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 lends 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

$$e^{j2\pi fnT} = \exp\left[j\frac{2\pi}{N}nk\right].$$

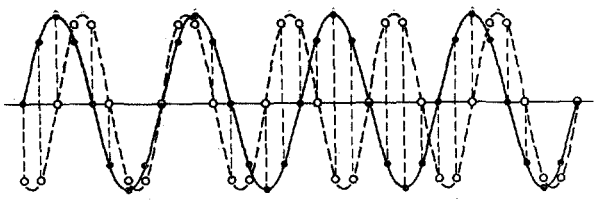


Fig. 1. Digital frequency determination.

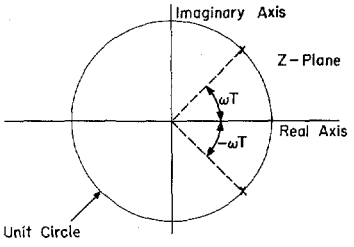


Fig. 2. Pole position of the digital recursion for sinusoidal samples.

$$y_n = (2\cos\omega T)\,y_{n-1} - y_{n-2}$$
$$y_0 = \cos\phi$$
$$y_{-1} = \cos[\phi - \omega T].$$

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

$$\exp[jo], \quad \exp\left[j\frac{2\pi}{N}\right], \quad \exp\left[j\frac{2\pi}{N}2\right], \cdots, \exp\left[j\frac{2\pi}{N}(N-1)\right].$$

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.

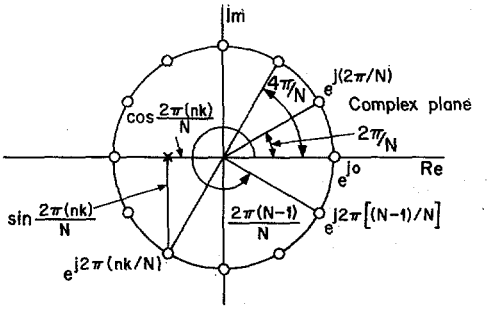


Fig. 3. Equispaced samples on the unit circle.

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 $0 \leq Y \leq N-1$, 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 $0 \leq Y \leq 2^b-1$ 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

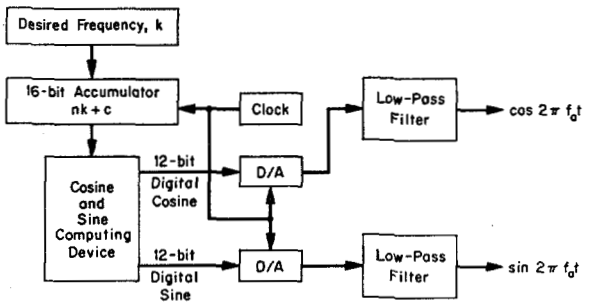


Fig. 4. Synthesizer block diagram.

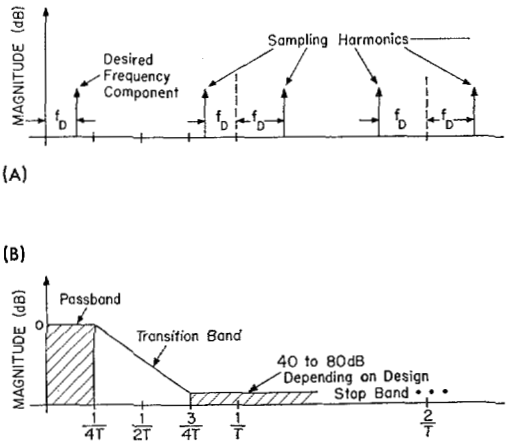


Fig. 5. (A) Sampled sinusoid spectrum. (B) Smoothing filter response.

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^{15} 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 $2\pi/2^{15}$ (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

$$\left(\exp\left[j\frac{2\pi}{2^{15}}f\right]\right)\left(\exp\left[j\frac{2\pi}{2^{15}}2^7e\right]\right) = \left(\exp\left[j\frac{2\pi}{2^{15}}f\right]\right)\left(\exp\left[j\frac{2\pi}{2^8}e\right]\right).$$

The computation of the complex exponential is then reduced to two table look-ups $\exp[j(2\pi/2^{15})f]$, $\exp[j(2\pi/2^8)e]$ 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

$$\begin{aligned} \sin(x + y) &= \sin x \cos y + \cos x \sin y \\ \cos(x + y) &= \cos x \cos y - \sin x \sin y \end{aligned}$$

with

$$x = (2\pi/2^8)e, \qquad y = (2\pi/2^{15})f$$

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.

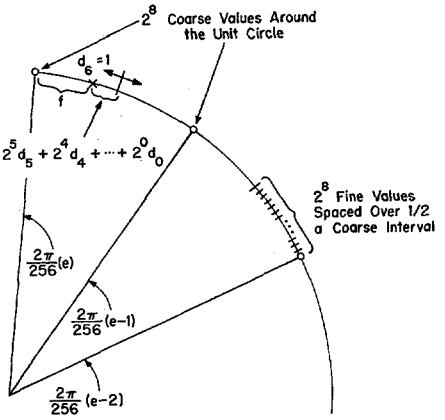


Fig. 6.

