

CONTRASTING SINEWAVE GENERATION IN THE ANALOG AND DIGITAL CASES

-by Bernie Hutchins

OBTAINING SINUSOIDAL WAVEFORMS

Nothing is more fundamental to true (non-sampling) music sound synthesis than oscillators. Very early electronic music studios had banks of sinewave oscillators in an additive arrangement. The so-called "subtractive" or voltage-controlled synthesis techniques pioneered by Robert Moog and others used oscillators that produced more complex waveforms that were then filtered down (subtracted) to produce desired, time varying spectra. Digital synthesis often involves digital sinewave oscillators which can be both trivial and subtle.

Probably the most familiar oscillator visible in everyday life is the pendulum (Fig. 1). A pendulum swings back and forth periodically. It is a familiar problem in elementary physics to show that the motion is the solution to a second-order differential equation. (Even this simple system requires an interesting approximation). The pendulum (or better still, a mass and spring system) provides sinusoidal or "simple harmonic motion."

But, we may well protest, a pendulum does not keep oscillating. It slows down and then eventually stops. That is, unless you supply additional energy at appropriate times. This is because there is friction in real cases and this dissipates energy. The oscillator is kept going by making up for the lost energy, usually by replacing the energy lost once each cycle. This is how a pendulum clock is kept going, by providing energy from a wound spring, the potential energy of a weight, or perhaps a

battery. It is the same way that a child on a swing is kept contented by a gentle push upon each return to the parent.

Accordingly, obtaining sinusoidal waveforms as the natural response of second-order systems (supplied with a little extra energy) offers a relatively simple approach for both analog and digital synthesis. Surprisingly, this is very seldom used in analog synthesis, and only sparingly in digital synthesis. One reason for this in the analog case is the problem of obtaining a stable oscillation. Another is that the sine wave, used by itself rather than in combinations of different frequencies, is not a particularly useful musical waveform (lacking harmonics, it doesn't have anything to filter, and is somewhat flute-like by itself). A third reason is that a sinewave is extremely difficult to waveshape into either a sawtooth or a triangle (a square wave is trivially obtained with a comparator).

More common is a "computational" approach. Here time advances and we compute the waveform as a function of time. In analog synthesis however, it is not the usual practice to compute sinusoidal functions of advancing time. Instead, we compute linear functions of advancing time - voltage proportional to time itself. These are ramps. Linear ramps are generated by driving constant currents into capacitors. Ramps are nice; but they run off scale and are not periodic. To make them periodic, they must either be reset when they try to exceed a certain limit (a sawtooth based oscillator), or turned around when they try to exceed certain amplitude limits (a triangle based oscillator)(Fig. 2). The triangle and sawtooth waveforms are fairly easily shaped, either one into the other. A square wave is obtained from either the saw or the triangle, and a sinewave approximation is obtained by shaping the triangle with a non-linear device (Fig. 3).

NETWORKS AND COMPLEX PLANES

While a pendulum or similar mechanical device provides a very useful intuitive grasp of analog sinewave oscillators, we need to get somewhat closer to electronic versions of these

oscillators. There are a good number of analog sinewave oscillator networks (circuits), but few are used in synthesizers. Perhaps the closest circuit that is commonly found in a synthesizer that resembles a sinewave oscillator is a state-variable filter. Fortuitously, this same network also models the generic form of a sinewave oscillator. Fig. 4A shows an interconnection of two analog integrators and a summer, which many will recognize as a state-variable filter (the exact arrangement and notation has been changed slightly to better emphasize the strong resemblance to a second order digital network).

The particular input/output arrangement shown is for a bandpass response. The transfer function is:

$$T_B(s) = V_B(s)/V_{in}(s) = s / [s^2 + (1/Q)s + 1] \quad (1)$$

From this, it is clear that if $Q \rightarrow \infty$ the poles of the network will occur when:

$$s^2 + 1 = 0 \quad (2)$$

which has solutions $s = \pm j$, that is, poles on the $j\omega$ -axis (Fig. 4B), and accordingly, a sinewave oscillator results. In an ideal case, making this network oscillate would seem to be just a matter of removing the middle feedback link ($-1/Q$). Accordingly, when this link is non-zero, it is seen to provide damping.

