by putting together hardware and making the measurements. Hands-on experience is the great tutor here.

Below is a list of typical measurements that represents some of the more nonintuitive phenomena. The time and spectra plots are followed by explanations.

All the hardware and drawings are courtesy of Sciteq Electronics. For binary models the model DDS-1, a 20-MHz, 32-bit accumulator, was used; and for BCD models, model VDS-3 was used, a 10-MHz all-BCD direct digital synthesizer, with 10-MHz BCD size accumulator (the accumulator size is 1×10^{10}, and at a clock rate of 10 MHz it generates a step size of 1 ms).

−1. Time-domain representation of the unfiltered output of a binary direct digital synthesizer, clocked at 20 MHz, with an output frequency of 2.5 MHz. The cycle contains exactly eight states. (See Fig. 4-52.)

−2. Same as −1 but with the detail of one transition to demonstrate the DAC transition response. (See Fig. 4-53.)

−3. Same, but the frequency is controlled to 7/16 or 9/16 of the clock. Note that both show the same time-domain charac-

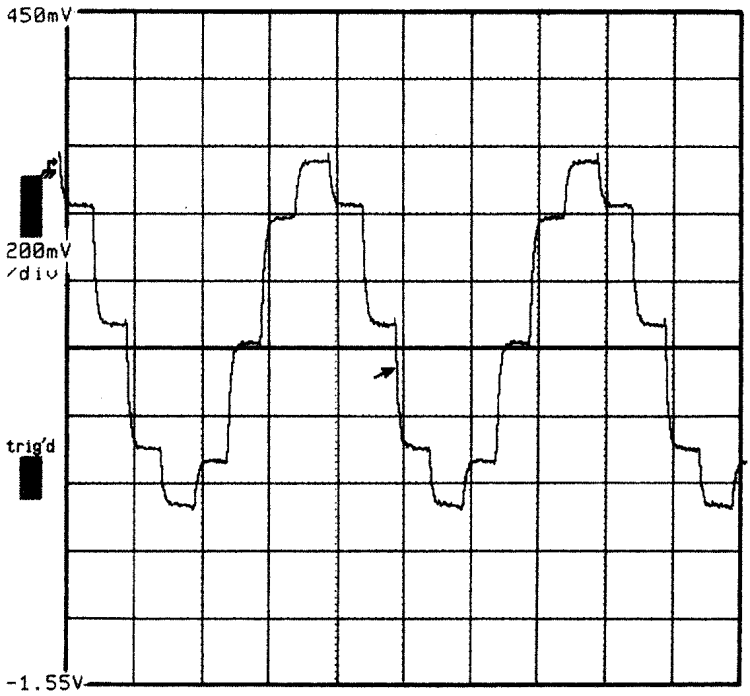


Figure 4-52

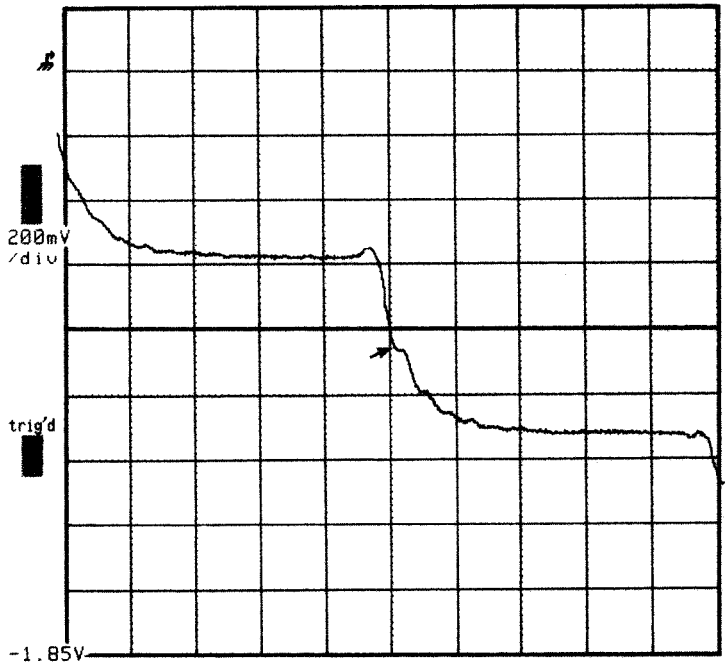


Figure 4-53

teristics, since 9/16 of the clock is above Nyquist and therefore generates both 9/16 and 1 − 9/16 = 7/16. It is hard to see these two frequencies in their time-domain presentations. (See Fig. 4-54.)

−*4*. Same, but with frequency control to 65/128 of the clock. This generates both 65/128 and 63/128 of the clock and resembles a double sideband (DSB) modulation of $F_{ck}/128$ on an $F_{ck}/2$ carrier. However, it is not exactly DSB because the amplitudes of the two frequencies are off slightly [(sin x)/x]. (See Fig. 4-55.)

−*5*. Switching transient for a BCD direct digital synthesizer showing a 2-MHz transient. The phase ringing is clear. Switching time is approximately 1.5 μs. (See Fig. 4-56.)

−*6*. Same, but with a frequency transient of 400 kHz. Much less ringing. (See Fig. 4-57.)

−*8*. Demonstration of a smooth frequency transition. Perfect for sweeping or MSK modulation. (See Fig. 4-58.)

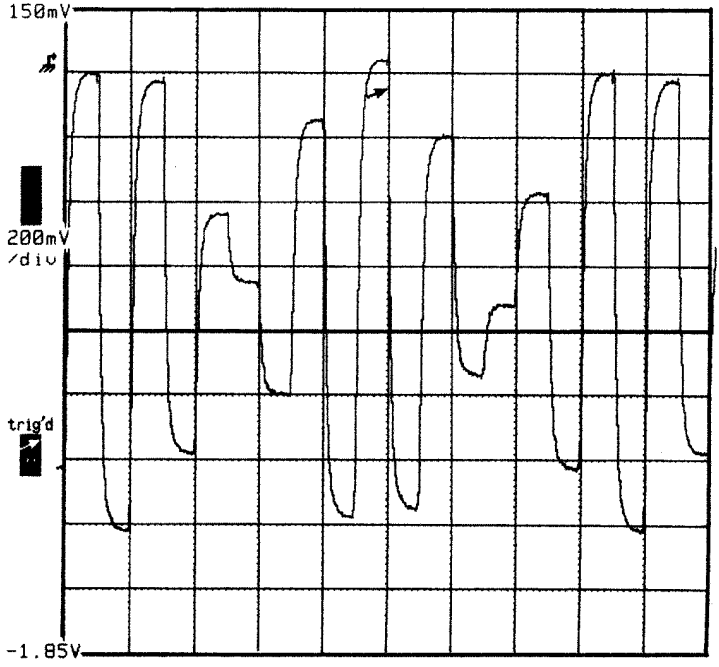


Figure 4-54

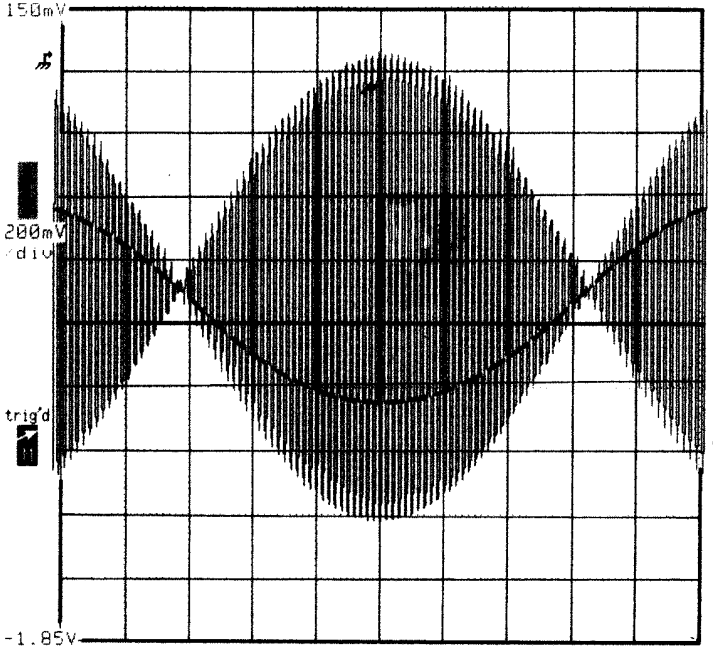


Figure 4-55

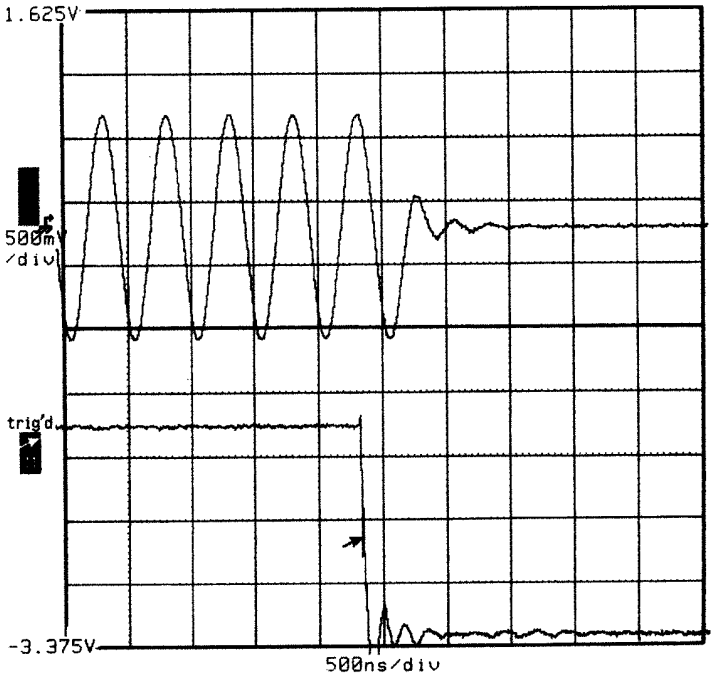


Figure 4-56

—*9*. DDS-1, normally clocked at 20 MHz and using an 8-MHz filter, is clocked here at 2 MHz, so that all the aliasing shows. (See Fig. 4-59.) The first line is the fundamental, the second its second harmonic, the third the aliasing, the fourth the clock leakage, etc. This is a distorted spectrum since the power level to the input mixer causes intermodulations.

—*10*. Same, but the input attenuation to the spectrum analyzer mixer was increased from 10 to 30 dB. Now the readings are correct. (See Fig. 4-60.)

—*11*. One-third clock output is one of the worst frequencies for a direct digital synthesizer, as the signal second harmonic intermodulates with the clock. Here is a view close to ⅓. (At exactly ⅓ they all fall on one another, and the phenomenon cannot be seen.) Again, there is a distortion (attenuation is set to 10 dB). (See Fig. 4-61.)

—*12*. Same as —11, with attenuation set to 30 dB. (See Fig. 4-62.)

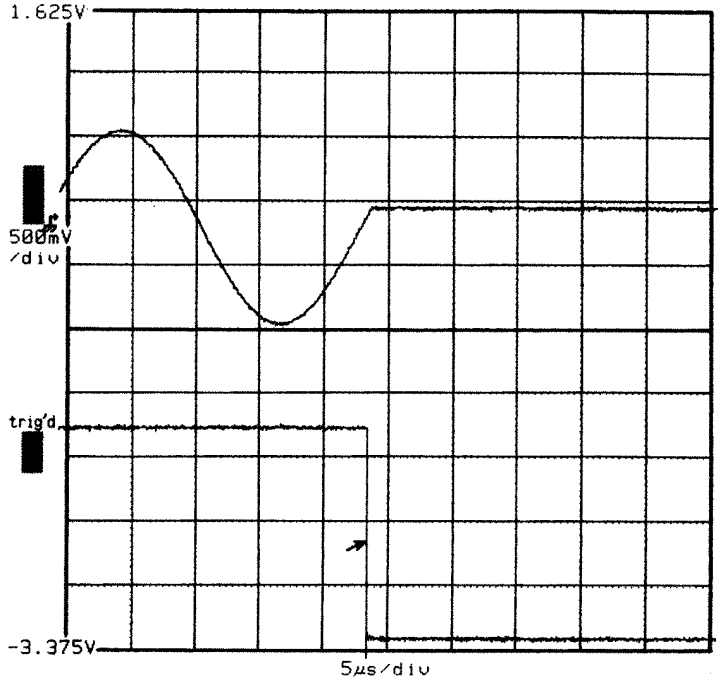


Figure 4-57

−*13*. Similar view, but now the DDS is run with a 20-MHz clock and an 8-MHz filter. There is only one signal since all the aliasing signals are filtered. (See Fig. 4-63.)

−*14*. Same as −13, with attenuation set to 30 dB. (See Fig. 4-64.)

−*15*. Another "bad" frequency, at close to $F_{ck}/4$. (See Fig. 4-65.)

−*16*. A BCD direct digital synthesizer, clocked at 10 MHz, controlled to 1,000,001 MHz. The 1-kHz spurious signals show. (See Fig. 4-66.)

−*17*. Same as −16, controlled to 1,000,100 Hz. The 2-kHz spurious signals show. (See Fig. 4-67.)

−*18*. Same as −16, with control set to 100,100 Hz (⅒ of before). (See Fig. 4-68.)

−*19*. Same as −16, with control to 100,200 Hz. (See Fig. 4-69.)

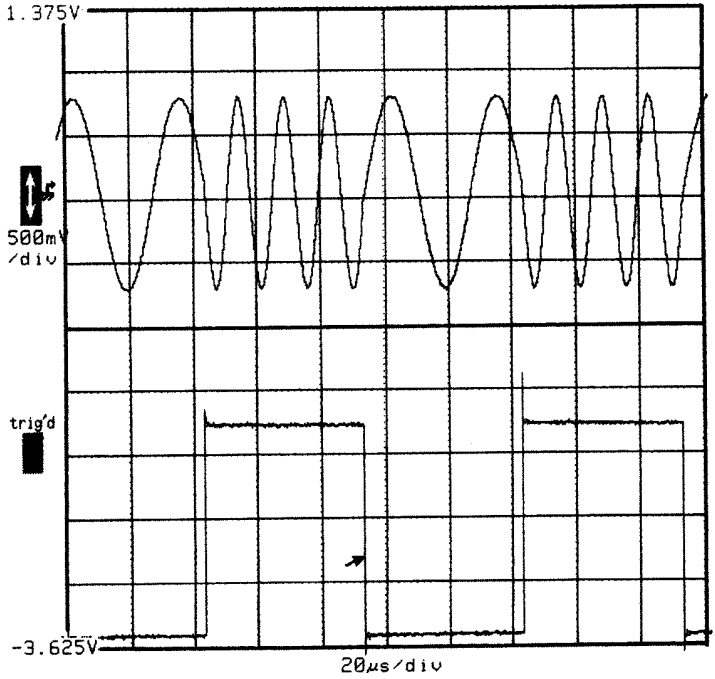


Figure 4-58

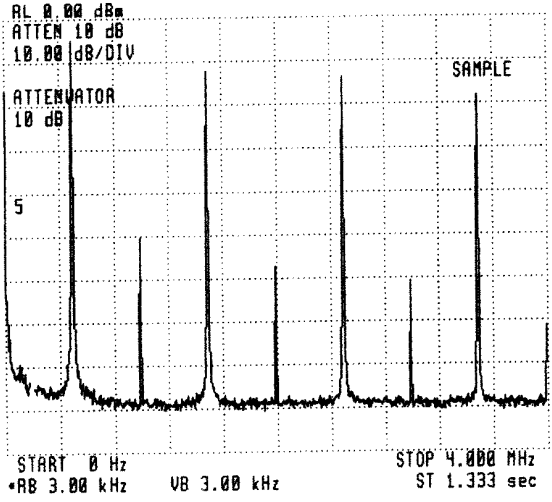


Figure 4-59

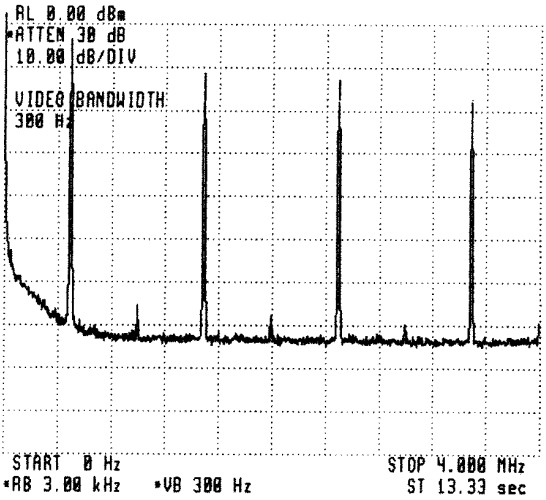


Figure 4-60

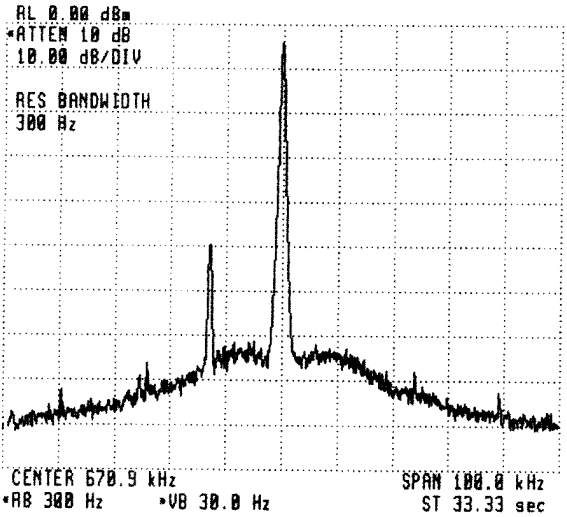


Figure 4-61

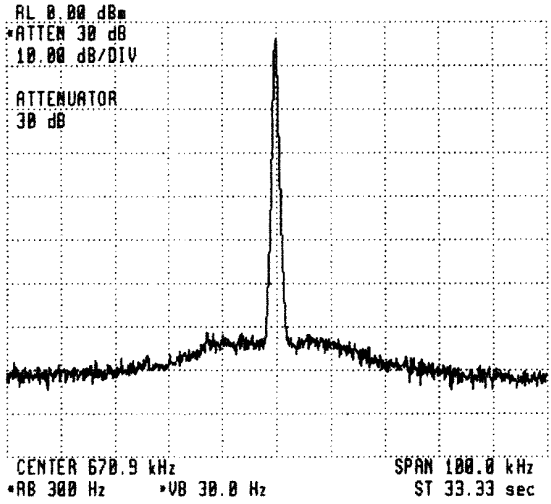


Figure 4-62

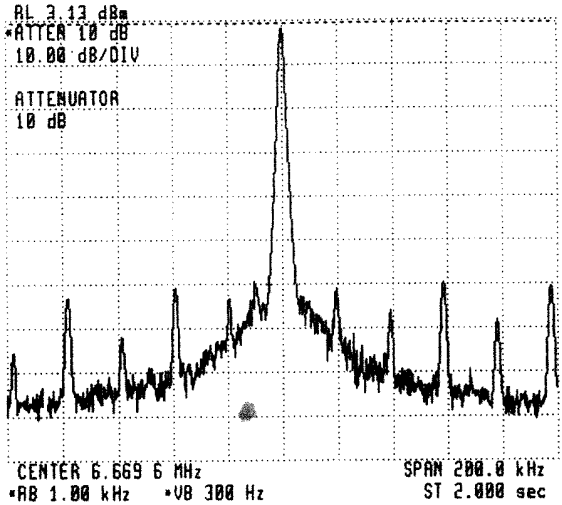


Figure 4-63

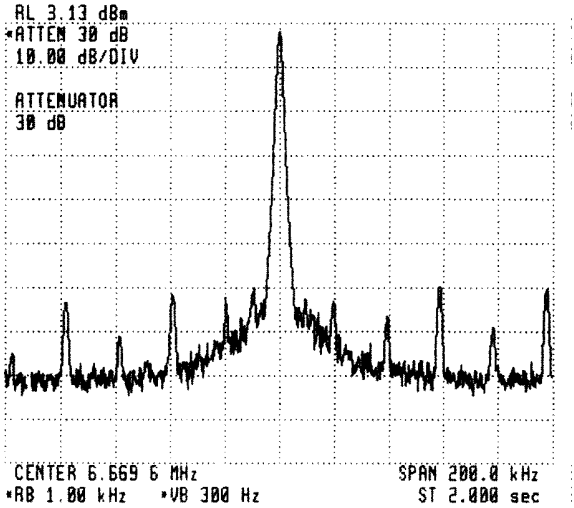


Figure 4-64

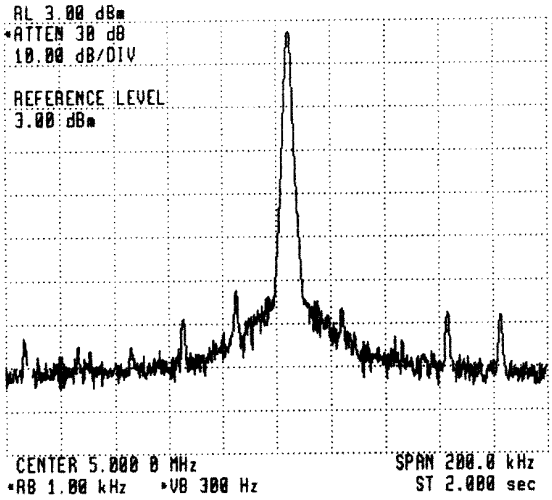


Figure 4-65

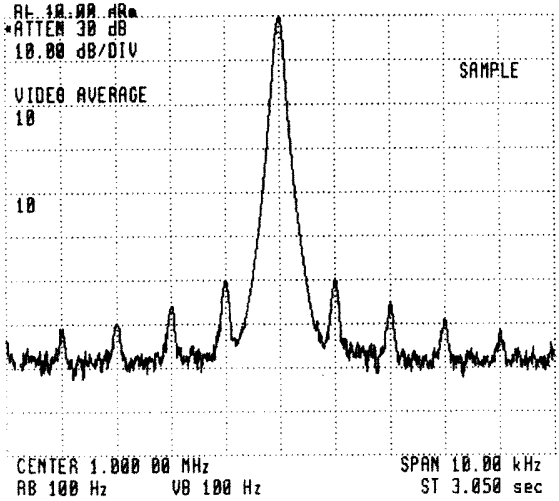


Figure 4-66

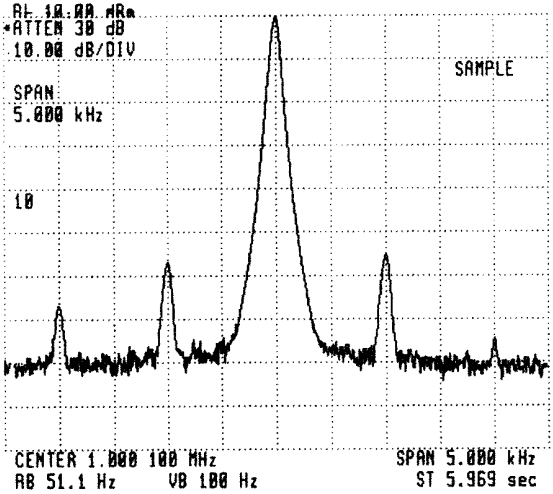


Figure 4-67

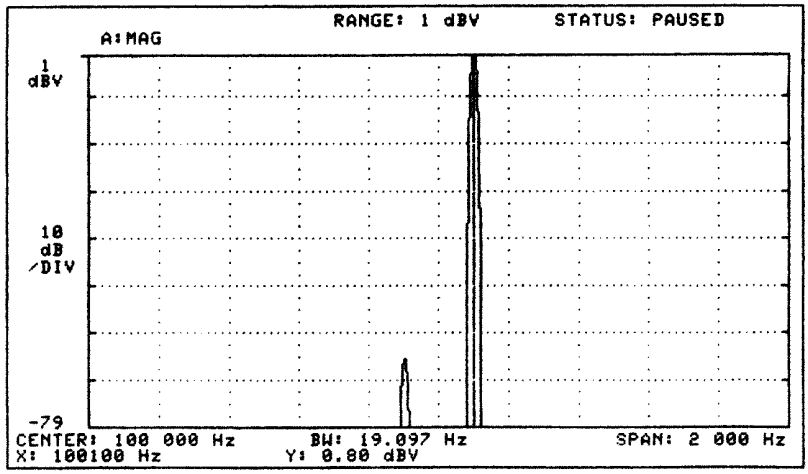


Figure 4-68

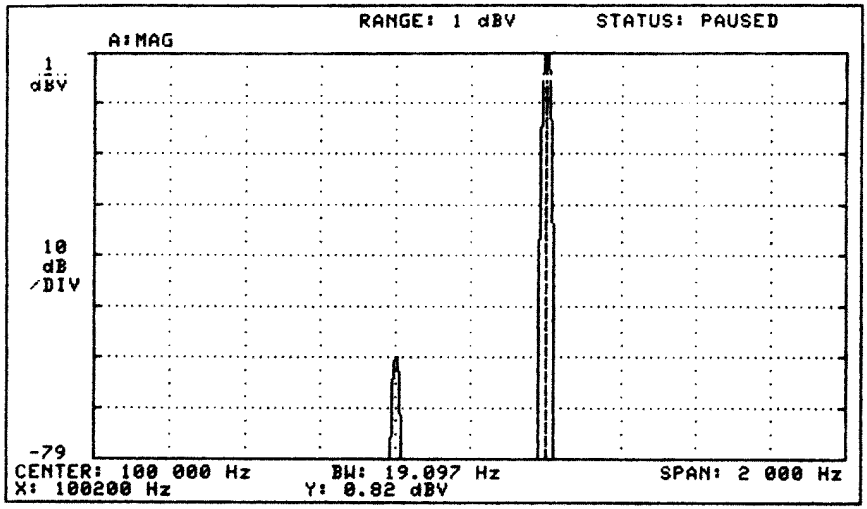


Figure 4-69

Sample-and-hold effect on DDS performance

See Figs. 4-70 to 4-74. The odd-numbered figures represent performance without a sample-and-hold device, and the even-numbered figures are the same but with a sample-and-hold device. Performance improvements of up to 15 dB have been realized!

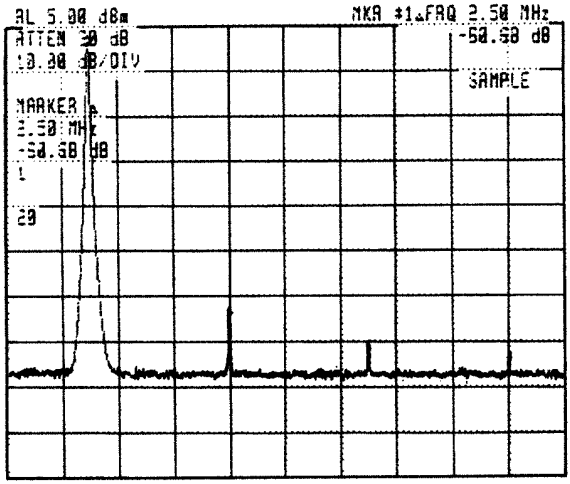


Figure 4-70

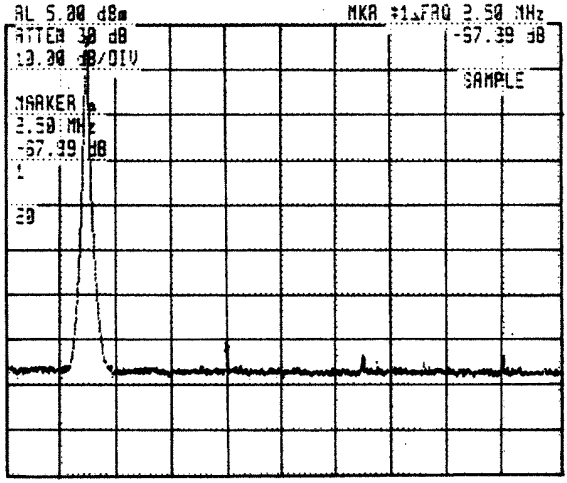


Figure 4-71

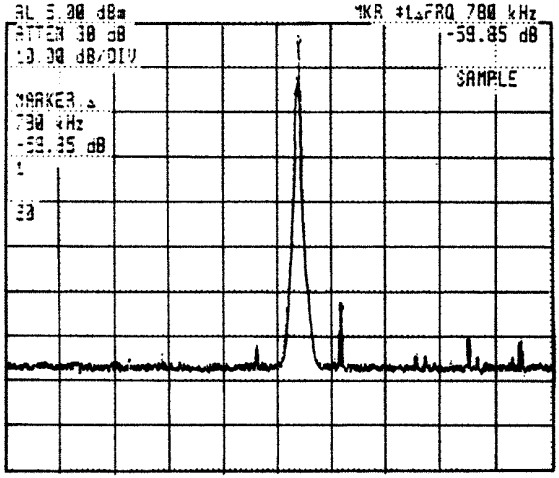


Figure 4-72

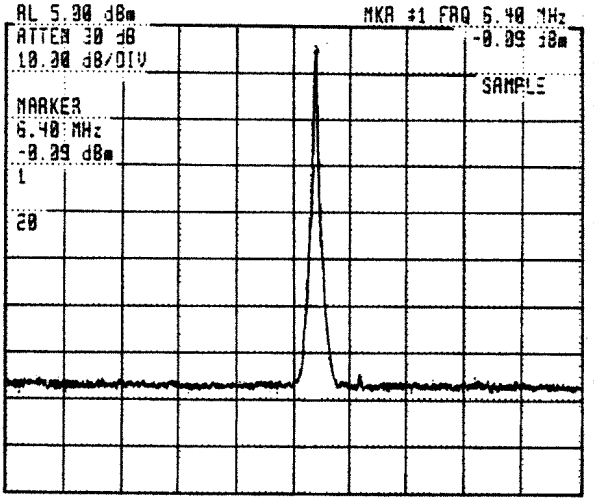


Figure 4-73

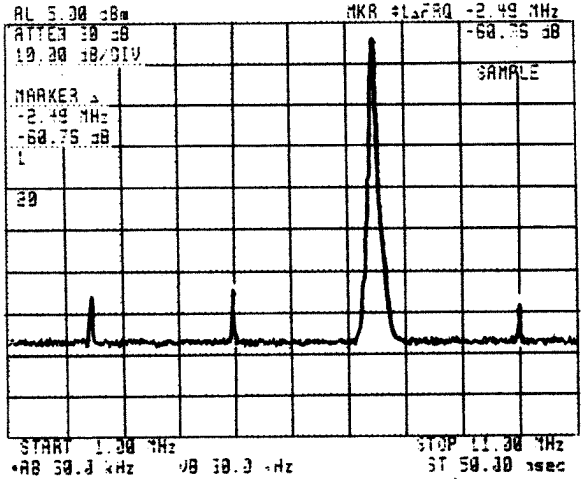


Figure 4-74 DDS artifacts, for output frequency of $0.375F_{ck}$ (3/8), show spurious signals (or subharmonics) at 1/8 and 2/8 of F_{ck}. F_{ck} = 20 MHz, F_{out} = 7.5 MHz.

Appendix 4C Sampling Theorem

The sampling theorem can be proved in many ways. We will follow Ref. 47. Suppose a stochastic signal $x(t)$ whose Fourier transform $X(\omega)$ exists and follows:

$$X(\omega) = 0 \qquad \text{for } |\omega| > B \tag{4-59}$$

Then $x(t)$ can be represented by its sample $x[k/(2B)]$ according to

$$x(t) = \sum_{k=-\infty}^{\infty} x\left(\frac{k}{2B}\right) v\left(t - \frac{k}{2B}\right) \tag{4-60}$$

and the weighting function $v(t)$

$$v(t) = \frac{\sin 2\pi Bt}{2\pi Bt} \tag{4-61}$$

is the impulse response of the ideal (noncausal) rectangular filter with bandwidth B:

$$V(f) = \begin{cases} \dfrac{1}{2B} & -B < f < B \\ 0 & \text{elsewhere} \end{cases} \tag{4-62}$$

Let's define a new function $u(t)$, given by

$$u(t) = \sum_{k=-\infty}^{\infty} \delta\left(t - \frac{k}{2B}\right) = \int_{-\infty}^{\infty} U(f)e^{j\omega t}\, df \tag{4-63}$$

$$U(f) = \sum_{k=-\infty}^{\infty} 2B\ \delta(f - 2kB) \tag{4-64}$$

Since $u(t)$ is periodic, it has a Fourier transform given by

$$u(t) = \sum_{k=-\infty}^{\infty} C_k e^{j\omega k/T} \tag{4-65}$$

and

$$C_k = \frac{1}{T}\int_{-T/2}^{T/2} u(t)e^{-j\omega k/T}\, dt = \frac{1}{T} \tag{4-66}$$

Now, we see that

$$X(f) = [X(f) * U(f)]V(f) \tag{4-67}$$

and we take the inverse Fourier transform of $X(f)$ to get

$$\begin{aligned} x(t) &= [x(t)u(t)] * v(t) \\ &= v(t) * \sum_{k=-\infty}^{\infty} x(t)\ \delta\left(t - \frac{k}{2B}\right) \end{aligned} \tag{4-68}$$

Now further manipulation yields

$$\begin{aligned} x(t) &= \sum_{k=-\infty}^{\infty} \sum_{-\infty}^{\infty} x(z)\ \delta\left(z - \frac{k}{2B}\right) v(t - z)\, dz \\ &= \sum_{k=-\infty}^{\infty} x\left(\frac{k}{2B}\right) v\left(t - \frac{k}{2B}\right) \end{aligned} \tag{4-69}$$

This completes the proof.

APPENDIX 4D - The effect of phase noise on data conversion devices

When sampling an analog signal for data conversion, clock phase noise can affect the accuracy of the measurement because of the inherent jitter in the clock. Let's calculate the phase noise requirement from sampling a signal with F_s bandwidth at F_c clock (sampling interval T_c=1/F_c). The ideal samples of a sine, represented by N bits, are given by: $S(t)=2^{N-1}\sin(2\pi\cdot t/T_s)$ at discrete time $t=i\cdot T_c$. If our timing is off (there is jitter in the clock), we can calculate the time error (T_j) that will cause a one digit error as follows:

Near zero, sin $x = x$; therefore, if instead of $t = 0$ we sample at T_j, the jitter necessary to cause an error of 1 is:

$2^{N-1}2\pi T_j/T_s = 1$ where $T_j = T_s/2\pi 2^{N-1}$ peak (the rms value will be this value multiplied by $\sqrt{2}$ because sensitivity is lower at higher phase values). We can now calculate the maximum phase jitter allowed by the clock:

$$\varphi_j = 2\pi T_j f_c = \frac{(f_c/f_s)}{2^{N-1}} \text{ rad, rms}$$

The table below calculates this jitter as a function of N:

N	8	10	12	14	16
$(f_s/f_c)\phi_j$	.008 rad	2 m rad	.5m	.125m	.06m

where m rad signifies radian/1000.

For example, for N=16, F_s = 1 MHz (1 μsec), F_c=10 MHz (.1 μsec) $T_j \approx 1000/32000 = 1/32$ nsec = 31.25 psec.

As a general rule of thumb, the following equation is practical:

$$\phi_j \approx (f_c/f_s)\cdot 2^{-(N+1)}$$

As expected, an increase in f_c relative to f_s (more samples per cycle) relaxes the requirement linearly, while an increase in N tightens it exponentially.

References

1. J. Tierney, C. M. Rader, and B. Gold, "A Digital Frequency Synthesizer," *IEEE Transaction of Audio Electroacoustics,* pp. 48–57, 1971.
2. H. T. Nicholas, "The Determination of the Output Spectrum of Direct Digital Synthesizers in the Presence of Phase Accumulator Truncation," Master's thesis, University of California at Los Angeles, 1985.
3. H. T. Nicholas and H. Samueli, "An Analysis of the Output Spectrum of DDFS in the Presence of Phase Accumulator Truncation," *Proceedings of the 41st Annual Frequency Control Symposium,* May 1987, pp. 495–502.
4. D. A. Sunderland, R. A. Strauch, S. S. Wharfield, H. T. Peterson, and C. R. Cole, "CMOS/SOS Frequency Synthesizer LSI Circuit for Spread Spectrum Communications," *IEEE Journal of Solid State Circuits,* August 1984, pp. 497–505.
5. Robert J. Bosselears, Phase-locked loop including arithmetic unit, U.S. patent 3913028, October 1975.
6. Francisco Cercas, M. Tomlinson, and A. Albuquerque, Designing with DDS.
7. Francisco Cercas, "DDFS for Frequency Hopped Spread Spectrum Systems," Master's thesis, IST-Lisbon, 1988.
8. J. Gorski-Popiel, ed., *Frequency Synthesis—Techniques and Applications,* IEEE Press, 1975.
9. A. Bramble, "Direct Digital Frequency Synthesis," *Proceedings of the 35th Annual Frequency Control Symposium,* May 1981, pp. 406–414.
10. B. Goldberg, Digital FS having multiple processing paths, U.S. patent 4958310, September 1990.
11. B. Goldberg, Digital FS, U.S. patent 4752902, June 1988.
12. H. Nicholas, H. Samueli, and B. Kim, "The Optimization of DDFS Performance in the Presence of Finite Word Length Effects," 42d Annual Frequency Control Symposium, 1988, pp. 357–363.
13. L. Rabiner and B. Gold, *Theory and Applications of Digital Signal Processing,* Vol. 4, Prentice-Hall, Englewood Cliffs, NJ, 1975.
14. E. J. Nossen, Digital FS, U.S. patent 4206425, June 1980.
15. B. Bjerede and G. Fisher, "A New Phase Accumulator Approach to FS," *Proceedings of IEEE,* May 1976.
16. C. E. Wheatley, Digital FS with random jittering for reducing discrete spectral spurs, U.S. patent 4410954, October 1983.
17. C. E. Wheatley and D. E. Phillips, "Spurious Suppression in DDS," *Proceedings of the 35th Annual Frequency Control Symposium,* May 1981.
18. C. E. Wheatley, "Spurious Suppression in DDS Using Accumulator Dither," Rockwell International, document WP81-3055.
19. F. Williams, "A Digital FS," *QST,* April 1984, pp. 24–30.
20. Roger Hosking, "FS—Direct, Indirect or Direct Digital," *Electronic Product,* December 1973.
21. R. Zavrel, "Digital Modulation Using the MNCO," *RF Design,* March 1988, pp. 27–32.
22. Leland Jackson, Digital FS, U.S. patent 3735269, May 1973.
23. E. M. Mattison and L. M. Coyle, "Phase Noise in DDS," *42d Annual Frequency Control Symposium,* 1988.
24. R. Kerr and L. Weaver, Pseudorandom dither for FS noise, U.S. patent 4901265, February 1990.
25. Robert P. Gilmore, DDS-driven PLL frequency synthesizer with hard limiter, U.S. patent 5028887, July 1991.

26. Frank Goodenough, "18-Bit Audio DACs Cut PCB Space Dramatically," *Electronic Design,* August 1990.
27. Hughes Aircraft and V. S. Reinhardt, Spurless fractional divider direct digital synthesizer and method, U.S. patent 4815018.
28. Hughes Aircraft and K. A. Essenwanger, direct digital synthesizer with selectable randomized accumulator.
29. I. Fobbester, "Spur Reduction in DDS," *Electronic Product Design,* June 1992, pp. 23–24.
30. M. P. Wilson, "Spurious Reduction Techniques for DDS," Colloquium on DDFS, University of Bradford, November 1991, *IEE Digest,* No. 1991/172.
31. M. Bozie, "Spurious Redistributing DDS," *IEE Digest,* 1991, p. 172.
32. COMSAT LABS, Evaluation of 16 KBPS voice processor, February 1991.
33. V. F. Kroupa.
34. B. Miller and B. Conley, "A Multiple Modulator Fractional Divider," 42d annual meeting.
35. V. Kroupa, "Spectra of Pulse Rate Frequency Synthesizers," *Proceedings of the IEEE,* vol. 67, December 1979, pp. 1680–1682.
36. Y. Matsuya et al., "A 16-Bit Oversampling A/D Conversion Technology Using Triple Integration Noise Shaping," *IEEE Journal of Solid State Circuits,* December 1987, pp. 921–929.
37. D. R. Welland et al., "A Stereo 16-Bit Delta Sigma A/D Converter for Digital Audio," *J. of the Audio Engineering Society,* June 1989, pp. 476–484.
38. R. M. Gray, "Oversampled Sigma-Delta Modulation," *IEEE Transaction of Communication,* May 1987, pp. 481–489.
39. Sciteq Electronics, San Diego, DDS-1 application notes.
40. Plessey, UK, "High Speed DDS: SP2002," data sheet.
41. Analog Devices, A Technical Tutorial on Digital Synthesis, 1999.
42. Analog Devices, "AD9850: Complete-DDS System" data sheet.
43. Stanford Telecom, "STEL-1177, STEL 2373," data sheet.
44. L. J. Kushner, "The Composite DDS: A New DDS Architecture," *Proceedings of the 1993 IEEE International Frequency Control Symposium,* June 2–4, 1993, pp. 255–260.
45. RF Design Staff, "Development of a New Direct VHF Synthesizer," December 1987, pp. 22–26.
46. Fred Williams, "A Digital Frequency Synthesizer," TRW LSI Product Division, La Jolla, Calif., April 1984, pp. 24–30.
47. Peter Asbeck, University of California at San Diego, private communication, February 1994.
48. J. M. Wozencraft and I. M. Jacobs, *Principles of Communication Engineering*, Wiley, New York, 1965, pp. 678–683.

This is a blank page.

Chapter

5

Phase-Locked Loop Synthesizers

Phase-locked loop (*PLL*) synthesizers are by far the most popular of all frequency synthesis techniques. Of the total market, in numbers of units used, PLL technology is probably more than 99 percent of the market and in dollar value probably more than 98 percent. A brief summary of the technique and the governing equations was presented in Chap. 1. Detailed reviews are available in Refs. 1 to 7. There is a great variety of excellent PLL books.

PLL, which started as a mainly analog technique, has been transformed to a major digital discipline as well, especially in the last 10 to 15 years. The reasons are mainly technological and market-driven. The market requirements are clear, with PLL synthesizers operating in every cellular or wireless phone and every other conceivable radio application, including hi-fis and car radios. With the advance of very high-speed dividers and highly integrated single-chip synthesizers and with the development of fractional-*N* synthesis, digital technology now operates in harmony with analog, *radio frequency* (*RF*), and microwave technologies.

The main focus of this chapter is to review the digital techniques but also mention progress achieved in overall PLL circuitry and architectures. Of special interest will be the review of fractional-*N* synthesizer concepts and architecture. This technique, which is the main breakthrough in PLL technology in the last 12 years or so (on top of technological advances and chip densities and functionality), is still not widely used. It resembles

direct digital synthesis (DDS) but operates as part of the feedback loop rather than as an independent signal generation discipline. The patenting of various fractional-*N* synthesis architectures and the relative complexity of the implementation have been a factor in the slow penetration of this important technique and its proliferation to the designer's market. On the other hand, there has been a reluctance to use the technique for reasons not clear to us. However, 1994 was the year when single-chip fractional-*N* PLL chips were introduced to the market, and we forecast a major shift to fractional-*N* synthesis applications, replacing classic PLL synthesis. Fractional-*N* PLL can be simple and easy to use. It should not be more expensive than classical PLL. With these points in mind, why not use the extra performance? Therefore, we are certain that the change will come very soon now.

5-1 Main Components of PLL Synthesis

Basically, a PLL circuit contains a voltage controlled oscillator (VCO), a divider, a phase detector, and a loop network or a loop filter. This is shown in Fig. 5-1. Most PLL circuits (not all) include a programmable divider and, of course, a reference input. The PLL transfer function was shown in (Chap. 1):

$$\frac{\varphi_o}{\varphi_i} = \frac{KH(s)}{s + KH(s)/N} \tag{5-1}$$

where $H(s)$ is the loop network transfer function.

The calculation of the performance of the PLL synthesizer depends on all its components. The VCO, the phase detector, the

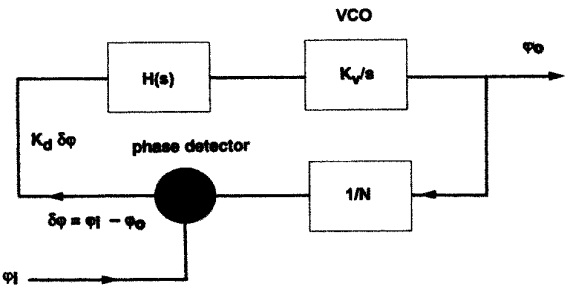


Figure 5-1 PLL block diagram.

reference generation, the loop filter—all these affect the performance, and all have to be weighted carefully for a demanding design.

5-1-1 Voltage controlled oscillators

There are a great variety of VCO implementations. At relatively low frequencies and in the VHF range, the most popular is a variant of the standard Colpitts design. A few examples are shown in Figs. 5-2, 5-3, and 5-4.

This design achieves wide bandwidth and reasonable phase noise performance. Obviously the noise generated at the output is

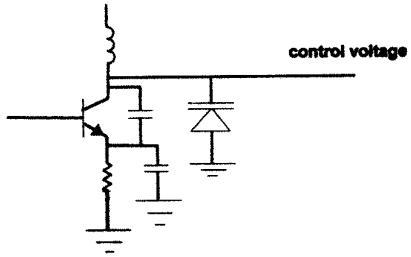


Figure 5-2 Colpitts-type VCO.

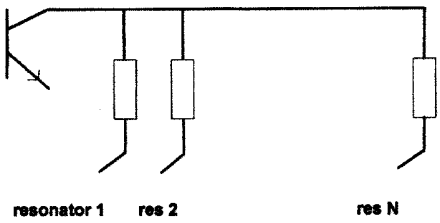


Figure 5-3 VCO using switched resonators.

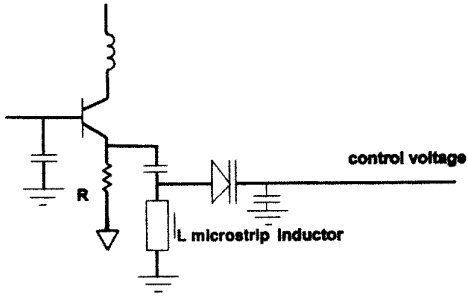


Figure 5-4 Negative-resistance VCO.

proportional to the VCO sensitivity. For a given loop amplifier whose analog circuit exhibits a certain noise characteristic, say, x nV/Hz$^{1/2}$, the output noise will be lower for a VCO with a lower K_v. So as a rule of thumb, it is desirable to operate with the lowest K_v possible. On the other hand, a low K_v value limits the output bandwidth. If one oscillator has $K_v = 6.3 \times 10^6$ rad/V · s, if another has $K_v = 63 \times 10^6$ rad/V · s, and if the tuning voltage available to the designer is 20 V, then the first will cover 20 MHz while the second can cover 200 MHz. It then becomes an engineering compromise between performance and complexity. General-purpose designs tend to use a single, as simple as possible, VCO. More demanding designs use either multiple VCOs or multiple resonators (as shown in Fig. 5-3). As an example, in the HP8662A, one of the industry's best, the main oscillators consist of five binary weighted resonators, covering together $2^5 = 32$ bands (see Ref. 27). This allows wide bandwidth coverage and low sensitivity at a very high increase of cost, size, and complexity.

In the UHF up to the lower S band, at 400 to 3500 MHz, the most popular VCOs in use today are the Colpitts and negative-resistance types. A typical structure is shown in Fig. 5-4. The term *negative resistance* came from the method of designing and analyzing these oscillators, by using their S parameters and noticing that s_{11} is a negative number.

A typical run is shown in Fig. 5-5. Between 2 and 18 GHz, the majority of the VCOs in use are either varactor-tuned *field-effect transistors* (FETs), achieving approximately an octave-band range though they usually do less, or transistor oscillators followed by multipliers and YIG tuned oscillators that achieve multioctave range. For YIG oscillators (and bandpass filters) a common bandwidth is 2 to 8 GHz and 8 to 18 GHz.

Above 20 GHz, most signals are generated by either multiplication or the use of negative-resistance diode oscillators, mainly Impatt or Gunn. Both techniques are utilized, up to about 110 GHz. Compared to multipliers, the diode VCOs have the advantage of generating substantial output power directly without the need to multiply and amplify. Obviously, there are a very large variety of oscillator designs and much literature, from audio to microwave, and the design parameters depend on the application.

As a general rule of thumb, the phase noise performance of a VCO depends on the active device noise figure, input-to-output

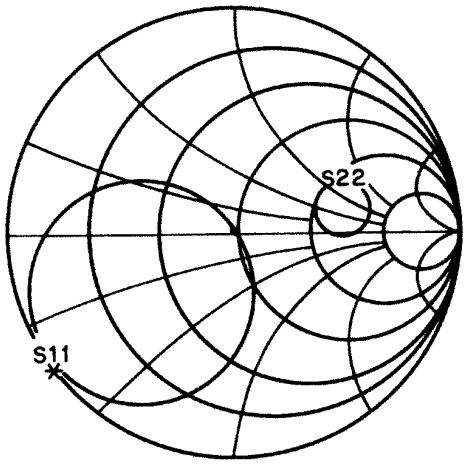


Figure 5-5 S_{11} (open loop) for negative-resistance VCO; F = 1200 MHz.

power ratio, and the quality Q of the resonator, with the latter being most important. The output power usually needs to be optimized since loading the resonance tank circuit (as in the Colpitts design) affects the loaded Q of the resonator. In the negative-resistance case, the output power is not derived directly from the resonator, and this provides a level of isolation between the resonator circuit and the output port.

A more detailed discussion is presented in Chap. 2. Typical general performance is shown in Fig. 2-8. Multipliers use nonlinear devices. The simplest forms are diodes and diode bridges, as shown in Fig. 5-6.

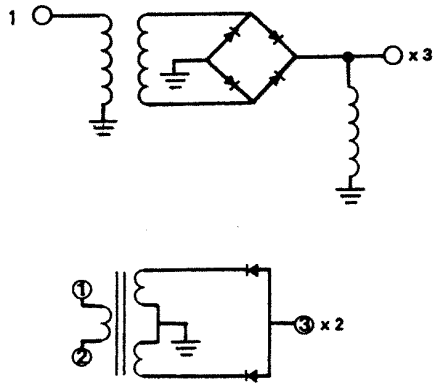


Figure 5-6 Multiple by 2 or 3.