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.

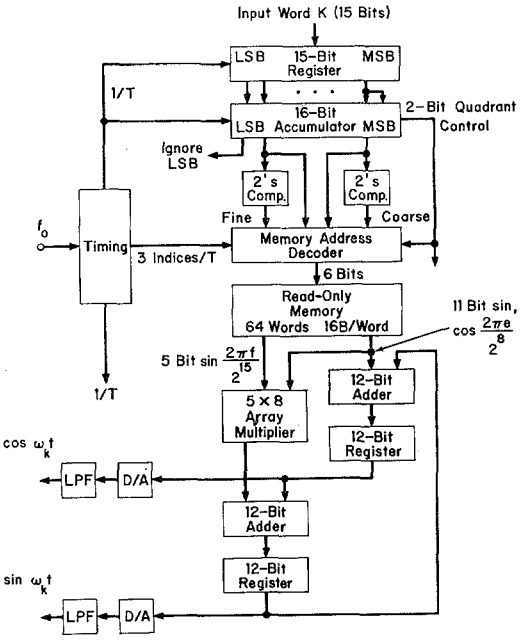


Fig. 7.

Output Noise and Time Response

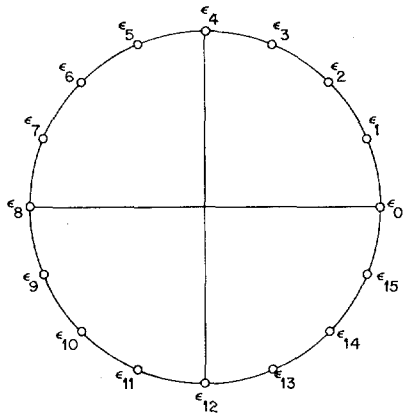
The synthesizer noise output consists of three separate effects. The first contribution results from truncation or roundoff in computation of the sample value. The second effect is from the output digital to analog converter. Sampling harmonics passing through the smoothing filter contribute the third 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 $k=2^a$, 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)	• T/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 1 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

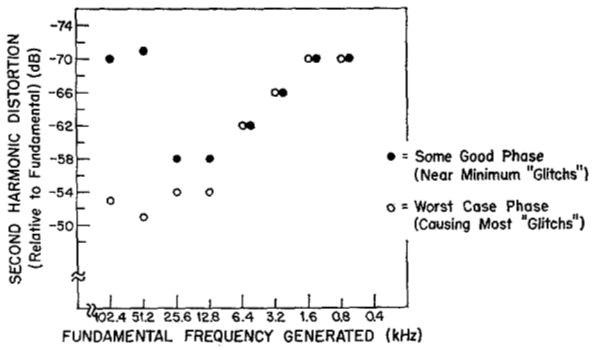


Fig. 9.

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 ($\sim 1/4T$). 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 (same

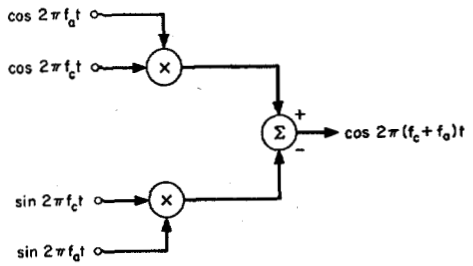
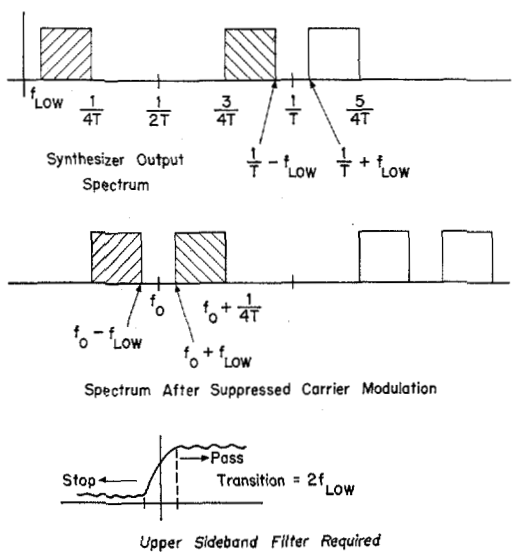
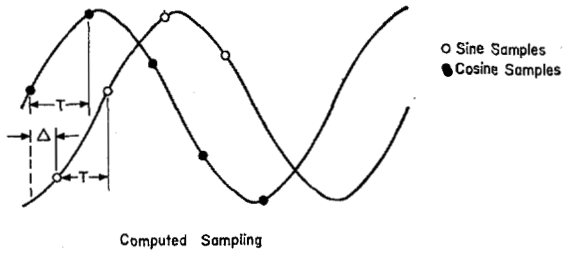


Fig. 10. Single sideband bandwidth doubling and frequency translation.