{ Persons familiar with analog VCF's are perhaps familiar with the Q of some early versions of the state-variable filter (multi-mode) going to infinity at high frequencies, resulting in filter oscillation (although not with a waveform as nice as a sine wave). In an ordinary, non-voltage controlled state-variable filter, pulling out the $1/Q$ path generally did not result in instability, but just a very high- Q filter. With the control elements in there (usually an OTA of the CA3080 type), there was enough additional phase shift to de-stabilize the filter, and this phase shift increases with higher frequencies. This is why you will often see small capacitors, say about 10 pf, across the input resistors to these

control stages (Fig. 5). This adds a zeros to the loop which compensates for the excess phase. Reference [1] }

The digital oscillator network [2,3] shown in Fig. 4C is drawn pretty much as a standard second-order digital filter with two feedbacks. It resembles the analog oscillator quite closely. Here we have two delays and a summer. Note that the feedback from the bottom of both networks is a -1. For the analog sinewave oscillator, the upper feedback is $-1/Q=0$ while for the digital oscillator, it is $A_1=2 \cos(\omega_0 T)$ where ω_0 is the desired frequency of oscillation and $T = 1/f_s$ is the sampling period, with f_s being the sampling frequency. This network has a transfer function:

$$H(z) = z^{-1} / (1 - A_1 z^{-1} - A_0 z^{-2}) \quad (3)$$

which has poles (for the complex conjugate case of interest) at:

$$z_{p1}, z_{p2} = A_1/2 \pm j(-A_1^2 - 4A_0)^{1/2} / 2 \quad (4)$$

which, for $A_0=-1$, are easily seen to be at:

$$z_{p1}, z_{p2} = \cos(\omega_0 T) \pm j \sin(\omega_0 T) \quad (5)$$

So the poles are on the unit circle at an angle corresponding to the desired oscillation frequency (Fig 4D). For example, if the sampling frequency is 8000 Hz and the poles are at an angle of 45° with respect to the positive real axis, the frequency of oscillation (in Hz) would be 1000.

A CONUNDRUM ?

As mentioned, it is extremely difficult, with analog components, to put poles exactly on the $j\omega$ -axis and keep them there. With the digital sinewave oscillator, we need to put poles exactly on the unit circle and keep them there. Is this possible?

Well, keeping them there is not a problem, as there is no drift of any sort associated with numbers. Further, equation (5), coming from equation (3) with $A_0=-1$ seems to indicate that the pole radius is exactly one. Thus we understand the successful oscillation as a consequence of being able to set the bottom multiplier, A_0 , to exactly -1.

Now for the problem. While it is possible to multiply by -1 with no roundoff (changing a sign bit), it is not possible to multiply by the general values of the coefficient A_1 without roundoff. While it seems that A_1 only changes the frequency of oscillation, and has nothing to do with whether the oscillation is stable, we need to note that the roundoff error following A_1 is summed with the feedback through A_0 and is then placed in the top of the delay line. How does the oscillator "know" that the roundoff came from the A_1 multiplier and not from the A_0 multiplier!

Clearly, the oscillator does not "know" so we are led to suppose that somehow, the roundoff must average out. But, this is probably not a correct view, since we have the empirical result that these oscillators always work, and we would expect averaging to be inexact in at least some cases. It would seem that some type of subtle negative feedback is operating here.

COUPLED FORM DIGITAL OSCILLATOR

So it might seem that digital sinewave oscillators just "decide" to work despite some questions about roundoff. But, if we next consider the so-called "coupled form" for the digital oscillator (Fig. 6), we find this refusing to work. Realized with floating point arithmetic, it either dies or blows up. This we can understand in terms of our inability to put the poles exactly on the unit circle [specifying $\cos(\theta)$ and $\sin(\theta)$] with a finite number of bits. Realized with fixed point arithmetic for the signal, we find to our surprise that the oscillator may stabilize, but not with an amplitude corresponding to the initial state. Not untypically, it

may decay to only 10% of the initial amplitude before locking on. All of this suggests a limit-cycle behavior [4].