Sometimes, transistors or FETs are used, too, operating in their nonlinear region. In microwave applications, varactor-tuned diodes as well as saturated FETs serve as very high-efficiency multipliers (multiples of 3, 4, 5 are quite common); however, most of these active multipliers are narrowband (below 10 percent bandwidth). When saturated active devices are used, more bandwidth can be achieved since the diodes are tricky to impedance-match.

Multiplication has been a cardinal microwave synthesis tool. However, lately, very high-speed dividers, up to 15 GHz, are available (see Fig. 5-46), and this helps to synthesize at higher frequencies. Digital technology is continuously making inroads into the microwave domain. One disadvantage of multipliers is the need for excellent filtering, to filter out subharmonics. Suppose that a signal is synthesized at L band, 1.5 to 1.6 GHz, and then tripled to C band, to generate 4.5 to 4.8 GHz. In the process of multiplying the frequencies, 1.5 to 1.6 and 3.0 to 3.2 GHz will have to be filtered out heavily, for these are spurious signals.

For comb generation, i.e., where F_i is used as an input and the output is NF_i, for $N = 1$ to 40, snap diodes [or *step recovery diodes* (*SRDs*)] are used and generate a signal that looks like an impulse and thus is saturated by its harmonics (see Fig. 5-7).

It must be emphasized that multiplications (by N) bear the penalty of loss of 20 log N dB in spurious signal and phase noise performance.

5-1-2 Analog phase detector

We will concentrate our discussion on digital phase detectors and review analog phase detectors briefly.

Almost any nonlinear analog device can be used as a phase detector. However, the use of analog phase detectors is limited because of their output characteristics. Suppose we use the most common mixer as a phase detector. The device is demonstrated in Fig. 5-8. Ideally, the device is a multiplier, and therefore it performs a transfer characteristic of

$$\begin{aligned} V_{\text{out}} &= A_1 \cos(\omega_1 t + \varphi_1) A_2 \cos(\omega_2 t + \varphi_2) \\ &= 0.5 A_1 A_2 [\cos(\omega_1 t + \omega_2 t + \varphi_1 + \varphi_2) \\ &\quad - \cos(\omega_1 t - \omega_2 t + \varphi_1 - \varphi_2)] \end{aligned} \tag{5-2}$$

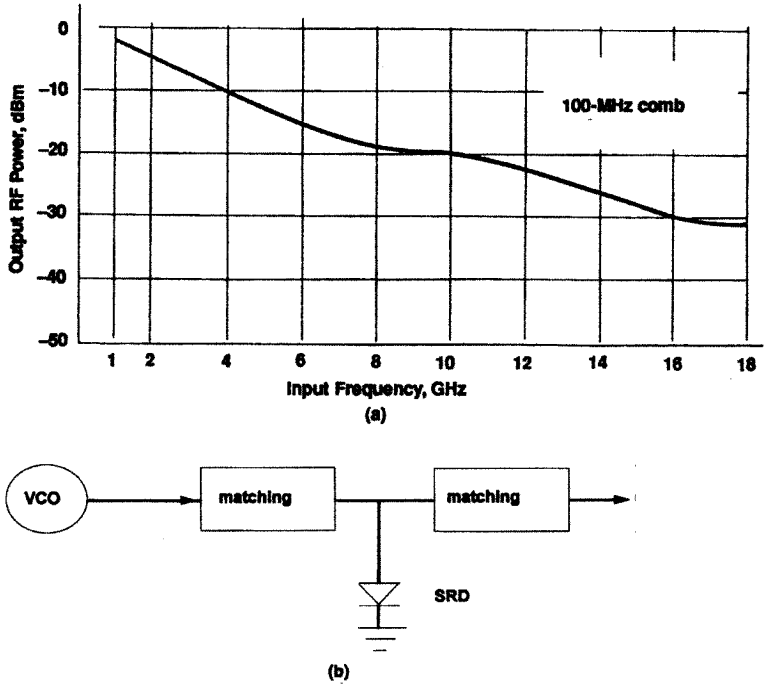


Figure 5-7 (*a*) Typical comb generator power; (*b*) comb generator using a step recovery diode.

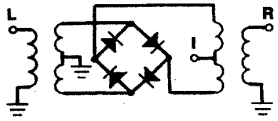


Figure 5-8 Double-balanced mixer used as a phase detector.

where A_1 and A_2 are the signals' amplitudes and $\omega_1 t + \varphi_1$ and $\omega_2 t + \varphi_2$ are their corresponding phases. Usually, the term $\omega_1 + \omega_2$ can be filtered out. Thus the phase detector output can be rewritten as

$$V_{\text{out}} = K_d \cos(\omega_1 t - \omega_2 t + \varphi_1 - \varphi_2) \tag{5-3}$$

The output of the detector is a sine (or cosine) waveform, and it will produce a dc output if $\omega_1 = \omega_2$, whose value depends on the value of $\varphi_1 - \varphi_2$ (see Fig. 5-9).

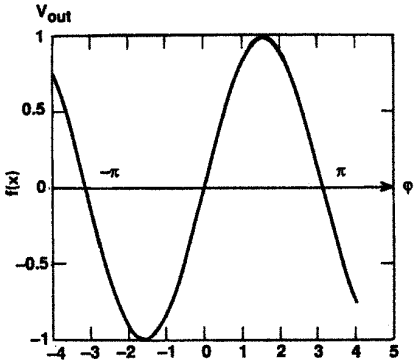


Figure 5-9 Mixer output waveform when used as a phase detector.

In the case that ω_1 does not equal ω_2 (that is, the loop is not locked), an error voltage will develop according to $\sin(\omega_1 t - \omega_2 t)$. If $\text{abs}(\omega_1 - \omega_2)$ is large enough (meaning that it is filtered by the PLL filter and integrator), as is the common case, then the output of the phase detector shall be attenuated by other components in the PLL circuit, and therefore the error voltage will be tiny and not enough to pull the circuit into a locked condition in a reasonable amount of time. The lockup time then becomes prohibitively long. A detailed analysis of the behavior of such a PLL circuit has been presented by the classic work of Viterbi (Ref. 7).

As a demonstration, suppose that the block diagram of a tentative PLL circuit is as shown in Fig. 5-10. Suppose that the circuit is locked for $N = 100$, and the output produces 100 MHz. Now the frequency needs to change to 200 MHz, and so N is changed to 200. Instantly the output of F_v becomes 0.5 MHz, and the output of the phase detector generates a sine wave whose output frequency is $1 - 0.5 = 0.5$ MHz. If the network $F(s)$ has a low-pass response that cuts off at 10 kHz, the error signal will be attenuated enormously and the time to lock will be intolerable.

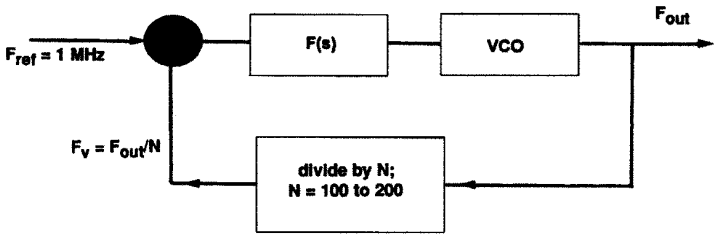


Figure 5-10 A 100- to 200-MHz PLL synthesizer.

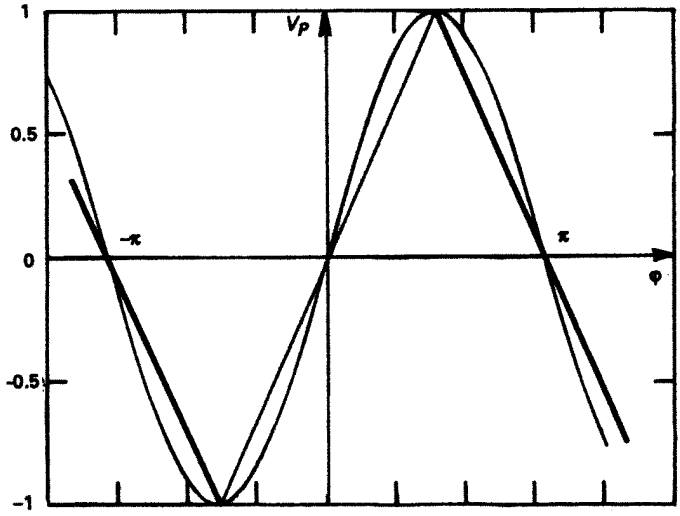


Figure 5-11 Different phase detector characteristics.

A very similar response appears at the output of a phase detector comprised of an exclusive-or gate, except that instead of a sine waveform, the output is a triangle, as shown in Fig. 5-11. The problem in these phase detectors lies in the fact that it is not possible to distinguish between a given phase and that phase $\pm 2\pi N$. Mixers are very low-noise detectors and therefore are desired and should be used, but mainly if the frequency range to cover is low. For example, if some fixed references need to be generated by using crystal VCOs [or other very high-Q resonators which also limit the possible output frequencies such as crystals or *surface acoustic wave* (*SAW*) resonators] and locking them to some crystal references, then a mixer is a very good choice since digital phase detectors are noisier.

In the example shown in Fig. 5-12, the voltage controlled crystal oscillator (VCXO) can change its output only ± 5 kHz around 83 MHz, so the error generated will diminish in a reasonable time, and since it is a fixed frequency, it need not change during operation. Note that the largest frequency error at the detector is $\pm 5000/83 = 60.24$ Hz.

The noise generated by the mixer phase detector is excellent since there are no active devices in the circuit and noise levels (very low) of -165 dBC/Hz are common. This is approximately 9 dB above kT, the "natural noise," -174 dBm/Hz, at room temper-

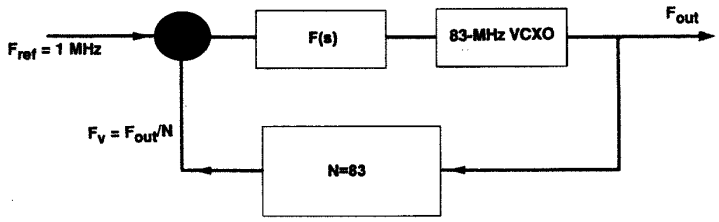


Figure 5-12 An 83-MHz crystal PLL.

ature. Such a level of performance is very hard to achieve with active digital devices.

In the example shown in Fig. 5-10, if we knew that only 1-MHz steps would be applied, then the frequency error to the phase detector would not exceed 10 kHz and a mixer phase detector could be used. Sometimes the VCO is pretuned to a frequency that is close to the desired one so that such a type of phase detector could be used, and the possibility of a large frequency error is avoided.

5-1-3 Digital phase detector 1

These types of phase detectors are digital circuits. They receive digital inputs and produce an output that reflects the phase between the (usually) rising edge of their input signals. A typical transfer characteristic is shown in Fig. 5-13. This is very desirable since an unlocked condition (as opposed to the analog phase detector) is characterized by a different output than a lock condition. However, because of the digital nature of the device, the phase

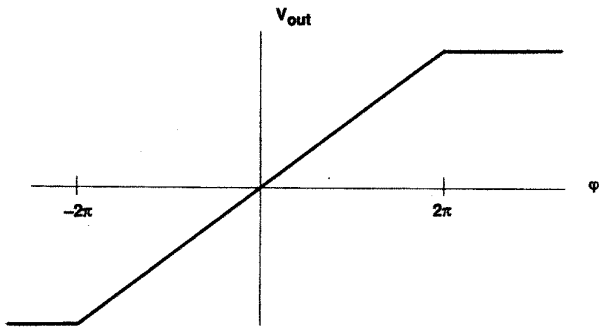


Figure 5-13 Phase frequency detector 1.

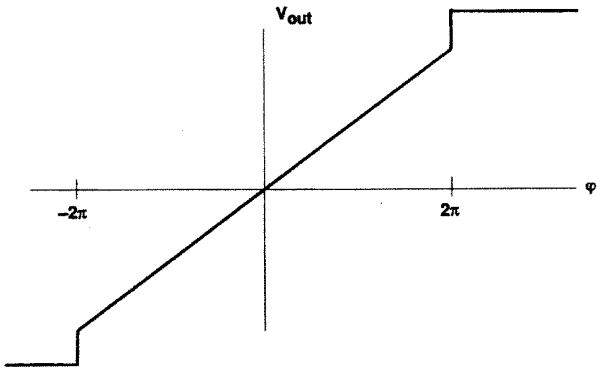


Figure 5-14 Phase frequency detector 2 (with speed up).

comparison is performed only at the signal's zero crossing, or when it changes state from 0 to 1 (or 1 to 0).

Ideally, we might wish a transfer function that looks like that in Fig. 5-14, and different circuits perform an equivalent function. The function should be such that when it is in the unlocked condition, a large error should be developed and steer the VCO in the right direction, i.e., toward lock.

The loop equation will now take two forms. One occurs in the out-of-lock state, and the other occurs when it is close to the locked state. Thus the loop equations derived in Chap. 1 are correct for the case of close to lock, which was an assumption in the development of the equations. However, when it is out of lock, the phase detector generates a fixed voltage ($\pm A$ V), and this is integrated twice (in the loop filter) until it saturates the analog circuitry controlling the VCO, thus providing the faster path to the lock state.

For the standard second-order loop, the equation is given (by viewing the equivalent block diagram shown in Fig. 5-15) by

$$\frac{\varphi_o(s)}{\varphi_i(s)} = \frac{KH(s)}{s + KH(s)} \qquad H(s) = \frac{R_2}{R_1} + \frac{1}{sCR_1} \tag{5-4}$$

The solution of this second-order differential equation depends on ω_n and ξ. For an underdamped case where $\xi < 0.5$, the solution is oscillatory, as shown in Fig. 5-16 (see Chap. 1). For an overdamped case where $\xi > 0.5$, the asymptotic solution is shown in the same figure. (All solutions are in the linear region of the

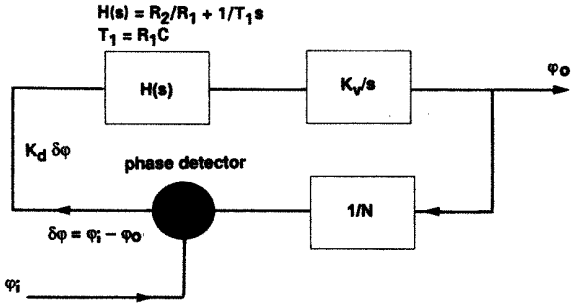


Figure 5-15 PLL block diagram.

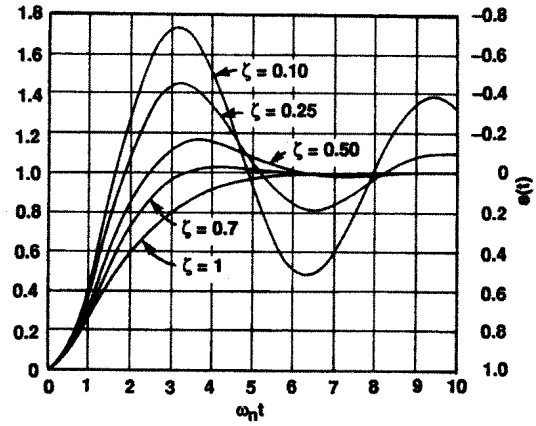


Figure 5-16 Phase settling as function of the damping factor.

phase detector.) In both cases the behavior of the phase is settling asymptotically.

However, the out-of-lock condition changes the equation to

$$\varphi_o = \frac{\pm AK_v}{s[R_2/R_1 + 1/(R_1Cs)]} \tag{5-5}$$

This is a double integration characteristic that accelerates the phase to the state when the absolute value of the phase error will be less than 2π and will bring the equation back to the previous case.

The acceleration of the phase in the out-of-lock state is very desirable since the lock time in most cases needs to be as short as possible. A demonstration of a computer simulation of tentative

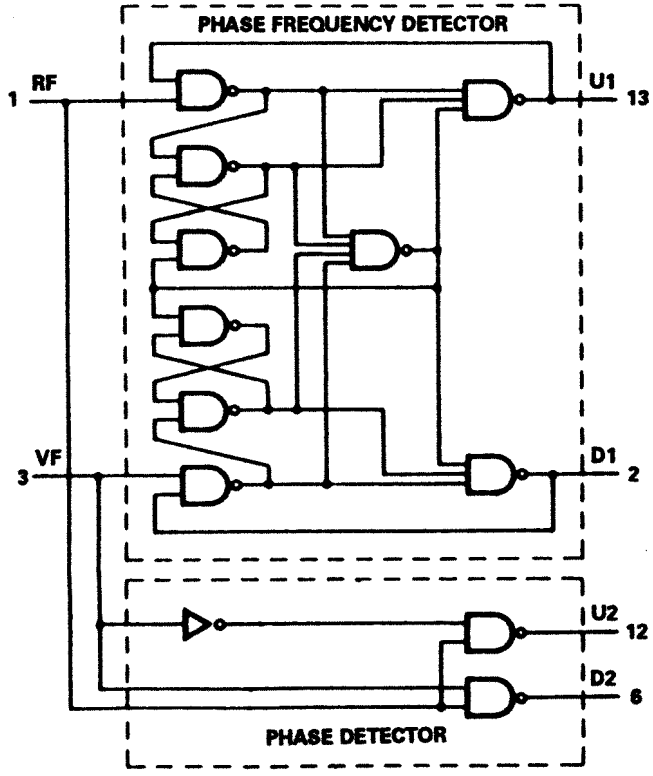


Figure 5-17 MC 4044 or 4344 phase detector. (*Courtesy of Motorola.*)

designs is available now in a great variety of computer software that enables total simulation of the loop behavior, including all nonlinearities, additional poles in the loop, and finite speed of all components involved.

The first configuration of such a phase detector is shown in Fig. 5-17. Since these phase detectors can sense when the frequencies are different, they are also referred to as *phase frequency detectors.* Typical waveforms are shown in Fig. 5-18.

5-1-4 Digital phase detector 2

A very popular phase frequency detector is the same type as Motorola part MC12040. The block diagram and truth tables are given in Fig. 5-19 and Table 5-1, respectively. The same circuit is also implemented in Motorola or Fairchild part MC4044. Its characteristics are similar to those shown in Fig. 5-13.

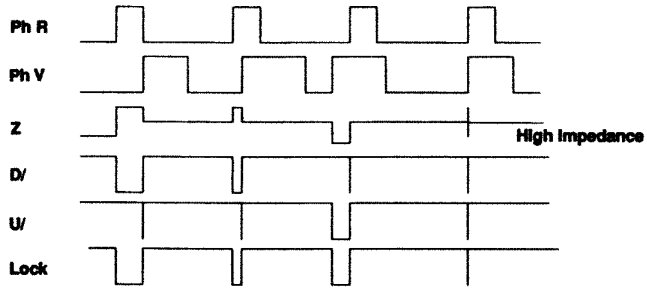


Figure 5-18 Phase detector outputs. Ph R is the reference signal; Ph V is the variable.

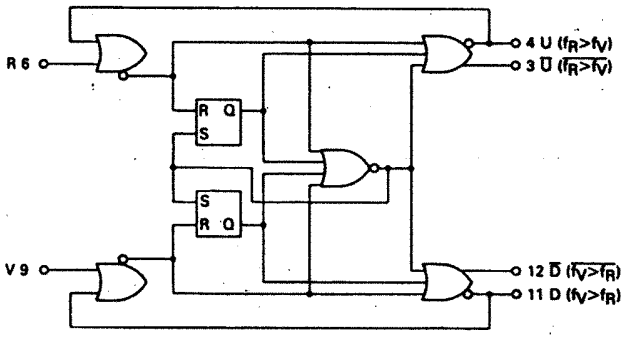


Figure 5-19 Digital phase frequency detector. (*Courtesy of Motorola.*)

5-1-5 Digital phase detector 3

Another very popular digital phase detector is a combination of two edge-triggered flip-flops and an AND gate, shown in Fig. 5-20. Waveforms in lock condition are shown. Logic state diagrams are shown in Fig 5-18.

Very similar performance and characteristics can be achieved with the IMI 4385, a CMOS phase frequency detector, which provides also a lock indicator and a tristate output.

In PLL ASICs, and specifically in wireless applications, the most popular Phase Frequency Detector (PFD) is a variant of Figure 5-20, shown in Figure 5-20a.

The reference frequency F_r is compared to the feedback signal coming from the VCO after division, F_{vco}/N. However, the output is connected to two current sources, one that can drive (+) and the other sink (–). This configuration is very convenient because it does not require any active devices to convert the phase error to voltage, and allows generation of the analog signal (the output) using digital devices (each having an ON state and a high-impedance state). In lock, the waveforms look like tiny "spikes" at the reference frequency F_r. See Waveforms in Figure 5-20b and c.

5-1-6 Digital/analog phase detector 4

Another innovative but not very popular phase detector is shown in Fig. 5-21. It is truly an analog/digital phase detector. The ramp generator generates a ramp signal whose period is the period of the reference signal. The F_v signal (the one coming from the VCO) is

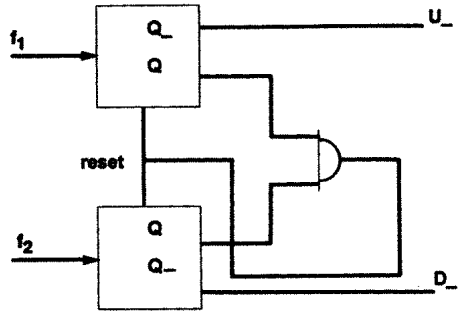


Figure 5-20 Digital phase frequency detector.

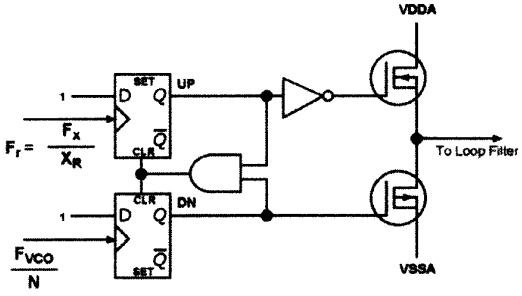


Figure 5-20a Phase frequency detector used in wireless ASICs.

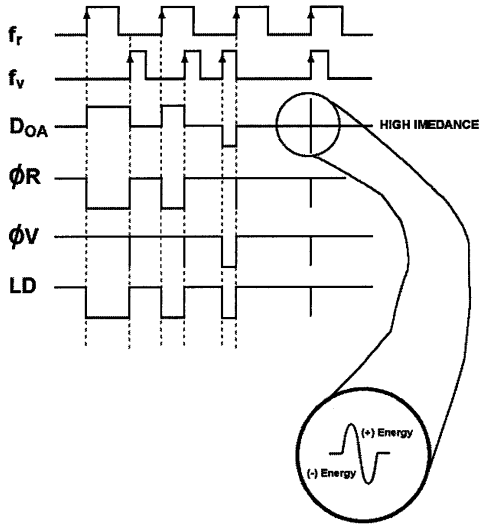


Figure 5-20b Waveforms in lock condition.

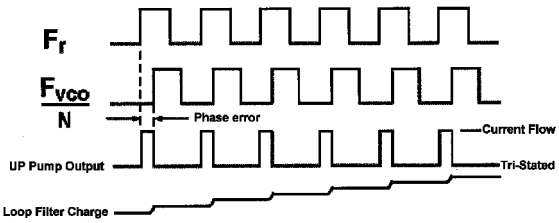


Figure 5-20c F_r and $\frac{F_{vco}}{N}$ waveforms.

TABLE 5-1 Truth Table of Phase Frequency Detector

Input		Output			
R	V	U	D	$\overline{U}$	$\overline{D}$
0	0	X	X	X	X
0	1	X	X	X	X
1	1	X	X	X	X
0	1	X	X	X	X
1	1	1	0	0	1
0	1	1	0	0	1
1	1	1	0	0	1
1	0	1	0	0	1
1	1	0	0	1	1
1	0	0	0	1	1
1	1	0	1	1	0
1	0	0	1	1	0
1	1	0	1	1	0
0	1	0	1	1	0
1	1	0	0	1	1

X = don't care.

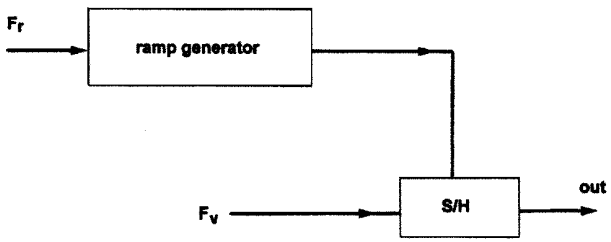


Figure 5-21 Sweep sample-and-hold (S/H) phase detector.

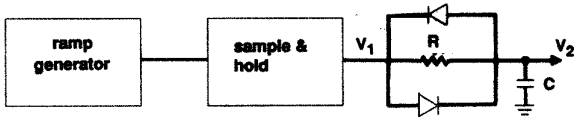


Figure 5-22 This is the same as Fig. 5-21 with a speedup circuit.

sampling and holding this signal at the time when it changes states from 0 to 1. Since the ramp covers the voltage range necessary to control the VCO across the desired frequency range, lock conditions will be achieved when this signal is sampled at the right time, therefore generating the right voltage to generate a lock. The loop filter is now different since the ramp performs the function of an integrator and is already built into the phase detector function.

This circuit has good noise characteristics, but its frequency of operation is limited to about 5 MHz. It is not a simple task to design a linear ramp circuit, ramping, say, 0 to 20 V at a speed of above 5-MHz rate. There are also problems of shielding this large ramp in a practical circuit. However, for the added complexity, one gets good noise performance and a relatively simple speedup mechanism, as shown in Fig. 5-22. The diodes are speeding up the charge of the hold capacitor C, but stop to function when V_1 gets closer in value to V_2, which is the lock condition. This a simple way of dynamically changing the loop parameters (changing R). Design parameters for loop equations are given in Eqs. (5-6) to (5-10).

5-1-7 Dividers

The dividers are an integral part of the PLL circuit. As a rule of thumb, the division ratio must be minimized to reduce phase noise, as we will see in Sec. 5-2.

A great variety of divider chips are available to the designer. While Mitel, Fujitsu, NEC, Motorola, and National SC (see Sec. 5-4) offer many combinations, designers need to weigh their needs and choose the best fit for a specific job. It is necessary to evaluate the divider phase noise performance, power dissipation, control interface, and so on.

With very few exceptions, the dividers are digital devices and operate up to 15 GHz! Some are fixed dividers (also called *prescalers*), others are programmable; and the more interesting

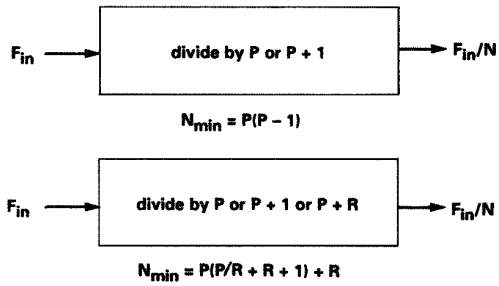


Figure 5-23 Dual- and trimodulus dividers.

ones are dual-modulus, capable of dividing by P or $P + 1$ (and sometimes P or $P + R$, for example, 128/144; see Fujitsu P/N MB 510), or three-modulus. The reason is clear since a synthesizer usually covers some bandwidth and the total division ratio is a programmable number from, say, N_1 to N_2. A dual-modulus device is a very convenient means to produce a range of continuous division. A minimum divide ratio is shown in Fig. 5-23.

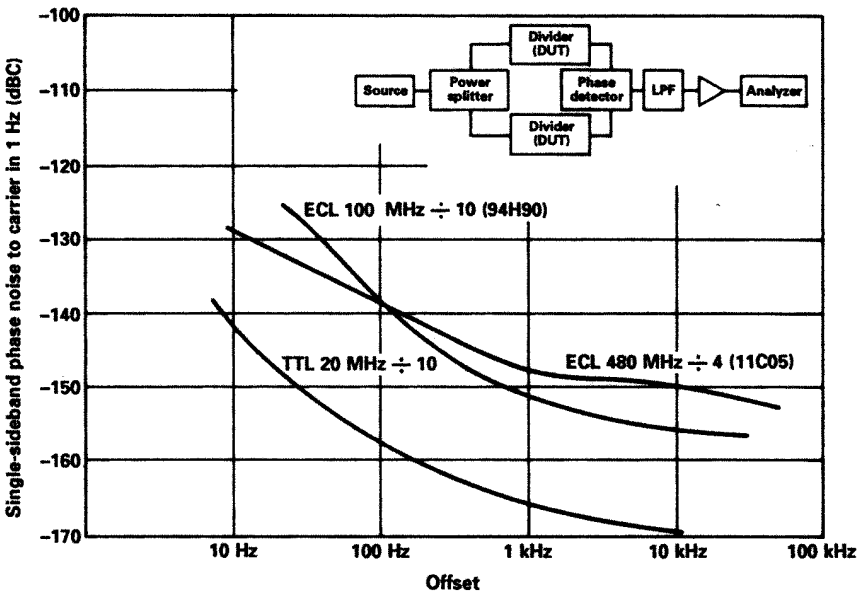


Figure 5-24 TTL/ECL divider phase noise; speed is below 500 MHz. (*Courtesy of Hewlett Packard.*)

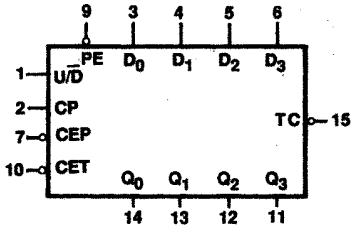


Figure 5-25 Programmable counter.

The general phase noise performance of digital dividers has been summarized in Ref. 27 and is shown in Fig. 5-24. When a fixed number of dividers is necessary, a programmable counter can be used, as shown in Fig. 5-25.

The $U/\overline{D}$ (count up or down) is connected to TTL low, which makes the counter a countdown mode. When a control hexadecimal word is connected to the input, say C hexadecimal, which is 12 decimal, the $\overline{PE}$ function will preset the counter to C hexadecimal when the counter counts down to 0, and the $\overline{TC}$ goes to 0. If we start the count at this point, the counter will count down 12 clock ticks and will use an extra count for the preset state. Thus the count in this configuration is always to $N + 1$ if it was controlled to N. This is a common counter configuration used in synthesizer designs. The extra count can usually be compensated for in a more complex design. A BCD counter is shown in Fig. 5-26.

In most designs, especially when the frequencies out of the VCO are high, it is very convenient to use a dual-modulus counter, such as divide by $^{10}/_{11}$, $^{16}/_{17}$, $^{20}/_{21}$, $^{40}/_{41}$, etc. A common design is shown in Fig. 5-27. To divide by 125,765 so that the output frequency will be 125.765 MHz, the divide by $^{10}/_{11}$ is set to divide by 11 for 5 times and by 10 for 12,571 times. The result is to divide by $12{,}571(10) + 5(11) = 125{,}765$ (M controls the 11, N the 10).

Note that in most counters, even those that are synchronous, TC is not a synchronous function. That causes multiple transients in the output and requires some type of registration.

In other designs, the translation from the command word to the counter's input can be manipulated by using a PROM or an EPROM device. This offers a lot of flexibility if the output frequency needs to be adjusted across a wide frequency band and if it is necessary to use different devices such as divide by $^{10}/_{11}$ or $^{5}/_{6}$ in a generic design. It also allows the designer a control mechanism convenient to (and sometimes required by) the user.

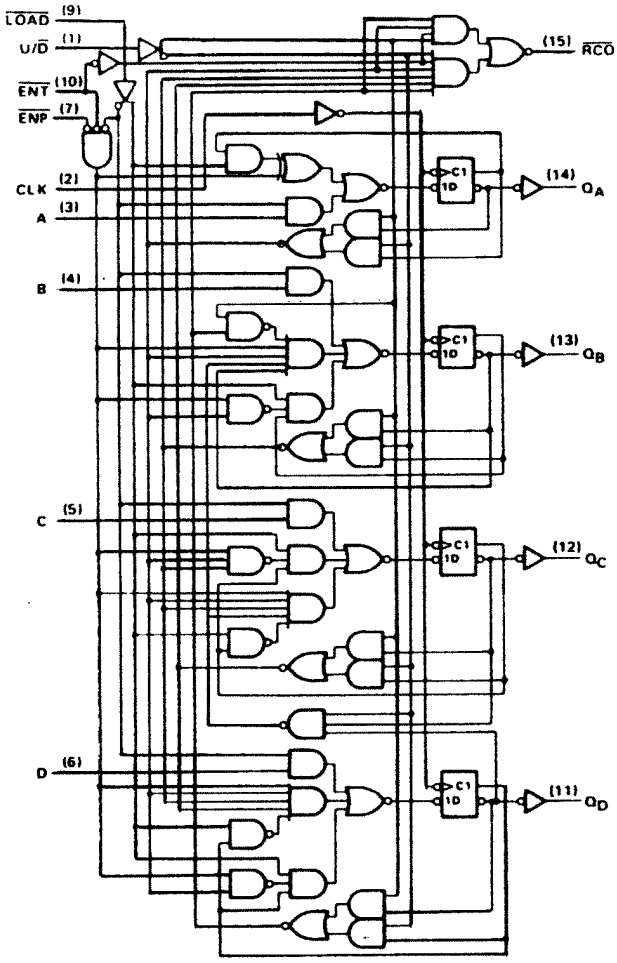


Figure 5-26 Programmable BCD counter. (*Courtesy of Signetics.*)

Some complex divider chains, part of general synthesizer design, can be found in instruments mentioned in Chap. 9.

Many new devices have integrated functions built in and thus can generate a variety of reference frequencies, have a wide range of division ratios, include one or more phase detectors, and operate up to 5 GHz, although the majority of the "single-chip" synthesizers operate up to about 30 MHz and require a dual modulus or a high-frequency divider to interface between their functions and the high-frequency VCO. These products are surveyed in

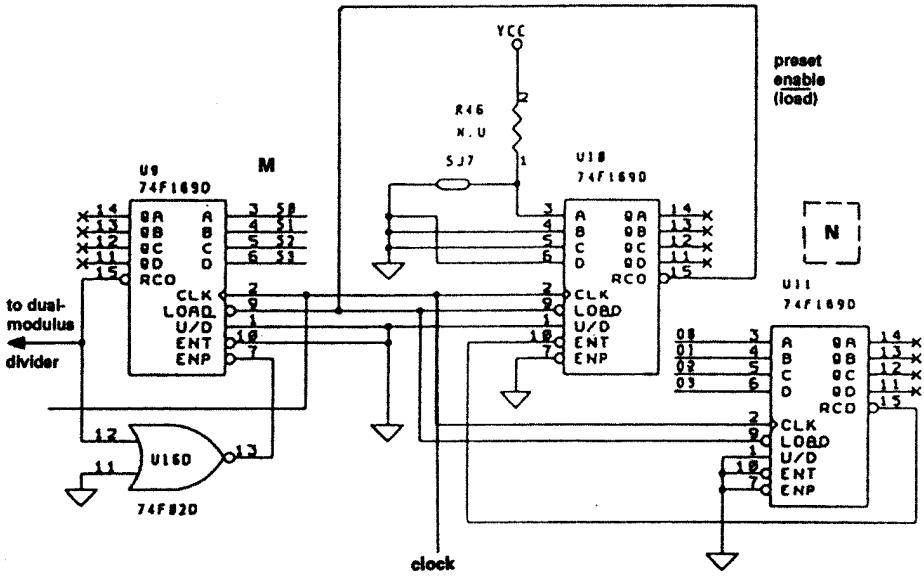


Figure 5-27 Complex programmable counter. Total count is $M(P + 1) + (N + 1 - M)P$, where P and $P + 1$ are the values of the dual modulus. N can be as large as is necessary; it is two stages here.

Sec. 5-4. Some devices, such as the Fujitsu MB15XX family, have all the functions, including the reference divider, dual modulus, and phase detector integrated in the device.

The minimum number for continuous division in an $N/(N + 1)$ device is $N^2 - N$ or $N(N - 1)$. For example, in a 10/11 device, the minimum continuous number is 90, and the number 89 cannot be generated since there is no combination of 10 and 11 that can generate 89. For a three-state divider (P or $P + 1$ or $P + R$), the general equation for the lowest divide ratio is

$$N_{\min} = P(P/R + R + 1) + R$$

5-2 Performance Evaluation

For the general PLL synthesizer having the block diagram shown in Fig. 5-28, the various transfer functions $H(s) = \varphi_o(s)/\varphi_i(s)$ are given below.

$$H_1(s) = \frac{K_d K_v F(s)}{s + K_v K_d F(s)/N} \qquad \text{from input to point 1} \qquad (5\text{-}6)$$

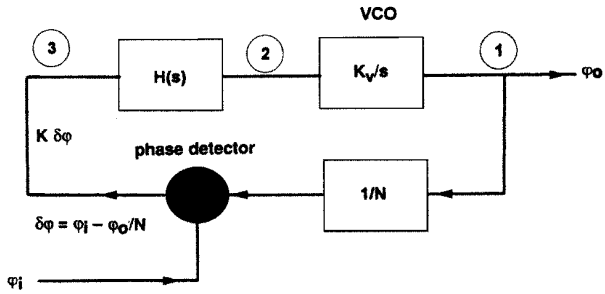


Figure 5-28 Single-loop PLL diagram with divider.

$$H_2(s) = \frac{K_d F(s)}{s + K_v K_d F(s)/N} \qquad \text{from input to point 2} \qquad (5\text{-}7)$$

$$H_3(s) = \frac{K_d}{s + K_v K_d F(s)/N} \qquad \text{from input to point 3} \qquad (5\text{-}8)$$

$$H_4(s) = \frac{K_v F(s)}{s + K_v K_d F(s)/N} \qquad \text{from point 3 to output} \qquad (5\text{-}9)$$

$$H_5(s) = \frac{K_v}{s + K_v K_d F(s)/N} \qquad \text{from point 2 to output} \qquad (5\text{-}10)$$

The transfer function $H_1(s)$ is the one that shows the output dependence on all components. The equation can be rewritten as

$$\varphi_o = \frac{\varphi_i N K_d K_v F(s)}{sN + K_v K_d F(s)} \qquad (5\text{-}11)$$

A plot of abs $H(s)$ for various ξ values (damping factors) is shown in Fig. 5-16, where ω/W_n was normalized to 1 and $F(s)$ is either 1 (or a fixed gain G) for a first-order loop or an integrator plus adder for a second-order loop.

Thus, for a first-order loop we have

$$H_1(s) = \frac{NK_v K_d}{Ns + K_v K_d} \qquad \text{equivalent for } W_n \text{ in this case is } K_v K_d$$

For the second-order loop we have shown that

$$H_1(s) = \frac{NK_v K_d(1 + s\tau_2)}{N\tau_1 s^2 + K_v K_d \tau_2 s + K_v K_d} \qquad (5\text{-}12)$$

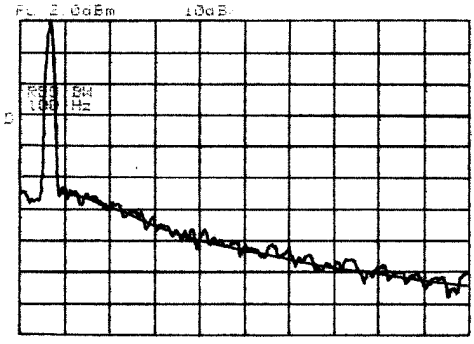


Figure 5-29 Phase noise sources at 1 GHz, 10 dB per division.

Inside the loop bandwidth, the phase noise will be that of the reference and the phase detector noise multiplied by N, and is thus corrupted by $20 \log N$ in decibels. Outside the loop bandwidth, the VCO noise will be dominant. Typical output spectra are shown in Figs. 5-29 and 5-30a. A typical drawing is shown in Fig. 5-30b. The various equations given above provide valuable insight into the operation of PLL and the effect of different noise sources.

There are three main noises that affect total output performance: reference noise, phase detector noise, and the VCO. The VCO is clearly very noisy close to the carrier, but becomes much better as the offset increases. We would therefore like to have the output acquire the characteristics of the crystal close to the carrier (and therefore acquire its stability and accuracy), while further away, the VCO should be dominant. This is exactly what PLL does for us.

The effects of the reference (crystal) and the phase detector are very similar and generally take the form:

$$H(s) = NKF(s)/sN+KF(s) = N(2\xi\omega ns+\omega n^2) / (s^2+2\xi\omega ns+\omega n^2).$$

At low frequencies, where $s<<\omega n$, the function value is N. The output will follow the input and noise will be multiplied N times or $20\log N$ dB. However, as the frequency offset increases, the transfer function looks like a low-pass filter and noise (from the crystal and PFD) is attenuated. For the VCO, the transfer function calculates to:

$$H(s) = s^2/(s^2+2\xi\omega ns+\omega n^2)$$

This equation has "high pass" characteristics. At low frequencies, $s<<\omega n$, the transfer function calculates to $s^2/\omega n^2$. There will be significant attenuation of the VCO noise close to the carrier! At higher offsets, the transfer function becomes 1, indicating that the output will follow the VCO noise characteristics.

A complete simulation for these three noise sources is shown in Figure 5-31, showing a 400-Hz bandwidth PLL, using Eagleware simulation CAD (see reference 36).

Curve *a* in Fig. 5-30a is of a synthesizer covering 800 to 1200 MHz and using 1 MHz as a reference (the division ratio *N* is approximately 1000). Even though 1 MHz is derived from an excellent 10-MHz crystal oscillator which is improved by another 20-dB by the divider (divide by 10), bear in mind that the divider has a noise floor so the calculated performance is limited by the phase detector and divider performance (see Fig. 5-24).

The second (and usually dominant) noise source is the phase detector, and it is approximately −145 dBC/Hz in many products. As a result, the real performance is much degraded relative to the

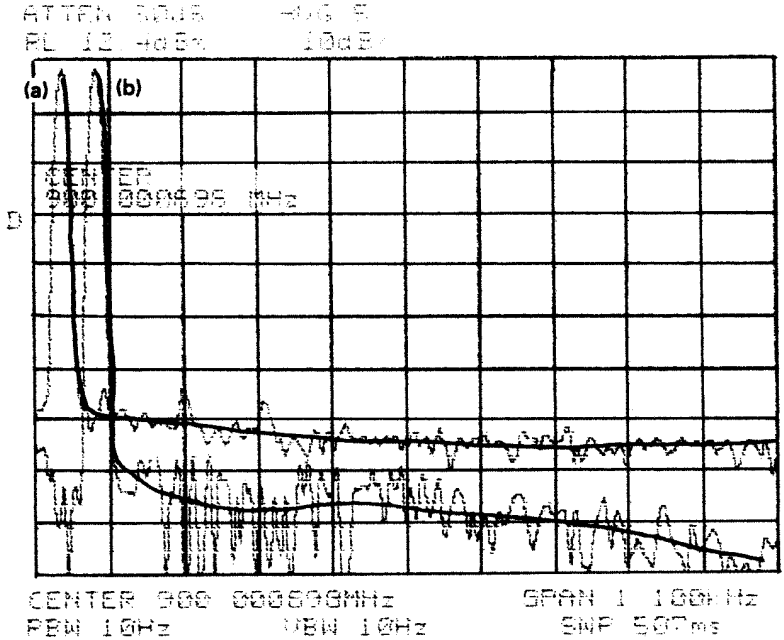


Figure 5-30a Phase noise performance of two lab instruments; 10 dB per division, span 1.1 kHz. The better performance is actually limited by the spectrum analyzer.

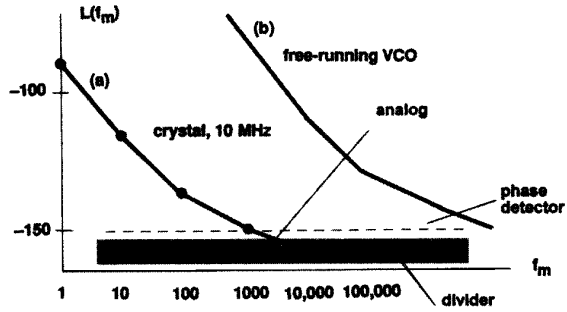
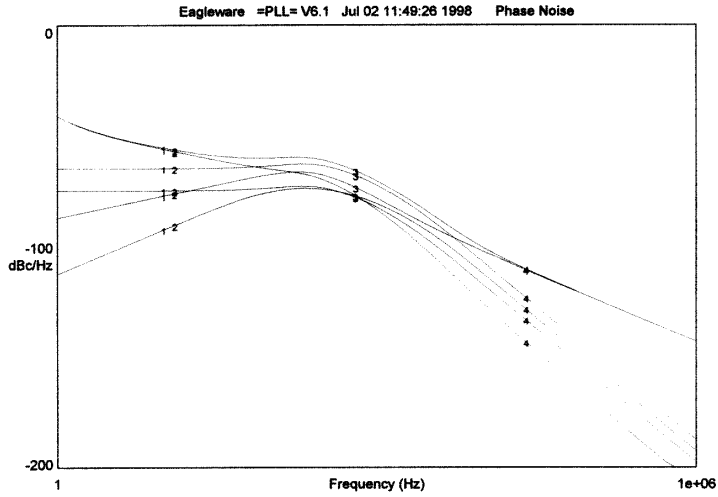


Figure 5-30b Phase noise sources.



Frequency	10	12.5893	630.957	25118.9
Ref	-56.7079	-57.7326	-77.8387	-143.669
PD	-65.2974	-65.2878	-68.593	-123.613
Integ	-92.9772	-91.0007	-77.9213	-128.659
Divider	-75.2974	-75.2878	-78.593	-133.613
VCO	-77.6728	-76.6659	-74.0447	-111.187
Total	-56.0616	-56.9177	-66.496	-110.847

Figure 5-31 Individual and composite phase noise.

first approximation (which would assume that the reference phase noise at 1 kHz offset from the carrier would reach −165 dBC/Hz). It is evident that the phase detector phase noise is dominant.

Curve *b* in Fig. 5-30a is a synthesizer covering 30 to 50 MHz and using 100 kHz as a reference (the division ratio N is approximately 400). The phase detector in this application is better-quality, and the noise performance is limited close to the carrier by

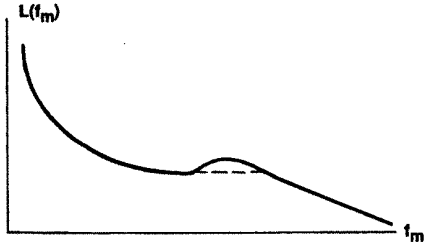


Figure 5-32 Phase noise for nonoptimal parameters.

a value of $-110 - 20 \log 400 = -110 - 52 = -162$ dBC/Hz. The performance in this case is affected by both the reference and the phase detector.

Figure 5-43 is a crystal oscillator at 160 MHz locked to a 1-MHz reference, using a mixer as a phase detector. Almost all the noise originates from the reference.

In many designs, the phase noise curve has a "shoulder." This is created because of the difficulty in controlling the loop parameters across the full range of operation. See Fig. 5-32. The problem is that both N and K_v change, and they add rather than cancel. For example, in the case of a PLL circuit that covers 500 to 1000 MHz, using 1 MHz as a reference, N changes 2:1 and K_v can change 2:1. Thus w_n (loop bandwidth) changes 2:1 [$w_n = \text{constant}(K_v/N)^{1/2}$] and ξ (the damping factor) changes 2:1.

These changes can be compensated at the cost of circuit complexity or parameter (K_v or K_d) control. The additional cost is not desired if it can be avoided. Sometimes the compensating circuit can reside in a ROM and feed the phase detector output voltage (or current) so that the total loop constant $K_v K_d/N$ stays constant. Other applications use nonlinear devices, as shown, e.g., in Fig. 5-33.

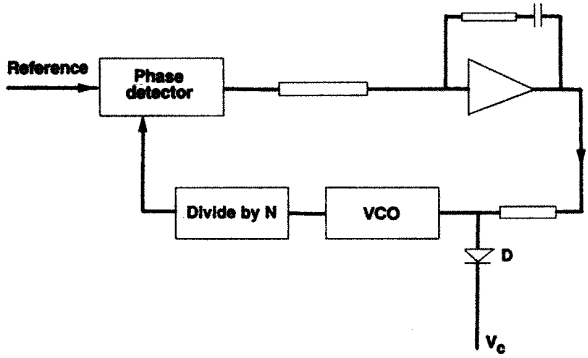


Figure 5-33 Loop linearization.

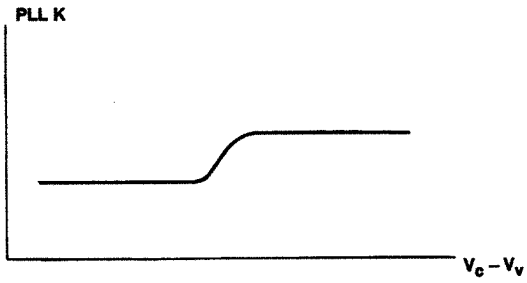


Figure 5-34 PLL loop linearization.

TABLE 5-2 Phase Noise and Floor of Typical Instrument Sources, PLL and Direct Analog, All Calibrated to 1-GHz Output

	Offset (Hz)					
Instrument	10	100	1000	10,000	100,000	1,000,000
Fluke 6060B	−65	−70	−75	−100	−120	−140
HP8657B	−75	−80	−95	−126	−148	−148
HP 8662	−100	−112	−120	−131	−132	−140
PTS300*	−85	−95	−105	−115	−120	−122
FS2000†	−85	−95	−110	−130	−138	−138

*Direct analog from PTS.
†Direct analog from Comstron.

Diode D is a Zener diode and is off if the voltage across it is below its breakdown voltage. But if $V_c - V_v$ is above this voltage, then the diode is on and the control voltage is divided. The complete characteristic is shown in Fig. 5-34. This is a simple but quite effective circuit. Obviously more than one diode can be used to achieve a better curve fitting to the desired characteristics.

Typical phase noise performances of various instruments (all normalized to 1-GHz output frequency) are shown in Table 5-2.