Fig. 11.





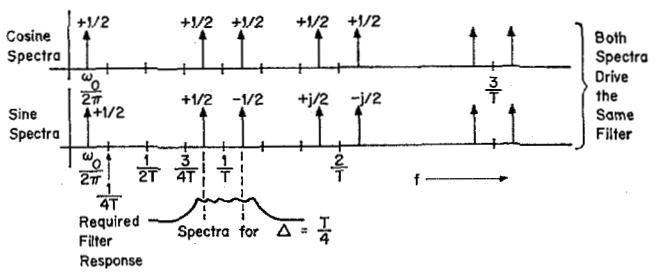


Fig. 12.

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:

$$\text{cosine spectrum} = \frac{1}{2}\sum_{n=-\infty}^{\infty}\left[\delta\left(\omega - \omega o - \frac{n2\pi}{T}\right) + \delta\left(\omega - \omega o - \frac{n2\pi}{T}\right)\right]$$

$$\text{sine spectrum} = \frac{1}{2}\sum_{n=-\infty}^{\infty}\left[\delta\left(\omega - \omega o - \frac{n2\pi}{T}\right) - \delta\left(\omega + \omega o - \frac{n2\pi}{T}\right)\right]\exp\left[-j\frac{n2\pi\Delta}{T}\right].$$

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.

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.

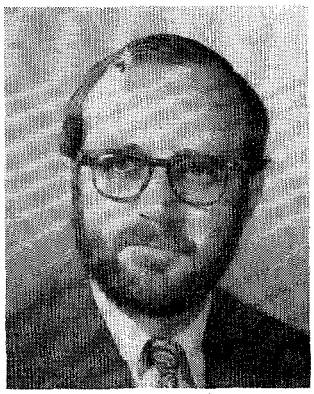
[4] G. C. Gillete, "The digiphase synthesizer," *Frequency Technol.*, Aug. 1969, pp. 25–29.

[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.

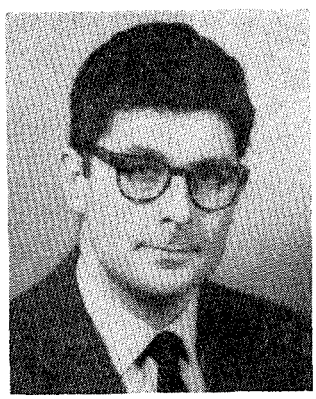
[7] A. W. Crooke, "A flexible digital waveform generator for use in matched filtering applications," presented at Arden House Workshop, Jan. 1970.

[8] H. Salinger, "Transients in frequency modulation," *Proc. IRE*, vol. 30, Aug. 1942, pp. 378–383.



Joseph Tierney (S'54–M'57) was born in New York, N. Y., on January 14, 1934. He received the S.B. and S.M. degrees in electrical engineering from the Massachusetts Institute of Technology, Cambridge, in 1956.

At M.I.T. he was a co-op student at the Bell Telephone Laboratories, and spent the year 1955–1956 as a Research Assistant at the M.I.T. Research Laboratory of Electronics. He has been with the M.I.T. Lincoln Laboratory, Lexington, Mass., since 1961 working in the areas of speech compression and satellite communications at both a systems and circuits level. For the past several years he has been involved in the design of digital signal processing devices.



Charles M. Rader (S'59–M'62) was born in Brooklyn, N. Y., on June 20, 1939. He received the B.E.E. and M.E.E. degrees from the Polytechnic Institute of Brooklyn, Brooklyn, N. Y., in 1960 and 1961, respectively.

He has been with the M.I.T. Lincoln Laboratory, Lexington, Mass., since 1961, where he has worked on techniques for speech bandwidth compression and computer simulation.

Mr. Rader is a member of Eta Kappa Nu, Tau Beta Pi, and the Acoustical Society of America.

Bernard Gold (M'49–SM'67) was born in New York, N. Y., on March 31, 1923. He received the B.S.E.E. degree from the City College of New York, N. Y., in 1944, and the Ph.D. degree in electrical engineering from the Polytechnic Institute of Brooklyn, Brooklyn, N. Y., in 1948.

From 1948 to 1953 he worked at the Avion Instrument Corp. and Hughes Aircraft Company on radar and missile system electronics. He has been with the M.I.T. Lincoln Laboratory, Lexington, Mass., since 1953, working on pattern recognition, noise theory, speech bandwidth compression, and digital signal processing techniques. In 1954–1955 he was a Research Fulbright Fellow in Italy, and in 1965–1966 he was a Visiting Professor of Electrical Engineering at M.I.T., Cambridge, Mass. He is the author of about 30 papers and coauthor of the book *Digital Processing of Signals.*

Dr. Gold is a member of the Acoustical Society of America, the International Scientific Radio Union, and the American Association for the Advancement of Science.