COMPUTING SINEWAVES WITH LOOK-UP TABLES

In an analog function generator or VCO, we saw that a computational approach rather than a second-order system approach was used to obtain sinewaves. A similar procedure in the digital case involves the look-up table. Here, time advances (driven by the digital system's clock) and in pace with it, we know the appropriate phase for a given frequency. This allows us to simply look up the corresponding samples from a table stored in memory. The same table (often only a single quadrant is used) can be used for as many sinewaves as we care to consider.

SOME SUMMARY COMMENTS

While sinewaves are clearly fundamental to music synthesis, analog sinewave oscillators are not especially easy to generate directly, or to work with; nor are they especially useful in isolation. Digital generation of sinewaves from a second-order network can be, however, very reliable. Yet one must still be very careful to choose an appropriate network, and to be prepared for subtle quirks in their performance. These isolated sinewaves are not especially useful either, but the potential for simultaneous generation of multiple sinewaves (by cutting and pasting lines of code) should not be overlooked.

REFERENCES

- [1] B. Hutchins, "Compensation of Linear Circuit Blocks for the Effects of Real Operational Amplifiers," Electronotes, Vol. 15, No. 164-167 (Special Issue F), July 1986, pp 47-65

- [2] B. Hutchins, "Design Considerations for Digital Sinewave Oscillators," *Electronotes Application Note No. 299*, Sept/Oct. 1987
- [3] B. Hutchins, "A View of a Digital Sinewave Oscillator ," *Electronotes Application Note No. 309*, Sept. 1990
- [4] B. Hutchins, "Frequency Errors and Distortion in Digital Sinewave Oscillators," *Electronotes*, Vol. 18, No. 186, Sept. 1995, pp 3-50

Questions Submitted About this Tutorial:

Q: I'm wondering if you have any ideas on the following: If given a sine and square of frequency F_1 (in phase), can you use these to generate sawtooth & triangle??

A: The short answer is no. The longer answer is that it is possible with certain extreme efforts or with some limitations. The problem is one of not having enough information available in the first waveform to tell you where you are in terms of phase. A square wave is either high or low, and knowing this tells you nothing about how far you are along in the waveform. [On the other hand, if you know the value of a sawtooth, you know exactly where you are in the waveform. A triangle also tells you where you are if you also keep track of the derivative (which is why an integrator-Schmitt-trigger oscillator is so useful).] You can integrate a square into a triangle, but of course, the amplitude of the triangle is frequency dependent.

Now, a sine wave could give you back all other waveforms if you were to somehow inverse shape it to a triangle, and differentiate that triangle. I can't think of a way of getting a precision inverse shaping directly, but putting a triangle-to-sine shaper in a feedback loop might be a possibility.