Another important characteristic is the switching speed. To increase the speed, the loop must be wide. This affects its ability to "clean" the signal (see Fig. 5-29). However, if the speed is necessary, the loop must be wider. Some other techniques to improve speed are as follows:

The VCO is pretuned. This is done so that when the frequency is changed, the VCO is pretuned close to the desired frequency, and then the phase lock mechanism completes the locking process. Since the VCO characteristics are known, it is possible to tune the

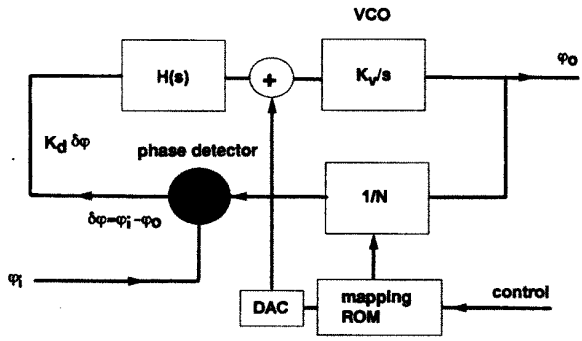


Figure 5-35 PLL circuit with speedup.

device quite close to the target value. Temperature has an effect, and it needs to be compensated, too. A tentative circuit is shown in Fig. 5-35. When tuned to a new frequency, the control signal is mapping the correcting voltage through a ROM which stores the VCO characteristics. The characteristics are also affected by the temperature, which is monitored by using a temperature-sensitive device (special diodes, thermistors, etc.). The loop filter $H(s)$ needs to correct only a small voltage error. This technique improves speed by up to 10:1.

Another technique is to change the loop bandwidth dynamically. When it is out of lock, make the loop very wide so it acquires fast. Then slow it for the best noise performance. The switching of the loop parameters sometimes causes transients that "throw" the loop off lock, and this has to be controlled. One simple way to improve speed is shown in Fig. 5-22 and 5-36.

In both cases, when the frequency error is large, the voltage across the diodes is high enough to turn one of them (either positive or negative voltage) on, thus reducing the effective value of R_1

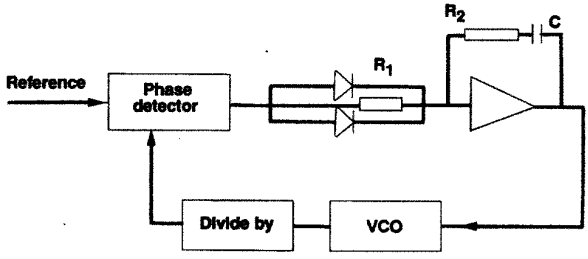


Figure 5-36 Speedup PLL.

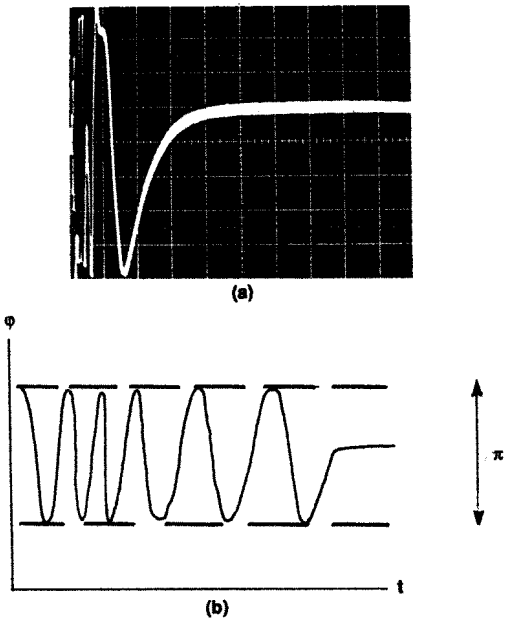


Figure 5-37 PLL transient (*a*) with and (*b*) without speedup.

to the resistance of the diode, which should be much lower than R_1. As a consequence, the capacitor can charge to close to the desired voltage fast; but as the control voltage to the VCO approaches the nominal value, the output of the phase detector starts to approach zero error, its steady state, and the voltage across the diodes falls below their turn-on voltage level. Then R_1 is the dominant value again.

A switching speed comparison with and without the diodes is shown in Fig. 5-37.

Spurious signal performance is a very critical parameter in synthesizer design. It depends on frequency planning, speed, resolution, shielding, and power supplies. Most of these issues are beyond the scope of this book, and the reader is referred to Refs. 1, 2, and 3.

The division ratio usually needs to be minimized, especially in high-quality synthesizers. The main reason is that inside the loop bandwidth the reference noise and the phase detector noise are corrupted by 20 log N dB. Many techniques are employed to reduce the division ratio. A major one is the *fractional-N technique,* and it is described in detail later in this chapter. But now let us discuss other rather common techniques in PLL for noise reduction.

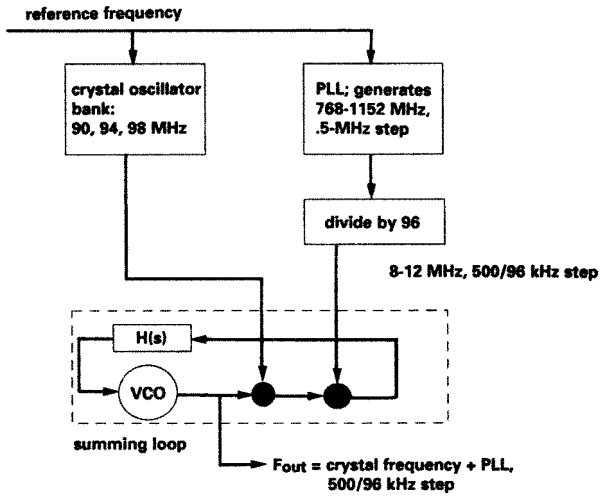


Figure 5-38 Crystal replacement unit architecture, representing typical multiloop design.

1. *Mix and countdown.* Suppose that it is necessary to generate a synthesized signal covering 5.0 to 5.2 GHz in 1-MHz steps. A single-loop design approach might look like that in Fig. 5-39. The reason that we have to put a fixed divide by 8 in front of the main divider is that the main divider needs to include a dual-modulus divider, and there are no such economical dividers above 2-GHz.

The introduction of the fixed divider forces the reference to be 125 kHz, and the total division ratio is in this case (5000 to 5200)8. The penalty for the use of such a high division ratio is 20

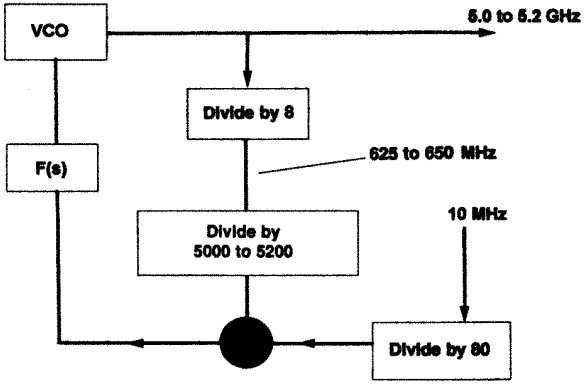


Figure 5-39 Regular single-loop PLL.

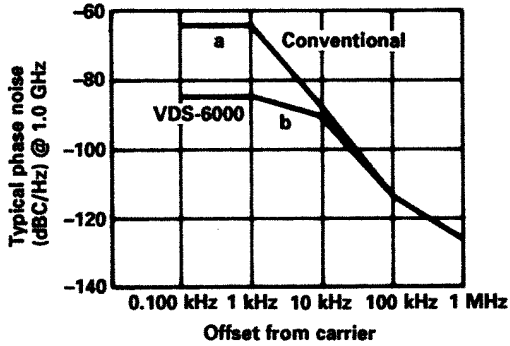


Figure 5-40 Typical conventional and fractional-*N* phase noise performances.

log (8 · 5200) = 92 dB, and the phase noise degradation within the loop BW will not be tolerable for most requirements. A calculated performance is shown in Fig. 5-40 at 1 GHz.

However, another tentative design is shown in Fig. 5-41. In this case a reference is generated while using a much lower division ratio, and because of the mixing, the division ratio in the main loop is reduced substantially. In this case the highest division ratio is in the 4.8-GHz reference, but it is only 480, so a total improvement of approximately 40 dB can be expected relative to the first design. Another advantage is that all the references to all loops are at much higher frequencies than before and are easier to filter out.

This technique is known as *mix and countdown* and is executed in a great variety of implementations by different manufacturers, mainly in microwave designs or when phase noise performance of a single loop is not sufficient.

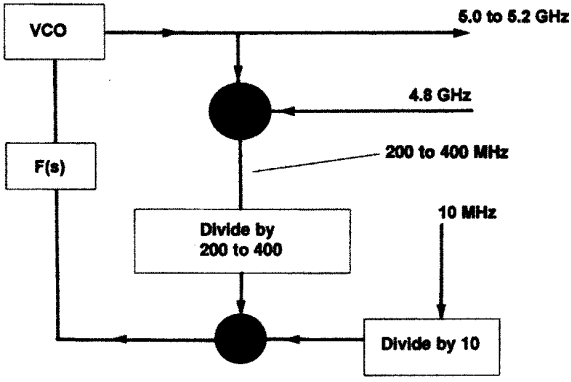


Figure 5-41 Mix and countdown PLL.

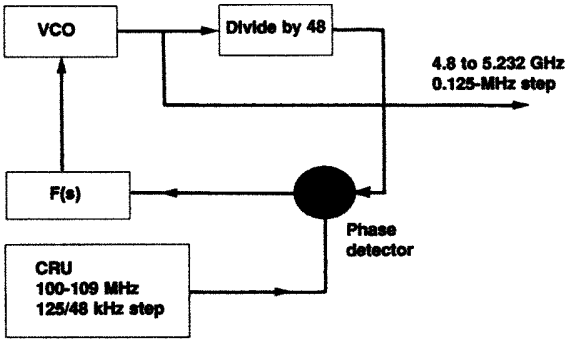


Figure 5-42 Crystal replacement unit PLL.

From time to time the reference is generated by using crystal oscillators to achieve excellent phase noise performance. See Fig. 5-42.

2. *Crystal multiplication.* For extremely demanding narrowband applications, PLL microwave designs use multicrystals and very low-division-ratio PLL circuits to achieve excellent phase noise, as shown in Fig. 5-42.

A tentative block diagram of an ultraclean reference signal [denoted here as the *crystal replacement unit*(*CRU*)] is shown in Figs. 5-38, 5-42, and 5-45. The circuit uses a combination of crystal oscillators and low-division-ratio PLL. The three to six crystals are switched in by PIN diodes, and all run off the same transistor oscillator. They are locked to a 2-MHz reference and generate, say, 90, 94, and 98 MHz, respectively. These signals are very clean because of the use of crystal oscillators. A high-division-ratio PLL generates 768 to 1152 MHz in 0.5-MHz steps (Fig. 5-38 shows this with 0.125-MHz steps), using division ratios of up to 1152*2 = 2304; but this signal is divided by 96, generating 8 to 12 MHz, so the total division ratio is only approximately 24. The two outputs are then combined in the third loop that generates 98 to 112 MHz. The phase noise performance of this design is shown in Fig. 5-43.

The multiplication of the signal to microwave is done by using either a fixed divider, as shown in Fig. 5-42, or a combination of multipliers and PLL (see Fig. 5-44). The VCO used in Fig. 5-44 is usually a cavity type, and the loop bandwidth is approximately 30 kHz. Beyond that bandwidth, the VCO phase noise is better than the multiplied reference; also see Fig. 5-45. High-speed dividers are shown in Fig. 5-46 and performance at microwave is shown in Fig. 5-47.

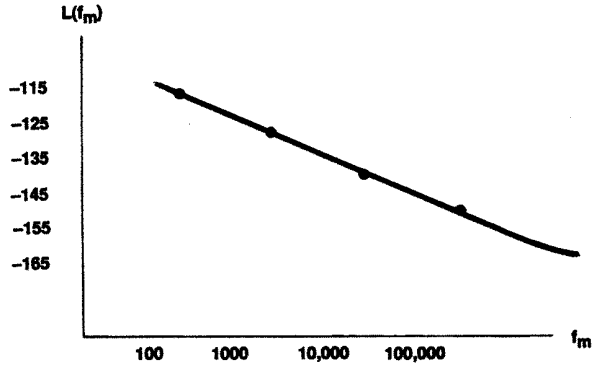


Figure 5-43 CRU phase noise requirement at 100 to 160 MHz.

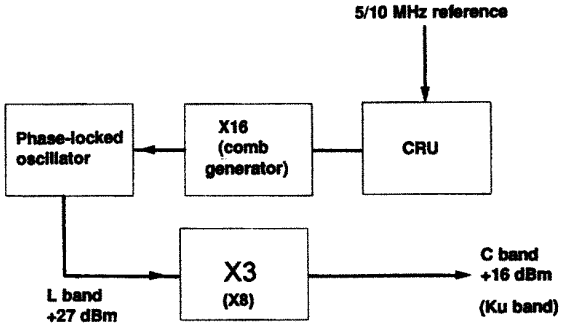


Figure 5-44 Microwave generation using CRU and multiplier.

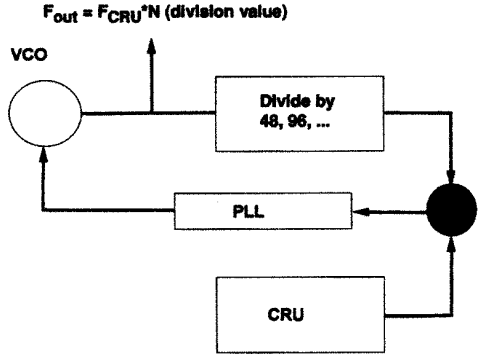


Figure 5-45 Microwave generation using CRU and PLL. Dividers operate up to 14 GHz, using GaAs technology.

STATIC PRESCALERS

SEI-12XX

SCITEQ Components Group		
	14 GHz DIVIDE-BY-2	SEI-1202
	14 GHz DIVIDE-BY-4	SEI-1204
	14 GHz DIVIDE-BY-8	SEI-1208
	14 GHz DIVIDE-BY-32	SEI-1232

FEATURES

- **Static Dividers**
- **Wide Frequency Range**: 2* — 14 GHz
- **Low Power Dissipation**
- **Single Supply**: +6V or -6V
- **Single-ended or Differential I/O**
- **Temperature Range**: T_A = -25°C to +85°C

DESCRIPTION

The SEI-12XX is a family of very high speed, low cost GaAs dividers, capable of operating up to 14 GHz. These devices are static dividers and therefore will divide over a wide frequency range up to 14 GHz. Input and output are either single-ended or differential, and nominal output power is +2 dBm.

The packaged versions of the prescalers are available in a 8-lead flatpak package. The standard part is the SEI-12XXP (positive bias voltage) but the SEI-12XXN (requires a negative bias voltage) is available as a special order.

Electrical Characteristics (T_A = +25°C, V_{cc} = +6V or V_{ee} = -6V)

Part Number		SEI-1202 (÷2)				SEI-1204 (÷4)				SEI-1208 (÷8)				SEI-1232 (÷32)			
Symbols	Parameters and Test Conditions	Units	Min	Typ	Max	Units	Min	Typ	Max	Units	Min	Typ	Max	Units	Min	Typ	Max
I_{ee}/ I_{cc}	Supply Current	mA		85		mA		80		mA		85		mA		95	95
f_{IN} (max)	Upper Limit of Input Frequency, P_{in} = +2 dBm	GHz	14			GHz	14			GHz	14			GHz	14		
f_{IN} (min)	Lower Limit of Input Frequency, P_{in} = +2 dBm	GHz			2*	GHz			2*	GHz			2*	GHz			2*
P_{in}	Input Power @ V_{cc} = +6.0V, f_{IN} = 2 GHz to 14 GHz	dBm	+2		+10	dBm	+2		+10	dBm	+2		+10	dBm	+2		+10
P_{out}	Output Power @ V_{cc} = +6.0V, f_{IN} = 14 GHz	dBm		+2		dBm		+2		dBm		+2		dBm		+2	
L	SSB Phase Noise @ 20 kHz offset from a 10 GHz carrier in a 1 Hz BW	dBc/ Hz		-138		dBc/ Hz		-138		dBc/ Hz		-138		dBc/ Hz		-138	

* f_{IN} (min) below 2 GHz can be divided if input slew rate is within certain specification.

Figure 5-46 Very high-speed GaAs prescalers; dividers by 2, 4, 8, and 32. (*Courtesy of Sciteq Electronics, Inc.*)

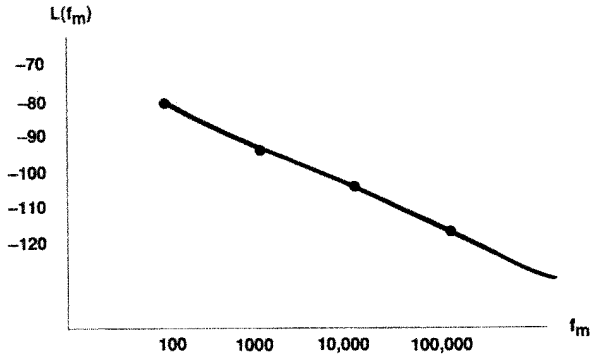


Figure 5-47 CRU phase noise after multiplied to C band.

5-2-1 Wireless PLL ASIC Configuration

Wireless applications require relatively narrow bandwidth, in the the 2–6% range. Such applications also require great economy and savings in power and size. As a consequence, most PLL ASICs have a current source output rather than voltage. Typical second, third and fourth order loop configurations are shown in Figure 5-48.

For the second order loop, the current source drives the RC network. Voltage (for output current I) is given by:

$$V = I\left(R + \frac{1}{sC}\right) = I\frac{(1+sRC)}{sC}$$

$$\frac{V}{I} = \frac{(1+sRC)}{sC}$$

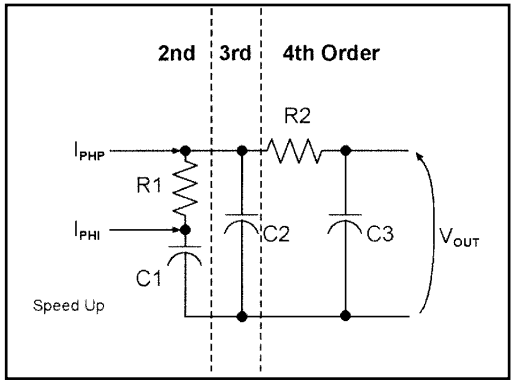


Figure 5-48 Loop filter for current source phase detector.

The transfer function is that of a perfect integrator (assuming that the current source is perfect). Therefore, the transfer function denominator will be very similar to that of Eq. (1-29). The denominator characteristic equation is given by:

$$1 + K_v K_\varphi \frac{1 + sR_1C_1}{s^2 NC_1} \text{ or:}$$

$$s^2 + K_v K_\varphi \frac{sR_1}{C_1} + \frac{K_v K_\varphi}{NC_1}$$

Therefore, for $K_v K_\varphi = K$

$$\omega n^2 = \frac{K}{NC_1} \qquad \xi = R_1 \frac{\left(KC_1 / N\right)^{.5}}{2}$$

Note that K_φ is given in ampere/radian.

Most wireless PLLs use a third order or fourth order loop, to attenuate the reference spurious. For a third order loop, C_2 is added. The impedance of the network then changes to:

$$Z = \frac{1 + sC_1R_1}{s^2 C_1C_2R_1} + s\left(C_1 + C_2\right)$$

$$\text{Define: } T_1 = \frac{R_1C_1C_2}{C_1 + C_2}$$

$$T_2 = R_1C_1$$

The open loop gain of this third order loop is given by:

$$\frac{-KT_1\left(1 + sT_2\right)}{T_2 s^2 C_2 N\left(1 + sT_1\right)}$$

While a second order loop is inherently stable as long as $\xi > 0$, for third order loops we require a phase margin, the excess phase from 180 degrees at the point the open loop gain is 1 (or 0 dB). Phase margin φ_M is given by:

$$\varphi_M = 180 + tg^{-1}(\omega T^2) - tg^{-1}(\omega T_1)$$

A typical drawing is shown in Figure 5-49.

A comprehensive simulation (from Eagleware software) is shown in Figure 5-50.

Fourth order loops are used to further attenuate the reference spurious; see Figure 5-49.

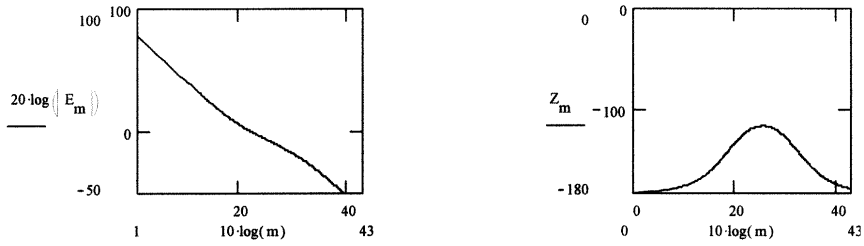
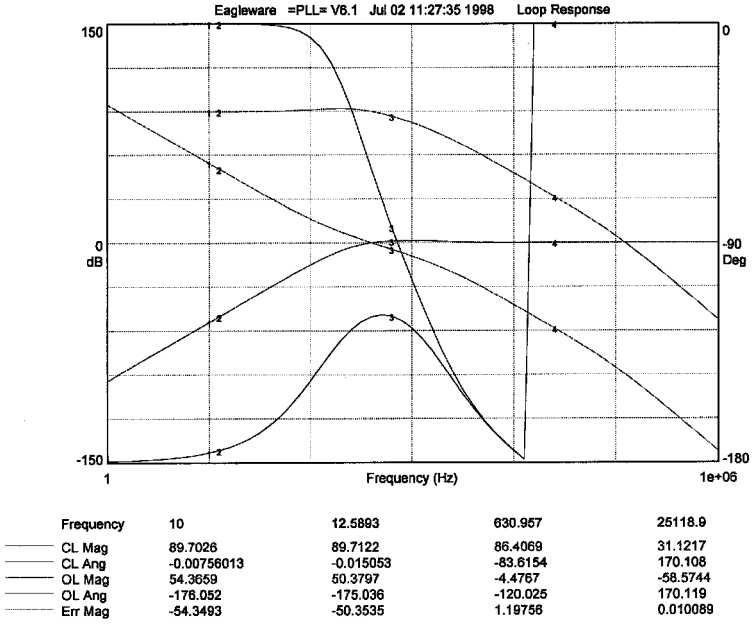


Figure 5-49 Open loop gain and phase margin for a third order loop.



Frequency	10	12.5893	630.957	25118.9
CL Mag	89.7026	89.7122	86.4069	31.1217
CL Ang	-0.00756013	-0.015053	-83.6154	170.108
OL Mag	54.3659	50.3797	-4.4767	-58.5744
OL Ang	-176.052	-175.036	-120.025	170.119
Err Mag	-54.3493	-50.3535	1.19756	0.010089

Figure 5-50 Eagleware simulation.

5-3 Fractional-*N* Synthesizers

As was shown above, it is a general rule and requirement in PLL designs to try to use the highest-frequency reference and the lowest division ratios. Also, the higher-frequency reference is easier to filter out and yields a lower division ratio, which produces better phase noise, lower spurious signal response, and better switching time. However, in many cases these requirements conflict with the rest of the design parameters, since high resolution (fine step size) requires low reference frequency and therefore high division ratios.

If the reference frequency is high, then the frequency resolution or the step size is high. To achieve small step size, it is necessary to either use low reference frequencies or resort to multiloop designs. This increases the complexity, size, and cost, but this was (and still is) the common basic technique and involves nested or coupled multiloop designs.

The tentative relationship of the output frequency to thereference in a multiloop design will be of the nature $F_o = NF_{\text{ref}} + (K/F)F_{\text{ref}}$, since a multiloop design allows the use of higher reference frequencies than a single-loop design. In fact, the cardinal reason for using multiloop designs is to reduce the division ratio by using higher reference frequencies and division to allow better resolution, as the term $(K/F)F_{\text{ref}}$ shows.

Another alternative, which has been explored fully only in the last decade or so, allows N (the division ratio) to take fractional rational values.

Dual loops perform a very similar function but in a complicated way. Suppose that in a dual-loop design, the secondary loop covers 5 to 5.99 MHz with 10-kHz step and is obtained by generating a 50- to 59.9-MHz signal having 100-kHz step and dividing by 10. If the 50- to 59.9-MHz loop uses 100 kHz as a reference, then the output (5 to 5.99) actually produces a total division ratio of 50 to 59.9, thus the equivalent of a fractional number. The trick is to produce the same function by using a single loop.

So far, all the designs that we demonstrated assumed the divider ratio N to be a whole number. But if N could include a whole number plus a fraction, it would be possible to generate step sizes that are smaller than the reference frequency.

Fractional division has already been demonstrated in the introduction to DDS [which is capable of generating frequencies given by WF_{ck}/ACM (maximum states of the accumulator)], and Secs. 4-2 and 4-3, and the same principles apply to PLL. Fractional-N PLL synthesis can and should be viewed as a combination of PLL and DDS principles.

The basic idea is to allow the division ratio to take the form

$$N = N_1 + \frac{K}{F} \tag{5-13}$$

where K can take the values 0 to $F - 1$, N_1 is an integer, and F is designated as the fractionality. If the reference frequency to the loop is F_r, the synthesizer resolution becomes F_r / F rather than F_r, since the output frequency will be given by $NF_r = F_r(N_1 + K/F)$. This, of course, has a major effect on the signal quality, i.e., the division ratio, since the resolution does not depend on the reference frequency on a one-to-one basis any more.

Suppose that we use a PLL circuit (see Fig. 5-27) with a 1-MHz reference signal. In a single-loop design, this will imply a step size of 1 MHz or more. If we need to achieve a step size of, say, 1 kHz, a second loop will have to be introduced and maybe even a third.

However, if it is possible to devise a divider that can divide by numbers that have a resolution of 0.001, that is, the counter and divider can generate a divide number given by abc.def (or in the equation given above, 5 to 10 MHz, the case of $F = 1000$), then the output resolution will be given by $F_r / F = 0.001$ MHz and the required resolution of 1 kHz will be achieved by using a single loop and a reference frequency of 1 MHz.

There are two types of fractional-N synthesizers. One uses only digital techniques with modest resolution improvements (in practice an improvement of 5 to 100 relative to the "standard" single-loop design). The second type uses both analog and digital techniques, resembling DDS technology in many aspects and allowing arbitrary improvement in resolution. In fact, such fractional-N synthesis can be viewed as inserting DDS inside the PLL, producing resolution improvements of arbitrarily high numbers. Many fractional-N synthesis instruments use 0.1 or 1 MHz as a reference and generate a 0.1- to 0.000001-Hz step size! Stated differently, the resolution is an *arbitrarily* small part of the reference frequency. The major difference between DDS and fractional-N synthesis is that in DDS the sig-

nal is fully generated and the sine lookup table is necessary, while in fractional-*N* synthesis, only the signal's phase is manipulated and therefore the sine-wave amplitude conversion is not necessary.

Another difference is that DDS generates all the spectrum it is capable of, while in fractional-*N* PLL the spectrum is limited by the PLL loop bandwidth, and this element helps reduce spurious signals and noise as the loop filter cleans all spurious signals beyond its bandwidth.

Inside a PLL, it is only necessary to generate the fractional-*N* phase (though in analog form), not the sine amplitude.

The first type of fractional-*N* synthesizer, which we will designate as fractional-*N* of the first order, uses digital techniques only, is of course a subset of the second, and is very useful as it is. Its features are very similar to those of a direct digital synthesizer that uses a 1-bit DAC, or an all-digital DDS (see Sec. 4-2). The second type, using both analog and digital circuits, resembles the more conventional DDS circuit, with the problems associated in combining analog and digital circuits.

Fractional-*N* synthesis is not yet as popular a technique as it should be, mainly because of the implementation "difficulties" and perhaps the lack of understanding and the usual resentment toward mixing analog and digital techniques. But its advantages are so powerful that we predict it will gain significant popularity very soon, as DDS did, and in a short time. The introduction of fractional-*N* PLL chips is now imminent and will probably expedite this process, too.

The first type is relatively simple to implement, but even this has not yet gained its deserved attention and popularity. We will emphasize first the all-digital approach because the nature of fractional-*N* PLL can be fully understood and comprehended by demonstration of this technique, and it is easier to follow its logic and circuitry than that of the complete fractional synthesis, involving both analog and digital components.

Also, we should mention that the artifacts introduced by fractional-*N* PLL can be manipulated by digital techniques very similarly to the way it is done in DDS, either by dithering or by oversampling and wave shaping. It is expected that much R&D work will be done, both theoretical and for hardware implementation. PLL is such a dominant technology that the advantages offered by fractional-*N* synthesis cannot be ignored by the industry any longer.

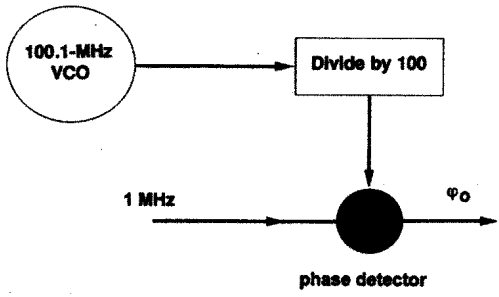


Figure 5-51 Fractional-*N* PLL principles.

5-3-1 Fractional-*N* synthesis of the first order

The principle of the problem and the proposed solution can be demonstrated, as shown in Fig. 5-51. Suppose that a frequency of 100.1 MHz needs to be generated, but a 1-MHz signal is used in the reference input to the phase detector. If the division ratio of the divider is 100, then in open-loop architecture, an error signal will develop in the output of the phase detector, as the feedback frequency from the divider generates 1.001 MHz, which beats against the 1-MHz reference, shown in Fig. 5-52.

The output of the divider is 1.001 MHz while the reference signal is 1 MHz. A linear phase error is being developed. The error will reach 2π after 1000 cycles of the VCO (or 10 cycles of the reference). To cancel this error so that the loop can be closed and locked, it is necessary to remove 1 clock tick from the output of the VCO every 1000 ticks. This means that it will be necessary to divide the VCO output by 1001 in 10 reference ticks.

This function can be implemented in many ways, e.g., divide 9 times by 100 but on the tenth time divide by 101, or divide 99 times by 10 and then 1 time by 11. Any other correct combination that achieves a divide by 1001 will do. (Above we demonstrated the use of a 10/11 or 100/101 dual modulus, but any other that performs the arithmetic will do.) Note that the dual modulus chosen has to meet the criterion $N(N - 1) < 1000$. (Most single-loop PLL chips on the market have very high dual-modulus dividers, such as 128/129 and 64/65, and force the design to use high division ratios. This, we predict, will change soon to accommodate fractional designs.) Such a procedure will swallow the extra clock tick and will generate an average division ratio of $N = 100.1$, which is a fraction.

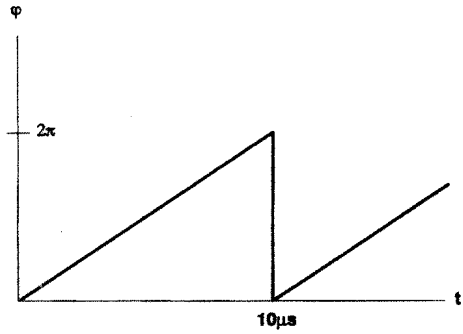


Figure 5-52 Phase detector output.

As can be seen, in order to produce this division ratio, the divider is not static any more, and it must have a dynamic (so to speak) and possess a procedure that requires changing the value of the division dynamically. This is, then, a new concept for division and indeed opens new opportunities for synthesis design. This operation is relatively simple to implement; an example is shown in Fig. 5-53. Basically, it is necessary to create a mechanism that will change the division ratio within a "total new cycle"—in the example above, 1001 clock ticks. Few components can be used to implement this function: an accumulator, a rate counter, and a counter/adder scheme. As an example, a simple circuit that executes this procedure is shown in Fig. 5-54. The circuit uses a dual-modulus device dividing by 10/11. Counters *M* and *N* are standard TTL programmable counters.

If we wish to divide by 100, counter *M* is programmed to 0 and counter *N* to 9 (remember, this counter counts to 9 + 1; see Figs. 5-27 and 5-54). Since counter *M* is programmed to 0, the 10/11 control is controlling counter *N* to divide by 10 only, all the time. Counter *N* counts from 9 to 0 (9 clock ticks from the output of counter *N*) and adds an extra count for the preset enable that presets counters *M* and *N* to their original programmed input state. The preset enable adds a clock tick, so the total number of clock ticks of counter *N* is 10; and since counter 1 divides by 10 only, the total division ratio is $10 \cdot 10 = 100$ as desired.

If we wish to count to 101, counter *M* will be programmed to 1, and counter *N* to 9. Thus counter *M* will program the dual modulus to divide 1 time by 11, and counter *N* will continue to count 10 including the preset (9 clock ticks from 9 to 0 plus 1 clock tick for

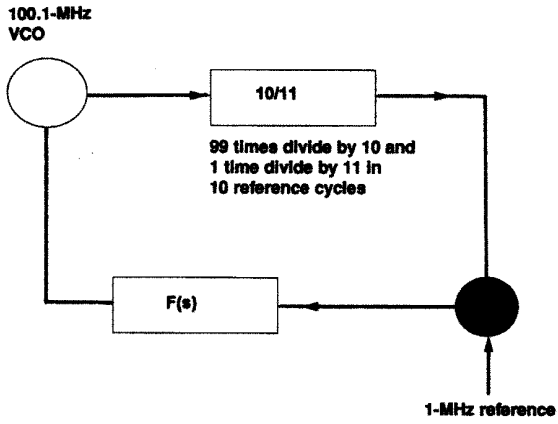


Figure 5-53 Principles of fractional PLL synthesis.

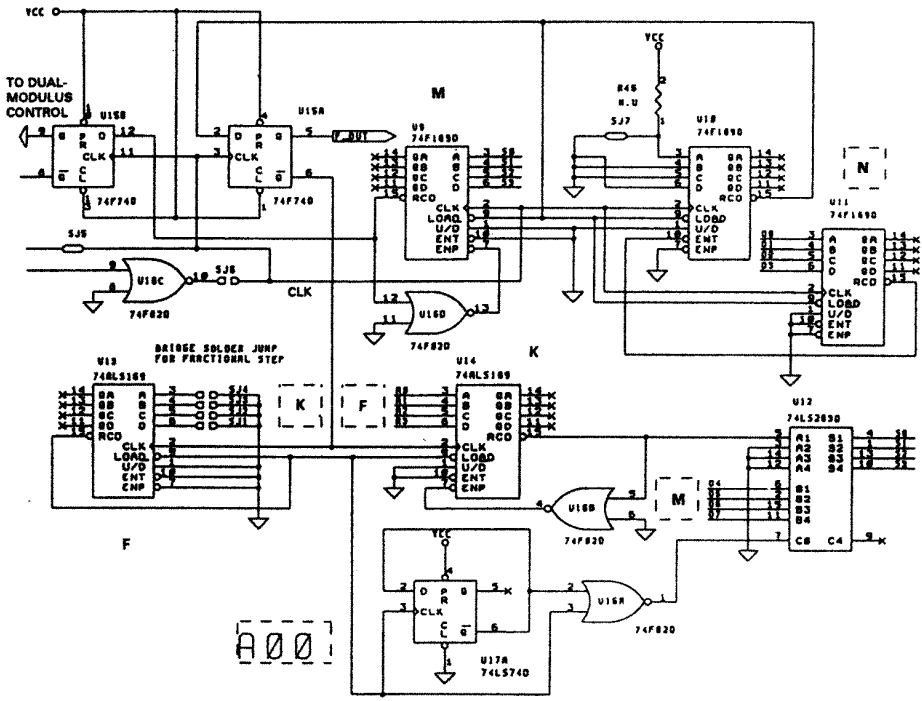


Figure 5-54 Fractional divider generates average division, given by $M(P + 1) + (N + 1 - M + K/F)P$. (*Courtesy of Sciteq.*)

the preset enable), so the total division ratio is $1 \cdot 11 + 9 \cdot 10 = 101$. For division by 102, counter M is programmed to 2 and counter N to 10, and the total division ratio is $2 \cdot 11 + 8 \cdot 10 = 102$ and so on for any number (above 90).

So far, this is a standard procedure for single-loop PLL circuits. However, for demonstration of the fractional principle, to the above standard scheme we added counters K and F and adder u12. The clock to counters K and F is the main counter's preset enable pulse, which determines the counter periodicity and serves as the "total" counter output. After all, this complete circuit operates as a *complex* counter, and this signal serves as its output. (Compare Fig. 5-27 to Fig. 5-54.)

So now the scheme receives a new periodicity since counters K and F divide the preset enable by 10 again (we assumed for simplicity that they count to 10). The overall time periodicity of the circuit has been multiplied by 10, or generally by the size of counter F (or an accumulator).

Now suppose we control counter M to 0, counter N to 9, counter K to 1, and counter F to 10. Without the additional circuitry, the basic counter will divide by 100, as demonstrated above. However, with the additional circuit, after the first clock into K, count output terminal count ($\overline{\text{TC}}$) goes from state 1 to state 0 for 1 clock out of 10. This state propagates up via the adder and controls counter M to 1 (1 time out of 10). So in the first cycle, the output from adder u12 is 1 and the division is 11. However, for the other 9 cycles, the output of adder u12 is 0, and the division will be 100. This completes a total cycle for the whole configuration, and all devices are at their initial states, ready to repeat exactly the same cycle as described herein. This then completes the new "true periodicity" of this circuit, and all is as in the beginning and the sequence repeats itself.

A new periodicity has been demonstrated that is 10 times longer than before, and the total count is $100 \cdot 9 + 1 \cdot 101 = 1001$. Since the *true* periodicity—the one connected to the phase detector—is the main preset enable, and since there were 10 such ticks (counter F was set to count to 10) and the total division was 1001, the average division is $1001/10 = 100.1$. We therefore generated two types of periodicity here. One periodicity is of the main counter (here counters M and N), and a second is the periodicity of the secondary (here counters F and K) which determines the level of fractionality, so to say. Writing 2 into counter K produces

a division of 100.2, etc., up to 9 in this case, that is, 100.9.

The result is in agreement with what we wish to achieve. In the open-loop configuration we saw that the input to the phase detector was 1 and 1.001 MHz accordingly. If we had not used any correction, then the phase error would have accumulated linearly to 2π after 1000 VCO ticks or after 100 reference ticks. The introduction of the fractional divider eliminates 1 VCO clock tick and resets the phase error back to 0, as shown in Fig. 5-55. What should also be expected is that the periodicity of the process increases 10-fold from 1 μs (1 MHz) to 10 μs (0.1 MHz). This is, of course, because the secondary periodicity increased the main one (10-fold in this demonstration).

The net effect is therefore to increase the resolution without changing the reference frequency or the main division ratio. Note that the waveform at u12 has a similar character as a single-bit direct digital synthesizer (see Sec. 4-2). The output has 9 steady pulses and 1 that is shorter by 1 clock tick. This will therefore generate a spectrum containing 100-kHz lines rather than the 1 MHz of the original circuit.

A general formulation can be written now for the case when the dual modulus is $P/(P + 1)$, the $P + 1$ control counter loaded to M, the P control counter loaded to N, the accumulator or counter is

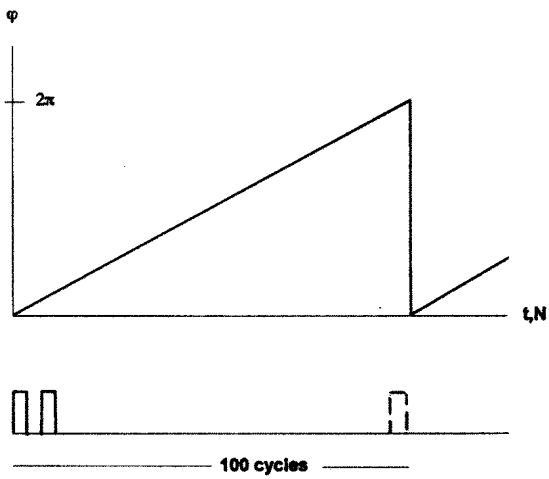


Figure 5-55 Elements of fractional design.

size F, and its input control is K. Then the relation of the output frequency F_o to the reference frequency F_r is given by

$$F_o = F_r\left[M(P+1)+(N+1-M)P+\frac{K}{F}\right] \qquad (5\text{-}14)$$

and after rearrangement we get

$$F_o = F_r\left(NP + M + 1 + \frac{K}{F}\right)$$

The case demonstrated herein shows an increase of 10:1 in the synthesizer resolution or a case of $F = 10$ for Eq. (5-13). Obviously the process can be repeated to increase the resolution to any desirable ratio. For example, 6 decades of fractional division will improve the resolution 1,000,000 times and in this case will yield a frequency resolution of 1 Hz. However, this result requires careful examination. Our technique is quite crude, and the phase correction, as shown in Fig. 5-55, is very abrupt.

Since the output of the phase detector, followed by the loop filter, drives the VCO, one can expect a high level of spurious signal output. The reason is that instead of generating a smooth continuous change in phase (a dc value in the VCO control output), we "faked" the PLL circuit and generated an abrupt change that caused the average value of the phase detector output (a pulse) to be correct, but not its instantaneous value. This is quite similar to the performance of a single-bit direct digital synthesizer.

As in the case of a single-bit direct digital synthesizer, only if we put a narrowband filter at the output frequency will it filter out the spurious signals. Otherwise the spurious signals will show. If the loop is very wideband, i.e., the loop filter is wide relative to F_r/F, then the spurious signal level will be high, as in a single-bit direct digital synthesizer, and can be calculated by the Fourier transform of the transfer function. This is so because the loop transfer function cannot filter the spurious signals.

Another way of looking at this is as follows: The desired control voltage (to the VCO) increases between 100-MHz and 100.1-MHz output in some incremental dc level that ideally is given by the average of the phase detector ramp signal times the loop transfer function from the phase detector to the output, or simply 2π(0.1

MHz)/K_v (K_v measures the VCO sensitivity in rad/(s · V). So to increase the output frequency by 0.1 MHz, it is necessary to feed the VCO with an increase in control voltage, as mentioned above.

Another interpretation of the principle is as follows: If the PLL bandwidth were infinite, we would have generated 100 MHz nine times and 101 MHz one time. The average is 100.1, but the instantaneous frequencies are different from the desired. This model can be applied to the loop filter to receive a closed-loop transient solution that will obviously show the fractional spurious signals.

What we have done is generate a pulse signal whose average through the circuit generates the desired correction, but instead of a fixed dc signal we generate a step with the expectation that the loop filter will average this waveform to the desired result. And so if we can filter, "smooth," or average this signal enough, we can filter out the spurious signals. This implies that the loop bandwidth must be substantially lower than the periodicity of the ramp signal, that is, $W_n << F_r/F$ (where W_n is the natural frequency of the closed-loop PLL circuit).

Thus by using this technique, it will not be possible to increase the resolution of this design to 1 Hz since the loop will have to filter 1-Hz spurious signals, and this is not practical. However, a 10- to 20-kHz (and above) signal resolution (or that order of magnitude) can be filtered out in many applications, and such a design will improve resolution 10- to 100-fold at the minor cost of a relatively simple digital complication.

Very similarly to DDS of the second order, the spurious signals can be attenuated by using a programmable delay line (see Chap. 4). However, so far not many designs have been implemented to take advantage of this "twist" in the design, and almost all single-chip synthesizers (see Sec. 5-4) just use straight, very high-division ratio single-loop PLL circuits with the penalties associated with such designs (relatively poor close-in phase noise performance). Another interesting advantage is the fact that the reference is much higher than usually can be used and therefore offers a switching speed advantage.

We expect to see a rapid proliferation of this technique in synthesizer design in the very near future. The additional advantages are substantial, and the additional complexities and costs marginal. A typical design and its phase noise advantage relative to standard PLL synthesis are shown in Figs. 5-56 and 5-57.

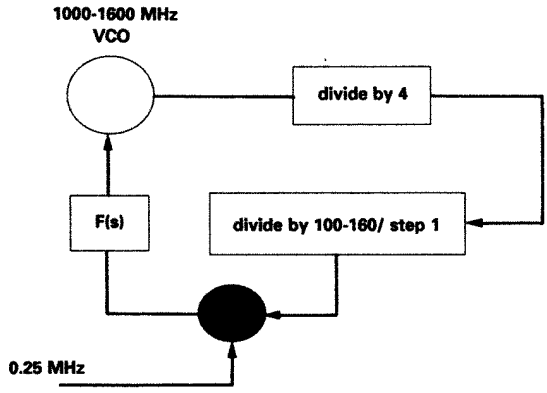


Figure 5-56 Single-loop PLL.

The detailed design shown in Fig. 5-54 is mainly for demonstration purposes, and a great variety of other hardware implementations are possible. The only element common to all the designs is the need for a dual-modulus divider (which is required in most PLL circuits anyhow) and an arithmetic function (the adder in this case) to manipulate the necessary arithmetic.

The same function can be executed by the use of accumulators rather than counters; see Fig. 5-58. The accumulator size is equivalent to counter *F,* and *K* is the number of carry outputs in the cycle.

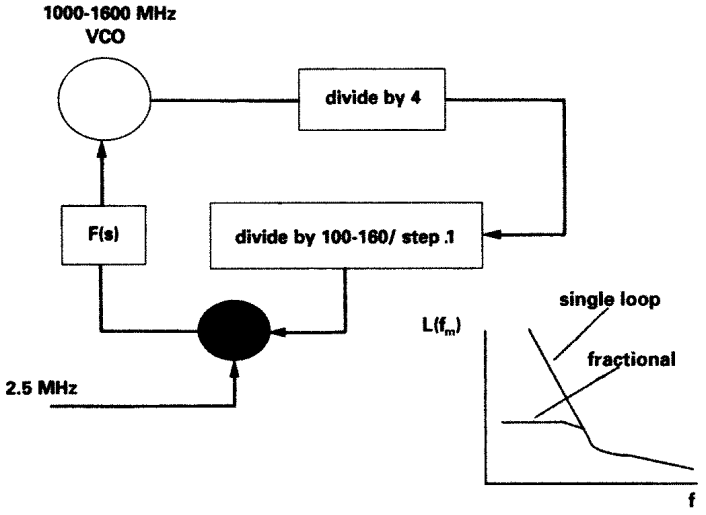
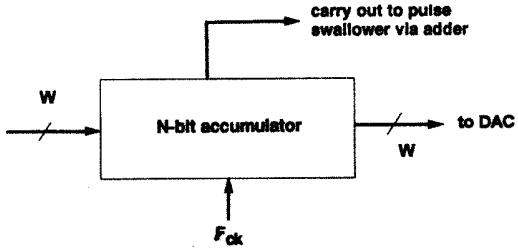
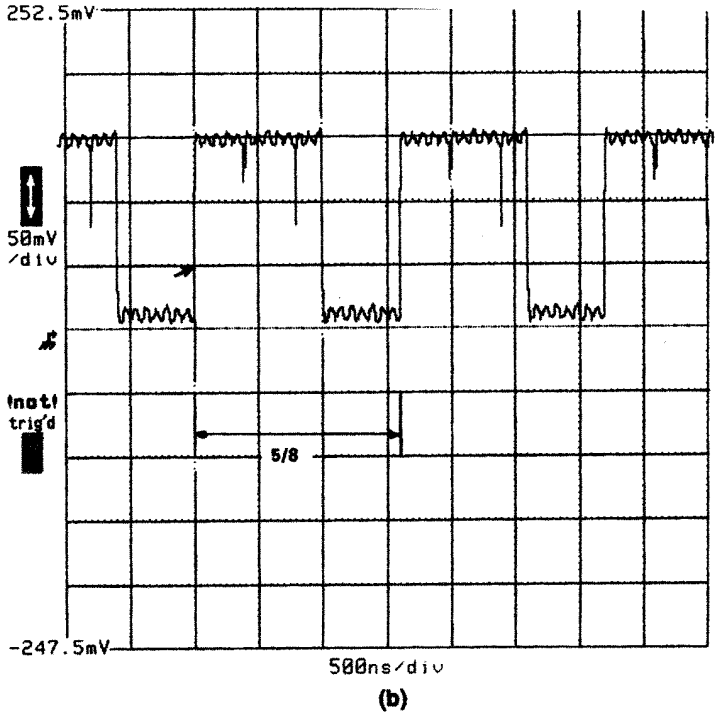


Figure 5-57 Single-loop PLL using fractional-*N* synthesis.



(a)



(b)

Figure 5-58 (*a*) Accumulator in a fractional-*N* design. Fractional waveforms (counters) (*b*) for counter output for $K/F = 5/8$ and (*c*) for accumulator output for $K/F = 4/10$.

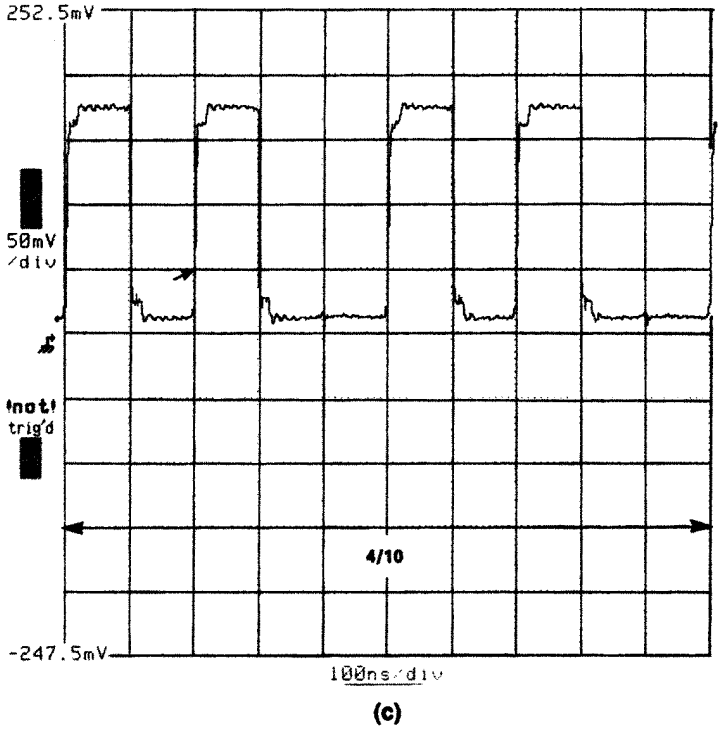


Figure 5-58 (*Continued*)

A rate counter can be used too, and we will demonstrate the same functionality with programmable accumulators (see Ref. 29).

The difference between using counters and using accumulators is that the counters generate a pulse whose duty cycle is K/F, while the accumulator generates a K/F ratio but not with uniform structure (see Fig. 5-58*b* and *c*).

Note that although the function is the same (i.e., the fractionality or the ratio K/F is the same), the fractional signals will be different when accumulators, rate counters, and counters are used. The counters will generate the corrective fractional signal at the beginning of the cycle, and this signal will have a duty cycle equal to the fraction K/F. For the accumulator, the number of corrections will be the same, also K out of F, but they appear at different times in the cycle. For example, if $F = 16$ and $K = 2$, for the counter, the corrective signal will look like a pulse with a width of $\frac{2}{16}$. When an accumulator is used, two pulses will show, one after tick 7 and one

after tick 15. The spectral shapes of these two signals will be different. A scope demonstration is shown in Fig. 5-58*b*, for the counter output of $K/F = \frac{5}{8}$, and Fig. 5-58*c* shows the accumulator output for $K/F = \frac{4}{10}$. The difference is significant.