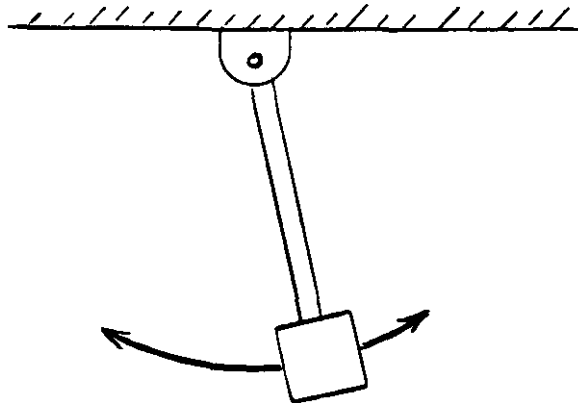


Fig. 1 A Simple Pendulum as a Simple Harmonic Oscillator

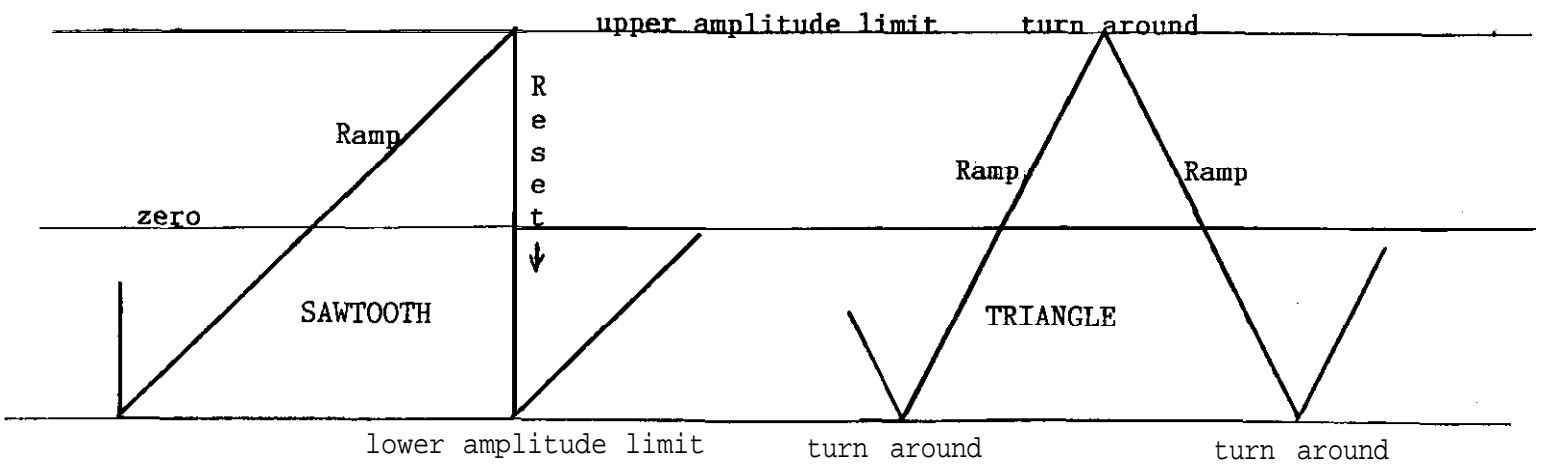


Fig. 2 Sawtooth and Triangle Obtained from Ramps

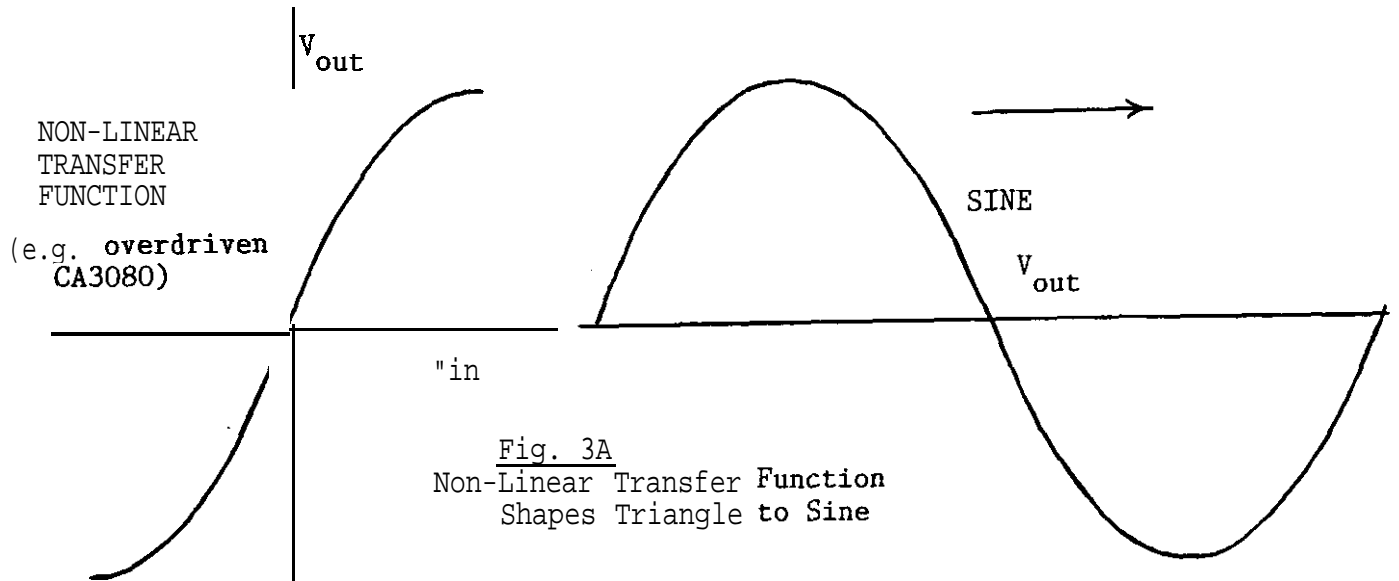


Fig. 3A
Non-Linear Transfer Function
Shapes Triangle to Sine

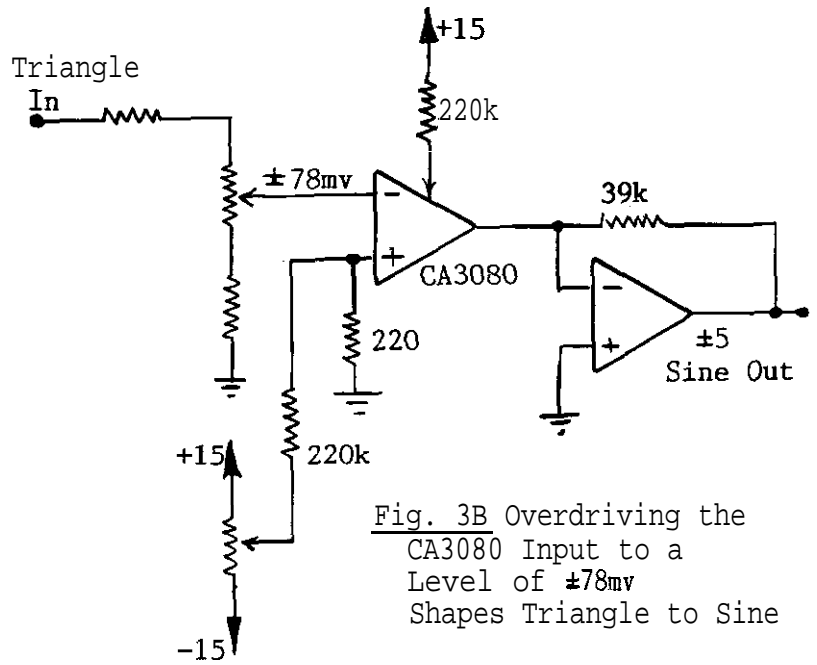
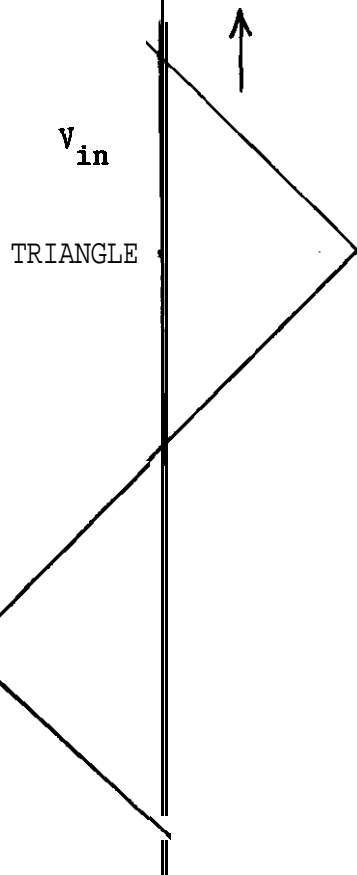
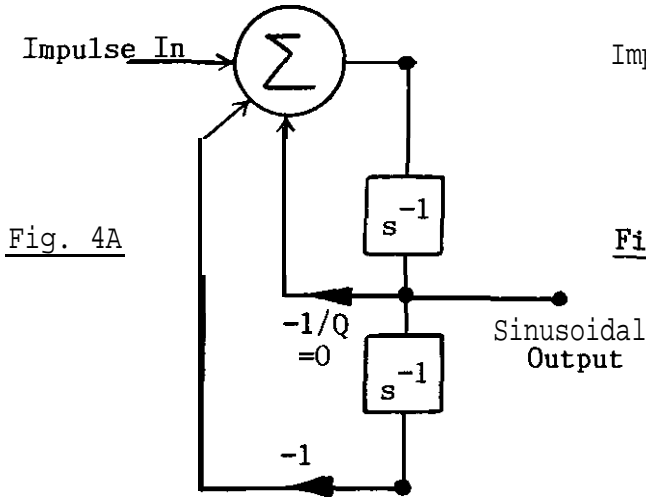
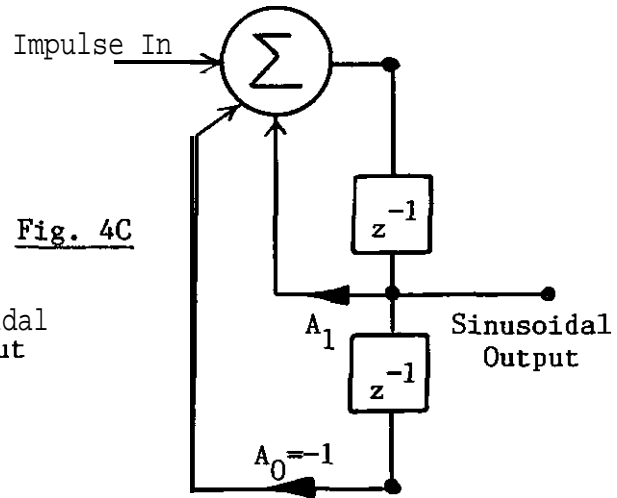