Almost all instrument makers, and most literature and patents, assume the use of an accumulator. In principle, there is little difference between the two methods, but it is our opinion that the accumulator is an optimum solution as it does not allow the VCO to develop a phase error of more than 2π and maximizes the number of transitions in the dual-modulus control. The accumulator generates lower spurious signals, except at $K = 1$, where it equals that of the counters. The output spectrum from the counters is easy to calculate since it is that of a pulse with varying width [basically a $(\sin x)/x$ function with varying sidelobes according to the relative duty cycle of the pulse] while the accumulator (or rate counter) output spectra are more complicated to calculate and vary from control to control. For example, for $K/F = \frac{1}{16}, \frac{2}{16}, \frac{3}{16}, \frac{5}{16}$ the counter and accumulator outputs look like these:

Counter:	1000000000000000	$\frac{1}{16}$
	1100000000000000	$\frac{2}{16}$
	1110000000000000	$\frac{3}{16}$
	1111100000000000	$\frac{5}{16}$
Accumulator:	0000000000000001	$\frac{1}{16}$
	0000000100000001	$\frac{2}{16}$
	0000010000100001	$\frac{3}{16}$
	0001001001001001	$\frac{5}{16}$

For $K = 2$, the accumulator has no energy at $F_r/16$, only at $F_r/8$!

Although the average values of these signals are the same, their spectra are very different. This must be taken into consideration when the filters needed to filter out the spurious lines are designed (usually two notch filters to take out the first and second harmonics).

The use of an accumulator enables the implementation of a complete (digital and analog) fractional implementation, and therefore it has been the preferred method traditionally.

Another difference between the accumulator and the counter output is that while the counter method creates a spurious signal

at the base fractional frequency whose energy increases with the fractional value (the width of the pulse increases linearly with the fractional value), the accumulator creates as many transitions as possible. The accumulator therefore creates spectra that are easier to filter than the counter, because the multiple transitions generate energy at higher multiples of the base fractional offset and therefore lower multiples at the base fractional offset where the filtering is the hardest. And in fact, for some fractional numbers, say 1/16, while the counter method creates a pulse of width 4, the accumulator creates a pattern of 4 pulses, each separated by 4 clocks, and therefore has no energy at the base fractional offset! This is an advantage of the accumulator method.

Note that by using programmable array logic and such other devices, the whole fractional circuitry can be implemented in a single device operating at speeds exceeding 200 MHz (as of 1998). One design uses Xilinx's FPGA to incorporate all functions. Usually the fractional part of the circuit operates at moderate frequencies since its clock input is the output of the main PLL counter, and typical speeds almost always will be below 5 MHz.

When an accumulator is used, the resemblance to a direct digital synthesizer is very striking. The carry output of the accumulator will be used to control the adder, which then controls the dual-modulus divider. But the contents of the accumulator (say, in the case of $F = 10$) generate the phase error since it steps from 0 to 9 (or in phase, 0 to $0.9 \cdot 2\pi$) linearly and follows the error shown in Fig. 5-49. This is similar to looking at the output of the counter in the previous example rather than only at the TC function. Note also that the accumulator bits represent the phase error, and show a linear ramp, very similar to the phase error ramp demonstrated in the open-loop analysis. Since the accumulator output implements the phase error, as in the case of a direct digital synthesizer, this signal could be used to approximate the error smoothly, given that the phase error is available as the output of the counter or the accumulator.

To summarize, digital fractional circuits are therefore easy to implement. One reason why PLL chip designers use very high dual-modulus dividers is to attempt to have the majority of the other counters run at low frequency and save power. When a cellular signal is generated, a division of greater than 4000 is required so that the dual modulus 64/65 can be used. But if a frac-

tional design is implemented, the division ratio is much lower, say, 900; then a smaller dual modulus is required, say, 32/33 or 16/17 [remember that the lowest usable divider for a $P/(P + 1)$ dual modulus is $P(P + 1)$]. The technology is already available today to use low dual-modulus numbers, achieve high levels of fractionality, and use very little power, via bipolar CMOS technology.

Very interesting development work has been done by Hewlett Packard, where an all-digital fractional with a spurious cancellation circuit has been designed. The principle of operation is to create a sequence, by the use of a combination of delta and sigma modulators, whose average is equal to the desired fractional ratio, K/F, but which is modulated with a "random" sequence whose spectrum has very low energy close to dc; the majority of the energy is pushed to higher frequencies. The phase-locked mechanism therefore cancels the excess noise at the high frequency while within the loop bandwidth, approximately 1 to 2 kHz, the noise is equivalent to −142 dBC/Hz. (See Refs. 17 and 31.) The use of DSP, especially the accumulator structure, is referred to sometimes as "noise shaping" or "wave shaping."

5-3-2 Fractional-*N* synthesis of the second order

The next step is the full implementation of the fractional-*N* principle, involving analog circuitry and providing the full advantage of achieving very fine resolution. We refer to this as the *fractional-N method of the second order.*

The implementation of this technique has many resemblances to DDS technology. It achieves very high resolution via mixed digital and analog techniques and provides phase-continuous switching. However, like DDS, this technique has its own "nightmares," and as of now, not many designers have achieved mastery over it. Compared to the method described above, which can be referred to as an *all-digital fractional-N method* (of the first order) and is clearly very simple to implement, the full fractional-*N* circuit is complex and requires expertise and experience. On top of this, many techniques have been patented, since patents have become (always, but more so lately) a method of protection in the hands of many businesses to protect their intellectual assets.

Spurious signal performance is limited to approximately 60 to

65 dB (very much like DDS); however, since it is a part of a PLL circuit, this technique is capable of cleaning wideband spurious signals since the PLL acts as a narrow bandpass tracking filter. However, the cost of narrowing bandwidth is that switching speed is much slower than that of DDS.

We resume the technical discussion where we left it since the pulse swallower divider is a part of the fractional-*N* circuit anyhow. As we saw before, the main problem in achieving high resolution and low level of spurious signals is the pulse signal generated in the output of the phase detector. What is needed is to remove the pulse in concert with adding the analog ramp (as shown in the open-loop design) so that only the dc component is left (after the loop filter) to control the VCO, and instead of letting the loop filter "smooth" the abrupt phase step, generate a "smooth" signal. If this could be achieved, the resolution could be improved arbitrarily and spurious signals could be kept low, even when the loop bandwidth is much wider than the resolution step and the loop filter does not have the task of removing the fractional-*N* spurious signals.

This then becomes the equivalent of implementing a full direct digital synthesizer (rather than a single-bit direct digital synthesizer) inside the PLL circuit.

What is required, then, is to construct an inverted ramp and add it to the output of the phase detector to cancel its ramp; i.e., generate an analog ramp and add the analog signal to the output of the phase detector. This analog signal will then gradually build the 360° error and at the end will fall back to zero phase while the pulse swallower then removes the extra clock pulse. If we resort to the previous example, viewing again the phase detector output, an analog circuit will have to generate the sawtooth signal at the exact frequency and amplitude for a "perfect" cancellation (see Fig. 5-59).

Now, as we saw in the open-loop circuit, the error signal coming out of the phase detector is an analog ramp. We now insert an analog circuit, fed by the digital data from the fractional-*N* divider followed by a DAC to convert the digital signal to analog. The input to this circuit is available from either the counters or the accumulator used to perform the digital part of the circuit. For this demonstration, we use the accumulator implementation. (The accumulator's error is never bigger than 2π.)

So the signal path is as follows: The carry output bit is driving the adder, as in the previous design (Fig. 5-54), and replaces coun-

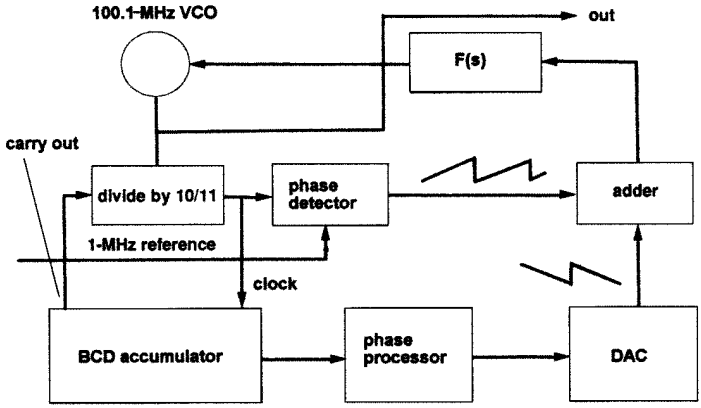


Figure 5-59 Fractional architecture and waveforms.

ters F and K. The accumulator carry output replaces the $\overline{\text{TC}}$ bit of counter K. So far the operation is similar to that of fractional-N of the first order. However, now the contents of the accumulator representing the phase error, in this case counting from 0 to 9, are fed to the DAC, which generates a ramping signal (equal to the one generated by the phase detector, in open-loop circuits). The analog signal, generated by the contents of the accumulator, represents the exact phase error beyond the carry output.

The DAC output controlled by the contents of the accumulator is always a quantized waveform since it is operating from a digital control and is being clocked at specific clock rates. The phase detector output is a linear ramp, also quantized in time since the phase detector operates only on the reference input edges.

However, since the circuit updates the loop only at the reference frequency rate, 1 MHz in this case, the error signal can and must be sampled. This is a very important point to understand. Since the phase detector samples the output of the programmable divider only in discrete points in time, the *only* information available from the phase detector is its output at these times. It is thus necessary to match the analog value of the outputs from the phase detector and the DAC at *only* the sampling time.

Suppose that we load the fractional-N digital control word into an accumulator and clock the accumulator with the pulse remover output (equivalent to the reference frequency in the locked condition); see Fig. 5-60. The output of the accumulator will increment

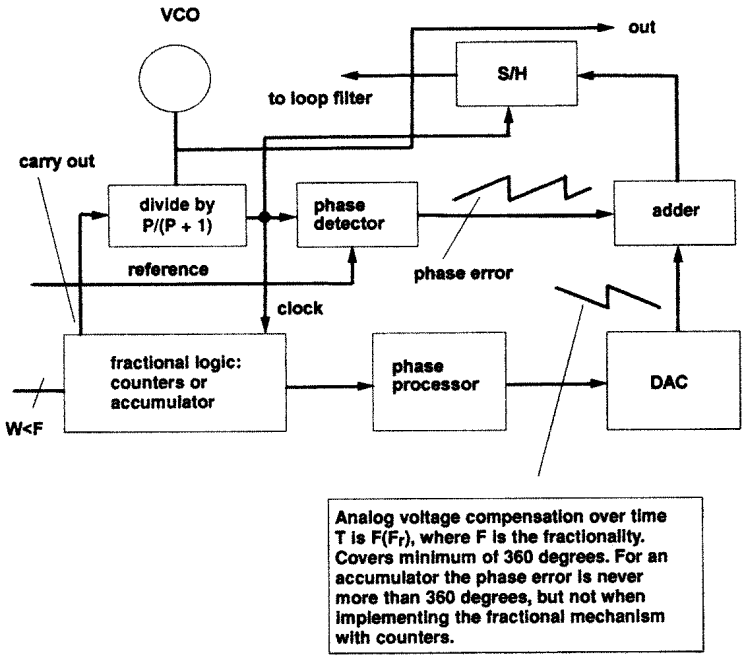


Figure 5-60 Generic fractional architecture including sample-and-hold (S/H) device. $P/(P + 1)$ is the dual modulus.

exactly at the rate of the phase detector output. If the output of the accumulator is fed to the DAC, the output ramp from the DAC will be equal in frequency to that of the phase detector output, and its amplitude will have to be controlled to an equal value (see Fig. 5-59). By inverting the signal polarity and adding to the output of the phase detector, we'll cause a cancellation. There are many variations and implementations of this scheme (and many patents have been issued).

Since the output of the DAC is equal to the value of the ramp only at the sampling points, it is necessary and generally helpful to follow the combining device by a sample-and-hold device, so that the staircase signal, which would cause spurious output if it were to drive the VCO, turns into a smooth dc signal instead, as shown in Fig. 5-59. This is an important function in this implementation, and the sample-and-hold device following the analog adder/canceler improved spur rejection.

The difficulty here is both in the timing and in keeping the analog voltage from the phase detector and the analog equalizer circuit equal, so the cancellation is as close to perfect as possible across the operating temperature range and the aging of the devices.

The execution of these circuits is not simple, but it is rewarding. Fractional-*N* is thus the most elegant integration of PLL and DDS technologies. It is also the point of contact between PLL and DDS, and therefore holds the promise for great utilization of DDS technology since PLL is the controlling technology for frequency synthesis. The integration of these functions on a single ASIC device is relatively complex, but has seen a remarkable evolution in the last four to five years. It is available from Texas Instruments (TRF2050, TRF3040), Philips (SA7025, 7026, 8035, 8025), National Semiconductor (LMX 2350/2352) and Peregrine Semiconductor (PE 3292). Fractional-*N* is clearly the evolutionary path for PLL technology.

Fractional-*N* synthesis is said to have been invented at Racal Dana (to the best of our knowledge) and was called *digi-phase* (see Ref. 18). Apparently, the importance of the technique was not fully exploited by Racal Dana; Hewlett Packard introduced its version of fractional-*N* in the HP3225A and HP-8662A synthesizers. This technique is now in use by the majority of instrument makers.

5-4 Fractional-*N* Synthesis of the Third Order

The most advanced fractional-*N* technology is now emerging, although it is still young and early. This technology uses oversampling and noise shaping, in a process used by data conversion devices, with multistage sigma-delta modulators. It allows arbitrary resolution and excellent phase noise. The following is a short description of its operational principles.

We have already mentioned delta modulators in Chapter 4. For sampled data, it is easy to show that quantization error energy decreases with increasing sampling frequency f_c. The mathematical relation is given by:

$En \approx Er\sqrt{2}/f_c$ and Er is the quantization error given by Eq. (4-50):

$Er=\Delta^2/12$ where Δ=the quantization step.

A delta-sigma modulator is shown in Figure 5-61a.

The additional integrator produces an overall transfer function given by:

$$Y(z) = X(z)\, z^{-1} + \mathrm{Eq}(1 - z^{-1})$$

The output is delayed by one clock, but the quantization error is multiplied by a transfer function that rejects low frequencies. Therefore, the noise, which originally is white, becomes colored with energy "pushed" to higher frequencies.

This is ideal. If we now interpret F_c as the reference frequency, then the loop bandwidth is much lower. For F_c=1 MHz and loop of, say, 1 kHz, the ratio is 1000. Noise shaping will allow low noise close to the carrier while the bulk of the noise at high frequencies will be rejected by the loop! This technology is used in almost all CD players, where the clock is in the 12–20 MHz range, and noise above 15 kHz is not heard by our ears. See noise shaping $(1–z^{-1})$ in Figure 5-61b.

The schemes used in audio and PLL circuits employ higher order devices, usually at least three levels. A tentative structure is shown in Figure 5-61d. This structure was proposed by Hewlett-Packard (see reference 32) and others. The basic structure, similar to a DDS, was an accumulator, with the carry out serving as the only output. The structure generates "random" signals, with a ranging level of –3 to +3, such that their average is X_{frac}/ACM, but their spectrum has very little "noise" in the loop bandwidth. A typical sequence is shown in Figure 5-61c. Because ACM can be very large, this structure allows arbitrary resolution (similar to DDS) with a cleaning loop, and no spurious! When the reference frequency is high, the loop can be wide and can achieve excellent switching speed. Excellent results have been reported by Marconi, HP and Rhode; see Figure 5-61e. Incremental phase noise depends on clock speed and phase detector noise, but can be as low as –145 dBC/Hz with a clock speed of >10 MHz!

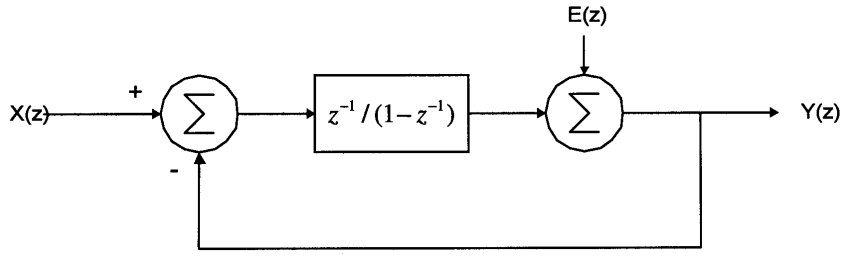


Figure 5-61a Delta-sigma modulator

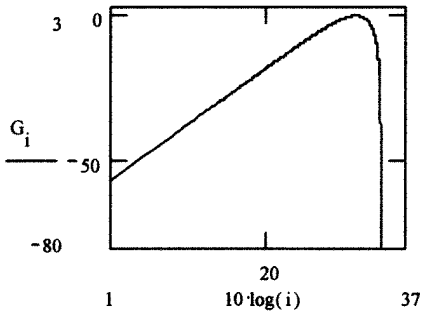


Figure 5-61b Single stage sigma-delta noise rejection.

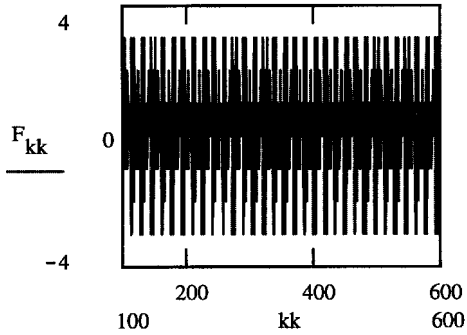


Figure 5-61c Sigma Delta Sequence, average is 1/16.

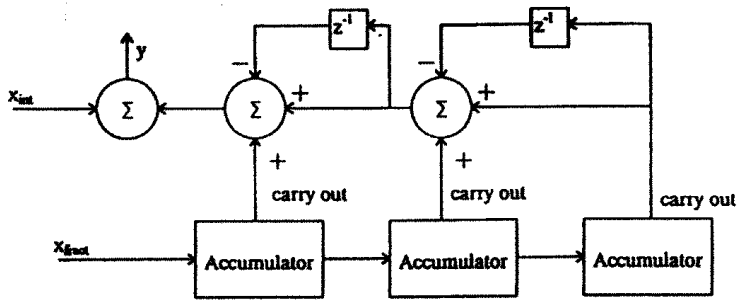


Figure 5-61d Third order delta-sigma modulator for fractional- N

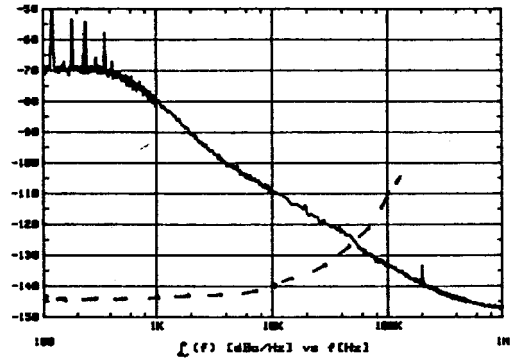


Figure 5-61e Third order fractional-N, simulated and measured using 200 kHz reference (HP).

5-5 DDS-Based PLL

There is a growing interest in a new kind of a PLL circuit family that involves another combination of PLL and DDS technology. The signal phase is built in an accumulator, there is an arithmetic device that performs the phase comparison operation, a DAC is used as a multilevel analog phase detector followed by a PLL analog network, and its output controls a VCO, as shown in Fig. 5-62; see Ref. 8.

When the total additions of accumulator 1, denoted by *A,* are equal to those of accumulator 2, denoted by *B,* the contents of the *arithmetic logic unit* (*ALU*) remain mainly constant and the circuit is in the lock stable-state condition.

If both accumulators are the same size ACM, then according to simple arithmetic, since the average values of the inputs to the ALU have to be equal, we receive

$$\frac{aF_r}{\text{ACM}} = \frac{F_{\text{out}}b}{\text{ACM}} \qquad \text{ACM} = \text{accumulator size}$$

Therefore

$$F_{\text{out}} = \frac{F_r a}{b} \tag{5-15}$$

where F_{out} is the VCO (the desired) output and F_r is the input clock to accumulator A.

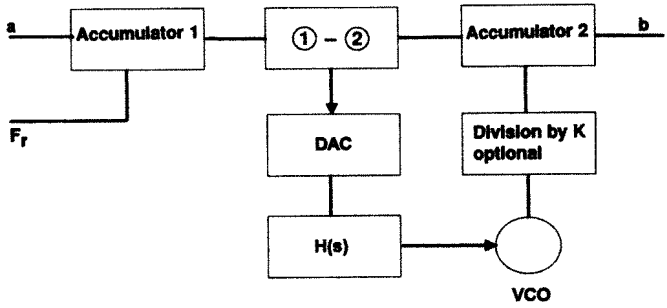


Figure 5-62 Accumulator PLL system.

This creates another way of achieving very high (in fact, arbitrary) resolution, depending only on the size of the accumulators. Suppose for simplicity that $F_r = 10$ MHz and that the accumulator is a BCD type and equal to 10000000. The resolution that can be achieved is clearly F_r/ACM, which equals 1 Hz in this case.

The interesting part of such implementations is that the phase detector is now comparing the phases not only at the zero crossings, but at all phases down to the quantization levels. Basically all PLL equations hold, and performance of better than −60-dB spurious signals has been reported.

To intuit the process, let's continue with the example above and assume that $a = 1000001$ and $b = 1000000$. To be locked, the VCO must be at an output frequency of 1,000,001 Hz.

Accumulator A will accumulate its input a, which consists of the 1-MHz signal but also the 1 Hz that will propagate up and after 10,000,000 clock ticks will generate an extra carry output. Accumulator B increments only the 1-MHz digit so its output is completely periodic with a period of 10 cycles exactly, with a period given by 1/1,000,001 s. The periodicity of accumulator A is 10,000,001 clock ticks. Suppose that the ALU performs its operation only using the three upper decades, that is, 0 to 999. It will take the 1-Hz digit 10,000 clocks to propagate up to the LSB of output A. This means that in the open-loop circuit analysis, the state of A will change every 10,000 clock ticks, or every 1 mHz. This is, of course, very like a similarly structured direct digital synthesizer. The quantization here (as usual) is of dual nature. One is caused by the finite value of the phase representation (i.e., number of digits in A and B) and the other by the quantization of the time, for both the accumulators and the ALU.

Since the accumulators run on two different time bases, one using F_r and the other using F_{out}, attention must be paid at the input to the ALU so that the setup and hold times of registers are not violated and the ALU does not go into a metastable state.

Another explanation of the operation of this PLL/DDS structure is to break it into two separate channels, as shown in Fig. 5-63. While the phase output of A accumulates up, the phase output of B accumulates down (because it is subtracted). A stable state (lock) will be achieved when the phase outputs cancel.

The frequency can be multiplied up by putting a fixed divider K between the VCO and accumulator B. All the equations will be the

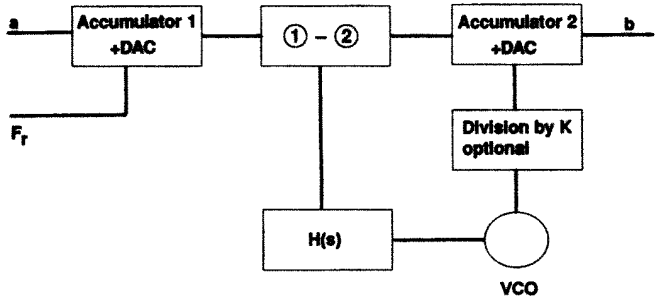


Figure 5-63 Accumulator PLL system—analog version.

same, only the output frequency will increase by the division ratio K, and the spurious signals will increase by 20 log K inside the loop bandwidth.

This design is also equivalent to a direct digital synthesizer inside the PLL, generates very similar artifacts, and, like fractional-N synthesis, does not require the ROM. One major advantage of this scheme is that a narrowband loop can "clean" the majority of the wideband spurious signals inherent in DDS, very much as in fractional-N circuitry.

Another DDS-based PLL uses a simple PLL circuit, usually employing a fixed divider, but driving the PLL reference with a direct digital synthesizer signal. This way if the divider is fixed at, say, 100 and the direct digital synthesizer covers 4 to 5 MHz, then the PLL output can generate 400 to 500 MHz. Since the direct digital synthesizer has very small step size, even after multiplication by 100, the output will have a fine step size. The problem in such designs arises because the spurious signals are also multiplied by 20 log N, that is, by the division ratio. If 100 is used, the output will suffer 40-dB deterioration in spurious signal performance.

Although it is possible for specific applications, the use of this technique is very limited.

5-5-1 Speed up

In the last few years, switching speed has become an increasingly important parameter. Frequency is hopped to fight multipath and fading, and as a networking protocol (Frequency Hop-

ping Spread Spectrum). Switching speed depends on the switching excursion. If we are required to settle to within *df*, while hopping *dF*, the solution of the time differential equation of the loop transfer function shows that:

$$T_{sw} = -\ln(\xi \cdot df/dF)/\xi\omega n \qquad (5\text{-}16)$$

Example: For $\omega n = 2\pi \cdot 200$ rad/sec, $\xi = 1$. To hop 20 MHz and settle to within 2 kHz of the final frequency will take:

$$T_{sw} \approx -\ln(2/20000)/1260 = 7.3 \text{ msec.}$$

Regular cellular PLL circuits, with a 30-kHz step, achieve switching speeds of a few milliseconds when required to hop across 20–25 MHz. For GSM applications where the step size is 200 kHz and the loop is wider, speeds of 500 μsec are common.

To improve speed without adding extra complexity, speed-up circuits have been devised. These circuits either generate more output current from the phase detector during the transient (thus increasing ωn temporarily), or bypass the loop network and charge the loop capacitor directly for the transient period only. See the circuits in the TRF2050 and Philips SA7025 diagrams (also called PHI function).

5-6 Single-Chip PLL Synthesis

Most single-chip PLL devices are complex digital *integrated circuits* (*ICs*), with built-in flexibility for generation of reference frequencies, crystal oscillators, very high-division ratios, and digital phase detectors for single-loop PLL circuits.

Now we review some of the more popular devices and some of the devices that we consider outstanding.

1. *Motorola MC145xxx family.* These devices are a complete PLL circuit that requires only a dual modulus, a VCO, and a loop filter. It is an evolving family with speed and functionality improvements. All controls are serial so the devices can be packed in 16-, 20-, and 24-pin devices. Some older parts in the family have parallel control.

2. *Fujitsu 15xx family.* This is a highly integrated family of complete PLL chips which include the dual modulus and operate

up to 1.2 GHz. A general block diagram is included in Fig. 5-64 and represents the general architecture of single-chip phase-locked loop ASICs.

The chip contains an input port for a reference input (or an external crystal), programmable reference divider, programmable main divider with a 64/65 or 128/129 dual modulus, and a dual-phase detector with a lock indicator function. The devices use very low power, some as low as 8 mA at 3-V supply, and have a serial interface to save packaging size and therefore cost.

3. *Qualcomm Q3036.* The Q3036 from Qualcomm is one of the more advanced and impressive devices. It is a 1.6-GHz silicon single-chip PLL. A block diagram of the device is shown in Fig. 5-64. The device is suitable for generating high frequencies but with large step size, mainly 1 MHz or higher, although other step sizes can be implemented.

The reference input is driving a programmable counter with a division ratio of 1:16. The VCO input is connected directly to a dual modulus 10/11 that operates at the input frequency, up to 1.6 GHz. The division ratio is 90:1295, and it is also possible to program division ratios of 2 to 128 but with an input frequency limited to 300 MHz (this is done by bypassing the dual-modulus prescaler internally).

The device includes the phase detector, lock indicator function, and control interface that is using either an 8-bit bus or a parallel loading. However, the divider output is not available to the user, and it seems to us that this device cannot be controlled to operate in a fractional-*N* mode. Power dissipation is 1.7 W typically.

4. *Plessey.* Part NJ88C30 is a member of a family of products which consist of a single-chip PLL synthesizer, generating frequencies up to 200 MHz. The company offers parts that run up to 2.5 GHz. A block diagram is shown in Fig. 5-65: it consists of an input for a crystal reference, a 16/17 dual-modulus divider, and control circuitry for controlling the division ratio for the reference and the input. The device includes a phase comparator as well. Control is serial, and the device is housed in a 14-pin IC, either DIP or SOIC.

Part SP8854 from Plessey is a similar device but operates up to 2.7 GHz. The block diagram is shown in Fig. 5-65.

5. *Signetics/Philips*. Two modern chips are the TSA5511 and the UMA1014T. The first is a 1.3-GHz single-chip synthesizer ASIC, and the second a 1.1-GHz device. Both use serial interface and come in surface-mounted packages. Current is 35 mA for the 5511 and 13 mA for the 1014.

6. *Philips PLN UMA7025.* This is a 1-GHz fractional-*N* PLL chip, with two levels of fractionality, 5 and 8. Interface is serial, and the device also includes a fixed-frequency PLL circuit with BiCMOS technology. The device uses a trimodulus divider, 64/65/72, to allow low division ratios (the minimum number is a little bit above 1000). The 8025 is a similar device, at 2000 MHz.

7. *Sciteq PLN SCI-1602.* This is a 2.5-GHz fractional-*N* PLL chip of the first order. It employs a fixed divide-by-2 or divide-by-4 prescaler followed by the dual modulus (16/17 up to 1250 MHz) and fractional circuit. Fractionality is 0 to 32 in SOS (Si on sapphire) technology. [In development.]

8. *TI TRF2050, 3040.* The TRF2050 is a 1.2-GHz fractional-*N* dual PLL ASIC and the TRF3040 is 2-GHz fractional-*N* + quadrature modulator device. See block diagrams in Figure 5-66.

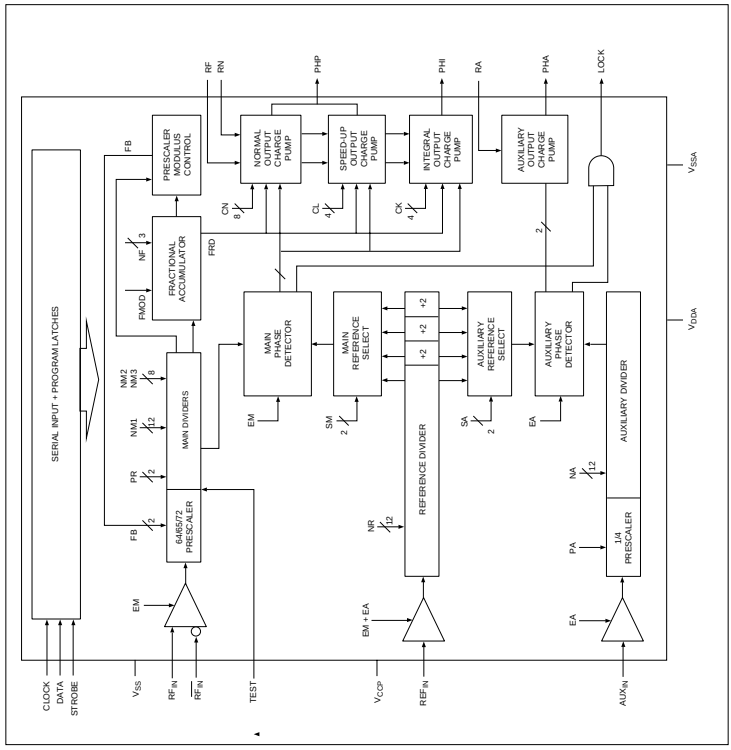
9. *PEREGRINE Semiconductor*. Using silicon on Sapphire (SOS) technology, Peregrine developed a very low-power, low-voltage dual fractional-*N* PLL ASIC, with the main covering 1.2 GHz and auxiliary 550 MHz. See block diagram in Figure 5-67.

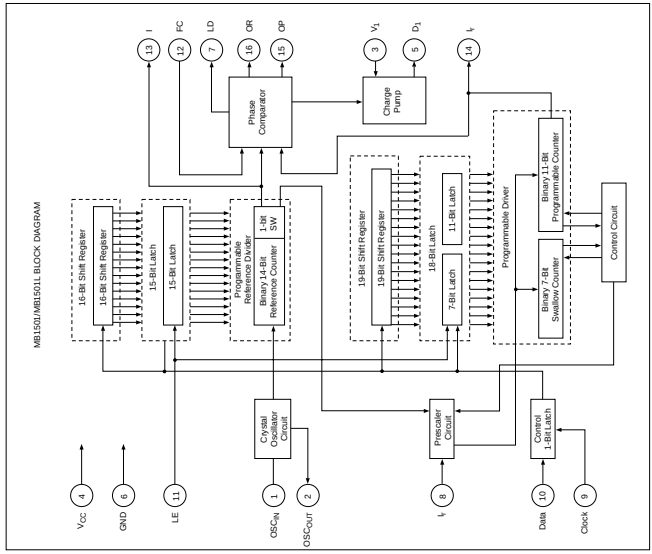
5-7 Conclusion

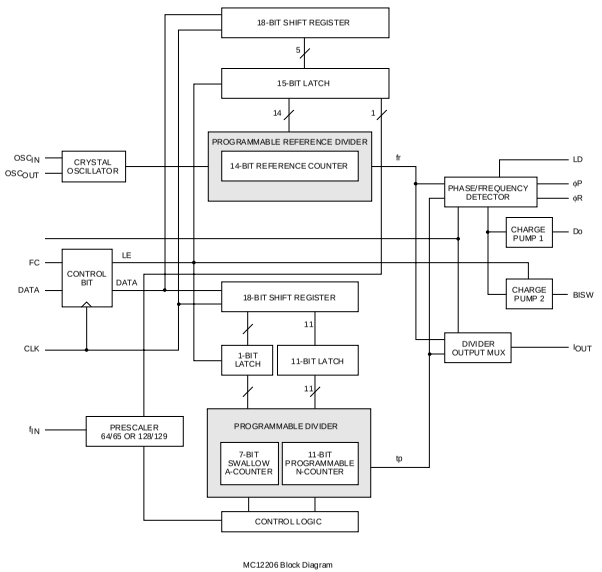
PLL synthesis is a mature and popular technology. PLL is used in all electronics, from consumer products to instrumentation, computers (disk drives), satellite communications terminals, and sophisticated radars and military hardware.

PLL is also "going" digital. A greater part of the circuit is now digital and is being implemented by low-power ASIC devices. More fractional-*N* and DSP will be used in the near future for most PLL designs because the advantages are very significant and the added complexity and cost are very marginal. Over the next two to five years, development of an all-digital, spurious-signal-free circuit will be the great challenge.

BLOCK DIAGRAM







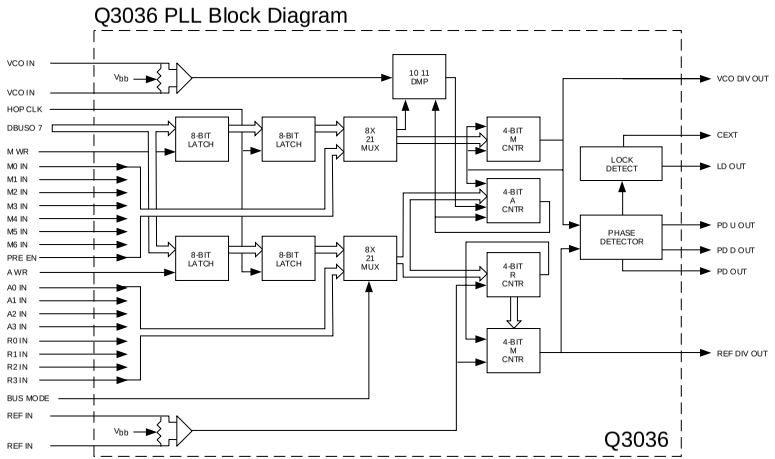


Figure 5-64 Popular PLL chips: 1501 (Fujitsu), 7025 (Philips), 12206 (Motorola), and 3036 (Qualcomm). (Courtesy of the manufacturers.)

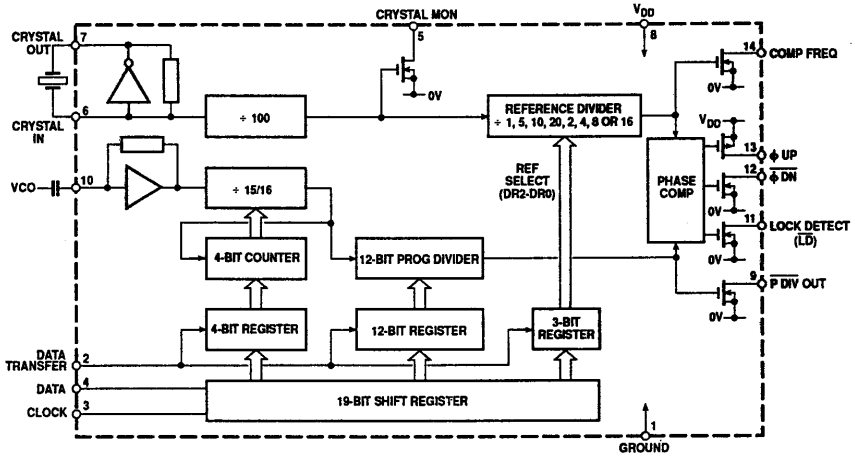


Figure 5-65, part a Block Diagram

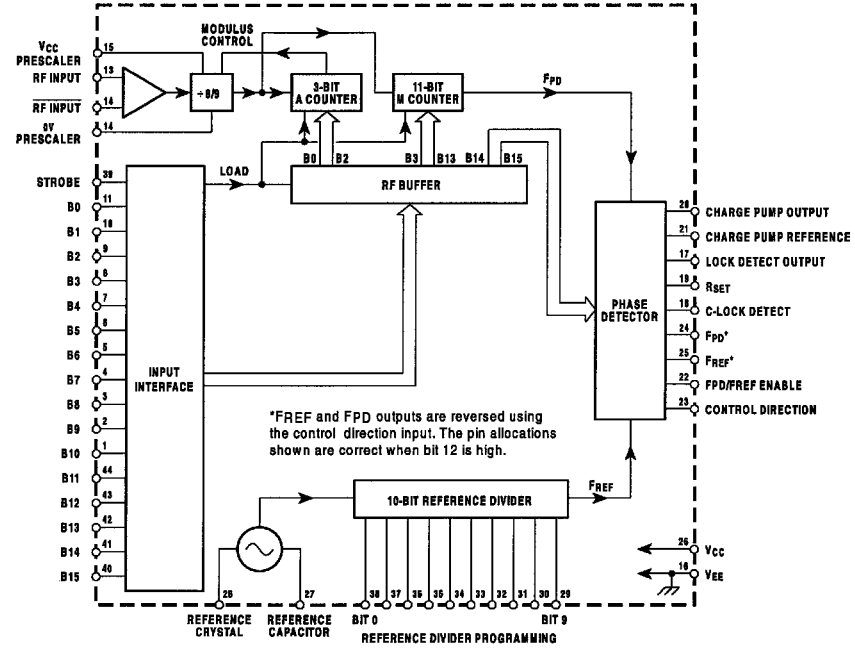


Figure 5-65, part b Block Diagram

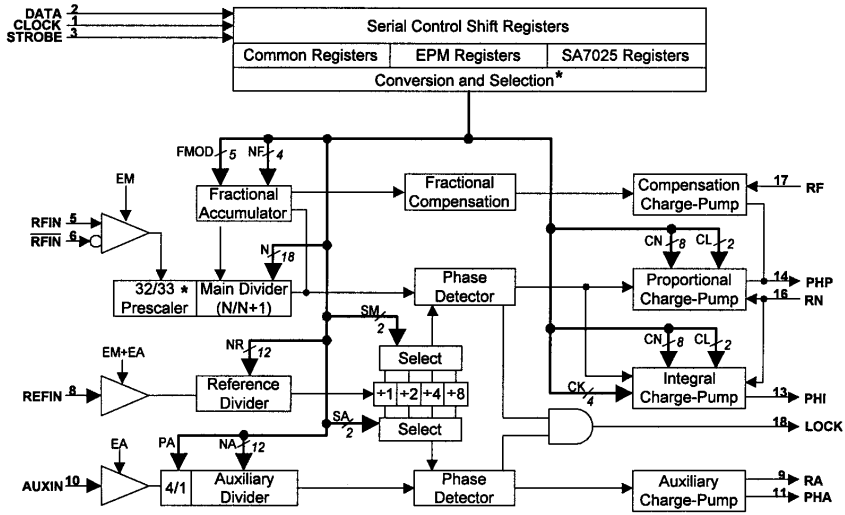


Figure 5-66 TRF2050 Diagram

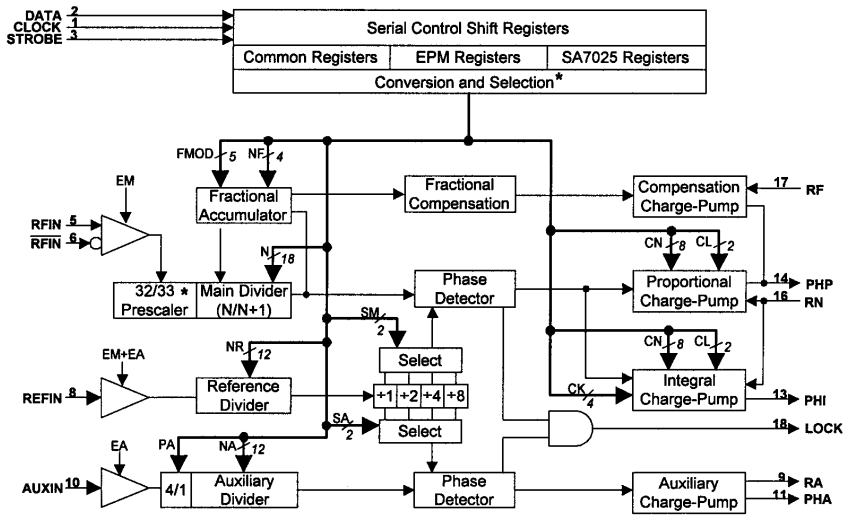


Figure 5-67 PE328A Block Diagram

References

1. Vadim Manassewitsch, *Frequency Synthesizers,* Wiley, New York, 1987.
2. Ulrich Rhode.
3. William F. Egan, *Frequency Synthesis by Phase Lock,* Robert Krieger Publishing, 1990.
4. Garry Gillette, "The Digiphase Synthesizer," *Frequency Technology,* August 1969, pp. 15–29.
5. F. Gardner, *Phaselock Techniques,* Wiley, New York, 1966.
6. A. Blanchard, *Phase Lock Loops,* Wiley, New York, 1976.
7. Andrew J. Viterbi, *Principles of Coherent Communications,* McGraw-Hill, New York, 1966.
8. Robert J. Bosselaers, PLL including an arithmetic unit, U.S. patent 3913028, October 1975.
9. Hewlett Packard and R. G. Cox, Frequency synthesizer, U.S. patent 3976945.
10. Hewlett Packard and C. A. Kingford-Smith, Device for synthesizing frequencies with fractional multiplier of a fundamental frequency, U.S. patent 3928813.
11. RCA and A. T. Crowley, PLL frequency synthesizer including fractional digital frequency divider, U.S. patent 4468632.
12. Hughes Aircraft Company and J. A. Crowford, Enhanced analog phase interpolation for fractional-*N* frequency synthesizer, U.S. patent 4586005.
13. U. S. Phillips and K. D. McCann, Frequency synthesizer having jitter compensation, U.S. patent 4599579.
14. Motorola and F. L. Martin, Frequency synthesizer with spur compensation, U.S. patent 4816774.
15. Motorola and F. L Martin, Frequency synthesizer with spur compensation, U.S. patent 4918403.
16. Motorola and W. P. Sheperd et al., Fractional-*N* synthesizer having modulation spur compensation, U.S. patent 5021754.
17. Hewlett Packard and B. M. Miller, Multiple modulation fractional-*N* divider, U.S. patent 5038117.
18. Racal Dana and M. A. Wheatley et al., Frequency-modulated PLL with fractional-*N* divider and jitter compensation, U.S. patent 5038120.
19. General Dynamics and W. G. Greken, Digital frequency synthesizer, U.S. patent 3882403.
20. Engelman Microwave and W. J. Tanis, Frequency synthesizer having fractional-*N* frequency divider in PLL, U.S. patent 3959737.
21. Marconi and N. G. Kingsbury, Frequency synthesizer with fractional- *N* division ratio and jitter compensation, U.S. patent 4179670.
22. Adret and J. Remy, Frequency synthesizer including a fractional- *N* multiplier, U.S. patent 4458329.
23. Plessey and T. Jackson, Frequency synthesizer of the fractional- *N* type, U.S. patent 4800342.
24. Plessey and C. Attenborough, Fractional-*N* frequency synthesizer with modulation compensation, U.S. patent 4686488.
25. Thomson-CSF and A. Albarello et al., Fractional-*N* division FS for digital angle modulation, U.S. patent 4492936.
26. RCA and R. O. Yeager, Fractional- *N* frequency divider, U.S. patent 4573176.

27. Hewlett Packard and A. P. Edwards, Low phase noise radiofrequency synthesizer, U.S. patent 4763083.
28. *Hewlett Packard Journal,* February 1981 (special issue on HP8662A).
29. "Phase Noise Characterization of Microwave Oscillators," Hewlett Packard product note 11729B-1.
30. B. G. Goldberg, U.S. patent 5224132, June 1993.
31. P. M. Wilson, "Spurious Reduction Techniques for DDS," Colloquium on DDFS, University of Bradford, November 1991 (IEE digest no. 1991/172).
32. B. Miller and B. Conley, "A Multiple Fractional Divider," 42nd AFCS.
33. Y. Matsuya et al., "A 16-Bit Oversampling A/D Conversion Technology Using Triple Integration Noise Shaping," *IEEE J. Solid State Circuits*, December 1987, pp. 921–929.
34. D. R. Welland et al., "A Stereo 16-Bit Delta Sigma A/D Converter for Digital Audio," *J. Audio Eng. Soc*., June 1989, pp. 476–484.
35. R. M. Gray, "Oversampled Sigma-Delta Modulation," *IEEE Trans. Comm. Theory,* May 1987, pp. 481–489.
36. Eagleware Corporation software CAD, PLL module, Windows/NT program. Stone Mountain, GA.

Chapter

6

Accumulators

The accumulator is one of the major elements of direct digital synthesizers. It serves as the phase generator and determines the resolution, frequency, and spurious signal formations. This chapter presents a short survey of accumulator implementations.

The two most common accumulators are the binary type and the decimal (BCD). The binary one is a "natural" from the hardware implementation standpoint. Logic circuitry is binary by nature, and it is therefore the most common way of executing accumulator circuitry. However, decimal or BCD is a "natural" for users, too. After all, we all use base 10 to count and live in a "decimal" arithmetic. Thus for most instrument makers and where human interface is required, BCD logic is preferred, and so we survey this technique, too.

As a consequence of the available hardware, the vast majority of direct digital synthesis (DDS) ASICs on the market use binary logic. The efficiency of the accumulator and the simplicity of the interface to the ROM present an advantage. However, BCD accumulators and BCD direct digital synthesizer ASICs and products are available, too.

6-1 Binary Accumulators

The binary accumulator is a digital integrator, performing the arithmetic function

$$S(n) = S(n - 1) + W \tag{6-1}$$

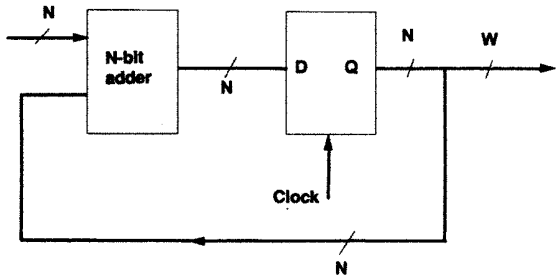


Figure 6-1 Accumulator block diagram.

where $S(n)$ is an N-bit word and W is the input control. The accumulator is usually constructed by adders and registers, as shown in Fig. 6-1. The register is a storage device which changes its output only when clocked.

The N-bit accumulator determines the output resolution frequency, given by

$$F_{\text{res}} = \frac{F_{\text{ck}}}{2^N} \tag{6-2}$$

To achieve some fixed-frequency resolution, the accumulator (number of bits N) must grow as F_{ck} grows. The minimum accumulator size is shown in Table 6-1 to meet <0.1-Hz, <1-kHz, and <1-MHz resolution.

Since all arithmetic operations are done modulo 2^N, any input W having a value $0 \le W \le 2^N - 1$ has an equivalent input given by $W^* = 2^N - W$ that will yield exactly the same DDS output frequency. This is apparent from the sampling theorem. These two control inputs, W and W^*, generate what we call the *main* and the *aliasing* frequencies, respectively. The sum of these two frequencies is always the size of the accumulator (2^N for a binary type).

TABLE 6-1 Accumulator Size Required to Achieve Resolution for Different Clock Rates, Binary and (BCD)

	Resolution		
Clock, MHz	0.1 Hz	1 kHz	1 MHz
10	27 (32)	14 (16)	4 (4)
100	30 (36)	17 (20)	7 (8)
1000	34 (40)	20 (24)	10 (12)

However, as already demonstrated in Chap. 4, the generated pair frequencies will be opposite in sign (but have the same frequency value).

To demonstrate the accumulator output, let's use a 4-bit accumulator (for which $N = 4$) and generate the following pattern for an input $W = 1$ [starting with $S(0) = 0$]:

0000
0001
0010
0011
0100
0101
0110
0111
1000
1001
1010
1011
1100
1101
1110
1111
0000
⋮

This will be considered as frequency 1, rotating clockwise. However, the same accumulator, for the input $W^* = 2^4 - 1 = 15$ decimal or 1111 = F (hexadecimal) will generate the following pattern:

0000
1111
1110
1101
1100
1011
1010
1001
1000
0111

0110
0101
0100
0011
0010
0001
0000
⋮

These two patterns are completely symmetric, and can be viewed as the same accumulation rate, one being clockwise and the other counterclockwise, or one being positive and the other negative. This is the nature of sampled data, and we mentioned the reasons in Chap. 4.

The accumulator output is interpreted as the sine-wave phase. Therefore, state 0 is zero phase and state $2^N - 1$ is 2π.

The accumulator is usually constructed from similar blocks, as shown in Fig. 6-2*a* and *b* (in this example for a 24-bit accumulator using 4-bit similar blocks). All blocks are similar, and the only connection between blocks is through the carry bits. The same accumulator could be constructed by using twenty-four 1-bit adders, twelve 2-bit adders, three 8-bit adders, four 6-bit adders, and other combinations.

The resolution of the accumulator can be doubled easily by toggling the lowest carry input bit as shown in Fig. 6-3. In this case, for reset = 0 and carry input = 0, the accumulator operates normally. However, for reset = 1, the carry input toggles between 0 and 1 periodically, with the equivalent effect of adding ½ LSB weight to the accumulator, which means an additional bit of resolution. This is similar to a fractionality of ½! Such a feature can be extended; and if another accumulator runs with its carry output connected to the carry input of the original accumulator, this will provide arbitrarily additional resolution. Another unique way of adding resolution will be mentioned in Sec. 6-5.

In the design shown here, speed is limited by the critical path, which will be the carry functions. This function needs to propagate up the whole accumulator ladder. Complex carry look-ahead functions can be added (see Fig. 6-2*b*), but in most cases a pipelined structure is used. The fastest 24-bit straightforward design we have seen in silicon runs at 120 MHz, though much higher speeds can be achieved at the cost of carry look-ahead complexity. This

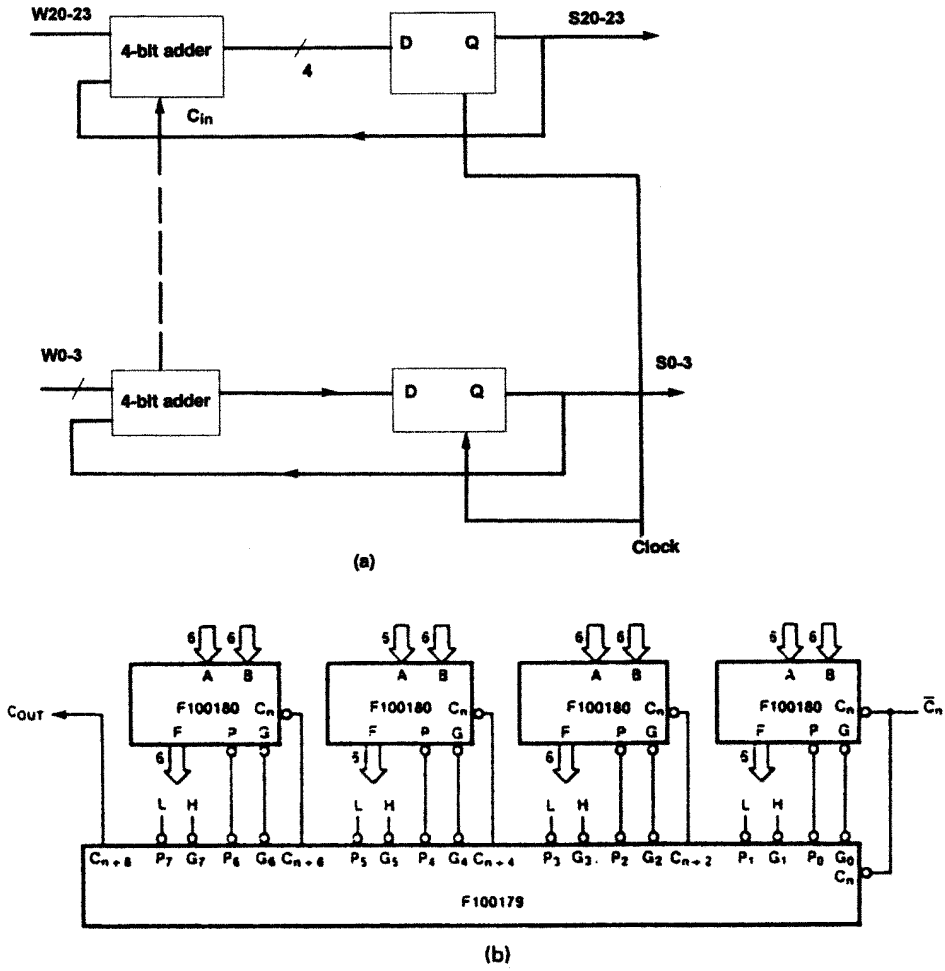


Figure 6-2 (*a*) A 24-bit accumulator using 6-bit adders; (*b*) a 24-bit accumulator using carry lock-ahead logic. (*Courtesy of Signetics.*)

design does not use pipelining and executes the function by using three 8-bit adders and complex carry look-ahead functions, and therefore it completes the accumulation in 1 clock tick (8 ns).

A general block diagram showing the use of standard 100K *emitter-coupled logic* (*ECL*) arithmetic parts, using adders and complex carry look-ahead functions (available in all logic families as standard devices), is shown in Fig. 6-2*b*. The F100179 was designed to operate with four adders (in this case, each is a 6-bit adder), and this design is capable of adding a 24-bit word in 1 clock

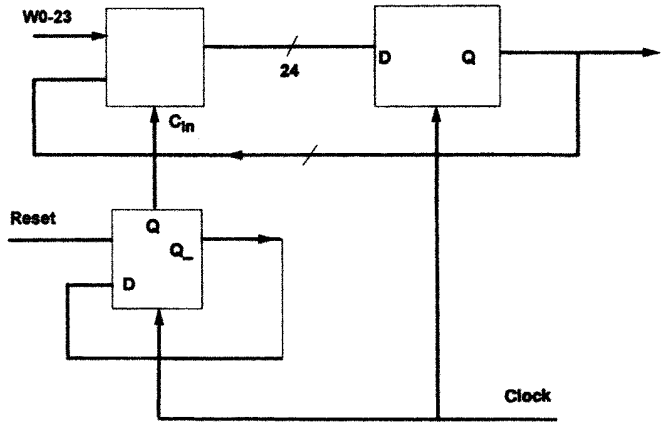


Figure 6-3 LSB extension.

cycle at speeds approaching 80 MHz. If higher speeds are necessary, a pipeline structure is usually employed. This allows one to simplify the design and gain speed. The pipeline requires partitioning of the accumulator, and the size of the adders is a design parameter; the principle is demonstrated in Fig. 6-4 again for a 24-bit accumulator using 4-bit adders.

Now, after employing the pipeline structure, the critical path is in the basic block, and there is no requirement for complex carry look-ahead functions. If the basic block can operate at a specific

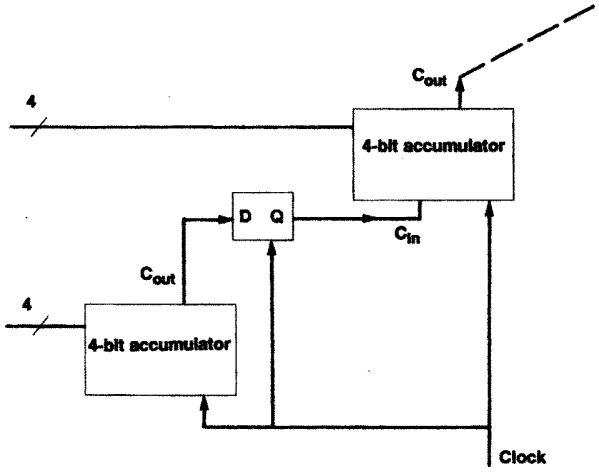


Figure 6-4 Accumulator pipeline structure.

speed, the whole accumulator can operate at this speed. But there is a price to be paid. First and foremost, there is an increase in complexity. In addition, when a new frequency W is loaded, it will reach the higher stages before the carry of the lower stages propagates to them. This will create a transient in the output, until the accumulator has been cleared of the remnants of the previous W. In many applications, this is not important, and the user does not care about the behavior of the phase in such a short transient. After all, if the accumulator runs at a 100-MHz clock speed and uses 16 levels of pipeline, then the transient will last $16 \cdot 10 = 160$ ns, and this is very fast for most applications.

However, other applications need to use the DDS feature of switching from frequency to frequency smoothly, without a transient. This is important when a smooth sweep needs to be generated [for testing, for linear FM (chirp) radars, spread-spectrum communications, simulation of Doppler signals, smooth phase modulations like MSK, and such].

To remedy this problem, two main techniques have been devised. One uses a timing circuit in the input and loads the input bits sequentially with the propagation of the pipe, as shown in Fig. 6-5.

The output has to be compensated, too, and this is done by delaying the signals through equalizing registers (D flip-flops). This structure simplifies the input because a lot of delay elements (usually D flip-flops) are replaced by a simple timing circuit. Also note that since the accumulator output is usually limited to a much smaller bit count than N (its size S, the number of bits connected to the ROM, is hardly ever more than 16 bits wide), the

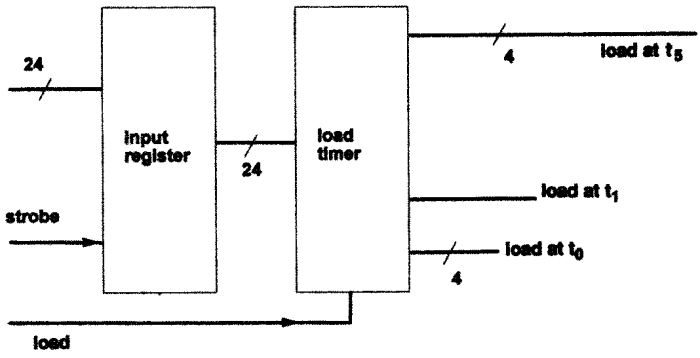


Figure 6-5 Sequential loading of accumulator.

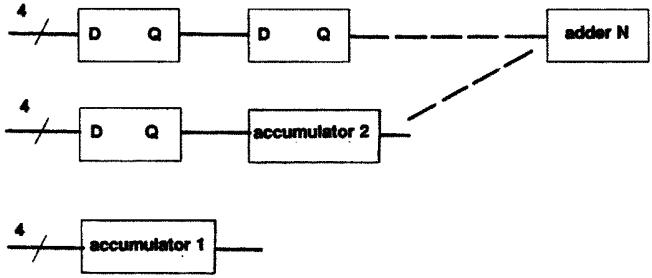


Figure 6-6 Full pipeline accumulator.

complexity of the output-compensating logic is much lower than would be required in the input.

The cost of this simplified implementation is that frequency can be updated only at F_{ck}/P, where P is the number of pipelined stages. If this cannot be tolerated, then the input must be fully compensated, as shown in Fig. 6-6. This architecture allows one to change the input at the clock rate. In both architectures a delay is generated in the pipeline and is equal to P/F_{ck} s.

The state of the art of DDS accumulators available on the market is 400 MHz in CMOS and 1500 MHz in bipolar silicon and GaAs implementations.

A linear FM product, manufactured by Sciteq Electronics, DCP-1, uses a dual accumulator device and updates, fully compensated, the accumulator output every 2 ns, permitting a 500-MHz clock speed. This is the fastest fully compensated product known to us.

A shortcoming of binary accumulation is the step size obtained. If, e.g., a 40-MHz signal is used to clock a 24-bit binary accumulator, then the bit weight is shown in Table 6-2. The numbers generated are not round, but because of the simplicity and economy of

TABLE 6-2 Bit Weight for Binary Accumulator

F_{ck} = 40 MHz

bit 23 (MSB)	20 MHz (Nyquist)
bit 22	10
bit 21	5
bit 20	2.5
bit 19	1.25
⋮	⋮
bit 0	$40{,}000{,}000/2^{24} = 2.384$ Hz

DDS, and binary designs, some manufacturers use very large accumulators, say $N = 48$, and use a microprocessor to control approximations to decimal numbers with such accuracy that it is next to impossible to detect the error. For example, with a 48-bit accumulator at a clock rate of 40 MHz, any decimal number can be approximated to

$$\frac{40 \times 10^6}{2^{48}} = 1.42 \times 10^{-9}\ \text{Hz} = 1.42\ \text{nHz} \qquad (6\text{-}3)$$

That is lower than 2 nHz (a signal that has a cycle of 0.5 billion hours)!

Another way to generate round numbers is to clock the accumulator with a binary clock. For example, clocking a 24-bit accumulator with a clock $F_{ck} = 2^{24}$ Hz yields exactly 1-Hz resolution. This is not convenient if the clock has to be derived from a 5- or 10-MHz reference.

6-2 Decimal Accumulators

Binary-coded decimal (BCD) is the logic we are naturally comfortable with. After all, most of us can do some simple arithmetic in hexadecimal, but otherwise we use BCD. Almost without exception, all instrumentation uses BCD. Few techniques have been devised to use decimal DDS designs. The first one (part of Ref. 1) uses a BCD all-digital logic and is shown in Fig. 6-7.

Every stage uses a BCD adder whose block diagram is shown in Fig. 6-8 and whose function is shown is Fig. 6-9. The last stage (the MSB) can be a BCD stage or a binary. As an example, if the last stage is BCD, the whole device is purely decimal and the clock has to be 1 or 10 or 100 MHz. But if the last stage is binary, the clock can be 1.6 or 16 or 160 MHz. If only 3 bits are used for the last MSBs, 8 or 80 MHz can be used for a clock (see STEL P/N 1176). For some designs this is an advantage.

Another method to produce round frequencies with decimal clocks (part of Ref. 2) is by using binary adders and accumulators which count to their terminal state and then add the difference to a decimal number. As an example, if 10-bit adders are used, the adder accumulates to 1023 (3FF hexadecimal); but instead of starting next at 0, it adds 24 each time the adder overflows, thus counting 1000 steps exactly. See Ref. 1. A block diagram is shown in Fig. 6-10.

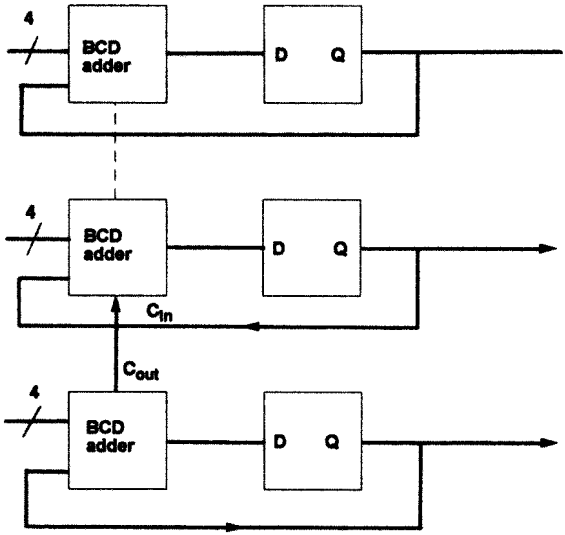


Figure 6-7 BCD (decimal) accumulator.

6-3 Interface to ROM

One of the difficulties of using a pure BCD structure is the inefficiency of the ROM. The architecture above helps resolve this issue.

This architecture (Jackson) has the advantage of easily converting back to binary and is a more efficient interface to the memory. Note that if, say, in the previous architecture (pure BCD) three decades are connected to the direct digital synthesizer memory (ROM), it requires a 12-bit input ROM (4K ROM) which will be addressed only in 1000 addresses, so only 25 percent of the ROM is utilized. In the above-mentioned architecture (complementation of binary), almost 100 percent of the ROM is utilized.

However, if the MSBs of an otherwise BCD accumulator are binary, say, 4 bits, then the clock can be 16 MHz and 12 MSBs driving the ROM will map 1600 points rather than 1000. The output bits of a BCD accumulator (for $W = 3$) are shown in Fig. 6-11.

A problem associated with the Jackson architecture is in the input control. Usually BCD designs require BCD input control. This works well for the pure BCD execution. However, the Jackson method requires binary inputs, and if BCD is required, an array of logic converting binary to BCD is needed.

Thus, each method has advantages and disadvantages.

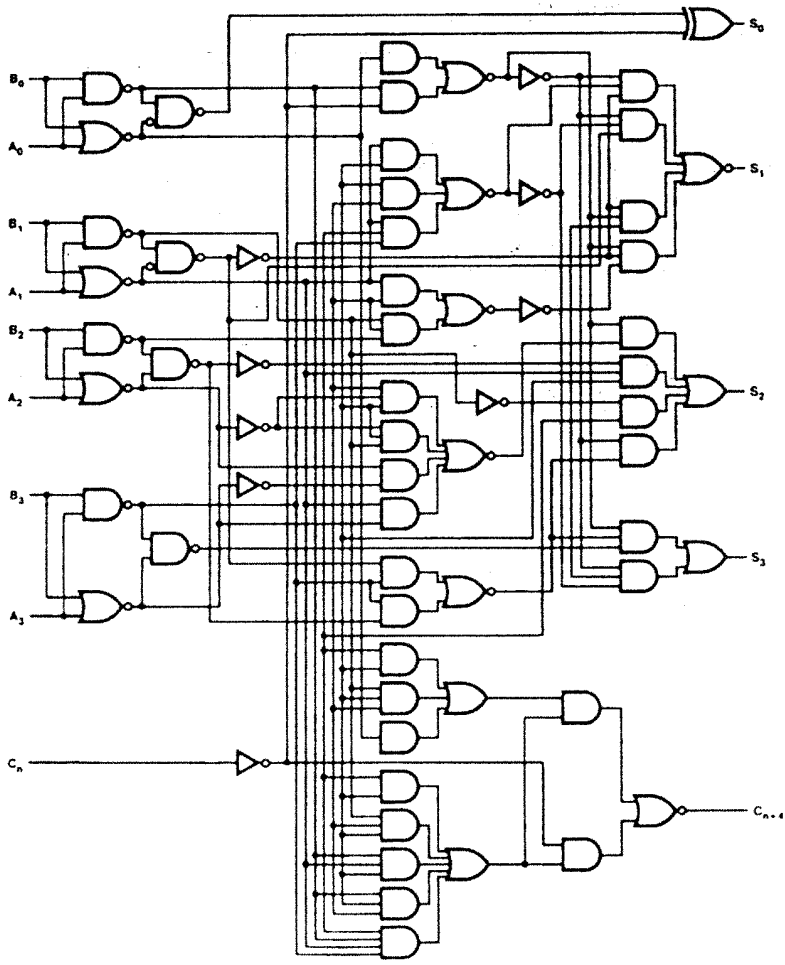


Figure 6-8 BCD adder logic.

$$a + b = \begin{cases} a + b & \text{if } a + b < 10 \\ a + b - 10 + \text{carry output} & \text{if } a + b > 9 \end{cases}$$

Figure 6-9 BCD adder function. (Courtesy of Signetics.)

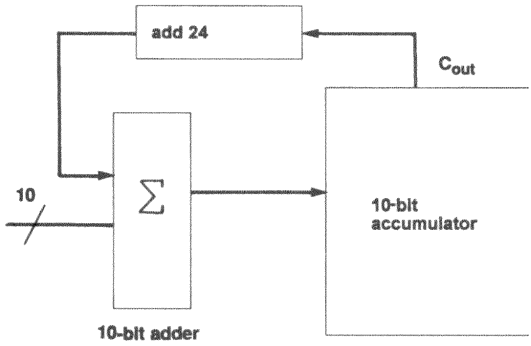


Figure 6-10 Jackson's BCD/binary accumulator. A 10-bit binary adder adds 24 every time the carry output is activated, therefore accumulating to 1000.

S(n)		C_{out}	S(n)		C_{out}
0	0000	0	8	1000	0
3	0011	0	1	0001	1
6	0110	0	4	0100	0
9	1001	0	7	0111	0
2	0010	1	0	0001	1
5	0101	0			

Figure 6-11 Output S(n) and carry output C_{out} for a BCD accumulator with input W = 3.

6-4 Accumulator DDS

As mentioned in Chap. 4, sometimes the accumulator is used to generate the required output. The average frequency of the accumulator carry output is always the correct frequency, given by $F_{ck}W/ACM$. The problem is that the instantaneous frequency is wrong and creates a high level of spurious signals, as shown in App. 4B. For time-keeping purposes (i.e., the average frequency is the only important parameter), this technique is good enough, since the instantaneous frequency does not matter. In such a case, the accumulator has the accuracy and the resolution of the total direct digital synthesizer.

6-5 Phase Adder and Accumulator Segmentation

Most DDS accumulators are followed by an adder. Since the accumulator output is interpreted as the signal phase, the phase adder allows the production of high-accuracy phase modulation. In a binary implementation, the MSB will have a weight of 180°, the second MSB will have a weight of 90°, and generally the Nth bit has a phase weight of $360 \cdot 2^{-N}$. This function usually operates as

a phase modulator for different modulation schemes and produces accuracy not available with analog methods.

However, some interesting features can be added when this function is available. If this input is now activated with the output of another accumulator (instead of the fixed data mentioned before), it will create a linear phase ramp. In this sense it is equivalent to programming a new frequency with the original accumulator. If, e.g., we use a standard 24-bit accumulator running at 400 MHz and we activate the second MSB, then the output frequency rate will be such that it will generate a 100-MHz signal. If we now activate the LSB, the output frequency will increase by $400 \times 10^6/2^{24} = 23.84$ Hz. We could achieve the same function by adding an accumulator with this rate to the phase input port, i.e., complete its cycle at the rate of 23.84 Hz. The difference is that while in the original accumulator the whole logic had to run at 400 MHz, the new accumulator can run at much slower speed! It is useful mainly when very high-speed logic is used. The reason is clear. For the original accumulator, an LSB has to clock many times until it gets to the output bits. But since the new configuration allows the new accumulator access to the output phase bits immediately, their speed can be slowed by the ratio of their "original" location to the new one. This ratio can be very significant!

Let's look at some numbers. Suppose that the 24-bit accumulator outputs 12 bits to the ROM. Suppose also that we want to run the secondary accumulator at $400/16 = 25$ MHz. Now we can add to the phase input another accumulator that runs at a clock speed 2^{-4} slower than the original. Because the second accumulator needs to generate 12 bits of phase, only the 12th MSB can be activated; therefore 11 MSB bits cannot be used for frequency generation (but can be used for phase control instead). Now the maximum frequency generated from the secondary accumulator can generate $400/(2^4 \cdot 2^{12})$ MHz, while the first accumulator has to generate the rest of the frequencies, i.e., it has to have 16 bits. Instead of a very demanding 400-MHz 24-bit base logic with adders and registers running at this speed, with tough timing specifications and a lot of power, we can make do with a 16-bit 400-MHz accumulator and a 24-bit secondary that runs at low clock speed. The secondary is clearly a CMOS part—simple to design and drawing almost no power. There is the cost of additional low-frequency unused bits, but in many designs this arrangement can be very significant, especially for power conservation. This config-

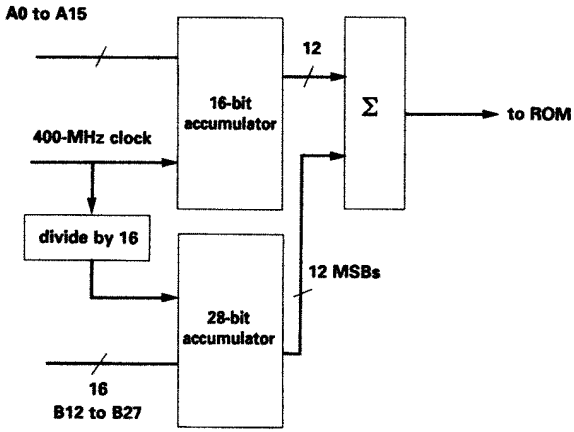


Figure 6-12 Segmented accumulator general structure. $B(0)$ to $B(11)$ are not used and $B(12)$ is MSB; the frequency generated is clock/$(16 \cdot 2^{13})$, equivalent to bit 17 of the main (A) accumulator.

uration can also save a lot of pipeline logic and therefore power. There has not been much use made of this fragmentation and there has been no mention in the literature, but it is elegant and working and therefore worth the effort. The same configuration can be used for chirp generation and in fact any other periodic function; however, the chirp implementation creates a small error. We have designated this technology as *accumulator segmentation* since this has not been published in other sources. A general configuration with 32 bits is shown in Fig. 6-12.

6-6 Conclusion

The accumulator is clearly a cardinal part of DDS technology. The addition of phase modulation is a simple but important feature. Both binary and BCD accumulator structures are being used, and additional features (speed, resolution) can be devised via the carry input and phase input ports to improve power utilization, thus reducing complexity and cost and improving overall performance.

References

1. Leland B. Jackson, Digital frequency synthesizer, U.S. patent 3735269, 1973.
2. B. G. Goldberg, Digital frequency synthesizer, U.S. patent 4752902, 1988.
3. Signetics Corporation, TTL manual, 1986.

Chapter

7

Lookup Table and Sine ROM Compression

In a direct digital synthesizer, the ROM is used as a lookup table to convert its phase input digital data bits to output digital amplitude data bits, to drive the DAC.

As mentioned above (see Chap. 4), the output of the accumulator is used as the address input of the lookup table and represents the sine-wave phase. This phase information needs to be converted to its amplitude value to drive the DAC and achieve the required analog output.

Any lookup table logic format will do, as long as it performs the transformation $\varphi \rightarrow \sin \varphi$ correctly (meaning that any logic configuration can be used). However, the majority of DDS designs, especially those that map more than 10 phase input bits to 8-bit amplitude output, use some form of memory devices, which will be referred to here as the ROM.

Up to approximately 15 years ago, memory devices were quite expensive and limited in size. Thus speed and compression algorithms (to reduce the actual size of the memory) became important. However, very little progress was made in theoretical work and compression efficiency, since the DDS industry was in its infancy. In the meantime, during the 1980s, memory technology matured and achieved low cost and large scale. For many designers, memory size became irrelevant, especially when operation was below 20- to 30-MHz speed. However, with the advance of very high-speed DDS, at and above 1000-MHz clock speed, memory size became again a critical component. Compressing the size

of the ROM is important for low- and high-speed DDS applications. Application-specific integrated circuits (ASICs) integrate the complete digital portion of DDS circuits, and production costs are proportional to dice size. With very high speeds this is even more critical because of complexity and heat-dissipation issues.

Thus, ROM compression offers economy and efficiency, reduction in die size, and better yield; and if it is available, it will be welcome in any design.

During the 1980s, some important theoretical work was done to achieve a very high level of signal compression to reduce the complexity of the ROM. Since the time when the work was completed, its use has spread to all (low- and medium-speed) devices. Reducing the ASIC density always affects die size, yield, and eventually cost. Also, for engineers with a theoretical bent, this represents an intellectual challenge; it is interesting, challenging, and rewarding. A level of understanding of the basic signal structure and a lot of computer optimization are involved. Few in the industry and the academy have spent substantial effort in this area and found it interesting and rewarding. Since the basic structure of the waveform is investigated, computer simulation is a powerful tool in helping optimize the results.

This chapter presents surveys of ROM algorithms and ROM compression techniques. Many of the ROM compression algorithms are protected by patents. However, most of the information is available in the public literature, and it is expected that better efficiencies in the compression levels will be seen in the future. All DDS ASICs on the market use some form of compression.

7-1 ROM Algorithm

A general expression for the transformation that is performed in the ROM, i.e., the mapping of φ to $\sin \varphi$, is as follows: If W is the word size of the input (2^W combinations) and D is the output word (see Fig. 7-1), then

$$s(j) = 2^{D-1} + \text{int}\left[(2^{D-1} - 1)\sin\frac{2\pi j}{2^W} + 0.5\right] \qquad j = 0 \text{ to } 2^W - 1 \qquad (7\text{-}1)$$

Since the arithmetic functions operate in this case with only positive numbers, all values are offset around 2^{D-1}, which is the DAC

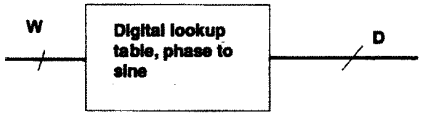


Figure 7-1 Digital lookup table; $S(j) = 2^{D-1} + 2^{D-1} \sin(2\pi i/2W)$.

center point. A few DAC manufacturers offer 2s complement as the interface logic, but the majority of the DACs operate in a true binary logic form. Therefore we will assume that the output of the ROM is monotonic from 000 to 111 (usually 0 to −1.0 V).

The sine multiplier is $2^{D-1} - 1$ so that the maximum peak-to-peak output value is $2^D - 2$; thus the maximum address value to the DAC will not exceed $2^D - 1$. The number of total addresses to the DAC is therefore $2^D - 2$.

In Eq. (7-1) the "int" function and the 0.5 inside the brackets are added so that the number will be quantized rather than truncated, the idea being to round the output value to the closest integer and generate a quantization error of not more than ±½ LSB. In some BASIC programs, the command "int" produces a quantized function, and the addition of the 0.5 is not necessary; however, QUICKBASIC uses "int" for truncation.

Some minor variations of this formula are possible, but basically they yield almost the same results. Note that this is a quantized version of a sine waveform, and there is always a level of error caused by the quantization effects relative to an ideal sine wave.

As noted earlier in Chap. 4, a good rule of thumb is to use W bigger than or equal to $D + 2$. This rule of thumb ensures no extra amplitude quantization; i.e., all addresses to the DAC will be visited. To gain an insight to this claim, let's review the output of the ROM near zero phase, where the rate of change is maximum. There, the output of the ROM is

$$(2^{D-1} - 1) \sin \frac{2\pi i}{2^W} \tag{7-2a}$$

which can be approximated as

$$\frac{2^{D-1} \cdot 2\pi i}{2^W} = 2^{D-W}\pi \tag{7-2b}$$

TABLE 7-1 ROM Size without Compression

$D \backslash W$	10	12	14	16
8	8,192	32,768	131,072	524,288
10	10,240	40,960	163,840	655,360
12	12,288	49,152	196,608	786,432

In the case where $D = W$, the ROM output will move from 0 at $i = 0$ to 3 at $i = 1$; therefore, addresses 1 and 2 will never be visited, and there are many others, too.

For $W = D + 1$, the ROM output will move from 0 at $i = 0$ to 2 for $i = 1$; and again, address 1—and many other addresses, too—will never be visited. For $W = D + 2$, the ROM output will visit all addresses (some more than once) and therefore avoid extra amplitude quantization.

The memory size for mapping various W values to D, assuming $W = D + 2$, is shown in Table 7-1. Clearly, the size of the memory becomes prohibitively large as W and D increase.

To map $W = 14$ input bits to $D = 12$ output bits in this straightforward mapping lookup requires $2^{14}(12) \approx 196{,}000$ memory bits. This is a large amount of memory even if the clock rate is only 20 MHz, and CMOS or bipolar ROM devices can be used.

Note that a cosine function can be loaded with equal results, except the output will be shifted 90° relative to that of the sine waveform.

When BCD logic is used (decimal DDS), the programming is more complicated since the input is represented by BCD numbers and therefore every digit (4 bits) represents a value from 0 to 9, and the other 6 hexadecimal states are never accessed. In such a case, there is a great waste of memory. For example, if 12 bits of the accumulator are connected to the input of the ROM, instead of representing in binary 2^{12} addresses, it represents only 0 to 999, or exactly 1000 states in BCD. Thus only 25 percent of the memory is used, and 75 percent of the ROM is never accessed. An example is shown in Table 7-2. It is also presented, through program BCDROM.01, on a disk the reader may obtain from the author (see Chap. 10). In this program, i is the input address to the ROM and D is the output according to the equation

$$D = 512 + \text{int}\left(511 \sin \frac{2\pi i}{1000} + 0.5\right)$$

TABLE 7-2 Binary-Coded Decimal ROM Mapping

i	D	
0	0	0°
1	3	
2	6	
3	A (10 BCD)	
4	D (13 BCD)	
5	10 (20 BCD)	
6	13	
7	16	
8	1A	
9	1D (29 BCD)	
10 (16 binary)	20 (32 BCD)	
11 (17 binary)	23	
12	26	
⋮	⋮	
249 (585 binary)	511	90°
⋮	⋮	
999 (2457 in binary)	−3	360°

All values are actually offset by 512, so address 999 will produce $D = 512 - 3 = 509$.

For $i = 1$, $D = 512 + 3$; for $i = 2$, $D = 512 + 6$; etc.

If the operation of the direct digital synthesizer is pure BCD, this waste cannot be avoided. However, another way of operating in a BCD manner was mentioned in Chap. 4 (using binary devices and preload), and by using this approach, the memory waste can be minimized since all operations are actually binary (see Ref. 15).

Let's review first the error that is generated by quantization for an ideal ROM without any compression at all. This can be calculated in two ways: either by performing a Fourier analysis [the *fast Fourier transform* (*FFT*)] of the ROM output [Eq. (7-1)] and picking a worst-case spurious signal or by calculating the mean square error. The results (all results now assume binary format) are shown in Table 7-3 for the cases $W = 10$, 12, 14 and $D = 8$, 10, 12, and all results are in decibels relative to the carrier power. As

TABLE 7-3

$D \backslash W$	10	12	14
8	−54, −47.6		
10		−62, −59.7	
12		−74.1, −71.2	−74, −72.1

can be seen from the simulation results, the rule of thumb that the worst spurious signal is approximately $-6D$ dB is validated.

Each entry in the table shows two numbers, both obtained by simulation. The first is the "ideal" quantized signal to mean square error, and the second is the worst-case spurious signal. This table was generated by a straight analysis (computer simulation) of the ideal quantized sine waveform.

To shrink the size of the ROM, compression algorithms apply. Compression adds errors and will degrade these results somewhat. As will be shown later, for the compression algorithms presented here, the loss is only 2 to 3 dB in mean square error.

7-2 Quadrant Compression

The size of the ROM can be reduced by more than 75 percent by taking advantage of the fact that only one quadrant of the sine (or cosine) needs to be stored, since the sine has the symmetric property, as shown in Fig. 7-2.

Thus, for $0 \leq a \leq 90$, $\sin(90 - a) = \sin(90 + a)$, $\sin(270 - a) = \sin(270 + a)$, $\sin a = -\sin(-a)$, and

$$\sin a = -\sin(180 + a) \tag{7-3}$$

The presentation of $\sin a$ only across the first quadrant $0 < a < 90°$ is sufficient to reconstruct all quadrants from the first.

Thus the 2 MSBs of W (the input word) are needed only to control the quadrants, and the values of the first quadrant need to be manipulated as shown in Table 7-4. Thus, given $\sin a$ over only the

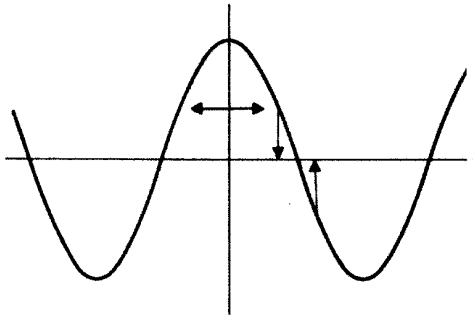


Figure 7-2 Sine quadrant symmetry.

TABLE 7-4 Quadrant Table

Phase	MSB	MSB − 1	Sine
$0 < A < 90$	1	0	$\sin A$
$90 < A < 180$	1	1	$\sin(90 - A)$
$180 < A < 270$	0	0	$-\sin A$
$270 < A < 360$	0	1	$-\sin(90 - A)$

first quadrant, the operations necessary to flip to the other quadrants are as follows:

Quadrant I: $\frac{1}{2} + \frac{1}{2}\sin i$ i is running index from 0 to $2^{W-2} - 1$

Quadrant II: $\frac{1}{2} + \frac{1}{2}\sin(i$ complemented) every bit inverted

Quadrant III: $\frac{1}{2} - \frac{1}{2}\sin(i$ complemented)

Quadrant IV: $\frac{1}{2} - \frac{1}{2}\sin i$

For all quadrants, i runs from 0 to $2^{W-2} - 1$.

This will operate in the following way: In the first quadrant, the sine starts at 0.5 (1000 binary) and curves up to 1 (1111). Then the address to the ROM goes from the maximum state to zero state (a new quadrant starts), but since all the bits are inverted, it will now curve back down the same way it curved up (111 to 1000). In the third quadrant, it starts again at 0.5, now 01111 , but because of the inversion, the value of the sine is maximized and the total sum declines to 0 (000) when it climbs back in the fourth quadrant (from 000 to 01111). Since most DACs operate in negative logic, that is, 000 is 0 V and 111 is −1.0 V, in reality, the output will be a negative voltage rather than a positive one, generating a signal that is 180° shifted relative to the one described above.

Therefore, to achieve the above equation, the MSB needs to be inverted, to the output, and the MSB and the MSB − 1 need to invert the sine function according to the quadrant they operate in. A typical block diagram is shown in Fig. 7-3.

The reduction of the ROM size is achieved because of the sine quadrant symmetry and because the inside ROM now maps $W - 2$ input bits to $D - 1$ output bits only, compared to W to D bits originally. The saving of ROM size is therefore

$$\frac{2^{W-2}(D-1)}{2^W D} = \frac{0.25(D-1)}{D} \approx 75\% \text{ saving} \qquad (7\text{-}4)$$

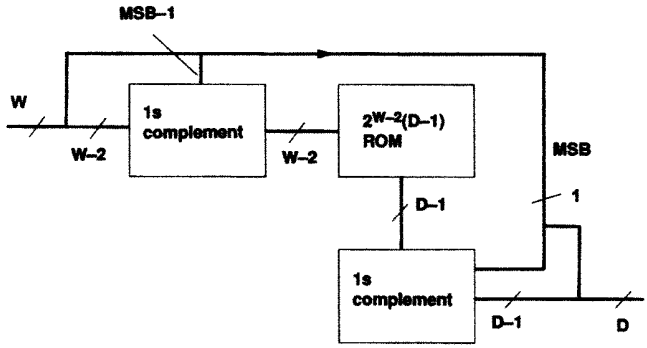


Figure 7-3 Sine quadrant compression.

The 1s complement is usually implemented by using XOR gates. So when the control input is 0, the output equals the input; and when the control input is 1, the output equals the input inverted. (See Fig. 7-4 for $W = 10$ and $D = 8$ bits.)

Table 7-5 shows the ROM size compared to Table 7-1. The MSB determines the sign of the output and indicates whether the phase is in the first two quadrants or in the second. The MSB − 1 is the quadrant bit and inverts the bits in the second and fourth quadrants.

Since all the numbers here are positive, it is convenient to measure positive or negative numbers relative to the value 2^{D-1}, which is shown in Eq. (7-1) as the offset. In this case the negative values are generated by inverting the MSB to the output, and they are indeed negative relative to the baseline value 2^{D-1}.

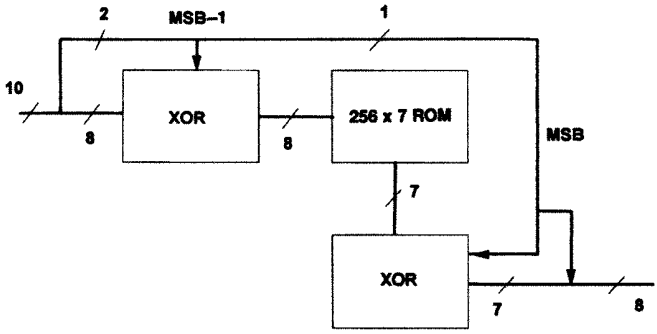


Figure 7-4 ROM for $W = 10$ and $D = 8$.

TABLE 7-5 ROM Size after Quadrature Compression

$D \backslash W$	10	12	14	16
8	1,792	7,168	28,672	114,688
10	2,304	9,216	36,864	147,456
12	2,816	11,264	45,056	180,224

The 1s complementer is a simple and common device; the operation requires two such inverters, one of depth $W - 2$ and the other of depth $D - 1$. This added complexity is minor relative to the very substantial reduction in memory size, and almost all DDS designs use the quadrant compression principle. Note that this compression, because of its built-in symmetry, generates a symmetric signal that by definition does not have even harmonics.

7-3 Compression Principles

Almost without exception, all other compression principles, at least the few published in the open literature, use two simple principles: the Taylor approximation and the fact that although the sine is a nonlinear function, it is still quite smooth and monotonic (referring here to the first quadrant only). See Ref. 3.

The Taylor approximation is given by

$$\sin(a + d) = \sin a + d \cos a - \frac{d^2 \sin a}{2} + \text{higher-order terms} \quad (7\text{-}5)$$

In fact, for all practical purposes, the first three terms are sufficient to understand and execute the principles of compression. When it comes down to designing an algorithm, a computer is used to optimize and "fine-tune" the values according to a given criterion (minimum mean-squared error, limit of error from the ideal at each point, and such).

The implementation of this simple principle is shown in Fig. 7-8.

Let's assume, then, that the quadrant will be divided into a number of sectors, and each sector will be represented by a straight line. This is a similar procedure to piecewise lineariza-

tion. The slopes of the straight lines are determined by the Taylor approximation values.

The great efficiency that can be achieved by these rather simple procedures is due to the fact the sine is a smooth function and rather simple to interpolate. If we divide the quadrant into, say, 64 sectors (it will always be divided into a binary number of sectors since all the addresses to the ROMs are binary numbers), then the value of the third term of the Taylor approximation is

$$\sin a \frac{d^2}{2} < \frac{d^2}{2} = \frac{(\pi/128)^2}{2} = 0.0003$$

which is equal to approximately 12 bits of accuracy at its maximum value. This means that if we divide the quadrant into enough sectors, we actually can do well using only the first two terms of the Taylor approximation.

7-4 Direct Taylor Approximation

Reference 4 discusses the direct implementation of the Taylor approximation (see Fig. 7-5), and its authors indicate the use of multipliers for the execution of the above equation (we have assumed that this multiplier is executed as a lookup table, another ROM). Since no details are given in the reference, it is hard to judge the effectiveness of the compression. No exact details are mentioned about the format of these ROMs.

Although the authors do not indicate the level of compression achieved, it is inferred that it was substantial; based on their discussion, we cannot estimate this number. The total memory size to convert 13 phase bits to 8 amplitude bits is given by

$$64 \cdot 12 \text{ (block 44)} + 64 \cdot 6 \text{ (block 46)} + 32 \cdot 5 \text{ (block 52)} + 2048 \cdot 7 \text{ (block 54)} \approx 15 \text{ kbits}$$

Based on these calculations, we have concluded that the drawing in Ref. 4 probably does not disclose details and that the authors do not achieve sufficient compression; later in this chapter, much higher ratios are demonstrated. We will also present detailed designs rather than general principles so that you can design a ROM compression algorithm quite easily for your own applications. However, Ref. 4 helps one to understand the basic

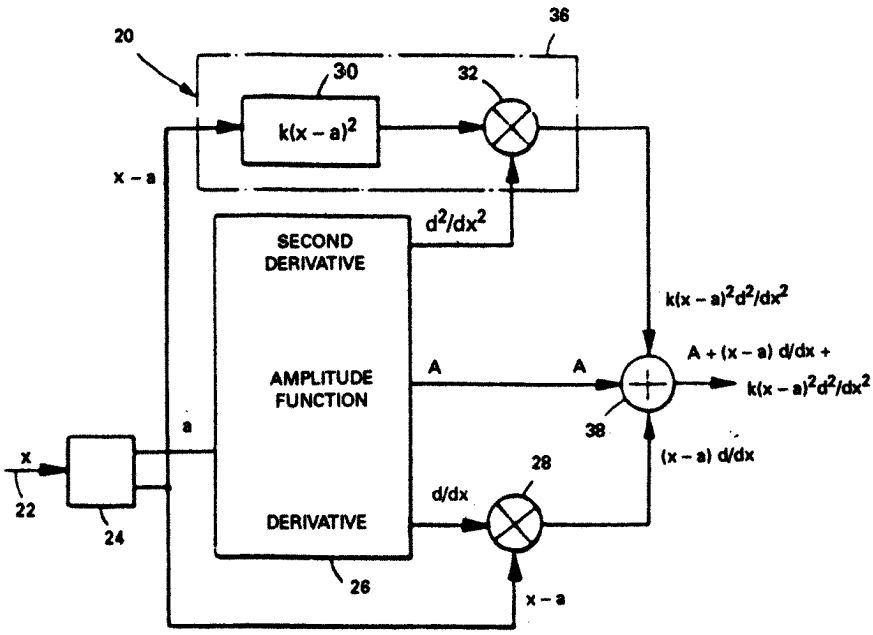


Figure 7-5 Taylor approximation—ROM compression. (*Courtesy of Qualcomm.*)

principles and various degrees of freedom available to the designer.

Insight can be gained by looking at some numbers. The steepest region of the sine function is around zero phase; however, this is also the region where the sine curvature (second derivative) is the lowest. As we progress into higher phase points, the steepness declines but the curvature increases. There is an element of compensation that does not indicate a special region to be more complicated to approximate than others.

Note, e.g., that from 0° to 30°, the sine changes from 0 to 0.5, that is, one-half of its peak value in only one-third of the peak phase. However, at 30° the error between a sine function and a straight line (which is $\pi/6$) is only 5 percent!

The second term of the Taylor approximation (a measure of the function's curvature) uses cos a as the multiplier. However, cos a changes very slowly near zero phase.

If we divide the 90° of the first quadrant into 32 equal segments, each of 2.8125°, the error in assuming cos a fixed across each of these regions for the first segment is given by $\cos(90n/32) -$

cos[90(n + 1)/32]. Thus in the first segment, for n = 0, if we use only one value for the first coefficient of the Taylor approximation, the error increases in the region from 0 to 0.0012. Or if the error is offset, it will develop from −0.0006 to 0.0006. So it can be seen that some elements of the sine function work for us. Near zero phase, the rate of change is the highest, but the waveform approximates nicely a straight line (low curvature) whereas near 90°, the rate of change is reduced to almost zero. This is demonstrated later in Fig. 7-8.

7-5 Hutchison Algorithm

This simple procedure and technique for compression was first suggested by Hutchison (Ref. 3). Suppose that we partitioned the first quadrant (from now on, we deal with only the first quadrant, since the other three can be derived easily from it) into a few (K_1) segments, so that K_1 is always a relatively low number (compared to W). Then it will be possible to present the sine by using two sub-ROMs and to save memory, since the first ROM will present the total ROM in fewer addresses than the original (we can refer to this ROM as the *coarse* ROM), and the second can use some form of linear interpolation for d cos a (see Fig. 7-6). We expect to achieve a level of compression since the sine is a smooth, "well-behaved" function.

In the case of Hutchison's algorithm, the fine ROM calculates the difference between the true value and the value of the sine at the K_1 points and does not allow any approximation; it is therefore a subset of the general approximation procedure.

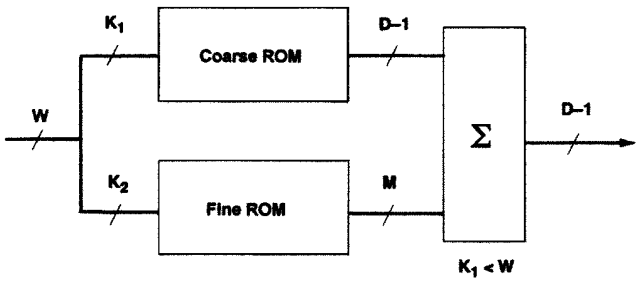


Figure 7-6 The Hutchison procedure.

Generally, the great gain in compression is possible because the sine is a smooth function and if it is segmented into small angular sectors, a linear approximation will suffice between the points, one that requires low dynamic range. This point must be emphasized because it is a cardinal tool for many designs and it can always squeeze a little more compression, as will be shown later.

It will be a worthwhile exercise for you to evaluate the errors in a few examples to get a feel for the level of error generated by the approximations.

Suppose $W = 10$ and $D = 8$. Suppose also that we divide the quadrant (90°) into 16 segments ($K_1 = 4$). Without compression, the complete $256 \cdot 7$ ROM (one quadrant) will be given by

$$S(i) = \text{int}\left(127 \sin \frac{2\pi i}{1024} + 0.5\right) \qquad \text{for } i = 0 \text{ to } 255 \quad (7\text{-}6)$$

The contents of this ROM are shown in Table 7-6.

First, let us consider the results if no errors are allowed. To continue with our example, suppose that $K_1 = 4$ and $K_2 = 8$, where K_2 is the number of bit addresses to the fine ROM. Note that even though we have not performed any approximation of the fine-tuned ROM, and we generated all 256 values, the output of the fine ROM needs only 4 bits of representation, since the highest value of this ROM (which represents the error between the sine function and the 16 points of approximation of the coarse ROM) is 12 (see address 15), and this value requires only 4 binary bits. Thus the introduction of the coarse ROM, which is relatively small, reduces the dynamic range of the fine ROM, and this is the major cause of the gain in compression. Later we will expand on this principle further. For address 16, or any address $16i$, the fine ROM value will be 0 again (the fine ROM has a sawtooth characteristic) since at this address the coarse ROM kicks in a new value, exactly the right value, so there is no need for any correction from the fine-tuned ROM. At all addresses that are multiples of 16, the fine-tuned ROM by definition will regress to 0; and since the error that needs to be corrected is less than 15, this fine ROM requires a ROM of size 256×4 only. This new ROM, given by the sum of the coarse and fine ROMs, as has been designed, generates exactly the values of the original one—without any

errors. Comprised of two sub-ROMs and an adder, the new ROM has a total size of 16(7) + 256(4) = 1136 bits, compared to the original one which contained 256(7) = 1792 bits. This represents a 37 percent reduction. See Table 7-6. The fine-tuned ROM is a linear function (quantized) whose slope decreases as the angle approaches 90°.

A compromise could be forged by allowing an error and using the same slope for every pair; i.e., use the same slope for the fine ROM for addresses 0 to 15 and 15 to 31, for 32 to 47 and 48 to 63, and so on. This will allow us to cut the number of input bits to the fine ROM by 1. More will be said about this later in the chapter.

This simple algorithm becomes more and more efficient as W and D increase. For example, for the very common case of $W = 14$ and $D = 12$, pick $K_1 = 8$ and $K_2 = 12$, instead of $2^{12}(11) = 45{,}056$ bits; a similar arrangement will call for a ROM size of $256(11) + 4096(4) = 19{,}200$, or a 57.4 percent reduction. Please note that K_1 and K_2 are design parameters that should be optimized for a requirement.

A general formula for the compression level, assuming that the fine ROM outputs only 4 bits, is given (for one quadrant) by the following:

Coarse ROM size: $$2^{W-4}(D-1) \tag{7-7}$$

Fine ROM size: $$2^{W}(4) \tag{7-8}$$

Compression level: $$\frac{2^{W-4}(D-1)+2^{W}(4)}{2^{W}(D-1)} = \frac{1}{16}+\frac{4}{D-1} \tag{7-9}$$

This simple procedure is referred to as the *Hutchison algorithm.* The algorithm saves approximately 50 percent (and more) of memory and does not introduce any errors.