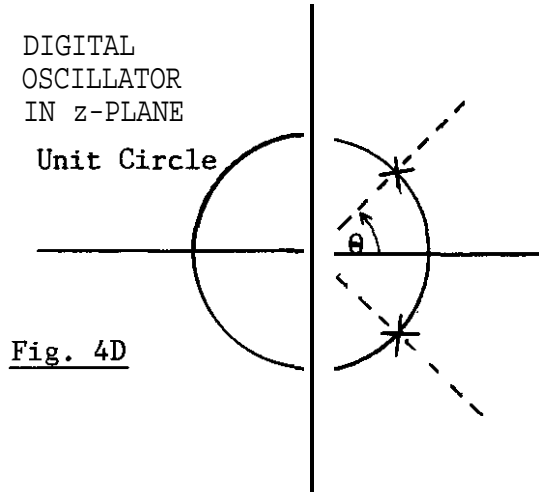
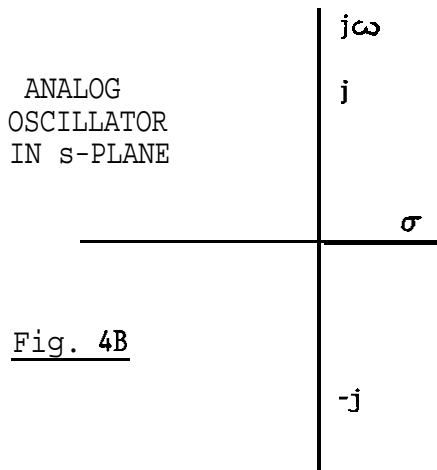
Fig. 3B Overdriving the
CA3080 Input to a
Level of $\pm 78\text{mv}$
Shapes Triangle to Sine