As already mentioned, if errors are allowed, i.e., the fine ROM is allowed to correct to only within $\pm e$ of the correct value (in most cases $e = 1$ or 2), then better compression effectiveness can be achieved. The majority of the work done in designing compression algorithms followed this approach, i.e., tolerate some error to achieve great compression. The reason for this lies in the fact that most of the noise is introduced by the DAC anyhow, and when a

TABLE 7-6 Hutchison Values

i	$S(i)$	Coarse ROM	Fine ROM
0	0	0	0
1	1	0	1
2	2	0	2
3	2	0	2
4	3	0	3
5	4	0	4
6	5	0	5
7	5	0	5
8	6	0	6
9	7	0	7
10	8	0	8
11	9	0	9
12	9	0	9
13	A	0	A
14	B	0	B
15	C	0	C
16	C	C	0
17	D	C	1
⋮			
32	19	19	0
33	20	19	1
⋮			
240	3E	3E	0
241	3E	3E	0
242	3E	3E	0
243	3F	3E	1
244	3F	3E	1
245	3F	3E	1
246	3F	3E	1
247	3F	3E	1
248	3F	3E	1
249	3F	3E	1
250	3F	3E	1
251	3F	3E	1
252	3F	3E	1
253	3F	3E	1
254	3F	3E	1
255	3F	3E	1

design is completed, its spurious signals can be checked by an FFT procedure; and if the resulting errors are much lower than those known to be introduced by the DAC, the errors are allowed. In the following we present a few such algorithms, which allow some errors (mostly ± 1 LSB) but achieve very high levels of compression.

7-6 Sunderland Algorithm

Sunderland and coworkers suggested a modification to the Hutchison algorithm that achieves a substantial reduction in total memory size; see Ref. 1. Their basic reasoning behind the improvement is based on breaking the phase angle into three parts, thus $\sin x = \sin(a + b + c)$. Again (a) represents the coarse ROM, and bands b and c are two controls of the fine-tuned ROMs so that $a < 90°$, $b < 90 \cdot 2^{-a}$ and $c < 90 \cdot 2^{-(a+b)}$. This is shown in Fig. 7-7. Sunderland showed that the trigonometric identity can be written as

$$\sin(a + b + c) = \sin(a + b)\cos c + \cos a \cos b \sin c - \sin a \sin b \sin c \qquad (7\text{-}10)$$

which is approximately

$$\sin(a + b) + \cos a \sin c \qquad (7\text{-}11)$$

This suggests that the addressing of the coarse ROM is bits a and b and of the fine ROM bits a and c.

This trigonometric argument is hardly necessary. The Taylor approximation and a review of the data provide a heuristic approach. As W and D grow, we are allowed to divide the sine function into more and more segments, and so the interpolation between adjacent points becomes more and more linear and easy to manipulate. This is demonstrated in Fig. 7-8.

However, this approach can be modeled on a computer to show that if the coarse ROM has 8 input bit W_4 to W_{11}, then the fine-tuned ROM needs only 8 input bits W_0 to W_3 and W_8 to W_{11}, and an

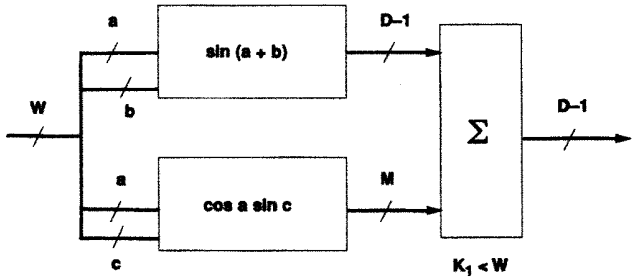


Figure 7-7 Sunderland procedure.

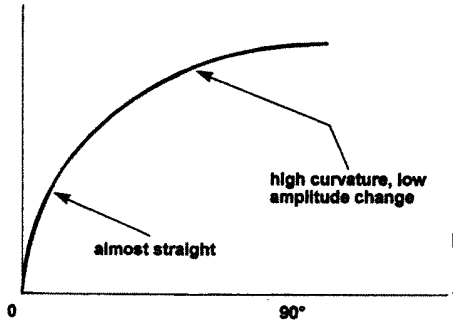


Figure 7-8 Sine characteristics.

approximation that yields an error of not more than ± 1 is achieved in the case of $W = 14$ and $D = 12$. Note that bits W_0 to W_3 are omitted from the coarse ROM and bits W_4 to W_7 from the fine ROM, with a minimal error. The block diagram is shown in Fig. 7-9. What happens is that the fine-tuned ROM will be updated only 16 times, and apparently this is enough. Cutting 4 address bits from the ROM reduces memory size substantially!

Another evident point is that such an algorithm becomes more powerful as W and D grow!

If we follow the Sunderland approach, he approximated Eq. (7-10) by (7-11), and the terms a, b, and c, indicate the addresses to the two ROMs. The execution of this function is shown in Fig. 7-10. A BASIC program simulating this procedure (ROMSUN.01) is included, and all 4096 values are given, on a disk the reader may

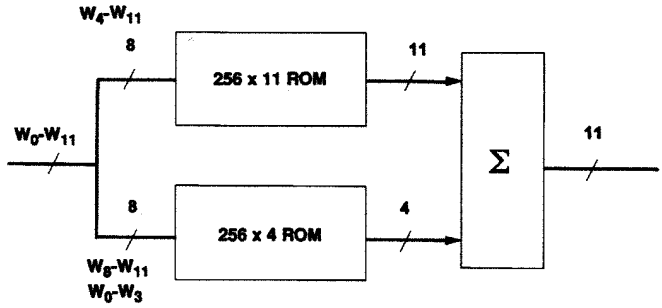


Figure 7-9 Sunderland procedure for 14 × 12 ROM, with one quadrant shown.

Mapping 14 phase bits to 12 amplitude bits using Sunderland algorithm:

$$\sin 1(i) = 2048 \cdot \sin [16i(360/2^{14})]$$

Values:

$S(0) = 0, S(1) = 12, S(2) = 25, S(3) = 37, S(4) = 50,$
$\ldots, S(253) = 2047, S(254) = 2047, S(255) = 2047$

$S(0) = 0°$ $S(255) = 90°$

Figure 7-10 ROM 1 used in the Sunderland algorithm.

obtain from the author (see Chap. 10). So for the fine ROM, 4 bits have been eliminated (W_4 to W_7) at almost no cost.

This is possible, of course, because the values of the Taylor coefficients change slowly, and what the algorithm does is to bunch groups of 16 together into a single value. This is why the fine ROM does not receive W_4 to W_7 as inputs at all with almost no penalty.

The size of the new ROM is now 256(11) + 256(4) = 256(15) = 3840 bits. Compared to the original ROM, a reduction of 91.5 percent has been achieved, or conversely a compression ratio of $2^{12}(11)/3840 = 11.33$ in storage requirements has been realized.

This memory size, though not trivial, can already be realized in the fastest ASIC logic families, at speeds up to 1000 MHz, or a memory access time less than 1 ns. It can be realized easily and cheaply by using low- and medium-speed (20- to 100-MHz) silicon CMOS logic. An FFT analysis of this algorithm yields spurious signal levels on the order of −85 dB. Note that this algorithm allowed the generation of some error, arbitrarily set (in the demonstration disk) to be not more than ±1.

It will be shown later that these errors are not significant in DDS applications since other sources in the circuit introduce much higher spurious signals anyhow.

A review of the operation of the fine ROM shows that a computer simulation can optimize the constants used in the linear interpolations. Since only the 4 MSBs W_8 to W_{11} control the slope of the approximation, their value can be optimized according to several criteria. The specific criterion in this application was to achieve minimum errors, of no more than ±1. Allowing larger errors enables shrinking of the memory size even further.

7-7 Variations and Randomization

As mentioned before, variations of the Sunderland architecture depend on the sizes of W and D and the design parameter criteria. An interesting variant is to reduce the size of the fine ROM by, say, one-half and approximate the second half as shown in Fig. 7-11. This means that the first half of a segment is calculated and the second half is added to it as a copy. If, e.g., the segment consists of the series

$$0, 1, 2, 2, 3, 4, 4, 5, 6, 7, 7, 8, 9, 10, 11, 12$$

then the new segment will be

$$12, 13, 14, 14, 15, 16, 16, 17, 18, 19, 19, 20, 21, 22, 23$$

A simulation of this procedure, ROMGG.01, is included on a disk the reader may obtain from the author (see Chap. 10).

Obviously the first half-segment is identical to the original, but the second is just a replica. The error cannot be kept to ± 1 anymore and errors up to ± 2 occur. The reason is very clear. All the ROM numbers are truncated, and so the fine ROM is not a true linear approximation (there is always a quantization error).

When approximating this line by two segments, we introduce another level of quantization and hence, from time to time, another bit of error. To improve this error, it has been suggested to

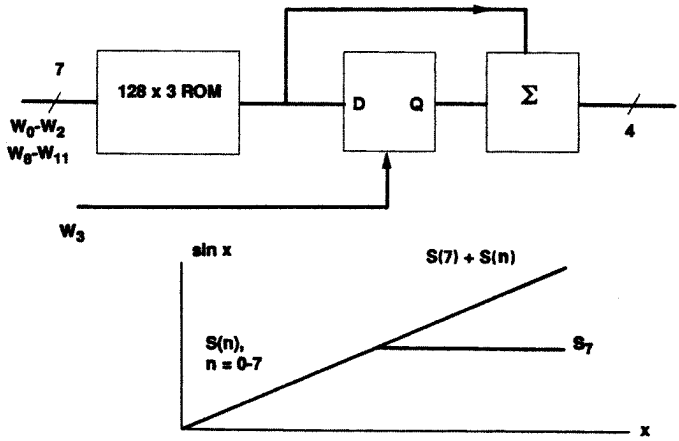


Figure 7-11 Reducing the size of fine ROM.

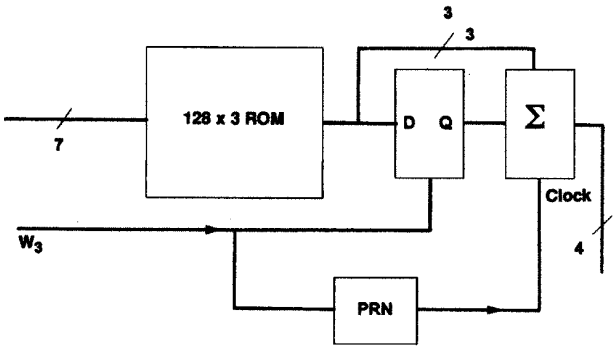


Figure 7-12 Randomizing fine ROM. PRN = pseudorandom numbers.

introduce a random number generator that affects only the LSB, as shown in Fig. 7-12. In this case, the new segment relative to the old one might look like this:

Old	0, 1, 2, 2, 3, 4, 4, 5, 5, 6, 7, 7, 8, 9, 9, 10
New	0, 1, 2, 2, 3, 4, 4, 5, 6, 7, 8, 8, 9, 10, 10, 11

which is much closer to the original.

The simulation and statistical errors achieved are presented in a program, ROMRAN.01, that is included on a disk the reader may obtain from the author (see Chap. 10).

The random number generator can be a simple pseudorandom sequence. As the simulation tables show, the randomization process now generates addresses with errors of up to ± 2 LSBs, and so the compression of the fine ROM by 2 : 1 cost in the increase of errors to 2 LSBs. The difference in the integrated mean square error has been simulated to be 2.2 dB, as shown in the simulation results. The error generated has a random (or pseudorandom) pattern; it is desired in some applications. The introduction of the randomization creates some increase in phase noise, but also distributes the spurious signal spectrum; see Wheatley (Chap. 4) and Kerr et al., Ref. 7.

Randomization is thus not all damaging, since it is also related to spurious signal reduction, as will be discussed here. In Ref. 7, a procedure is described to add randomization to a direct digital synthesizer by adding a random number generator output to the ROM out-

put. Since the random number is added to the output of the ROM, this does not affect the output frequency, which is solely determined by the accumulator, but does add dithering to the otherwise "clean" data. The outcome is quite similar to the randomization shown above. The idea here is that by adding the random numbers, the periodicity of the errors that cause the spurious signals is destroyed, and thus turns the spurious signal energy into noise.

In reality, the effectiveness of the randomization has been less than that expected from the theoretical reasoning, since a great part of the spurious signals is generated in the DAC via its nonlinearities and the intermodulation of its output with the clock signal leakage.

7-8 Auxiliary Function ROM Approximation

The Sunderland algorithm is a very effective one, and as demonstrated (using this algorithm), it achieves data compression of 11.7 in the case of mapping $W = 14$ to $D = 12$. However, one serious drawback of the algorithm is that the coarse ROM needs to produce $D - 1$ output bits. In the common case of $D = 12$, the coarse ROM is required to generate 11 outputs. This is quite unusual and inconvenient for memory devices, most of which produce binary, usually 1, 4, and 8 outputs. In standard CMOS designs and especially in a very high-speed design (where signal loading affects speed), it is extremely inconvenient to produce 11 outputs; usually 8 or 4 are desirable.

This can be solved in a simple way by breaking the course ROM into two ROMs, both having the same input but one, say, uses 8 outputs and the other 4, for a total of 11 outputs (or such numbers as required). However, the requirement to reduce the output bit count was a driver to yet another twist in the ROM compression efficiency.

Further insight reveals a further possible improvement in the ROM compression ratio. See Ref. 8.

The basic idea is not to map sin X, but rather to generate an auxiliary function $f(X)$, then approximate sin $X - f(X)$, and add them, getting sin X. This is a common technique in waveform compression. The cardinal driver of this approach is that even if $f(X)$ is mapped in very few addresses [and $f(X)$ is an approximation of sin X], then sin $X - f(X)$ has a much lower dynamic range than

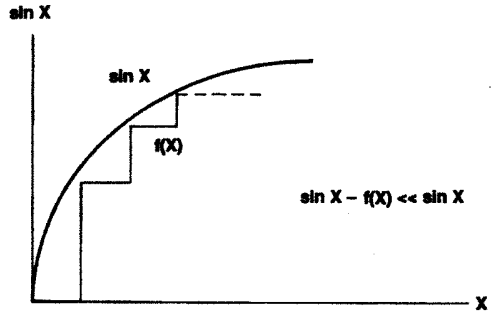


Figure 7-13 Auxillary function principle.

sin X. It therefore becomes possible to use ROMs whose output is 8 bits or less and achieve better compression ratios (Fig. 7-13).

In a way, it can be argued that the basic principle here is similar to that mentioned in the section describing the Hutchison algorithm, i.e., reducing the dynamic range of the required error correction signal (fine-tune ROM).

In the course of our investigation (which was performed mainly to reduce all output ROMs to less than 8 bits), the architecture achieved a better level of compression, relative to Sunderland. A tentative design is presented using three ROMs, as shown in Fig. 7-14 for the common case of $W = 14$ and $D = 12$.

A BASIC program, SINROM.03, is included on a disk the reader may obtain from the author (see Chap. 10).

Note that $f(X)$ is an approximation of sin X, but is not a unique function; and $f(X)$ can take many forms as long as it approximates

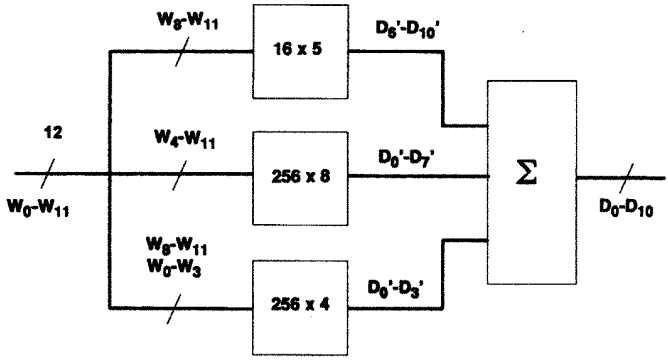


Figure 7-14 Auxilliary function mapping 14 × 12 ROM, one quadrant.

sin X at very few points (we have chosen 16) and with a very coarse level of approximation (in our example only 5 bits). However, this auxiliary function simplifies the design significantly and improves the compression efficiency. This is not claimed to be an optimal arrangement, but it is one that works with the constraint that all errors are not to be greater than ± 1. Many variations are possible here, and the example is for demonstration purposes only.

The total ROM size is now $16(5) + 256(8) + 256(4) = 3152$ bits. The compression ratio achieved is $2^{12} \cdot 11/3152 = 14.3$.

A few interesting observations should be noted:

- The coarse 4×5 ROM [the one that generates $f(X)$] can be executed by random logic rather than a memory device.
- The sum of the two smaller ROMs is always less than 255, so if an 8-bit adder is used as shown in Fig. 7-15, this adder never overflows and does not need a carry output function.
- Since ROM 1 is weighted $\times$ 64, only a 5-bit adder is necessary for the second adder and only two inputs are required from the lower bits adder.

Reference 9 mentions better compression ratios but does not present the details of the memory structure. The block diagram presented is quite complicated, and therefore it is difficult to follow all the details.

Obviously the above example was demonstrated for clarity, but the principle can be applied to any values of W and D; and it is nec-

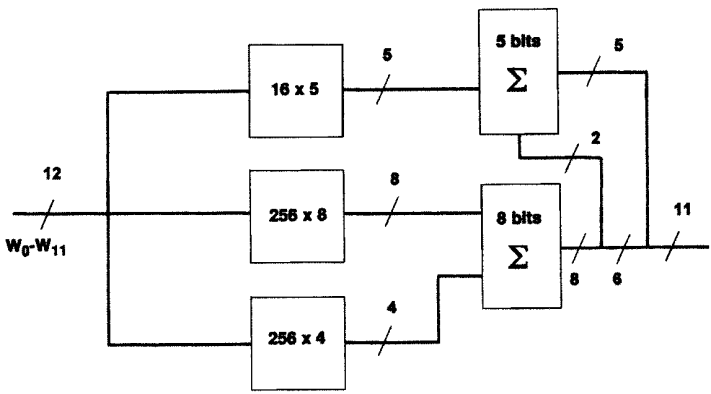


Figure 7-15 Adder arrangement for Fig. 7-13 is 14 $\times$ 12, one quadrant.

essary to follow design procedures and criteria for errors according to the specific requirements.

Another point is that the efficiency of the algorithm improves as W and D grow!

The total memory size in bits, for the compression of various W and D values, is as follows:

	W		
	12	13	14
$D = 12$	2176	2678	3152

The signal to mean square error for $W = 14$ to $D = 12$ is 70.4 dB and for $W = 12$ to $D = 12$ is 66.8 dB compared to the "ideals" (no errors) of 75.1 and 74.0 dB, respectively.

For the case of $W = 12$ and $D = 12$ (one that we designed), the ROM construction is given by 5×6 for coarse, 6×7 for the second, and 8×6 for fine, for a total ROM size of $192 + 448 + 1536 = 2176$ bits. A further reduction was achieved by applying an approximation to the fine ROM. Since it represents a linear quantized ramp, by breaking it into two ROMs, one 6×6 and the other 6×4 (very much like a Hutchison procedure), the total ROM size now becomes $192 + 448 + 640 = 1280$ bits. This approximation, which cut the size of the ROM by approximately another 40 percent, cost less than 2 dB in signal to mean error and has a worst-case spurious signal of -63 dBC.

As can be seen, these procedures have many degrees of freedom that need to be evaluated by the designer. This requires extensive computing and simulation; but if you use a 386 or 486 processor, a run takes but a few minutes. For those with a taste for this sort of work, it is fun and rewarding.

7-8-1 Spurious signal analysis

The performance of the compression can be analyzed by a computer, and the error distribution can be derived as follows:

- Reference 9 presented a vigorous analysis of ROM quantization.
- The simulation calculates only the number of errors and the sum of their squares.

- An FFT analysis is the only way we know of resolving the spurious signal levels.

Reference 9 reports spurious levels of −85 dBC, Sunderland reports −73 dBC, and our auxiliary method ($W = 14$ to $D = 12$) generates −75 dBC worst case.

Most designers in the field believe (from experience) that there is a correlation between the spurious signal levels and the sum of the errors (or the mean square error), but we have no mathematical proof of such a claim.

7-9 Coordinate Transformation (CORDIC)

There are other methods to produce the transformation of φ to sin φ, although the overwhelming use (up to date) in DDS applications is, as mentioned above, some type of memory device. A very innovative approach to this transformation was derived by Volder (see Ref. 5), and can be referred to as a *dynamic transfor - mation* rather than a lookup table.

This algorithm was designed for navigational computation and executes the following equations:

$$Y' = K(Y \cos z + X \sin z) \tag{7-12}$$

$$X' = K(X \cos z - Y \sin z) \tag{7-13}$$

and

$$R = K\sqrt{X^2 + Y^2} \tag{7-14}$$

$$\phi = \arctan \frac{Y}{X} \tag{7-15}$$

These are standard rotations and cartesian-to-polar transformations. (Indeed, this has become a very popular transformation technique, especially in video DSP applications and computer graphics, but it has not been applied much to DDS applications.) Volder termed it *co*ordinate *r*otation *di*gital *c*omputer (CORDIC), and this is the term used in the electronics and DSP industries.

The transformation was designed to go both ways, but we will concentrate on the rotation part since this is the transformation of interest in our application.

Volder required that the angular increments of rotation be computed in a decreasing order, that the first magnitude increment be 90°, and that the rest of the increments be according to the equation

$$a_i = \arctan 2^{-(i-2)} \tag{7-16}$$

This means that the phase shifts will be performed at binary divisions of 90° only.

Note that 180° is very simple to achieve since

$$\sin a_i = -\sin(a_i + 180) \tag{7-17}$$

$$\cos a_i = -\cos(a_i + 180) \tag{7-18}$$

Thus, a 180° shift implies only an inversion. A 90° shift is also simple since

$$\sin(a_i + 90) = \cos a_i \tag{7-19}$$

$$\cos(a_i + 90) = -\sin a_i \tag{7-20}$$

So, for the first two values of 180 and 90° shift, the transformation is simply

$$Y_1 = -Y_0 \quad \text{for } 180° \tag{7-21}$$

$$X_1 = -X_0 \tag{7-22}$$

and

$$Y_2 = \pm X_1 = R_1 \sin(\phi \pm 90) \quad \text{for } 90° \tag{7-23}$$

$$X_2 = \pm Y_1 = R_1 \cos(\phi \pm 90) \tag{7-24}$$

The rest of the calculations will be performed per phase increments of arctan $2^{-(i-2)}$, for $i > 1$, that is,

$$\arctan 1. = 45.° \qquad i = 2$$

$$\arctan 0.5 = 26.565\ ° \qquad i = 3$$

$$\arctan 0.25 = 14.036\ ° \qquad i = 4$$

and so on, up to the desired accuracy. In most DSP applications this will be carried out to 16-bit accuracy.

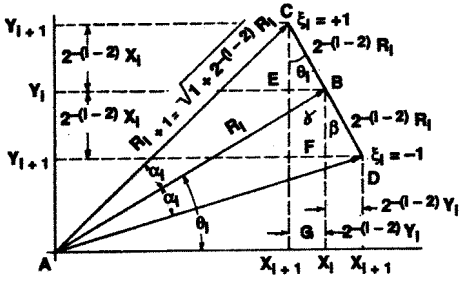


Figure 7-16 Geometry for CORDIC. Equations for CORDIC: $y' = K(y \cos \phi + x \sin \phi)$; $R = K\sqrt{X^2 + Y^2}$; $\phi = \arctan y/x$.

Volder noticed that if the rotations are restricted to these increments, these equations follow:

$$Y(i+1) = Y_i \pm X_i \cdot 2^{i-2} \tag{7-25}$$

$$X(i+1) = X_i \pm Y_i \cdot 2^{i-2} \tag{7-26}$$

This can be proven easily by viewing Fig. 7-16. Triangles *CEB* and *BFD* are equivalent, and so

$$EC = CB \cos \phi = 2^{i-2} R_i \cos \phi = 2^{i-2} X_i \tag{7-27}$$

$$EB = CB \sin \phi = 2^{i-2} R_i \sin \phi = 2^{i-2} Y_i \tag{7-28}$$

and the above equations follow. Note, now, that by restricting the angular rotation steps to Eq. (7-11), the right-hand term of Eqs. (7-12) may be obtained by two simultaneous shift and add operations. A block diagram is shown in Fig. 7-17.

So far, the algorithm produced a very simple procedure for rotation and generation of the quadrature components X and Y. Since any phase shift can be constructed (up to the desired resolution) by a combination of the phase increments, the mapping has now

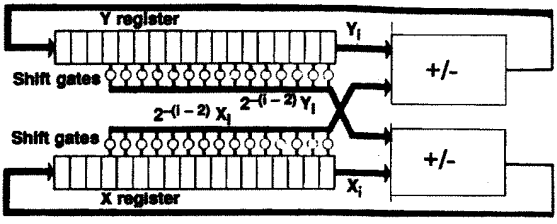


Figure 7-17 CORDIC block diagram.

TABLE 7-7 CORDIC *K* Value for *i* = 1, 10, 20

i	1	10	20
K	1.58113883008419	1.646759996375618	1.64657602581210816

become a simple procedure of shift and add. However, there is a problem.

The expressions $X(i + 1)$ and $Y(i + 1)$ are not perfect rotations because of the increase in amplitude by the term $\sqrt{1 + 2^{2(i-2)}}$; however, adding *or* subtracting a phase produces the same change in amplitude, so the amplitude distortion is the same for both phases.

In Ref. 5, Volder goes on to show the necessary corrections. For a given number of steps of rotation, say, n, the correction factor is given by

$$\sqrt{1 + 2^{-0}}\,\sqrt{1 + 2^{-2}}\,\sqrt{1 + 2^{-4}} \quad \sqrt{1 + 2^{-2(n-2)}} \qquad (7\text{-}29)$$

The values of the correction factor for $n = 1$ to 32 are shown in Table 7-7. The table shows that the correction value converges very fast, and for 16-bit accuracy, the first six terms achieve the necessary accuracy.

Our advice to practical designers is that it does not seem that the CORDIC algorithm is simpler to execute than the ROM implementation. However, CORDIC offers some unique advantages such as quadrature outputs (very simple to achieve in standard DDS designs, also shown in Sec. 4-8) and amplitude modulation (which requires the addition of a multiplier in standard DDS designs).

A device that produces the CORDIC operation has been produced as a monolithic IC by TRW, part number TMS-2340, and by Plessey, part number PDSP16630, also referred to as the *Pythagoras processor.*

7-10 Other Methods

There are other implementations of the transformation. For example, in the case of small ROMs, say, less than 8×8 bits, it is possible to replace the ROMs with logic. A computer software optimizes the Karnaugh map of all the states. It is also obvious that other implementations of the Taylor approximation, even if

it is desired to use three or four terms, can use memory rather than multipliers.

There is little doubt that better compression will be achieved, and in fact at least one case was reported (Ref. 9 reported compression ratio of 37:1 for $W = 15$, $D = 13$ with increased complexity of the arithmetic elements); however, its details have to be clarified. This design has been executed as an ASIC part by Analog Devices.

Some advanced theoretical work is required to bring the issue to an optimum solution per given error criteria, but progress so far has been remarkable. There is no closed-form solution to the relation between the ROM error and the spurious signal levels of the DDS output. It is a complex nonlinear problem that has not been solved yet.

Computing power offers a solution here, by brute-force FFT analysis. This means that a designer can evaluate the "cost" of a compression algorithm by running an FFT and calculating the spurious signal levels via computer simulation.

However, many who deal with the problem on a day-to-day basis indicate that there seems to be a direct relation between the total sum of errors and the spurious signal levels, and errors of more than ± 2 are not desired, as already indicated.

7-11 Conclusion

A variety of ROM compression algorithms have been presented along with the standard equations necessary to generate the ideal sine digital waveform for any input and output bit size.

In most cases, ROM compression involves a small sacrifice in theoretical performance; but since the main cause of DDS spurious signals is the DAC anyhow, it is understood that these errors are negligible in a real-world direct digital synthesizer.

The ROM programs for $W = 8$ and $D = 8$, $W = 10$ and $D = 8$, $W = 12$ and $D = 10$, $W = 14$ and $D = 12$, and $W = 16$ and $D = 12$ are included on the CDROM in the file ROM.W.D.

ROM compression improves its efficiency or its compression ratio with the increase of W and D, the input and output bit widths. Compression ratios of close to 15:1 are achieved and we expect that improvement will be made as more research is applied to this rather interesting aspect of DDS technology.

References

1. D. Sunderland, R. Strauch, S. Wharfield, H. Peterson, and C. Cole, "CMOS/SOS Frequency Synthesizer LSI Circuit for Spread Spectrum Communications," *IEEE Trans. Solid State Circuits,* SC-19, no. 4, 1984.
2. J. Gorsky-Popiel, *Frequency Synthesis and Applications*, IEEE Press, 1975.
3. B. H. Hutchison, Jr., *Frequency Synthesis and Applications,* IEEE Press, 1975.
4. L. Weaver and R. Kerr, High resolution phase to sine amplitude conversion, U.S. patent 4905177.
5. Jack E. Volder, "The CORDIC Trigonometric Computing Technique," *IRE Trans. on Electronic Computers,* September 1960, pp. 330–334.
6. D. H. Daggett, "Decimal-Binary Conversion in CORDIC," *IRE Trans. on Electronic Computers,* September 1960, pp. 335–339.
7. R. Kerr and L. Weaver, Pseudorandom dither for frequency synthesis noise, U.S. patent 4901265.
8. Henry Nicholas and H. Samueli, "An Analysis of the Output Spectrum of Direct Digital Synthesizers in the Presence of Phase Accumulator Truncation," 41st Annual Frequency Control Symposium, 1987, pp. 495–502.
9. Henry Samueli and H. Nicholas, "The Optimization of Direct Digital Frequency Synthesizer Performance in the Presence of Finite Word Length Effects," 42d Annual Frequency Control Symposium, 1988, pp. 357–363.
10. R. E. Lundgren and D. Snyder, "Designs and Architectures for EW/Communications DDS," R&D report, SLCET-TR-85-0424-1, U.S. Army Laboratory Command, Fort Monmouth, N.J., August 1986.
11. Albert L. Bramble, "Direct Digital Frequency Synthesis," *Proc. 35th Annual Frequency Control Symposium,* USERACOM, Fort Monmouth, N.J., May 1981, pp. 406–414.
12. Cercas Francisco et al., "Designing with Direct Digital Frequency Synthesizers," pp. 625–633.
13. J. Tierney et al., "A Digital Frequency Synthesizer," *IEEE Trans. on Audio and Electroacoustics,* March 1971, pp. 48–56.
14. Cercas Francisco, "DDS for a Frequency Hopped Spread Spectrum System," Master's thesis, Lisbon, 1988.
15. L. Jackson, U.S. patent 3735269, May 1973.
16. B. G. Goldberg, U.S. patent 5321642, June 1994.

Chapter

8

Digital-to-Analog Converters

The *digital-to-analog converter* (*DAC*) is the link that connects the digital signal generated, modulated, and manipulated in the direct digital synthesizer, and converts it into its analog form, the "real" signal. Naturally it is a device that contains and shares both digital and analog technologies. It is where digital and analog connect and merge to generate an analog signal.

There are a great variety of DACs for different applications. Most DACs have been developed for industrial, consumer, and video applications. Only recently (less than 12 years) did the direct digital synthesizer and the field of signal generation attract the attention of DAC manufacturers. The DAC function is very critical in all digital signal processing (DSP) applications, and a great variety of products are available, fitted to specific applications.

We concentrate here on the architectures that are specific to direct digital synthesis (DDS) applications and demonstrate the performance of a few state-of-the-art devices.

In this era of specialization, most manufacturers design and produce DACs for specific markets. For example, specific DACs are produced for computer graphics and CAD applications; there the package usually consists of three DACs for the three colors (red, green, and blue). Others produce products for industrial applications where speed and density are less of a concern, but cost is an overwhelming consideration. DAC technology has evolved and matured in the last 20 years and has reached an impressive level of performance. From 16- to 24-bit DACs for use

in CD players to very high-speed (and ultrahigh-speed) 8- to 12-bit DACs, mainly for high-resolution graphics, arbitrary waveform generators, and direct digital synthesizers, and the race is on. For DDS applications the DAC performance has not yet reached the expected performance, and in most existing designs, the DAC is the "Achilles heel" of the direct digital synthesizer. Our focus will be on high-speed, high-linearity, and high-accuracy DACs, since these are the three parameters that have the greatest effect on DDS performance.

8-1 DAC Performance Evaluation

There are two basic ways to evaluate the performance of a DDS design. One way is to specify performance of the digital part only, and the other way is to specify the analog output, including the DAC. In our opinion, the second way is the one that presents the whole picture. Since speed, phase noise, resolution, and modulation achieve excellent performance in DDS compared to any other synthesis technique, the only major problem still facing the DDS designer is the spurious signal response.

In a DSP technique like DDS, the nature of the quantization errors and their periodicity are a source of spurious signal generation. Since in most cases the digital part is separated from the DAC, it is very convenient for many manufacturers to define the spurious signal performance of the digital part. This is a dangerous and "deceiving" way of defining the spurious signals for the part they supply, because in most cases, these specifications have very little relevance to true performance. The problem is that the manufacturers present a digital calculated number that is correct, but cannot be executed because of deficiencies of the DAC.

The spurious signal result, of the digital signals produced before the connection to the DAC, is a calculated number. The digital data are presented and analyzed by a computer, via some discrete Fourier transform (DFT) or fast Fourier transform (FFT) procedure. All quantization errors are simulated, and an accurate performance level is calculated. It is a fair way of specifying the device as long as the user understands what the specification means. What it means is that if an ideal DAC existed, having infinite speed and accuracy and linearity, the specification would be met in the analog domain.

Based on 10-bit designs (the ROM maps 12 input bits to 10), typical spurious signal levels up to 65 to 70 dB can be achieved (worst case is −60 dB). If 12-bit designs are used (the ROM maps 14 input bits to 12 output bits) up to 75 to 80 dB (typical) can be achieved.

However, the second method of defining the spurious signal performance, the one that includes the DAC, is used by those of us who have to deliver a complete working part, one that generates a true analog sine wave. In these cases, the DAC rarely exceeds 65-dB performance. Being an analog device, the DAC suffers from finite linearity, finite speed, and finite accuracy. Moreover, the device usually generates its own artifacts, including overshoots, transitional spikes, and "glitch" products, all contributing to its limited performance.

As a rule of thumb, under very extensive testing and long experience, the worst-case spurious signal performance is given by approximately $-6N$, where N is the number of bits in the DAC.

A very important part of the DAC performance depends on its circuit layout. It is hard to overemphasize the criticality of the circuit layout, paying attention to signal path and radiation, proper analog and digital grounding (the more the better), and proper loading.

One key to a successful DDS design lies in a proper grounding, ground returns, isolation of the analog and digital grounds so that the digital ground does not interfere with the generation of the analog signal, and bypassing.

A few DAC manufacturers who have paid enough attention to this problem design the DAC in such a way that all digital inputs are physically separated from the analog part of the circuit. This helps create a solid ground plane for the device. Almost all DDS DACs have separate power supply and ground pins for the analog and digital parts of the integrated circuit (IC).

To understand the difficulty for the DAC, it is enough to remember that a direct digital synthesizer generates many frequencies, the main and many more aliasing frequencies, and is a sample device rich with clock energy. It is therefore natural to expect a level of intermodulation between these signals. In Chap. 4 we demonstrated the problem associated with measuring DDS performance because of saturation of the front end of the spectrum analyzer. The problem for the DAC is similar—how to linearly deal with a multitude of signals, many of which have a compara-

ble power level, without introducing too much intermodulation and nonlinear effects. Although the DAC is a sampling device (similar to a mixer), third-order and higher-order intermodulation products are never specified for it. Maybe it is a parameter that requires specification from the manufacturer. The measurement is very simple. Simply connect to the input a digital word which represents the sum of two closely related sine frequencies (sin $\omega_1 t$ + sin $\omega_2 t$), and measure the power at $2\omega_1 - \omega_2$ and $2\omega_2 - \omega_1$. Such a measurement is very simple to make in a direct digital synthesizer when the frequency is controlled to $F_{ck}/2 - d$. This way both $F_{ck}/2 - d$ and $F_{ck}/2 + d$ are being generated.

There are a few frequencies whose generation enables a fast evaluation of the DDS DAC performance. These frequencies do not always present the worst-case spurious signal generation, but they are close to worst case and give an excellent indication of the quality of the direct digital synthesizer.

One such frequency is at an output frequency that is close to $F_{ck}/3$, another is close to $F_{ck}/4$, and yet another is at 0.375 F_{ck}. To achieve a frequency output of $F_{ck}/3$ in a binary direct digital synthesizer, the necessary control is 0101010101 . However, we recommend generating a frequency that is a little lower (or higher) than exactly $F_{ck}/3$. The output will show both the desired output frequency and the intermodulation of the second harmonic with the clock. Since the DAC is an analog device, the clock signal always exists at the output port, and this signal intermodulates with the harmonics of the signal. A typical spectrum is shown in App. 4B. For example, for F_{ck} = 20 MHz and a control of 0101010100000 , the output frequency is given by

$$F_{out} = 20(0.25)(1 + \tfrac{1}{4} + \tfrac{1}{16} + \tfrac{1}{64}) = 6.640625 \text{ MHz} \qquad (8\text{-}1)$$

This is quite close to $F_{ck}/3$ = 6.666 MHz. The intermodulation signals will show at $\pm(\frac{20}{3} - 6.640625)$ MHz around the output frequency.

This is one very fast and very indicative measurement for the DAC linearity. For another simple and fast measurement, set the direct digital synthesizer to a relatively low output frequency, say, $F_{ck}/16$, and measure the harmonics (see App. 4B). This measurement will usually produce many results, because the 16 points at the output of the DAC do not have the same accuracy, and therefore different "constellations" will produce different results. Many

DDS users are surprised to dial a frequency, measure spurious signals, go to another frequency, come back to the first, and measure a different level of spurious signal. But this is exactly what one should expect, for going back to the original frequency will likely generate another set of DAC points. The same results will show if one runs a "theoretical," either calculated or the FFT of the ROM output, of, say, of 16 points. The quantization is not the same for all possible "16 constellations." The effect is sometimes referred to as *bubbling* spurs by DDS designers.

Another indicative frequency is the control 0110000 , for this generates an output given by $0.375\ F_{ck}$, but as indicated in Chap. 4, it has a periodicity of 8 clocks and generates all harmonics of $0.125F_{ck}$ (see App. 4B). This command word is a good indicator of the quality of the DAC grounding as well as its linearity.

In the following we present a review of DAC technology and topology, since this is crucial for a solid DDS design.

8-2 DAC Principles of Operation

The DAC receives as an input a digital word that represents a sample point of the digital signal, and the DAC produces an analog output, voltage or current, approximately equivalent to the value of the digital word. The output of the DAC is thus

$$I_{\text{out}} = \frac{I_{\text{ref}}W}{2^D} \qquad 0 < W < 2^D - 1 \tag{8-2}$$

where I_{out} is the output current, W is the instantaneous control word, D is the number of bits in the DAC, and I_{ref} is the maximum current output.

The analog resolution of the DAC is given by $I_{\text{ref}}/2^D$. In most high-quality DACs, the digital input is latched or strobed by the clock (STROBE, or CONVERT) input command. Many DACs require a differential clock input, sometimes referred to as *convert* and *convert-not*.

For DDS applications, where high speed, high linearity, and accuracy are the major requirements, almost all DACs are constructed by a set of accurate current sources that are added to or disconnected from (according to the input word) a common node, the output port. And DACs generate the output current directly related to the input word W and to a given reference I_{ref}, usually

given by 40 mA, to a load of 25 or 37 Ω. Since the current sources have high output impedance, the output node has high output impedance and requires a resistor (sometime embedded in the device, of 50 or 75 Ω), to set up a source impedance, and the same external load impedance.

A reference input voltage controls the peak-to-peak output current, usually set up to be 40 mA, or the peak-to-peak output voltage of 1 V (40 mA across 25 Ω). (The standard 1-V signal has been derived from video systems.) In many devices this reference voltage can be modulated to achieve an AM function. Attenuation of up to 40 dB can be achieved very accurately.

Since the current sources produce the current continuously, those sources not in use are steered to another, complementary port, and thus most DDS DACs generate I_{out} and $I_{out\text{-}not}$ ports. These outputs are equal but opposite in phase.

There are two main digital interfaces used in DACs. The most popular is binary (or inverse binary), and the other that often is available is the 2s complement. If two interfaces are available in a device, a set of control digital inputs determines the DAC mode of operation. Binary interface is a natural; however, since DACs are usually DSP parts and since 2s complement is a most common digital interface for representation of positive and negative numbers, many manufacturers provide both formats and save an additional interface for users who operate in the 2s complement format.

Most DDS applications, but not all, use straight binary interface. A general block diagram and specifically a direct digital synthesizer DAC block diagram is shown in Fig. 8-1. As seen in the figure, many high-quality DACs use segmentation of a few most significant bits (MSBs), usually 4 to 6 bits, in such a way that the dynamic range between the various current sources is minimized. As will be demonstrated here, the segmentation improves accuracy and therefore also production yield.

The principle of segmentation can be demonstrated as follows: In a 10-bit DAC, suppose that the MSB current source has a current weight of 20 mA. Then the LSB has a current weight of 20/512, which yields approximately 40 μA.

To keep the accuracy of the DAC, the MSB current source has to be accurate to at least 0.001; otherwise this error will mask the contribution of the LSB. Also, when one is generating (or discon-

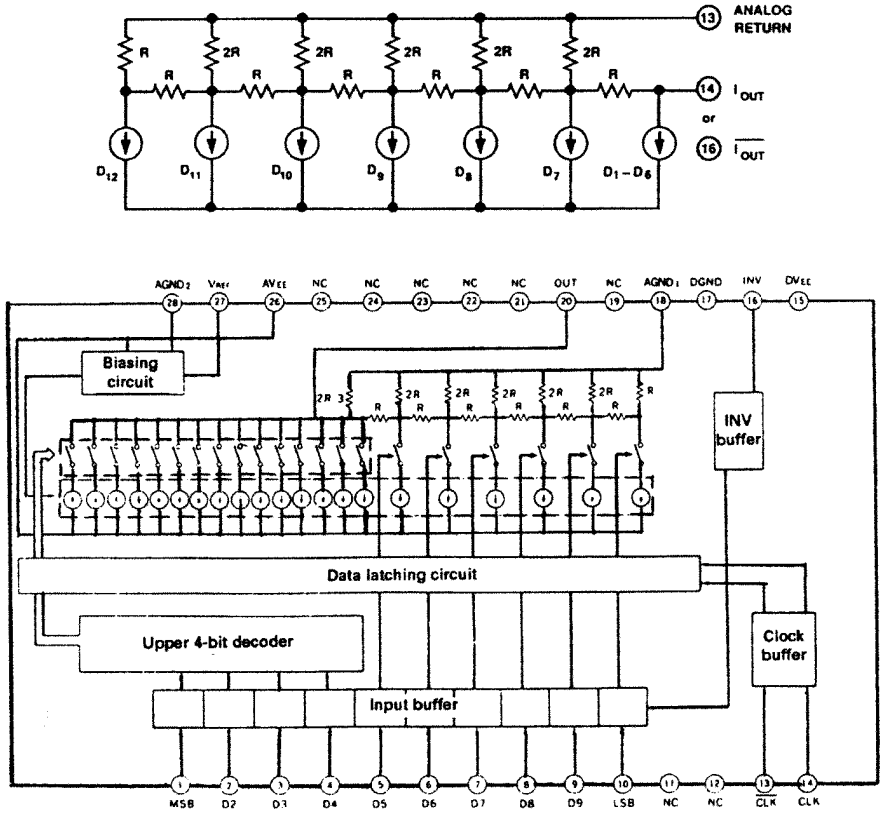


Figure 8-1 DAC general diagram and segmentation.

necting) the MSB current source, a 20-mA surge current switches in the circuit in the presence of other current sources that are much lower in magnitude. This is obviously a source of current surges and accuracy problems.

By segmenting the MSBs, part of the problem is attenuated. Suppose that we return to the above example, but we assume that the 4 MSBs have been segmented. The segmentation replaces four current sources with binary weights by 15 current sources that are equal. The largest current source in the circuit is now 1.25 mA. To generate the MSB, 8 of them will have to be connected in parallel, 4 for the second MSB, and 2 of the third MSBs. The decoding of the segments is performed in the segmentation logic control

decoder and is clearly a simple logic function. However, for this added complexity we gain very important advantages:

- The highest current source in the circuit is now only one-sixteenth of what it was before; therefore a relaxation in accuracy of an order of magnitude can be tolerated. This helps achieve better performance and produces much better production yields.
- Although the error of each 1.25-mA current source is within a given tolerance, the errors are quite random. Thus when more than one current source is added to create a higher-order bit, the errors tend to average (according to law of the large numbers).
- There is a reduction in the level of the surge currents.

These arguments are very important from the hardware standpoint, and almost all DDS DACs use this architecture to achieve better performance.

Another important parameter is the relative timing of the switching of the current sources. Ideally, all switches should open or close simultaneously. In practice, this is not possible because of limitations in the device speed, rise and fall times, and layout. To minimize this effect, all high-performing DDS DACs must use a strobe (or latch or clock) command that aligns the timing. The difference between a latched and nonlatched DAC can cost easily 10 dB in spurious signal performance.

Insight into the causes of the DAC errors can be gained by looking at the response of a DAC when changing states. A time-domain plot is shown in Fig. 8-2, where (*a*) is the ideal waveform in transition from state S_1 to S_2 and (*b*) shows the transient detail. In the transition the signal demonstrates finite rise time; overshoot; "glitch," which is a spurious signal spike related to the current surges; rise and fall time asymmetry; and ringing. In most cases, the transient is corrupted almost linearly in reference to the size of the step $S_2 - S_1$. Obviously, the transients of the MSBs and especially the MSB are most problematic, since this current source carries the most current. Many DAC specifications show a time-domain transient for the change from 1000 to 01111, because that amount of change is only 1 LSB, but all current sources change state, and this is usually a sensitive operating point.

Such a plot, as shown in Fig. 8-3, can be modeled and approximately describes the settling time of the DAC waveform. As is

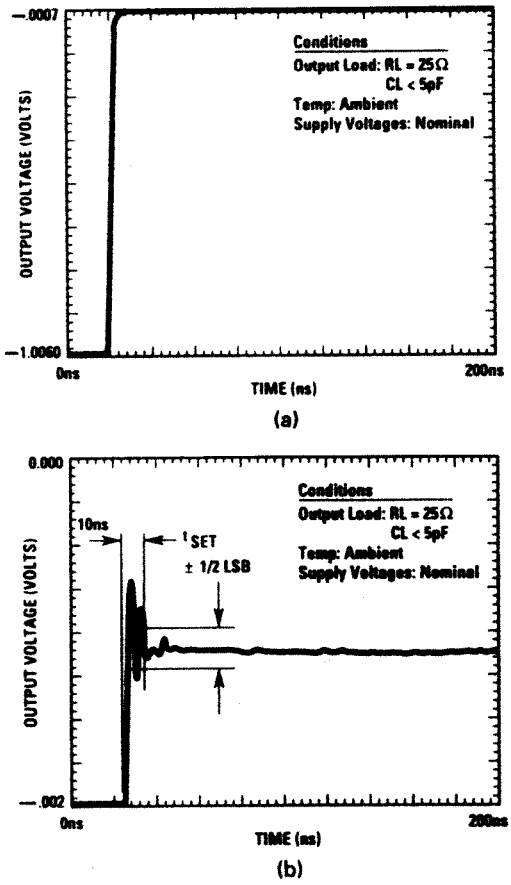


Figure 8-2 Time-domain plot of DAC, where (*a*) is ideal waveform in transition from S_1 to S_2 and (*b*) is the transient detail.

shown, the rise and fall settling times can be approximated by *RC,* with exponential convergence slope (sometimes with additional ringing), and full accuracy (settling to within $\pm\frac{1}{2}$ LSB) is achieved only after some finite time. Clearly if the DAC is commanded to change to a new state before it fully settles to the old one, accuracy will be lost. Since the slope changes depending on the value of the difference between the transitional states, the error is not uniform. Many DAC manufacturers specify their devices in an ambiguous way, e.g., specifying the device speed at

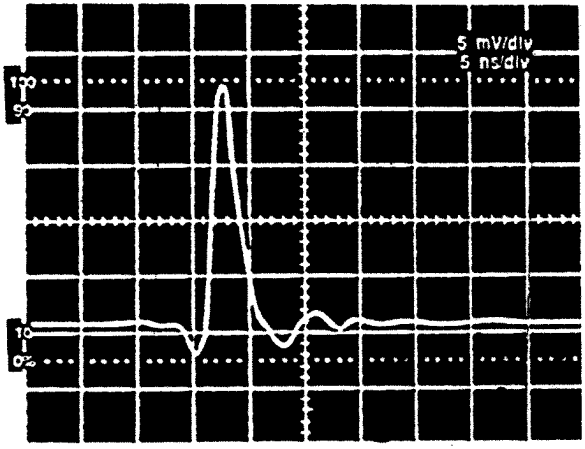


Figure 8-3 Glitch model and measurement.

50 MHz (that means that a new value can be updated every 20 ns) but specifying settling to ± ½LSB accuracy in 30 ns! Such an operation risks an update that is faster than the settling time. In such applications where $S_2 - S_1$ is small, e.g., when one is generating low frequencies, the operation might not suffer excessive errors. But otherwise when the DAC needs to perform large amplitude changes (when high frequencies are generated) from sample to sample, it will change states without ever settling! The error can be described as a loss in DAC output bits of resolution. If the DAC settles only to within ± 2 LSBs, this means that 2 bits of resolution were lost; and instead of using, say, a 10-bit DAC, the operation will be equivalent to using an 8-bit DAC, with the equivalent loss in performance. In such cases, it is strongly recommended to use a faster 8-bit DAC rather than a slower 10-bit device.

Another source of error is the asymmetry of the rise and fall times. This can be demonstrated easily for the generation of a square wave. An ideal DAC would generate a perfectly symmetric wave with no second (or any even) harmonic. But an asymmetry in rise and fall times creates a second harmonic.

One method to cure some of these problems is to put a transformer between I_{out} and $I_{out\text{-}not}$. Thus the symmetry improves, and some of the glitch signal cancels because of the common-mode rejection of the transformer. The results on this are not conclusive,

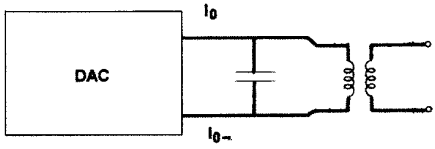


Figure 8-4 DAC output using trafo to balance the two complementary current sources, in order to reduce spurious signals.

but we have tried this in many designs, where this method can improve spurious signal performance by 2 to 6 dB (see Fig. 8-4).