ANALOG SINEWAVE OSCILLATOR



DIGITAL SINEWAVE OSCILLATOR



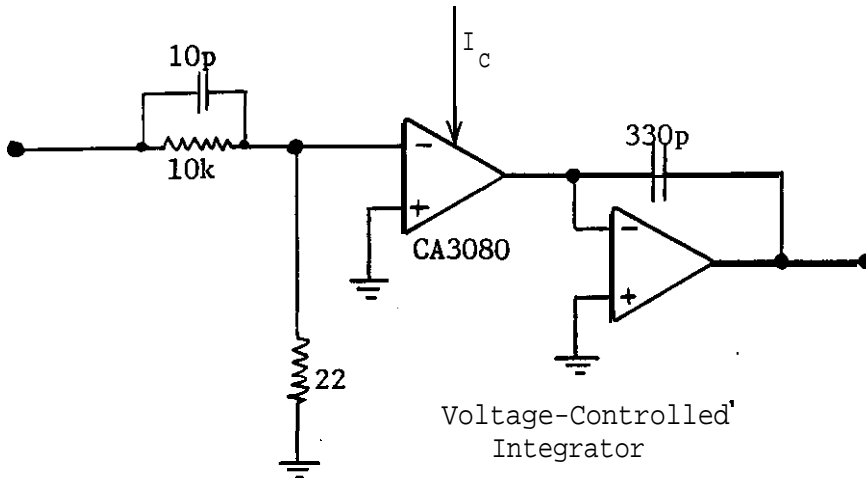


Fig. 5

10p Capacitor
Compensates VCF Stage
to Prevent Oscillation

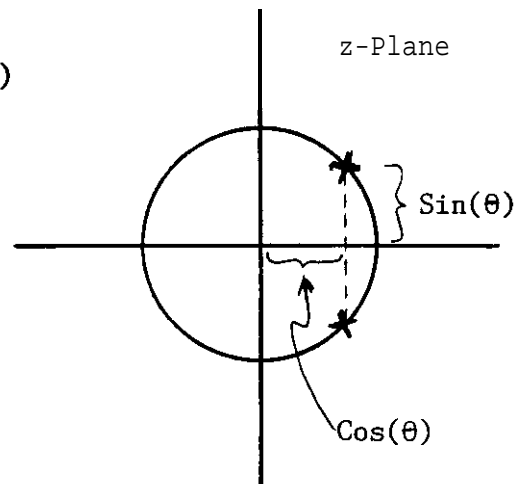
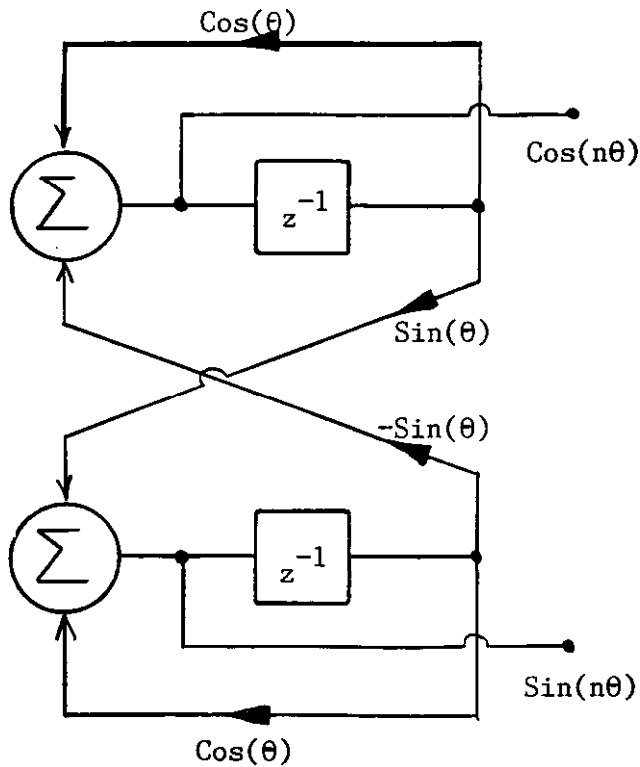


Fig. 6 "Coupled Form"
of Digital Sine Wave
Oscillator has
Implementation Problems