From time to time, the use of a sample-and-hold device complements the DAC operation. A fast sample-and-hold device can follow the DAC and sample its output after all the transients have settled, as shown in Fig. 8-5. The major problem here is that fast sample-and-hold (S/H) devices are complex, have linearity problems, and are expensive. The use of S/H devices cleans the signal and usually reduces the transient spurious signals (which usually produce wideband spurious signals), but obviously will not affect or improve the DAC linearity or intermodulation products. The overall improvements are therefore important only in applications where wideband spurious signals are a major issue. On the other hand, in applications where narrowband signals around the carrier are the main concern (such as embedding the direct digital synthesizer inside PLL designs, where the loop cleans the DDS output anyhow), the S/H is redundant. For a performance demonstration, see App. 4B.

It must be stressed again and again that the DAC is still the major contributor of errors in DDS designs, and therefore major attention must be paid to its selection, circuit layout, by-passing, and grounding. The digital part has already achieved a high level of integration and accuracy, but the DAC performance must be

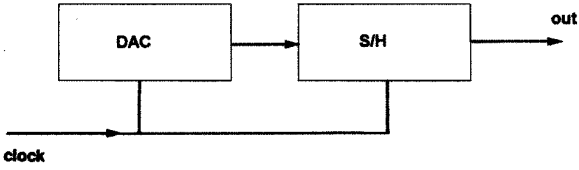


Figure 8-5 DAC followed by a sample-and-hold (S/H) device.

further improved. As of this date, there is no computer simulation program or tool that can simulate these transients for the different states of the DAC, and the quality of the DACs has to be evaluated by testing them in the circuit. Section 8-4 presents recommendations for what we experimentally found to be the best-performing DACs for low-, medium-, and high-speed operation and reviews their architectures.

8-3 DAC Parameters

The DAC is a complex integrated circuit (or hybrid), includes digital and analog circuitry, and requires excellent speed, linearity, and accuracy. DAC users have their own terminology, and the specification of typical DACs will be reviewed, since there is a fair amount of confusion regarding the importance and meaning of different parameters. The following is a description of DAC parameters and their definitions.

8-3-1 Update rate

Usually given in megahertz (or kilohertz) is the speed at which the DAC input can be changed. This parameter should also imply that at this rate, the analog waveforms have to completely settle. For example, if 500 MHz is specified as the update rate, the user assumes that the analog output settles at less than 2 ns. This parameter has been abused by many manufacturers who specify speed, say, at 50 MHz (20-ns update rate) but a settling time of 35 ns! These are conflicting specifications; therefore, always design in the more conservative number.

8-3-2 Resolution

The smallest analog increment corresponding to 1 LSB of the converter is the *resolution.* This number corresponds directly to the number of digital input control bits. Therefore, this term is usually given as *8-bit* or *12-bit* DAC, and it is understood to be the reciprocal of $2D$, where D is the width of the input word. The larger this number is, the better the accuracy and approximation to the ideal analog value. CD players use 14- to 16-bit DACs since excellent signal reproduction fidelity is required. In DDS applications, the most common D values are 8, 10, and 12.

8-3-3 Logic format

As mentioned above, the format usually is either binary (or inverse binary) or 2s complement (or inverse 2s complement). This defines the relation of the output analog voltage and the format of the digital input control.

A general equation for binary format, also referred to as *unipolar binary,* is

$$V_{\text{out}} = V_{\text{f.s.}} \sum_{1}^{n} \frac{A_n}{2^n}$$

where $V_{\text{f.s.}}$ is full-scale voltage, n is the number of bits, and A_n is the input control word (A_n is 0 or 1). The 2s complement is given by

$$V_{\text{out}} = V_{\text{f.s.}} \sum_{2}^{n} \frac{A_n}{2^n - A_1}$$

For a comparison we demonstrate the maximum, minimum and midpoint for both formats for a 12-bit representation:

	Binary	2s complement
Minimum	000000000000	100000000000
Midpoint	100000000000	0000000000000
Maximum	111111111111	0111111111111

8-3-4 Setup-and-hold time

Define the setup-and-hold time of the input register.

8-3-5 Rise, fall, and settling times

Rise and fall times are the 10 percent to 90 percent switching times required for a full-scale swing (from state 000… to 111…), but they have little value for DDS applications. Since the output of the DAC can be approximated by

$$V_{\text{out}} = V\left[1 - e\left(-\frac{t}{\tau}\right)\right] \qquad \tau \text{ is rise/fall time constant}$$

we will be much more interested in settling to within 1 LSB. In the case of a 10-bit DAC, the settling is from 0 to 0.9990; or in the case

of an 8-bit DAC, it will be from 0 to 0.9975. This is a parameter that will interest DDS designers, and they should compare this number to the update rate. These two numbers should be reciprocals. That means that a 5.6-ns device should not be specified (or operated) to operate at update rates of more than 1/5.6 ns = 178 MHz.

8-3-6 Propagation delay time

The *propagation delay time* is the time between the rise time of the clock input and the time that the DAC output changed by 1 LSB.

8-3-7 Differential linearity

This parameter measures and specifies the worst-case deviation from the ideal change of 1 LSB for an up or down ramp that goes through all the DAC states. For example, a DAC that produces a 1.5 LSB error has a 0.5 LSB differential nonlinearity. An error of 1 LSB or more can cause a nonmonotonic transfer function. This term is expressed either as fractions of the LSB or percentage of full scale.

The dc measurement of differential nonlinearity is a tedious but simple procedure. However, the dynamic measurement is practically impossible at speeds above 10 MHz. The reason is that this is a time-domain measurement, and no oscilloscope has that kind of measurement resolution, especially to measure 10- or 12-bit accuracy at tens or hundreds of megahertz of dynamic testing.

Most digital sampling scopes use analog-to-digital converters in their front end, and their dynamic range and accuracy are worse than those of the measured device. Often such a dynamic measurement introduces the tester's artifacts, too.

However, in dc measurements, many DACs achieve better accuracy than resolution; i.e., a 12-bit device achieves 14-bit error. There is no conflict in terms here. This is an ideal case for the designer.

8-3-8 Integral nonlinearity

This parameter measures the worst-case deviation from the ideal straight line connecting the two endpoints of zero to full scale. It can be expressed in LSBs or as a percentage of full scale.

8-3-9 Monotonicity

This specification ensures that the DAC is monotonic for increased (or decreased) input code and is an indicator of the linearity of the device.

8-3-10 Multiplying bandwidth

Many DACs allow one to change the value of the reference and produce an output that is linearly related to the change. The linearity of this process and its bandwidth are often specified. This is a parameter that is not commonly used in DDS applications, but it can be very accurate and economical as a part of the amplitude control section of a design. The bandwidth specifies the frequency at which the DAC gain is -3 dB relative to its gain without modulation.

8-3-11 Glitch energy

This parameter specifies the energy of the integrated transient spurious signal waveform, and it is defined as the difference between an ideal straight line and the real waveform (see Fig. 8-3). This is done in the time domain and measured in volt-seconds or rather picovolt-seconds. It is a complicated measurement to make or verify, and most high-quality DACs quote 10 to 30 pV s.

We could not correlate this specification with the DDS performance, and we have doubts about the accuracy of the measurement of this parameter. Usually DACs that specify more than 100 pV · s do not perform well in DDS circuits; but some of those that specify 20 or 30 pV s also exhibit rather poor performance.

Since it is always a dynamic measurement, it depends on the accuracy of the test equipment, as mentioned in Sec. 8-3-7. Most high-quality DACs use balance circuitry; some provide complementary clock inputs, clock and clock-not; and most also provide complementary output I_0 and $I_{0\text{-not}}$.

The loading of the two output ports affects the performance and usually requires the same termination; i.e., if the main output is terminated by 25 Ω, the complementary output is also terminated by a 25-Ω resistor to ground.

There are cases—and it depends on the device being used—where a balanced output, using the two outputs through either a

transformer or a balanced amplifier, improves the performance of the device. This was mentioned above. A schematic configuration of high-speed DAC is shown in Fig. 8-6 (see also Fig. 8-4).

8-3-12 Symmetry

The difference between the rise time and fall time symmetry affects linearity and therefore spurious signal performance. For example, a perfect square wave will have no even harmonics, but one with a different (final) rise and fall times will show even harmonics. This will translate to spurious signal response.

8-4 State-of-the-Art DACs

In the following we present a few DACs that perform well at low, medium, and high speeds. These recommendations are not the only devices to use. In fact, a few designers will probably argue about the choice, but this is the best we can do.

8-4-1 Low-speed operation

For low-speed operation, below 20 MHz, the TRW TDC 1012 is an excellent part. It is a 12-bit fully registered DAC, and it exhibits excellent linearity and accuracy. The device settles to 10-bit accuracy in 12 ns and to 0.5 LSB in 35 ns maximum. Harmonic performance is as good as -65 dB, and spurious signal output is as good as 65 to 70 dB.

The device is convenient to use in TTL circuits since it has TTL input levels (the TDC 1112 is the same device with ECL inputs) and interfaces directly to TTL or CMOS logic.

The input setup time is on the order of 25 ns, and from experience we recommend using up to 30 to 35 ns of data valid before strobing. The peak glitch area is specified at 20 pV s typically, and the differential linearity is ± 0.012 percent. As mentioned, the device is recommended for operation up to 20 MHz. The 1112 can be run up to 40 MHz.

The general structure is a segmentation of the first 6 MSBs (63 current sources, each at the value of the MSB weight divided by 32), followed by 64-LSB current sources.

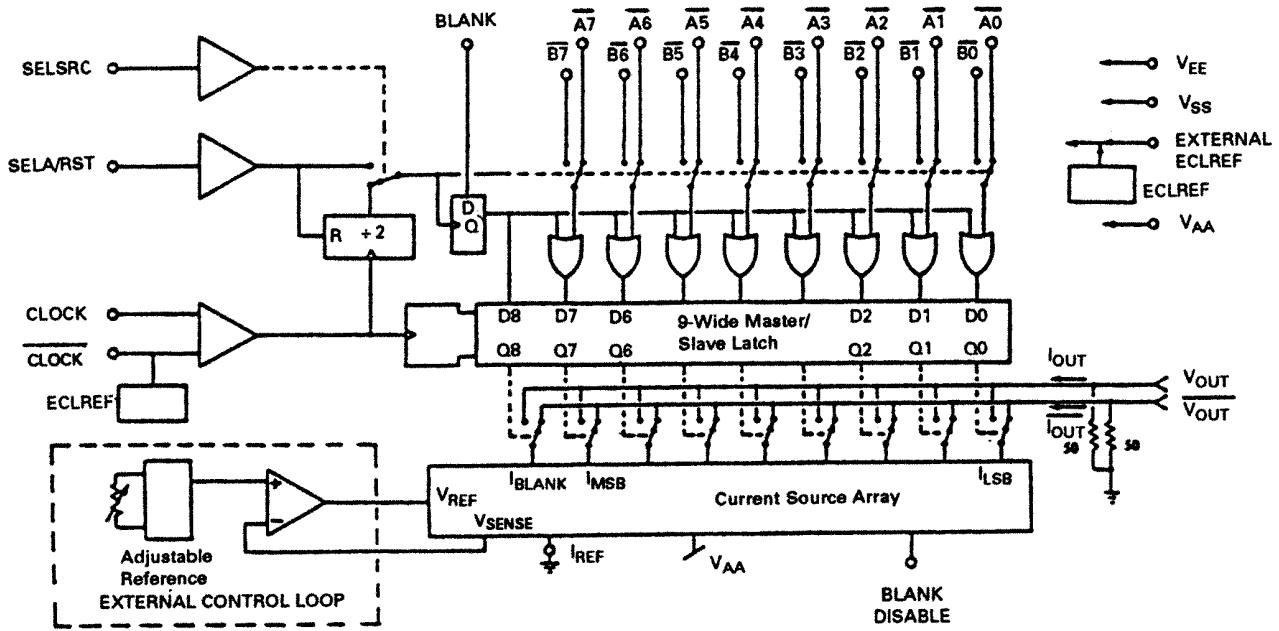


Figure 8-6 1000-MHz 8-bit DAC (TQS). (*Courtesy of TQS.*)

8-4-2 Medium-speed operation

For operation at medium speeds, around 100 MHz, Analog Devices' AD9760 is an excellent choice. It is a 10-bit device from AD's series of high performance, low power CMOS DACs and supports update rates up to 125 MSPS. The T×DAC family consists of pin compatible 8-, 10-, 12- and 14-bit DACs specifically optimized for the transmit signal path of communication systems. All of the devices share the same interface options, small outline package and pinout, providing an upward or downward component selection path based on performance, resolution and cost. The AD9760 is manufactured on an advanced CMOS process, using a segmented current source architecture combined with a proprietary switching technique to reduce spurious components and enhance dynamic performance. Edge-triggered input latches and a 1.2 V temperature compensated bandgap reference have been integrated to provide a complete monolithic DAC solution.

Another part is the Tektronix TKDA30,which exhibits 12 bits at 100 MHz, has 1/2-LSB differential and integral linearity, and has a settling time of 3 ns. This device has to be used at frequencies substantially above the recommended.

8-4-3 Very high-speed operation

For those applications demanding a few hundred megahertz operation, we recommend the Tri-Quint GaAs DAC model TQ6122 (8 bits). The DAC is an 8-bit part and exhibits excellent linearity up to 500-MHz speed. At 1000-MHz speed it exhibits the performance of a 7-bit device. Settling time is on the order of 2 ns to settle to 0.4 percent, or 1 LSB.

The device uses ECL level interface and has a differential linearity of $\pm$ 0.5 LSB. Glitch energy is specified to be below 10 pV s.

A block diagram is shown in Fig. 8-6. Note that the device uses only a single power supply (previous GaAs DACs and S/H devices used two and up to four power supplies). Power dissipation is 1.5 W nominal (previous GaAs DACs used up to 3.5 W). A 12-bit version is also available.

8-4-4 Other recommended DACs

- Analog Devices model AD9751 is a new high-speed 10-bit DAC in CMOS, with support of update rates up to 300 MSPS. Power dissipation is specified at less than 300 mW.
- SPT (Signal Processing Technology) model SPT5310, an ECL high-speed device, 250 MHz, 8 bits.
- Sony model CXA 1236Q, an ECL high-speed device, with multiplexer to support two logic inputs, in excess of 500 MHz, 8 bits.
- Fujitsu model MB40788, an ECL high-speed device, in excess of 100 MHz, 10 bits, and differential linearity of ± 0.2 percent.
- TRW model TDC1018, an ECL high-speed device, in excess of 100 MHz, 8 bits.

8-5 Sine-Wave DAC

No such device exists yet. After all, DAC designers want their devices to be as generic as possible. However, the level of specialization is already so high today that such a device, one that includes the ROM decoding in the DAC, should be contemplated. This integrated ROM/DAC takes as an input the output of the accumulator, or in other words, for an input ramp of 000 to 111 it produces a complete analog sine-wave cycle.

Such a device will serve the DDS market alone, but this market is already sizable and growing fast. Whether it is a simple integration of the ROM and the DAC, or some simplification that might be possible, such a device will be very desirable and welcome to the DDS market.

8-6 Multiplexing

It is possible to gain direct digital synthesizer logic speed and therefore bandwidth through multiplexing. Historically, the speed of DACs has always been substantially higher than that of logic circuits. There are therefore many applications that use DACs whose operational speed exceeds that of the logic circuits by 2 to 5 times and even more.

If done correctly, an architecture can be devised in such a way that some logic circuits will run in parallel, at a lower speed, and the digital data will be multiplexed at the input of the DAC. The fact is that a few DAC manufacturers have recognized this logic speed limitation and have included multiplexers in the front-end inputs of the DAC; see Sony model CXA1236Q, Tri-Quint model TQ6112, and some others.

As an example, if we run two direct digital synthesizer logic circuits in parallel and multiplex them sequentially, we can double the logic speed and therefore the bandwidth. A tentative scheme is shown in Figs. 8-7, 8-8, and 8-9 (see also Ref. 5). Obviously, the two DDSs will have to be synchronized so that the outcome produces the desired waveform.

Since each of the DDSs will generate only one-half of the waveform, both have to fit perfectly as they sequence. The general principle (for N logic paths) is shown in Fig. 8-7. One direct digital synthesizer is generating the even samples, and the other generates the odd. If we use N DDS logic circuits in parallel, the operation will be similar, but now each one generates only one Nth of the waveform. This requires the following operations:

1. The first direct digital synthesizer is loaded to a predetermined state, say 0. The second is then programmed to D, which is the frequency increment. The third one is loaded to $2D$, etc., and the Nth one to $(N - 1)D$.
2. Then the direct digital synthesizers are commanded to increment ND, see Ref. 5.

This way, the first direct digital synthesizer produces addresses 0, ND, $2ND$, $3ND$; the second direct digital synthesizer produces D, $(N + 1)D$, $(2N + 1)D$, ; the third generates $2D$, $(N + 2)D$, $(2N + 2)D$, ; and the Nth generates $(N - 1)D$, $(2N - 1)D$, $(3N - 1)D$, .

Since the multiplexer is synchronous with all the direct digital synthesizers, it will sample the outputs of synthesizers 1 to N sequentially and will produce, according to the above data, a sequence of 0, D, $2D$, $3D$, , $(N - 1)D$, ND, $(N + 1)D$, which is exactly the desired output.

The result is N direct digital synthesizers, operating in parallel, each one running at the clock rate but multiplexing at N times the clock rate and generating a signal of N times the bandwidth of

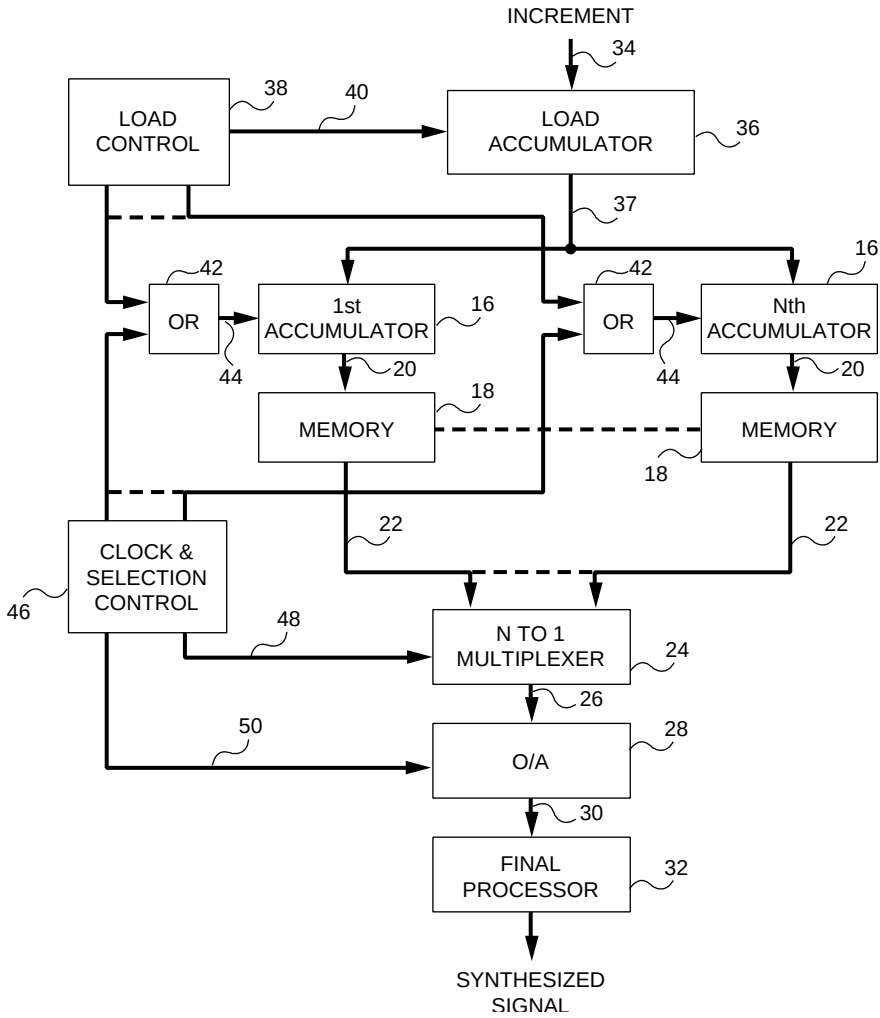


Figure 8-7 Multiplex logic, single-DAC direct digital synthesizer. (Courtesy of Sciteq.)

each direct digital synthesizer. The timing and loading of the individual direct digital synthesizer and the multiplexer are, of course, very critical.

This could be done by running N identical direct digital synthesizer circuits, generating analog outputs, and then multiplexing in the analog domain (Fig. 8-8); this is not advised. The reason is that it will be impossible to create N DACs that are exactly the same,

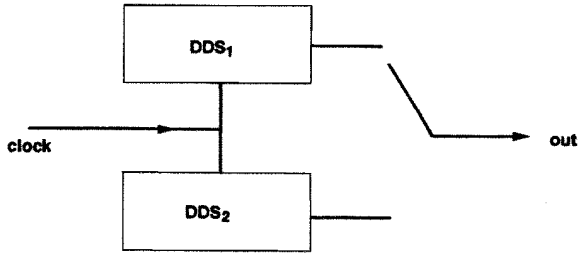


Figure 8-8 Multiplexed direct digital synthesizer, in the analog domain.

and the analog voltage discrepancies between the N DACs will degrade the performance badly.

An alternative scheme is to multiplex only the digital part, as mentioned above. This method is described in Ref. 5. Here all the parallel processing is done in the digital domain, and only one DAC is used; thus the analog errors between the DACs described above do not exist. The DAC being the critical part is the same for all paths, and only the digital inputs are multiplexed.

The penalty is in the requirement to use a DAC that is N times faster than the logic. This is not necessarily a problem since traditionally DACs operate much faster than the logic (accumulator speed is quite comparable to that of DACs but memory is still substantially slower). Most commercially available digital DDS chips operate between 20 and 60 MHz while good-quality 10-bit DACs operate above 200 MHz.

The above scheme can be executed in a different way, yielding the same results, as shown in Fig. 8-9. In this scheme, only one accumulator is used, and the rest are adders. In the beginning, the first stage, the accumulator, is loaded by an arbitrary number, say 0. The rest of the adders add D sequentially. So the output of the first adder is D, the second $2D$, etc. When the accumulator is incremented by ND, the adders follow, and the net result is the same as shown above.

The advantage of using this scheme is that all values are basically generated in a single accumulator, and the timing control is somehow easier to implement. However, there is some lack of commonality; one stage (the first) is different from the others. On the other hand, since all arithmetic is derived from this single first

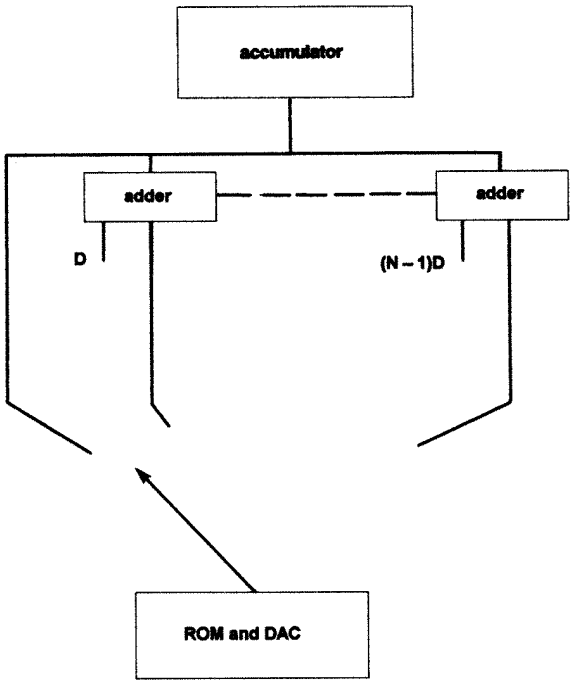


Figure 8-9 Multiplexed direct digital synthesizer.

stage, parameters such as phase continuity will be easier to achieve and control.

The cost of multiplexing generally is a factor of the amount of hardware used and tight timing control. However, the benefits are substantial and should be considered when necessary. Very high-speed DDS designs do exist already with very good performance; however, cost is always a consideration, and for some applications, multiplexing makes sense (see Sec. 4-9).

References

1. Analogic-Data conversion system digest, 1983.
2. Analog Devices data sheets for AD9760 and AD9751 (1999).
3. Sony data sheet for SX2020A.
4. TRW data sheet for TDC 1012 and 1018.
5. B. G. Goldberg, Digital frequency synthesizer having multiple processing paths, U.S. patent 4,958,310.

This is a blank page.

Chapter

9

Synthesizers in Use and Reference Generators

9-1 Synthesizers in Use

This chapter presents a short review of reference generators (mainly crystal oscillators and atomic standards) and a set of synthesizers, either instruments or just original equipment manufacturer (OEM) products whose designs are related to the subjects of this book and which are exceptional and durable. Most synthesizers described here have been on the market for a long time and have been very successful and innovative in their designs. The description of every design will demonstrate its principles of operation, assess its advantages, and evaluate its performance.

We concentrate mostly on designs which include either direct digital synthesis (DDS) or fractional-*N* synthesis. There are other fine synthesizers (mainly the PLL type), mostly with older designs, which did not take advantage of emerging technologies, and they will not be mentioned here, even though they constitute the majority of designs.

However, we feel that a technological change eventually will sweep this field, and in the near future a great many instruments and designs will use these new technologies.

Note that almost all designs are combinations of more than one synthesis technique. Hybridization of PLL and DDS or DA and DDS or fractional-*N* is so advantageous that it seems to be the dominant wave of current designs as well as of new designs likely to appear in the next 5 years.

All these instruments have been successful and therefore have been steadily improved and "fine-tuned" by their manufacturers to better performance and reduce cost and size.

9-1-1 Hewlett-Packard 3325B

This instrument, now on the market for more than 17 years, is a synthesizer that generates direct current to 21 MHz with 1-μHz resolution; and it is a second-generation model following the very successful HP3325A (Fig. 9-1). It is a fractional-*N* PLL circuit that uses 30 MHz as a crystal reference, generates a 30- to 51-MHz PLL signal, and by mixing the 30- to 51-MHz with the fixed 30-MHz generates direct current to 21 MHz. At the heart of the 3325B, there is a 30- to 51-MHz fractional-*N* PLL circuit using a 0.1-MHz reference.

The division ratio is therefore 300-510, and the step size is 1 μHz. Since the ratio of the step size to the reference is given by 0.000001/100000 = 0.00000000001, the fractionality is 10^{-11}. Spurious signal performance is specified at −70 dBC, and the phase noise 1 kHz from the carrier is approximately −110 dBC/Hz.

The HP3325B is a remarkably simple and excellent product. It is one of the first and most successful fractional-*N* synthesis designs.

9-1-1-1 General specifications

Sine output: 1 μHz to 20.999999999 MHz

Square wave: 1 μHz to 10.999999999 MHz

Triangle wave: 1 μHz to 10.999999999 kHz

Step size 1 mHz above 100 kHz and 1 μHz below 100 kHz

Phase noise: better than − 60 dB for integrated noise of ±15 kHz bandwidth, excluding ±1 Hz about the carrier

Phase shift: ±719.9°, 0.1° resolution

Allows phase modulation, AM, and frequency sweep

Functions such as square wave and triangle wave available, and FM sweep for which start, stop, and sweep times are controlled from front panel or via HP interface bus (GPIB)

Figure 9-1 HP3325B. (*Courtesy of Hewlett-Packard.*)

9-1-2 Hewlett-Packard 8662A

This instrument is one of the highest-quality synthesizers on the market and has been successful for 15 years. It generates 0.1- to 1280-MHz signals, with very complex multiloop design. The kernel is a 320- to 640-MHz synthesizer, using a fractional-*N* design for the high-resolution section. The close-in phase noise is achieved by locking a master oscillator to multiples of 10 MHz, thus achieving close-in phase noise of the crystal (which is multiplied and cleaned by crystals and SAW filters) and exceptional additive performance from the high-quality switched resonator VCOs and resolution from the fractional-*N* PLL. Frequency resolution is 0.1 Hz (0.2 Hz above 640 MHz). This instrument is used in the automatic phase noise measurement system produced by Hewlett-Packard.

9-1-3 Program Test Sources 310

The Program Test Sources (PTS) PTS310 model is one of the most remarkable synthesizer products; it is part of a PTS family of direct analog and digital synthesizers covering 0.1 to 1000 MHz. The PTS310 covers 1 to 310 MHz and combines direct analog and DDS technology. This family of synthesizers appeared on the market some 25 years ago, and it continues to evolve by expanding capabilities, bandwidth, and continuous refinements.

A photograph and frequency planning and block diagram are shown in Fig. 9-2. The latter consists of the following major blocks:

- Reference generator
- Direct digital synthesizer
- Up converter
- Drift-cancel loop and output stage

All frequencies are derived from a 5- to 10-MHz standard.

The direct digital synthesizer is clocked by a 16-MHz signal and comprises a BCD direct digital synthesizer using 4 bits of binary for the 4 MSBs. This circuit generates 4 to 6 MHz with 0.1- or 1-Hz step (and includes digital phase shifting) and is converted up to the 26- to 28-MHz range.

The reference generators then generate a 10-MHz signal in the 140- to 150-MHz range, and the rest of the bandwidth is achieved through the drift-cancel loop that converts the 140 to 150 MHz to 365 to 375 MHz and then converts back down to 1 to 310 MHz.

The switching speed is 2 μs for a 100-kHz step and has a maximum specification of 20 μs for any frequency excursion. Phase noise is excellent, approximately −110 at 1 kHz and −125 at 10 kHz.

The PTS family has been on the market for almost 20 years, with gradual improvements and refinements. These included mechanical changes, packaging, introduction of the direct digital synthesizer approximately 10 years ago, with excellent reliability and featuring extremely competitive performance and pricing.

The product started as a pure direct analog design based on the principles indicated in Sec. 1-4-2. In the last few years, PTS designers incorporated mechanical changes, bandwidth expansion, and the introduction of the direct digital synthesizer and save hardware without compromising performance.

The original designers made a point to design in low-power-dissipating parts and no fans (to avoid mechanical vibrations). Power supply voltages of + 5 and −12 V are used for all functions.

9-1-3-1 General specifications

Frequency range: 0.1 to 309.9999999 MHz

Step size: 0.1 Hz

Spurious signal: −70 dBC

Phase modulation: 0.36° step (this is BCD design)

Switching speed: 20 μs. Switching within 100 kHz is phase-continuous

Power consumption: less than 40 W

Size: 3.5-in rack-mounted chassis

9-1-4 Comstron/Aeroflex FS-2000

This high-frequency direct instrument is the Cadillac of direct analog synthesizers. The different models cover up to 1 MHz to 18 GHz, with excellent phase noise performance and 1 μs maximum switching time. New models of the FS-5000 recently introduced offer 0.2-μs switching time.

A block diagram and photograph are shown in Fig. 9-3.

9-1-4-1 Major specifications

Frequency range: 1–18,000 MHz

Resolution: 1 Hz

Speed: 0.10-, 0.2-, and 1-μs options!

Spurious signals: −80 dB below 2.4 GHz.

Phase noise at 1 GHz: −118 dBC/Hz at 10 kHz from carrier

9-1-5 Schomandl models ND500 and ND1000

Manufactured in Germany, these instruments cover 500 and 1000 MHz, respectively, similar to PTS in performance. (See Fig. 9-4.) They use direct analog and DDS technologies and have excellent phase noise and fast switching speed (see PTS specifications).

9-1-6 Stanford Research DS345

This instrument covers the range from 1 μHz to 30.2 MHz. It uses a 40-MHz clocked binary direct digital synthesizer, using 48 bits. The basic direct digital synthesizer covers 15 MHz, and the rest of the bandwidth is achieved by doubling and filtering. The software control allows BCD use since the resolution of the instrument is $80 \times 10^6/2^{48} = 0.14$ μHz, the error being on the order of

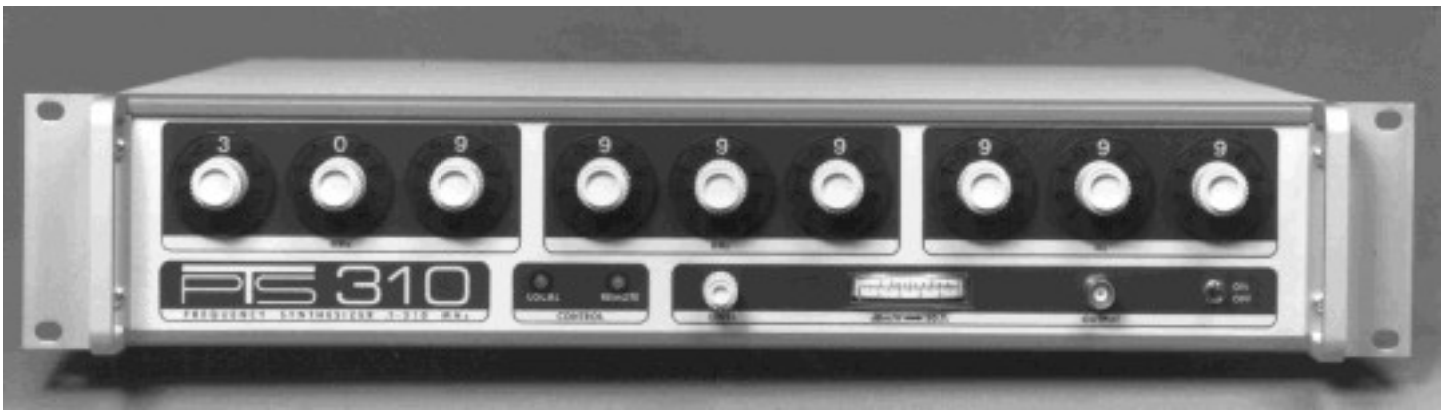


Figure 9-2 PTS310 and block diagram. (*Courtesy of Program Test Sources.*)

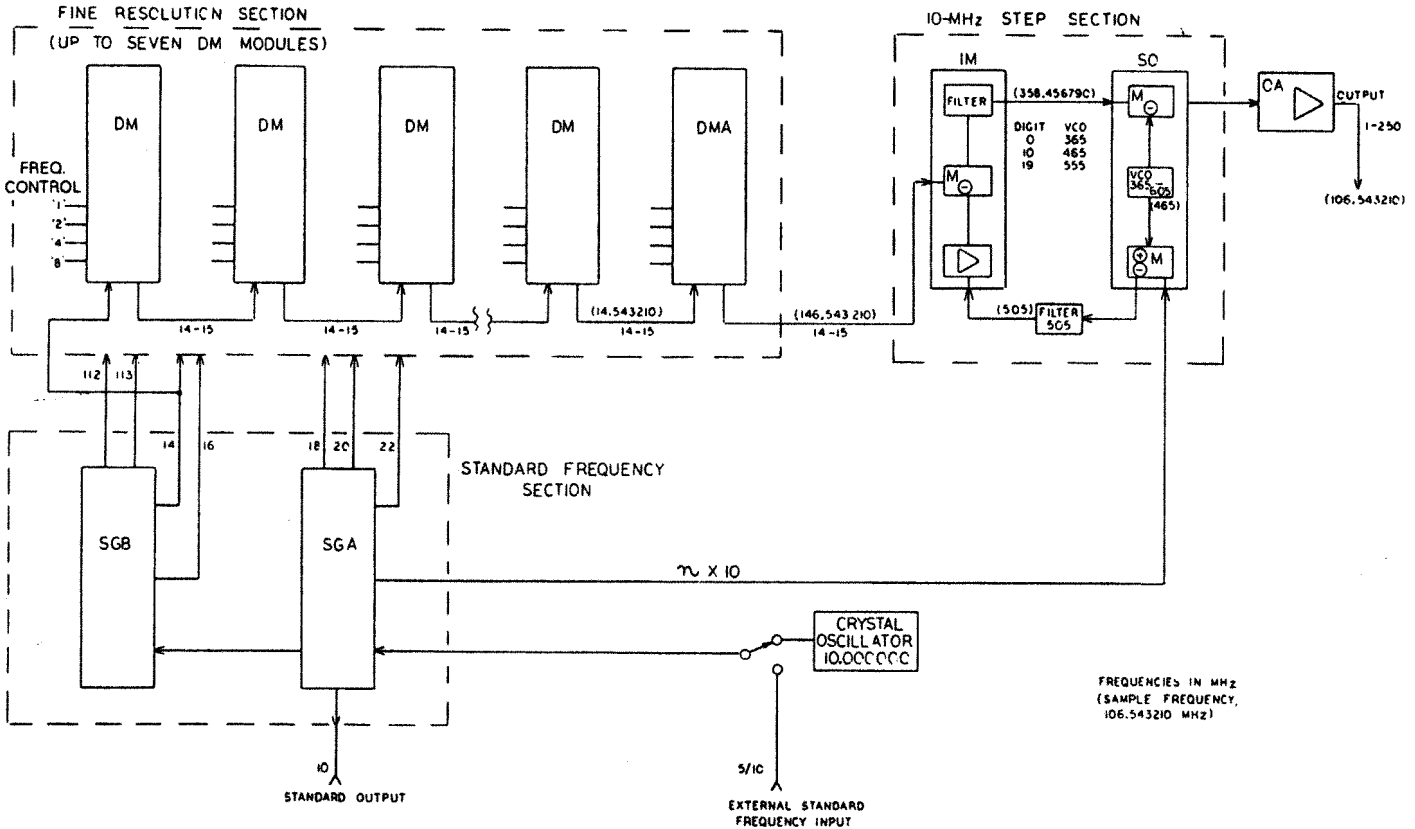


Figure 9-2 *(Continued)*

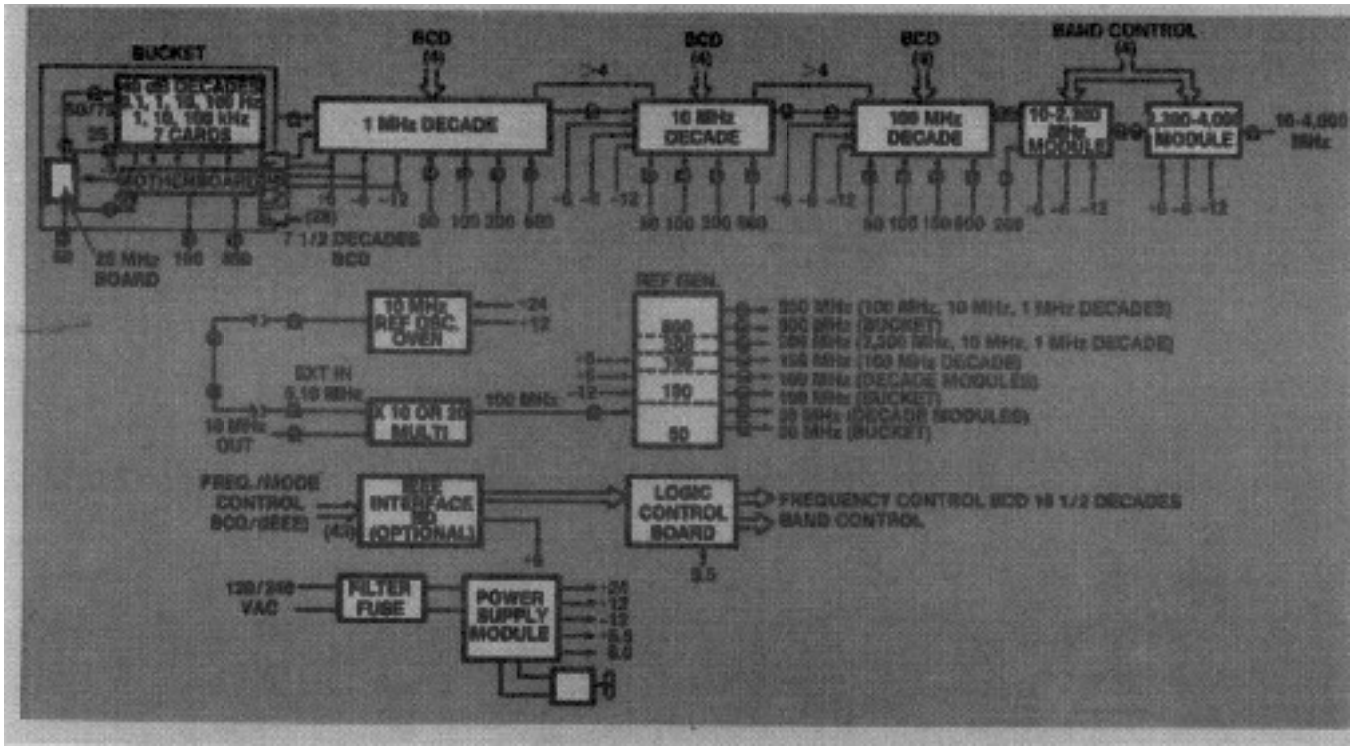


Figure 9-3 FS-2000 diagram and instrument. (*Courtesy of Comstron.*)

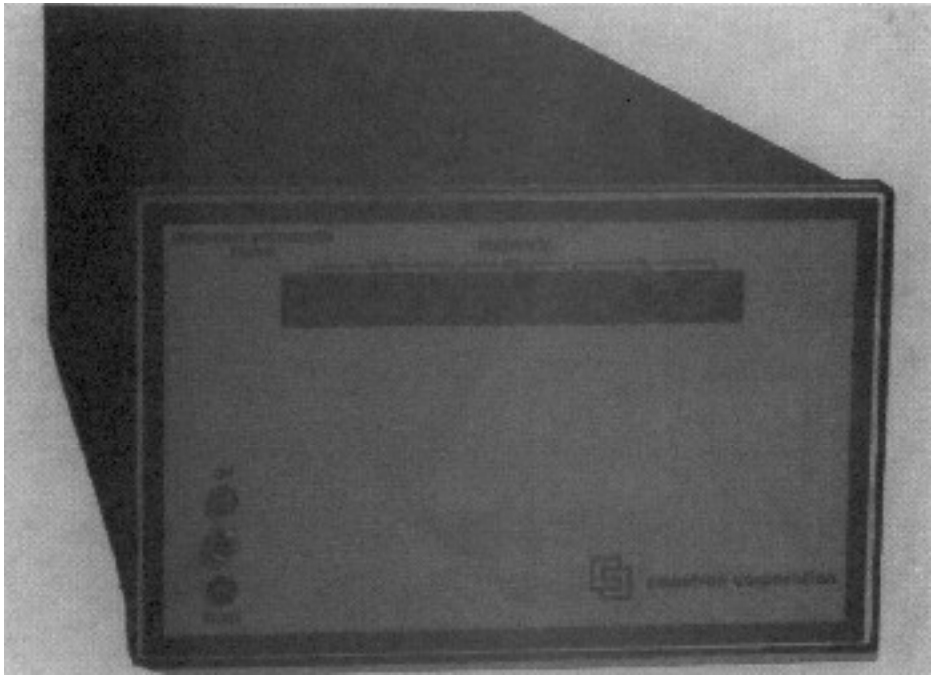


Figure 9-3 (*Continued*)

±0.1 μHz. The memory uses a RAM and therefore allows sine, square, ramp, and 16K ARB functions. Modulations are AM, PM, and frequency sweep. Spurious signals are −55 dBC, and subharmonics are −50 dBC.

9-2 Reference Generators

All frequency synthesizers, at least according to the definitions of this text, use a reference input from which all output frequencies are derived. The reference frequency determines the stability and accuracy of the output frequencies, since all are slaved to the reference.

Output phase noise might also relate to the quality of the reference, although often this is not the case. In PLL circuits, unlike DDS or DA, sometimes the phase detector and the analog circuitry noise can mask the crystal reference. In this chapter we present briefly the principles of reference generation and review the state of the art.

Figure 9-4 Schomandl ND500 (bottom) and ND1000 (top). (*Courtesy of Schomandl.*)

9-2-1 General review

Most synthesizers are specified for total accuracy. This is a complex parameter that breaks down to:

- The temperature range at which the accuracy is kept
- Accuracy per day/month/year
- Aging rate with time
- Stability per mechanical orientation and sensitivity to magnetic field

All these parameters relate to the long-term stability and fluctuations of the reference and affect the output frequency directly.

Another most important parameter is the *reference phase noise,* which affects the total phase noise performance, especially close to the carrier.

In direct synthesizers, the output phase noise spectrum follows that of the reference, because in most cases the reference frequencies being derived from the master reference, and used to synthesize the signal, are generated by multiplying or dividing the reference. Thus in a direct synthesizer we can expect the output phase noise to be corrupted by at least 20 log (F_{out}/F_{ref}). Thus, if the synthesizer uses 10 MHz as a reference and generates 230 MHz, we can expect a phase noise corruption of 20 log 23, or approximately 27-dB reference to the 10-MHz reference. If this reference has a phase noise characteristic of −145 dBC/Hz at 1 kHz from the carrier, the upper limit of the output performance for practical designs will be −145 + 27 = −118 dBC/Hz at this frequency.

In PLL circuits, the output can improve the 20 log N ratio because of the "cleaning" effect of the VCO. However, other noises add, as mentioned above.

The most popular reference is a crystal oscillator. A vast body of knowledge has been accumulated since the beginning of their use (approximately 50 years ago), and excellent performance levels can be achieved.

Crystal oscillators have extremely high Q, and this yields their accuracy, stability, and phase noise performance. Most crystal manufacturers cut fundamental crystals up to 35 to 40 MHz; above these numbers, overtones are used (odd numbers only, so third, fifth, and seventh overtones are common). However, in the last few years, few manufacturers achieve fundamental frequencies up to 300 MHz. This requires special mechanical structure and process since the crystal width is tiny. Fundamental frequencies up to 200 MHz are reasonably priced and affordable today.

Crystal oscillators using overtone mode are available up to 500 MHz. Beyond that, multiplers are used.

Another technique for generating references at high frequency is the use of *surface acoustic wave* (*SAW*) resonators, available now from a multitude of manufacturers. This is also a crystal, but the physical principles of its operation are different. SAW oscillators operate up to 1500 MHz.

The best accuracy and stability are achieved when atomic standards are used. In all cases, the atomic reference is followed by a

TABLE 9-1 Reference Oscillator Performance

	SAW	Crystal	Atomic (cesium)
Accuracy, ppm	10	0.0001	0.0000001
Stability, ppm	20	0.0002	0.0000001
Phase noise (10 MHz)			
at 1 Hz	—	−100	−100
at 10 Hz	—	−130	−130
at 100 Hz	−130	−150	−150
at 1 kHz	−140	−165	−160
at 10 kHz	−150	−170	−170
at 100 kHz	−160	−170	−170
Reproducibility, ppm	20	0.0001	0.0000005
Long-term (per day), ppm	5	0.00002	0.00000002
Cost ($)	50–600	100–2000	2000–50,000

crystal oscillator locked to it. The atomic standard provides the long-term accuracy, but is relatively noisy. The crystal oscillator locks to the atomic standard and provides the phase noise performance.

A state-of-the-art performance for the three references is shown in Table 9-1. Note that high-quality crystal oscillators are ovenized to keep the operating environment as stable as possible.

Atomic references are used only in very demanding applications. SAW devices are used when high-frequency references are necessary and are usually being locked to a crystal reference. As mentioned above, because of the inherent physical performance, price, and volume, the most common references are crystal oscillators.

9-2-2 Crystal oscillators

Crystal oscillators usually employ one active device (transistor or FET), the crystal resonator, and the support circuitry followed by a buffer amplifier to isolate the oscillator from the load. Typical circuits for fundamental and overtone designs are available in many referneces (see Refs. 1, 2, and 3).

Note that the overtone design oscillates at the overtone frequency rather than oscillating at the fundamental and multiplying. State-of-the-art performance is shown in Fig. 9-5, and a SAW oscillator diagram is shown in Fig. 9-6.

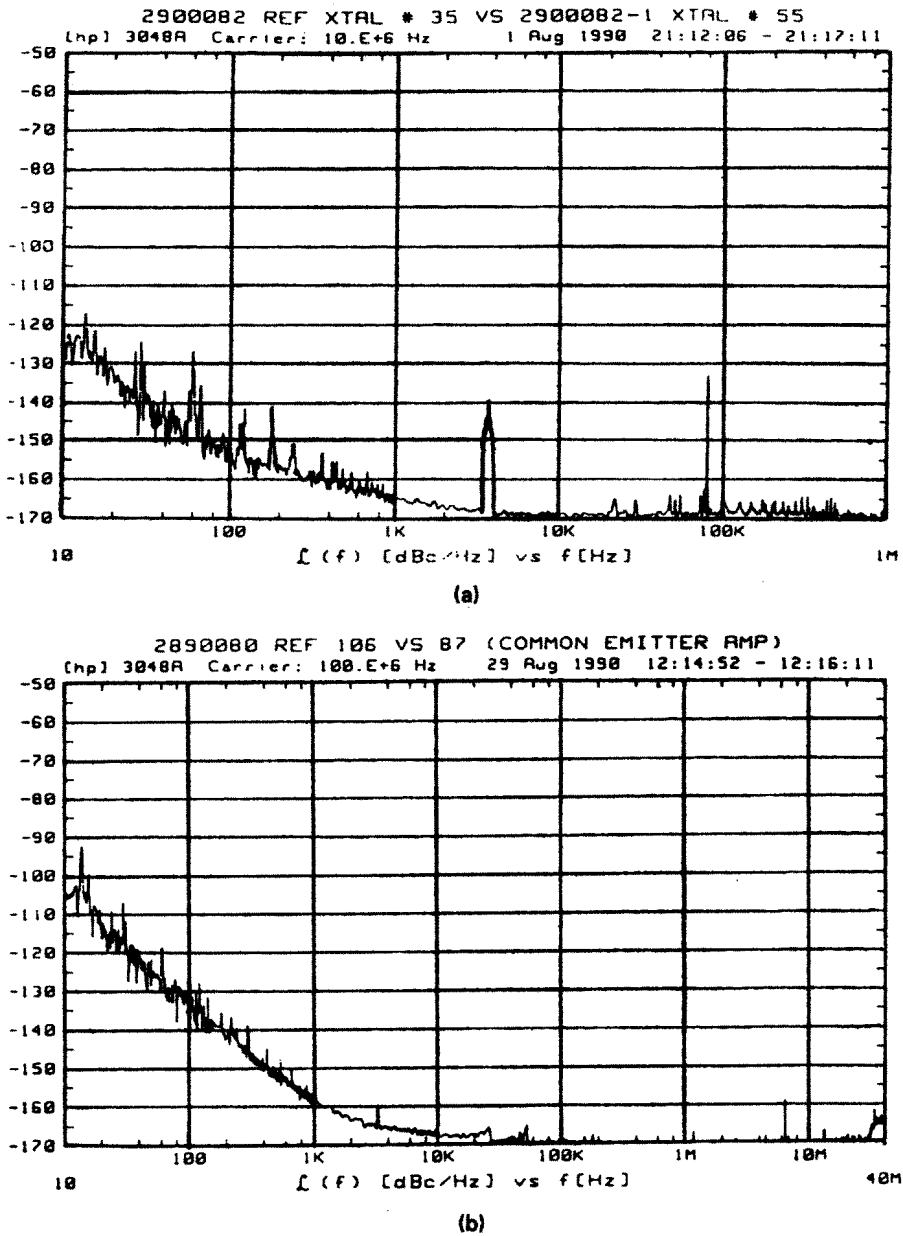
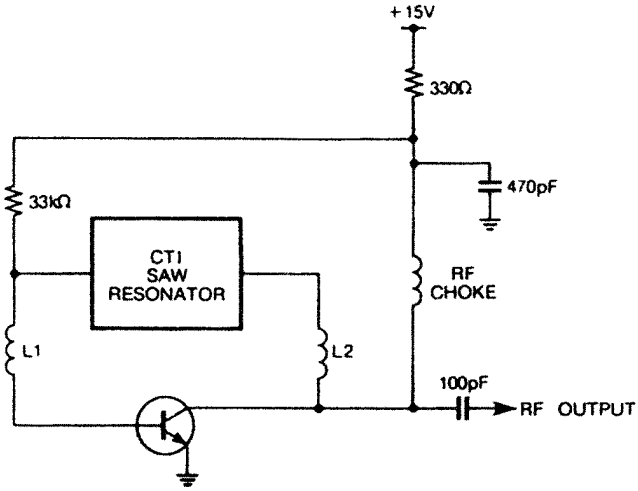
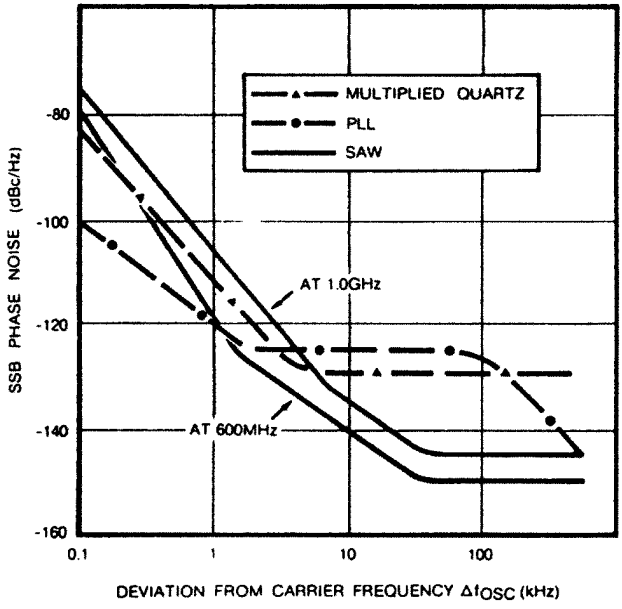


Figure 9-5 Crystal oscillator state-of-the-art phase noise performance at (*a*) 10 and (*b*) 100 MHz. Measured on HP3048 system. (*Courtesy of Piezo Crystal Co.*)



(a)



(b)

Figure 9-6 SAW (*a*) oscillator and (*b*) phase noise. (*Courtesy of Crystal Technology.*)

From time to time it is necessary to lock the reference to yet another one. This requires tuneability, and a voltage controlled crystal oscillator (VCXO) is used. For example, many high-frequency synthesizers use 100 MHz as the fundamental reference in the design. However, 100 MHz is not a "standard" standard, so the 100-MHz VCXO needs to be able to lock to 5- or 10-MHz references.

Research and development in crystal and atomic references continue constantly, to improve accuracy and reduce both power and cost.

9-3 Conclusion

Many instruments already use DDS and fractional-*N* synthesis techniques to improve performance and competitive position. These disciplines ensure producibility, pricing, resolution, speed, and digital design with its inherent advantages.

Increased proliferation of these technologies is expected, into the test instrumentation and for the OEM user, as their advantages improve competitive position and performance and enable accuracies and complex waveforms not attainable before.

References

1. Sciteq Electronics catalog.
2. Piezo Technology catalog.
3. Schomandl (Germany) catalog.
4. HP3325B manual.
5. HP8662A manual.
6. PTS310 manual.
7. FS-2000 manual.
8. DS 345 manual.

This is a blank page.

Chapter

10

Original Paper and Software

This chapter presents, at its end, the original paper that started the DDS technological evolution with some short footnotes; and then a description of the files, mainly memory and compression algorithms, on the accompanying CD-ROM.

10-1 Tierney, Rader, and Gold Article

The article was published almost 30 years ago: J. Tierney, C. M. Rader, and B. Gold, "A Digital Frequency Synthesizer," *IEEE Transactions on Audio and Electroacoustics,* vol. AU-19, No. 1, March 1971, pp. 48–56. It is a pioneering work and highly recommended. One has to bear in mind that the state of available hardware was very different, and therefore the design considerations changed over the years. Specifically, the ROM compression is very creative but not one that we use today.

The article starts with two approaches. The first is the solution of a differential equation whose Z transform has poles on the unit circle. This solution is a closed-loop one. The second approach is an open-loop solution, one that leads to the standard DDS approach. The block diagram shown in Fig. 4 of the article is almost exactly that of a standard direct digital synthesizer.

Because of the size of the memory required to transform 15 phase bits to 12 bits, Tierney et al. suggest a compression algorithm, based on lookup and interpolation, and use a combination

of memory and multipliers, as shown in the equations on page 51 of the article. The rest of the article deals with the quantization and DAC noise estimates and forecasts the utilization of phase control.

10-2 Description of Files on CDROM

Assorted Design Tools

All programs ending with .bas use Qbasic as the platform. Other files, not executable, are written for Microsoft Word 6.0 for the PC or were created in MathCAD (.mcd files). The CDROM contains the following files:

1. Sine1.bas, Cosi1.bas – creating sine and cosine quantized data having W input bits and D output bits.
2. Hutchsn.bas – Hutchison compression algorithm.
3. Sinsndrl.bas – Sunderland type compression.
4. Sinfin55.bas, Sinfin99.bas, Sinfin66.bas – Auxiliary function compression.
5. Phaznoiz.exe – executable file, translates L(fm) and spurious noise to degrees rms and time jitter. Program written by Brian Bannister and is an approximation.
6. Chirp2.bas – simulates a DDS chirp and compares the instantaneous frequency to the ideal.
7. Serial.exe – a generic serial frequency control for 3 wire format (used by almost all chip makers (Fujitsu, Motorola, National S.C, etc...)
8. Sinrnd66.bas – ROM compression with randomization.
9. Zverev.exe – executable file, calculates Cauer filters for 3,5,7, sections.
10. Snrquan.bas – calculates sine quantization rms error.
11. Dualmd16 or 32 or 64.bas – calculates dual modulus control.
12. 1618app1.doc – is a Word6.0 document application note for the SEI-1618 Sciteq PLL fractional-N chip.

13. Many files were created using MATHCAD. These are all followed by .mcd. The titles will help you find your way. "Combline" shows time and spectrum of various comblines, "bessel" is Bessel functions, "pll" are various PLL transfer functions and noise profiles. Of special interest will be pll3ERD and pll4ERD, which simulate transfer functions and phase margin of 3rd and 4th order PLL loops, using real world parameters. Also, note deltasigma and DS programs for fractional PLL of 3rd order.

Analog Devices DDS Tutorial

This tutorial in pdf format offers an excellent overview of the DDS field in addition to application notes useful to designers. (Used with permission of Analog Devices.)

PDF Version of This Book

Also included on the CDROM is a fully searchable version of this book in pdf format. This file is readable using the Adobe Acrobat Reader.

Other Books from LLH

Sample software versions of other technical references from LLH Technology Publishing are included on the CDROM, along with purchasing information.

Various friends and colleagues made a contribution to this disk. If you have something interesting related to FS that you wish to share with other designers, please let us know. Thanks.

giora18@tns.net

A Digital Frequency Synthesizer

JOSEPH TIERNEY, Member, IEEE
CHARLES M. RADER, Member, IEEE
BERNARD GOLD, Senior Member, IEEE
M.I.T. Lincoln Laboratory
Lexington, Mass. 02173

Abstract

A digital frequency synthesizer has been designed and constructed based on generating digital samples of $\exp[j(2\pi nk/N)]$ at time nT. The real and imaginary parts of this exponential form samples of quadrature sinusoids where the frequency index k is allowed to vary $(-N/4) \leq K < (N/4)$. The digital samples drive digital to analog converters followed by low-pass interpolating filters to produce analog sinusoids. The method is superior to digital difference equations with poles on the unit circle since the noise or numerical inaccuracy remains bounded.

The digital technique used consists of factoring the exponential into two table look-ups from an efficiently organized small READ-ONLY memory table and performing a complex multiply to produce the real and imaginary components. A small array multiplier efficiently organized performs the multiplications.

The technique leads itself to the production of phase coherent or phase controlled sinusoids because of the indexing arrangement used. In addition finer frequency steps than the READ-ONLY memory allows are available by expanding the indexing register at no increase in inaccuracy.

Manuscript received July 22, 1970.

This work was sponsored by the Department of the Air Force.

Introduction

The generation of many different frequencies from a single stable source frequency is commonly achieved with analog circuits [1], [2] or combinations of analog and digital circuits [3], [4]. All of these approaches [5] generate a large set of frequencies from a single source in the analog or continuous sense by division, phase lock, mixing, or a combination of such techniques. An attractive alternate approach is the generation of a set of sampled sinusoids by digital computation of some kind (including simple table look-up) at the sampling times. Fig. 1 indicates the sample trains produced for two different frequencies. The return to analog or continuous sinusoids is accomplished with simple realizable smoothing filters.

One can consider the conventional frequency synthesizer as processing a single source frequency to produce a large set of new frequencies. The digital frequency synthesizer uses a single frequency to establish a stable sampling time at which sample values are computed. The difference in approach is one of using the source as a frequency directly or as a time reference.

The Digital Approach

Given the problem of computing samples of sinusoids the most obvious choice is a digital recursion, a difference equation whose Z transform has poles on the unit circle. By starting such a recursion with the proper initial conditions one can produce sinusoidal samples. There are at least two problems with this approach (Fig. 2). The frequency of the sampled sinusoid produced by such a typical recursion is not linearly related to a settable coefficient but related by the function cos WT, where T is the sampling interval, W is the produced frequency, and 2 cos WT is the actual coefficient of the recursion. The noise produced by such a recursion is in general worse than can be obtained by other methods and may in some cases be a function of the number of iterations [6].

The chosen approach is simply a direct computation of the samples,

$$\cos(\omega nT + \phi), \qquad \sin(\omega nT + \phi) \qquad n = \text{time index.}$$

Consider values of

$$\cos 2\pi fnT = \text{Re}\,(e^{j2\pi fnT})$$

$$\sin 2\pi fnT = \text{Im}\,(e^{j2\pi fnT})$$

where $f = kfo$, fo = lowest computed frequency. That is, we can compute multiples of some lowest frequency,

$$f_o = \frac{1}{NT}$$

where N is a design parameter. Then the exponential becomes

$$e^{j2\pi fnT} = \exp\left[j\frac{2\pi}{N}nk\right].$$

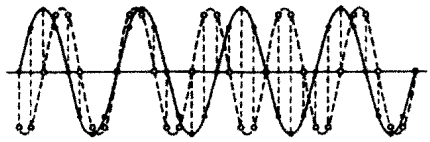


Fig. 1. Digital frequency determination.

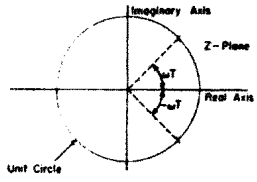


Fig. 2. Pole position of the digital recursion for sinusoidal samples.

$$y_n = (2\cos\omega T)\,y_{n-1} - y_{n-2}$$
$$y_0 = \cos\phi$$
$$y_{-1} = \cos[\phi - \omega T].$$

Samples of this complex exponential are equivalent to values of sine and cosine of the argument $2\pi fnT$ with

$$f = \frac{k}{NT}.$$

Computing samples of this complex exponential indexed on a frequency index k, and a time index n, is equivalent to computing coordinates of one of N equispaced points around the unit circle (Fig. 3) in the complex plane described by

$$\exp[j0], \quad \exp\left[j\frac{2\pi}{N}\right], \quad \exp\left[j\frac{2\pi}{N}2\right], \cdots, \exp\left[j\frac{2\pi}{N}(N-1)\right].$$

For a particular frequency index k, the argument of the exponential varies in increments of $(2\pi/N)k$ in successive time indices. The product nk is treated modulo N since $\exp[j(2\pi/N)[X+N)] = \exp[j(2\pi/N)X]$ for any X. The generation of samples of a complex sinusoid consists then of accumulating multiples of k (i.e., nk at time n, $[n+1]k$ at time $n+1$), and using the accumulated value to calculate $\exp[j(2\pi/N)nk]$.[1]

[1] If in accumulating multiples of k, the accumulator is not initially zero but contains some constant C, the argument of the exponential becomes $(2\pi/N)(nk+C)$ which affects the phase of the result, but not the frequency. This can be useful in phase control.

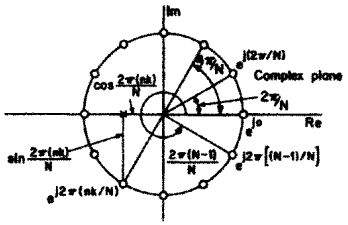


Fig. 3. Equispaced samples on the unit circle.

The Calculation

To determine the value $\exp[j(2\pi/N)[nk+C] = \exp[j(2\pi/N)Y]$ consider the simplest case, namely, a table storing N values from $\exp[j(2\pi 0/N)]$ to $\exp[j(2\pi(N-1)/N)]$. The computation then consists of using the value in the accumulator Y to index the table of N values producing a single complex sample [7]. As Y increases, the table is scanned producing sinusoidal samples. For a larger k value the table is covered faster (the interval between successive Ys, or nks is larger). Such an approach is suitable for small values of N, but more often we are interested in large N (a large set of frequencies) so that a single table look-up becomes impractical because of the size of the table.

For large N and $0 \le Y \le N-1$, Y may be broken into a sum of several words each of which represents a part of Y. If $Y=q+r+s$, then $\exp[j(2\pi Y/N)] = \exp[j(2\pi[q+r+s]/N)] = \exp[j(2\pi q/N)]\exp[j(2\pi r/N)]\exp[j(2\pi s/N)]$ where each factor takes on many fewer than N values and the overall storage has been reduced. For example, for N, a power of 2, say 2^b, then $0 \le Y \le 2^b-1$ and can be represented as a binary number b digits long. $Y=\alpha_0 2^0+\alpha_1 2^1+\alpha_2 2^2 + \cdots + \alpha_{b-1}2^{b-1}$. We can factor the exponential into b factors each of which is of the form $\exp[j(2\pi\alpha_i 2^i/N)]$ with $\alpha_i = 0$ or 1. Thus the table has been reduced from 2^b complex entries to b complex entries ($\log_2 2^b$) and we need $b-1$ complex multiplications to obtain the value $\exp[j(2\pi Y/N)] = \exp[j(2\pi/N)[nk+C]]$. Obviously the number of factors or the number of complex multiplies is one of the design parameters in this approach to frequency synthesis.

For our purposes factoring the complex exponential into two terms allows for a very efficient use of READ-ONLY memory. In addition we may take advantage of approximations to the sine and cosine of small angles as well as symmetries in the functions to effect further savings in READ-ONLY storage.

The Complete Synthesizer

The block diagram of a digital synthesizer which produces quadrature outputs is shown in Fig. 4. An input frequency control word k is stored in a register and used

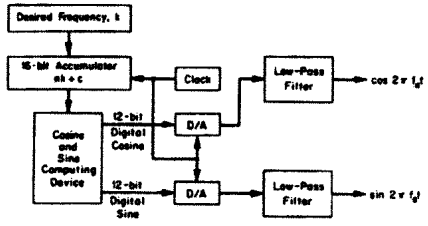


Fig. 4. Synthesizer block diagram.

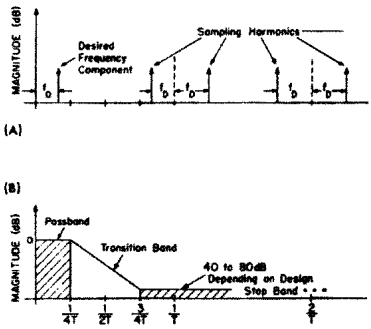


Fig. 5. (A) Sampled sinusoid spectrum. (B) Smoothing filter response.

to update an accumulator every T seconds. Each time the accumulator is changed, the value $nk+C$ is used to compute the real and imaginary parts of $\exp[j(2\pi/N)Y]$ by one of the methods proposed in the previous section. The computed values of $\sin(2\pi/N)Y$, $\cos(2\pi/N)Y$ are used to drive a pair of D–A converters of the proper word length to produce analog samples. These samples are interpolated by the output smoothing filters. At a sampling interval of T seconds the output spectrum before smoothing of a sampled sine or cosine would look like that shown in Fig. 5. The Nyquist condition would allow us to produce frequencies of up to but less than $1/2T$ which we could recover with ideal LP filters with cutoff at $1/2T$. However, for ease of filtering consider using only $1/4$ of the sampling frequency as the band limit. Then we would like an output smoothing filter which passes all frequencies up to $1/4T$ with some design ripple, to have a transition band in the interval from $1/4T$ to $3/4T$, and to have some out of band attenuation depending on the allowed sampling harmonics. Such a filter is shown in Fig. 5(B). For an accumulator which overflows at some N and a sampling interval T, a digital frequency synthesizer as shown in Fig. 4 would produce a lowest frequency of $1/NT$, and a highest frequency of $1/4T$ for a total of $N/4$ different frequencies. If we take advantage of the quadrature outputs frum this realization we can double the bandwidth of $1/4T$ since we can modulate a carrier to $\pm 1/4T$ for a total of $N/2$ frequencies. Note that a single output synthesizer can be implemented using only one D–A converter if so desired. There is no constraint to produce quadrature outputs although they may in some cases be desirable, as shown above.

A Design Example

Up to this point the discussion of index accumulation and calculation of sine and cosine has been general. We could use any arithmetic and any size N. Consider now a design example using binary arithmetic and standard binary logic. The design specifications are: 1) 2^{16} frequencies; 2) 409.6 kHz bandwidth; 3) 12.5 Hz frequency spacing (or lowest frequency); 4) 70 dB spectral purity (ratio of power in desired frequency to power in any other 100 Hz band). If we assume the quadrature outputs will be used to obtain a bandwidth of $1/2T$, then $T=1/2\times 409.6$ kHz$\cong 1.22$ μs. In addition $N/2=2^{15}$ or $N=2^{16}$ so that the accumulator will be a 16-bit binary register, and the frequency spacing of $1/NT=12.5$ Hz. If quadrature outputs are desired, the synthesizer must produce values of $\exp[j(2\pi/2^{16})Y]$, $0\leq Y\leq 2^{16}-1$, that is, one of 2^{16} points equispaced around the unit circle. Since the highest frequency allowed is $1/4T$, four points around the circle, the largest k value corresponding to this value of frequency is $2^{16}/4$ or 2^{14}. A two's complement negative frequency input will cause samples to occur in the opposite sense around the unit circle, which is meaningful as a negative frequency only if quadrature outputs are available.

The remaining problem is the computation of the samples. To meet the design requirement of 70 dB purity the computation must be carried to about 12 bits. That is, if a sample out of the computation consists of a sign and 11 binary digits and the accuracy is $\pm$ the last digit or $\pm 2^{-11}$, a worst case harmonic caused by such an error will be 66 dB down from the output. As shown in a later section this bound is quite pessimistic and 70 dB is more likely. Using this 12-bit arithmetic the following procedure, which takes considerable advantage of the symmetries of the sine and cosine, is used.

First note that neglecting the least significant bit in the 16-bit accumulator will cause an amplitude error of no more than $\sin(2\pi/2^{16})$ which is roughly 2^{-12}. In fact one could extend the accumulator register and the input frequency word on the low significance end, use these extra bits for accumulation, but ignore them for the computation of sine and cosine, and still be bounded by $\sin(2\pi/2^{16})$ as an amplitude error. This implies generating finer and finer frequency steps by adding only to the ac-

cumulator, a feature which is unique to this type of synthesis and efficient in terms of memory use. Computing sine and cosine of multiples of $2\pi/2^{16}$ (ignoring bit 16) is solved by table look-up and interpolation as follows. Consider Y represented in binary form as

$$\begin{aligned} Y &= 2^0d_0 + 2^1d_1 + 2^2d_2 + 2^3d_3 + 2^4d_4 + 2^5d_5 + 2^6d_6 \\ &\quad + 2^7(d_7 + 2^1d_8 + 2^2d_9 + 2^3d_{10} + 2^4d_{11} \\ &\quad + 2^5d_{12} + 2^6d_{13} + 2^7d_{14}) \\ &= 2^0d_0 + 2^1d_1 + 2^2d_2 + 2^3d_3 + 2^4d_4 + 2^5d_5 - 2^6d_6 \\ &\quad + 2^7(d_6 + d_7 + 2^1d_8 + 2^2d_9 + 2^3d_{10} + 2^4d_{11} \\ &\qquad + 2^5d_{12} + 2^6d_{13} + 2^7d_{14}) \end{aligned}$$

or

$$Y = f + 2^7e$$

where

$$\begin{aligned} f &= 2^0d_0 + 2^1d_1 + \cdots + 2^5d_5 - 2^6d_6 \\ e &= (d_6 + \cdots + 2^7d_{14}) \end{aligned}$$

so that the exponential can be factored as

$$\left(\exp\left[j\frac{2\pi}{2^{16}}f\right]\right)\left(\exp\left[j\frac{2\pi}{2^{16}}2^7e\right]\right) = \left(\exp\left[j\frac{2\pi}{2^{16}}f\right]\right)\left(\exp\left[j\frac{2\pi}{2^8}e\right]\right).$$

The computation of the complex exponential is then reduced to two table look-ups $\exp[j(2\pi/2^{16})f]$, $\exp[j(2\pi/2^8)e]$ and a complex multiply. The index e consists of the eight high-order bits of the accumulator rounded by the bit d_6, while the index f consists of the six lower bits d^5 through d_0 if d_6 is zero, or these bits two's complemented if d_6 is one. From the point of view of computing the value of a point at a particular angle $(2\pi/N)Y$, around the unit circle, the value of e determines which of 2^8 equally coarsely spaced points is nearest the desired point, and the value of f determines which of 64 possible angular corrections should be added to or subtracted from the coarse point to get the desired value. The angular correction is, of course, a complex multiplication. For this two factor approach the complex multiply is implementing the trigonometric identities

$$\sin(x + y) = \sin x \cos y + \cos x \sin y$$

$$\cos(x + y) = \cos x \cos y - \sin x \sin y$$

with

$$x = (2\pi/2^8)e, \qquad y = (2\pi/2^{16})f$$

where f may be positive or negative (according to bit d_6). Fig. 6 represents these operations on the unit circles.

To compute the value at X in Fig. 6, the eight high-order bits would be augmented by one (since $d_6 = 1$), giving e, and the value of f would be the two's complement of the distance above the center of the coarse interval. In other words we would go to a larger angle and decrement because $d_6 = 1$. If $d_6 = 0$ we would start at the lower angle and increment.

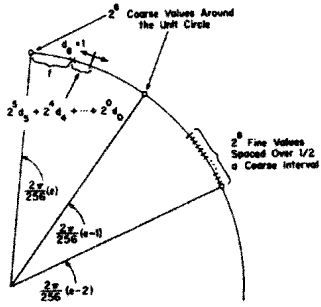


Fig. 6.

The bit d_6 is used to break the coarse intervals in half. If this bit is one, it means the computation needs the coarse value just bigger than the desired value. The fine correction is then clockwise. If d_6 is a zero, the coarse value just below the desired value will do, and the correction is counterclockwise.

Now we note that the cosine component of $\exp[j2\pi f/32\,768]$ is between 1.0 and 0.9999247, a difference in the 14th bit of its binary representation. Therefore we will approximate it by 1. The sine component is so small that its six most significant bits, not counting the sign bit, are equal to the sign bit, which is in turn equal to d_6. Thus the READ-ONLY memory indexed by f need only save the five least significant bits of the 11-bit plus sign representation of the sine corresponding to each of the 64 positive values of f; the sines corresponding to negative values of f can be found by taking the two's complement of f and changing the sign of the result. It is easier to take the one's complement, and this also results in an insignificant error.

The value of $\exp[j2\pi e/256]$ can also be found by look-up in a table with only 64 values corresponding to $\frac{1}{4}$ cycle of a sine wave. The sine and cosine components are addressed with the six least significant bits of e, and its two's complement, and the two most significant bits of e are used for exchanging the components and complementing either or both if necessary; mathematically, if the six least significant bits of e are g and the two most significant bits are h, we look up $\sin 2\pi(64-g)/256 + j \sin 2\pi g/256$ and multiply the result by j^h.

The final operation is the multiplication of the coarse estimate $\exp[j2\pi e/256]$ by the fine corrector $(1 + j \sin 2\pi f/32\,768)$. This requires two 8 by 5 bit multiplications and two additions. Two answers are kept to an accuracy of 11 bits plus sign and fed to the D–A converters.

The READ-ONLY memory requirements are 64 words of 5 bits fine angle and 11 bits coarse angle which can be combined into a 64 word by 16-bit memory which is accessed three times in a computation. The detailed block diagram of this synthesizer is shown in Fig. 7.

The array multiplier indicated in the block diagram is used to perform the 5 by 8 multiplication (5×8 rather than 5×11 because of the accuracy required). It can be implemented in several ways using the 4-bit TTL adder packages currently available. Using only the bits needed for the final accuracy of $\pm 2^{-12}$ and using a tree-like interconnection between partial sums the multiplication time is about 160 ns.

The digital-to-analog converters driven by the final 12-bit sine and cosine outputs are updated every 1.22 μs and hold the data words between transfer times. This sample and hold operation provides smoothing and filtering in addition to that provided by the output low-pass filters.

A synthesizer designed and built according to the above discussion requires about 85 TTL logic packages, and dissipates about 12 W. If quadrature outputs are not needed, a still simpler device will suffice.

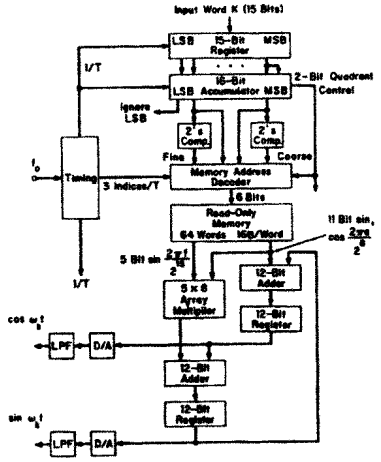


Fig. 7.

Output Noise and Time Response

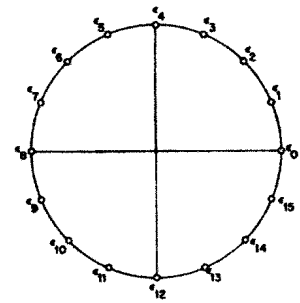
The synthesizer noise output consists of three separate effects. The first contribution results from truncation or roundoff in computation of the sample value. The second effect is from the output digital to analog converter. Sampling harmonics passing through the smoothing filter contribute the third effect.

Truncation Noise

Since the calculation for a particular sample (that is, a particular value in the index accumulator) is always the same, the output samples are those of an exact sinusoid with some deterministic noise sample train added to it due to truncation. For an arbitrary generated frequency the truncation noise will have a period equal to NT, the largest period possible so that the truncation noise consists of a line spectrum with frequency spacing $f_0 = 1/NT$. If the generated frequency k/NT has one or more factors of 2 in k, or $k = 2^l k^*$, then the truncation noise harmonics are multiples of $2^l f_0$. This is a consequence of N being a power of 2. The limiting case occurs for $k = 2^e$, some power of 2. In that case the truncation harmonics are harmonics of the generated frequency. These cases are demonstrated for 16 equispaced samples around a unit circle in Fig. 8. Each sample has associated with it a truncation error ϵ_i and each line below represents the sequence of sample errors generated as the index k increases. The three cases are shown. Lines 1, 2, 4 are cases of a power of 2 times lowest frequency generated. Notice that the error period is the same as the generated period. Lines 3, 5, 7 represent noise periods of $16T$, the longest possible because the frequency index K has no factors of 2. Line 6 shows one factor of 2. As a consequence of our particular arithmetic implementation the error waveform only contains odd harmonics, that is, it has the property $\epsilon(n) = -\epsilon[n+(P/2)]$ where P is the period.

To bound the noise contributed by the truncation error, consider a generated frequency which is an odd factor k

Fig. 8.



	Frequency		Noise Period
(1)	Lowest 1/16T	Noise period same as generated period	16T
(2)	1/8T	Noise period same as generated period	8T
(3)	3/16T	0 3 6 9 12 15 2 5 8 11 14 1 4 7 10 13	16T
(4)	1/4	Noise period same as generated period	4T
(5)	5/16T	0 5 10 15 4 9 14 3 8 13 2 7 12 1 6 11	16T
(6)	6/16T	0 6 12 2 8 14 4 10 0	8T
(7)	Highest 7/16T	0 7 14 5 12 3 10 1 8 15 6 13 4 11 2 9	16T

times $2\pi/NT$. In this case the error waveform is of period NT or N samples long. A Parseval's relation for a discrete Fourier transform over N samples can be written as

$$\sum_{i=0}^{N-1} |F_i|^2 = N \sum_{n=0}^{N-1} \epsilon_n^2$$

where F_i is a frequency amplitude defined by

$$F_i = \sum_{l=0}^{N-1} \epsilon_l \exp\left(-j \frac{2\pi}{N} il\right)$$

and ϵ_n is the error associated with a particular sample. The ϵ_n is hopefully a pseudorandom variable uniformly distributed over the interval -2^{-11} to $+2^{-11}$ because of the truncation. We may assume all of the error energy in one frequency to obtain a bound. For a particular

$$|F|^2 = N \sum_{n=0}^{N-1} \epsilon_n^2$$

and assuming worst case conditions on ϵ_n,

$$|F|^2 = N^2(2^{-11})^2 \quad \text{or} \quad |F| = N2^{-11}.$$

The desired generated frequency will have an amplitude of N in the discrete transform, so that the ratio of noise amplitude to signal amplitude is 2^{-11}, the case we referred to earlier. A more realistic "bound" for the cases when the noise period includes many samples should be

$$\sum_{i=0}^{N-1} |F_i|^2 = N^2\left(\frac{1}{N} \sum_{n=0}^{N-1} \epsilon_n^2\right)$$

where the quantity in parenthesis is the error variance. Since the variance is $[(2^{-11})^2/3]$ for uniformly distributed noise, we expect

$$\sum_{i=0}^{N-1} |F_i|^2 = N^2 \frac{(2^{-11})^2}{3}.$$

So the bound obtained assuming all the energy in one harmonic is $2^{-11}/\sqrt{3}$ noise amplitude to signal amplitude ratio or about -71 dB. If one assumes that the noise waveform is white in one period and using the fact that it contains only odd harmonics we have

$$\frac{N}{2} |F|^2 = N^2 \frac{(2^{-11})^2}{3}$$

so that noise to signal ratio $= \sqrt{2/3N}\, 2^{-11}$ which is very small for large N.

Digital to Analog Converter Noise

Even in the case of perfectly calculated samples driving the digital to analog converter, the output analog samples will produce noise from switching time disparities between bits, and differences in on and off switching. These so called "glitches" which occur at the transitions between sample outputs depend on the initial and final words upon which the converter is acting. A transition

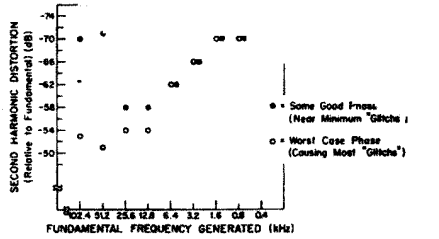


Fig. 9.

that changes many bits such as 10000 to 01111 will tend to produce more noise energy than a transition of only a few bits. This noise has the same periodicity properties that truncation noise has since it depends on transitions which are periodic if the sample train is periodic. However, all harmonics of the basic frequency are present since the "glitches" do not have odd symmetry. If one assumes a "glitch" amplitude can be as high as one-half of full scale out of the converter, then such a noise pulse of width Δ occurring even only once in each period will produce a noise-to-signal rate of approximately Δ/NT. For the higher generated frequencies such a ratio can be large. For example if $N=8$ (a frequency generated of 2^{13} $(2\pi/NT)$) and we want to maintain a noise-to-signal ratio of 10^{-3} (60 dB), the "glitch" duration must be about 8 ns or less. Such a requirement is difficult to achieve even with current techniques. Basically then the time of nonlinearity must be small compared to the roughly 1-μs sample interval for low noise effects. In our design example a 12-bit converter with settling time of about 300 ns (presumably "glitches" of any consequence occupy only a small portion of this overall transient) in an interval of 1.22 μs produced a distortion curve as shown in Fig. 9. Since this curve represents only second-harmonic distortion it is not a result of truncation, but strictly converter distortion. Higher harmonics were smaller or equal to those shown on the curve. For generated frequencies which were not powers of 2 as given on the graph, the distortion products were less, falling between the two cases given. The large disparity between "good" and "bad" phase for the higher generated frequencies is a consequence of the small number of samples per period so that no averaging takes place. Either a good set of samples does or does not occur so that the noise is high or low.

Even for odd generated frequencies the predominant noise tends to be harmonics of the generated frequency rather than harmonics of the lowest frequency. This indicates that certain more significant bits of the D-A converter are delayed more or less than others, and differences between on and off switch times are significant. In order to reduce these transition effects, one is forced to

gating or sampling devices on the output of the converter. However, for times of less than 1 μs and linearities extending into the greater than 60 dB range, such circuits require careful design. Using such a gate for shorting the converter output to ground during transition time, the design presented earlier produced a worst case harmonic of 55 dB, an improvement upon the curve presented for the converter alone.

Smoothing Filter Time and Frequency Response

The third source of noise contributions to the synthesizer output is sampling harmonics of the generated frequency. As mentioned earlier a final smoothing filter is needed to interpolate computed sinusoidal samples or smooth the sampled spectrum. Since this smoothing filter is the energy storage device in the synthesizer the problem of time response arises. From Fig. 5(B) as previously seen, a filter was needed to pass the band up to 204.8 kHz and reject the band above 614.4 kHz. with a transition region between. The first tradeoff encountered is that between rejection or attenuation at 614.4 kHz and the time response of the filter. Consider a low-pass filter of fifth order (five poles) and examine its attenuation at 614.4 kHz as well as the time response. For several filter types the following table compares attenuation and step response. (Sample and hold response will add 10 dB to these figures.)

Type	Attenuation at 614.4 kHz	Step Response
Bessel (max flat phase)	~10 dB	~1 μs
Butterworth (max flat amp)	~48 dB	~4 μs
Chebyshev (1.0 dB ripple)	~65 dB	~7 μs
Cauer (elliptic function filter) 1.0 dB ripple	~85 dB	~9 μs

For this comparison the filters are 1.0 dB down at 204.8 and the step response is measured from 10 percent of final value to a ± 10 percent window around final value. In other words overshoots must settle down to within a ± 10 percent window. The obvious point made by this table is the tradeoff just mentioned. In addition, as the filter type moves from Bessel down, the phase or envelope delay characteristics become poorer. A similar trade can be effected between width of passband and step responses for a fixed attenuation at 614.4 kHz. If a certain attenuation is desired at that frequency a Bessel filter has a smaller passband than does a Butterworth, and so on down the list. For a fixed order of filter, and a fixed attenuation to be achieved at 614.4 kHz, the widest passband is obtained by using the sharpest filter and this in turn has the poorest time response.

The response times used in the above discussion have been step responses to dc rather than steps of sinusoidal input. It is clear that the time response of a step transition from one frequency to another consists of the response to two frequency steps; one to cancel the original frequency the other to start the new frequency. The amplitude response to each of these frequency steps reflects the natural frequencies of the driven filter. Therefore, the dc step response is still a measure of the time for amplitude response to settle down.

However, it is often the instantaneous frequency out of the synthesizer that is of importance, and the time this measure takes to settle down to some meaningful value. Obviously the instantaneous frequency must be influenced by the natural frequencies of the smoothing filter and must become some steady state value after the filter response time [8].

If an exact response for the instantaneous frequency is necessary, careful computation is necessary. If a rough response time is adequate, the table values will suffice. It is true that the smaller the frequency change, the smaller the frequency and amplitude perturbation.

It is also possible to reduce switching effects in both amplitude and phase response while still using a sharp cutoff filter such as an elliptic filter. This is accomplished by extending its cutoff and therefore its poor phase response into the transition region where no synthesizer outputs occur. In this way poorer properties of the filter are never manifested and some advantage is still taken of the sharp cutoff. Given any set of amplitude or phase criteria, there are a large set of filters to chose from satisfying smoothing requirements.

Producing RF Frequencies

Since the basic digital synthesizer cannot produce RF frequencies directly because of D–A speed limitations rather than logic problems, other methods must be used to produce an RF synthesizer output.

The method of single sideband modulation as mentioned earlier is useful if quadrature outputs are available. This is shown in Fig. 10 and requires a quadrature carrier as well. To move from upper to lower sideband the sign of one of the synthesizer outputs must change. and this is easily done. Since this technique requires a balanced subtraction at the output it is difficult to achieve high suppression of the opposite side frequency of more than 40–50 dB over a wide range.

A second approach to modulating the synthesizer output involves a single synthesizer output rather than quadrature. If the device need not produce quadrature outputs, it can work at a higher sampling or computation rate and produce a wider band of output frequencies (~1/4T). Then the lower generated frequency is limited to produce a band from 1/4T down to some f_{low} which requires a modest filtering after modulation as shown in Fig. 11.

A third approach which is implemented at the digital level may be useful. This approach consists of computing quadrature outputs but at different sampling times (sam

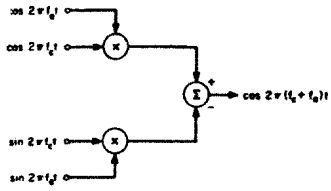
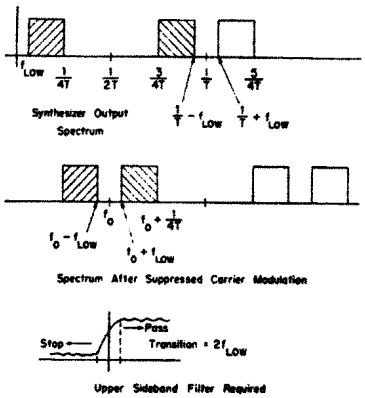
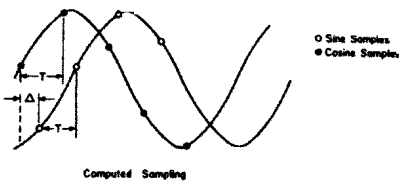


Fig. 10. Single sideband bandwidth doubling and frequency translation.

Fig. 11.





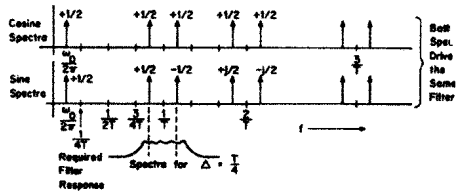


Fig. 12.

$$\text{sine spectrum} = \frac{1}{2}\sum_{n=-\infty}^{\infty}\left[\delta\left(\omega-\omega o-\frac{n2\pi}{T}\right)-\delta\left(\omega+\omega o-\frac{n2\pi}{T}\right)\right]\exp\left[-j\frac{n2\pi\Delta}{T}\right].$$

This technique depends on generating narrow enough pulses out of the digital-to-analog converter to produce energy at some $n2\pi/T$ sampling harmonic.

sampling interval) in order to cancel an upper or lower side frequency around some sampling harmonic (see Fig. 12). That is, if the cosine is computed at sampling times nT, then the sine must be computed at times $nT+\Delta$. To cancel the upper side frequency around the nth sampling harmonic $n2\pi/T$, Δ must be equal to $T/4n$, and the two computed samples are summed into the same smoothing filter (at their respective times). If the lower side frequency is to be eliminated, the difference between the two sample trains is used to drive the smoothing filter. This technique follows from the expressions for the spectra of sampled cosine computed at nT and sampled sine computed at $nT+\Delta$ as follows:

$$\text{cosine spectrum} = \frac{1}{2}\sum_{n=-\infty}^{\infty}\left[\delta\left(\omega-\omega o-\frac{n2\pi}{T}\right)+\delta\left(\omega-\omega o-\frac{n2\pi}{T}\right)\right]$$

Phase Control and Other Applications

The phase of the synthesizer output frequency at any computation time depends only on the stored value in the accumulator of Fig. 4 (ignoring any phase distortions introduced by the output smoothing filter). This allows for considerable phase control. In the normal mode of operation a change in frequency would be accomplished by changing only the input frequency word k and leaving unchanged the previous accumulator value. In this way the new frequency would be produced with no phase discontinuity since samples of the new frequency sinusoid would include the last sample of old frequency sinusoid. If on the other hand an arbitrary phase were desired at each change in frequency this means both a new frequency word k and a new accumulator value initial state. An example of such frequency switching would be resetting the accumulator (and phase) to zero whenever a new frequency is generated.

The ease with which phase can be controlled suggests many possible applications. The ability to frequency hop under phase control allows the implementation of new

kinds of coherent communication systems if the channel phase is well behaved.

The synthesizer in the phase continuous mode can be used as a very linear frequency modulator. The modulating signal would drive an analog-to-digital converter (unless it is digital to begin with) whose output would be the digital frequency control word. The use of a large encoding range and a large encoding word would reduce distortion and noise. If the converter worked synchronous to the frequency synthesizer clock, no phase discontinuities could occur and a large deviation signal could be obtained with very high linearity.

Another class of applications is generation of sweep signals of various kinds. If the input frequency control word is incremented at certain fixed time intervals, a continuous stepped frequency waveform is generated. In the limit, this becomes a continuous frequency sweep so long as the sweep signal spectrum is not distorted by the sampling. The use of wired or stored frequency sequences could produce very complex sweep patterns.

Finally, the algorithm itself may be used in certain computational environments to produce digital samples of sine and cosine or complex coefficients for use in discrete Fourier transforms.

Some Design Considerations

A synthesizer of the type described here has several parameters which lead to design specifications. One is the time to produce a sample. This is composed of several elements, the accumulator time, the table look-up time, the multiplication time, and the D–A converter time. These times may be added, in some realizations, or they may be overlapped somewhat depending on the logic implementation. In our limited experience, the D–A time has been the largest element, and it is easily overlapped with other times, so that the time to produce a sample is the D–A time.

The reciprocal of the time to produce a sample is the sampling rate. This must be at least twice the highest frequency produced (or the bandwidth when a synthesizer design is intended to produce intermediate frequency sine waves), but it should be higher in order to simplify the requirements on the output smoothing filter. As discussed earlier there is, for any fixed sampling rate, a tradeoff between the complexity of the output filter required and the maximum frequency obtainable. This tradeoff is only interesting over a range of about 2:1 in maximum frequency, further simplification in the output filter as the sampling rate is increased becomes very small. The output filter also affects the switching time of the synthesizer output. Insofar as the sampling rate chosen affects the filter requirement, it also affects the obtainable switching time. This is an area where further work would be useful.

Another important set of parameters is the number of complex multiplications allowed in composing a sample. This is related to the number of subtables of constants required. With N words of storage, broken into $(n+1)$ subtables and combined by n multiplications (words, storage, multiplies all complex), the number of different complex exponentials (proportional to the number of frequencies which can be obtained) can be shown to be $(N/(n+1))^{n+1}$. By choosing n to maximize the number of exponentials, one can obtain a very large number indeed. For example, if $N=32$, it is quite straightforward to obtain 2^{16} exponentials with seven multiplies. In practice, we expect that it will not be desirable to use so many multiplications and probably one multiply and two subtables will be the most common choice.

The arithmetic precision needed, both in the stored constants and in the complex multiplications, will typically be determined by the capabilities of the output D–A converter. If the D–A converter is capable of resolving k bits, the computation should not attempt to produce numbers accurate to very much more than k bits. Note that the number of different frequencies obtainable is still unlimited, since the technique of using nearest samples is still available. If the sample used differs from the exact sample by less than 2^{-k}, the technique of using the nearest sample will not have any important error associated with it. This error is clearly bounded by $\sin 2\pi 2^{-k} \approx 2\pi 2^{-k}$. This implies using about $(k+3)$ bits of the accumulator in the computation, and using the lower bits only for accumulation to produce finer frequency increments. Many of these considerations become straightforward for particular design requirements.

Conclusion

A unique approach to the problem of frequency synthesis has been presented based on a sampled data realization. The simple factoring algorithm explained allows for very efficient use of storage and simple digital implementation. Finally, the phase properties of the synthesized signal and the ease of phase control allow the device to be used for generation of signals more complex than simple sinusoids.

References

[1] V. E. Van Duzer, "A 0–50 MHz frequency synthesizer with excellent stability, fast switching and fine resolution," *Hewlett-Packard J.*, vol. 15, May 1964, pp. 1–8.

[2] A. Noyes, Jr., "Coherent decade frequency synthesizers," *The Experimenter*, vol. 38, no. 9, Sept. 1964.

[3] E. Renschler and B. Welling, "An integrated circuit phase-locked loop digital frequency synthesizer," Motorola Semiconductor Products, Inc., Application Note 463.

[4] G. C. Gillete, "The digiphase synthesizer," *Frequency Technol.*, Aug. 1969, pp. 25–29.

[5] J. Noordanus, "Frequency synthesizers—A survey of techniques," *IEEE Trans. Commun. Technol.*, vol. COM-17, Apr. 1969, pp. 257–271.

[6] B. Gold and C. M. Rader, *Digital Processing of Signals*. New York: McGraw-Hill, 1969.

[7] A. W. Crooke, "A flexible digital waveform generator for use in matched filtering applications," presented at Arden House Workshop, Jan. 1970.

[8] H. Salinger, "Transients in frequency modulation," *Proc. IRE*, vol. 30, Aug. 1942, pp. 378–383.

Index

More Great Books from LLH Technology Publishing

Digital Signal Processing Demystified

By James D. Broesch

INCLUDES WINDOWS 3.1/95 CD-ROM. A readable and practical introduction to the fundamentals of digital signal processing, including the design of digital filters. The interactive CD-ROM contains a powerful suite of experimental, educational, and design tools. A volume in the Engineering Mentor series.
1-878707-16-7—$49.95

Modeling Engineering Systems

PC-Based Techniques and Design Tools

By Jack W. Lewis

Teaches the fundamentals of math modeling and shows how to simulate any engineering system using a PC spreadsheet. A great hand-holding introduction to automatic control systems, with lots of illustrations and practical design examples. A volume in the Engineering Mentor series. 1-878707-08-6—$25.00

Video Demystified, Second Edition

A handbook for the Digital Engineer

By Keith Jack

INCLUDES WINDOWS/MAC CD-ROM. Completely updated new edition of the "bible" for digital video engineers and programmers. Over 800 pages of hard-to-find design info and video standard specifications. The CD-ROM contains valuable test files for video hardware and software designers.
1-878707-23-X—$59.95

Programming Microcontrollers in C

By Ted Van Sickle

Shows how to fully utilize the C language to exploit the power of the new generation of microcontrollers that doesn't have to be programmed in assembly language. Also contains a great C tutorial for those who need it. Many practical design examples in over 400 pages. 1-878707-14-0—$29.95

Controlling the World with Your PC

By Paul Bergsman

INCLUDES PC DISK. A wealth of circuits and programs that you can use to control the world! Connect to the parallel printer port of your PC and monitor fluid levels, control stepper motors, turn appliances on and off, and much more. The accompanying disk for the PCs contains all the software files in ready-to-use form. All schematics have been fully tested. Great for students, scientists, hobbyists. 1-878707-15-9–$35.00

Bebop to the Boolean Boogie

An Unconventional Guide to Electronics

Fundamentals, Components, and Processes

By Clive "Max" Maxfield

The essential reference on modern electronics, published to rave reviews from engineers, educators, and nontechnical types who need to work with technology. Covers all the basics from analog to digital, bits to bytes, to the latest advanced technologies. 500 pages of essential information presented with wit and style. Worth the price for the glossary alone! 1-878707-22-1—$35.00

The Forrest Mims Engineer's Notebook

By Forrest Mims III

A revised edition of a classic by the world's bestselling electronics author. Includes hundreds of circuits built from integrated circuits and other parts available from convenient sources. Also contains special tips on troubleshooting, circuit construction, and modifications. 1-878707-03-5—$19.95

Fibre Channel, Second Edition

Connection to the Future

By the Fibre Channel Association

A concise guide to the fundamentals of the popular ANSI Fibre Channel standard for high-speed computer interconnection. If wading through the entire Fibre Channel standard document seems too daunting, this is the book for you! It explains the applications, structure, features and terminology in plain English.
1-878707-45-0—$16.95

The Integrated Circuit Hobbyist's Handbook

By Thomas R. Powers

This practical circuit collection belongs on every electronics hobbyist's shelf! Covers the major types of ICs, and provides complete detail and theory about their operation. Also includes directions for building electronic devices that make use of ICs, as well as a massive listing of the most popular ICs in the world, thoroughly indexed by application. 1-878707-12-4—$19.95

The Art of Science: A Practical Guide to Experiments, Observations, and Handling Data

By Joseph J. Carr

A friendly and readable guide to the "nuts and bolts" of scientific inquiry. Examines the scientific process in detail, covering experimental technique, error analysis, statistics, graphing, and much more. A great reference for students, engineers, scientists.
1-878707-05-1—$19.95

Credit card holders can order by calling
1-800-247-6553
(1-419-281-1802 outside U.S.)
These titles are also available at your local bookstore.

VISIT OUR WEBSITE AT: www.LLH-Publishing.com!

This is a blank page.