Fundamentals of Phase Locked Loops (PLLs)

FUNDAMENTAL PHASE LOCKED LOOP ARCHITECTURE

A phase-locked loop is a feedback system combining a voltage controlled oscillator (VCO) and a phase comparator so connected that the oscillator maintains a constant phase angle relative to a reference signal. Phase-locked loops can be used, for example, to generate stable output high frequency signals from a fixed low-frequency signal.

Figure 1A shows the basic model for a PLL. The PLL can be analyzed as a negative feedback system using Laplace Transform theory with a forward gain term, G(s), and a feedback term, H(s), as shown in Figure 1B. The usual equations for a negative feedback system apply.

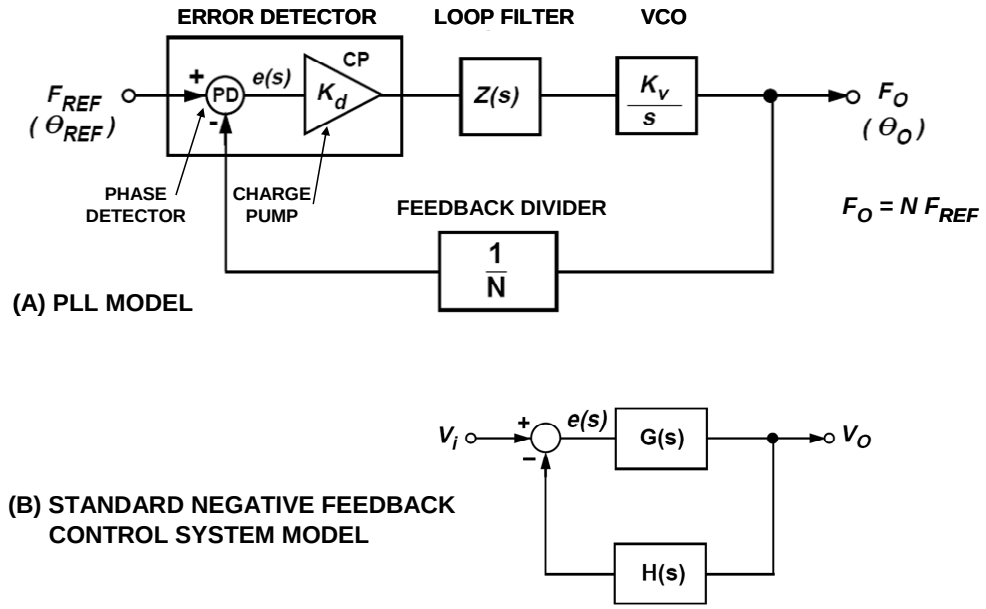


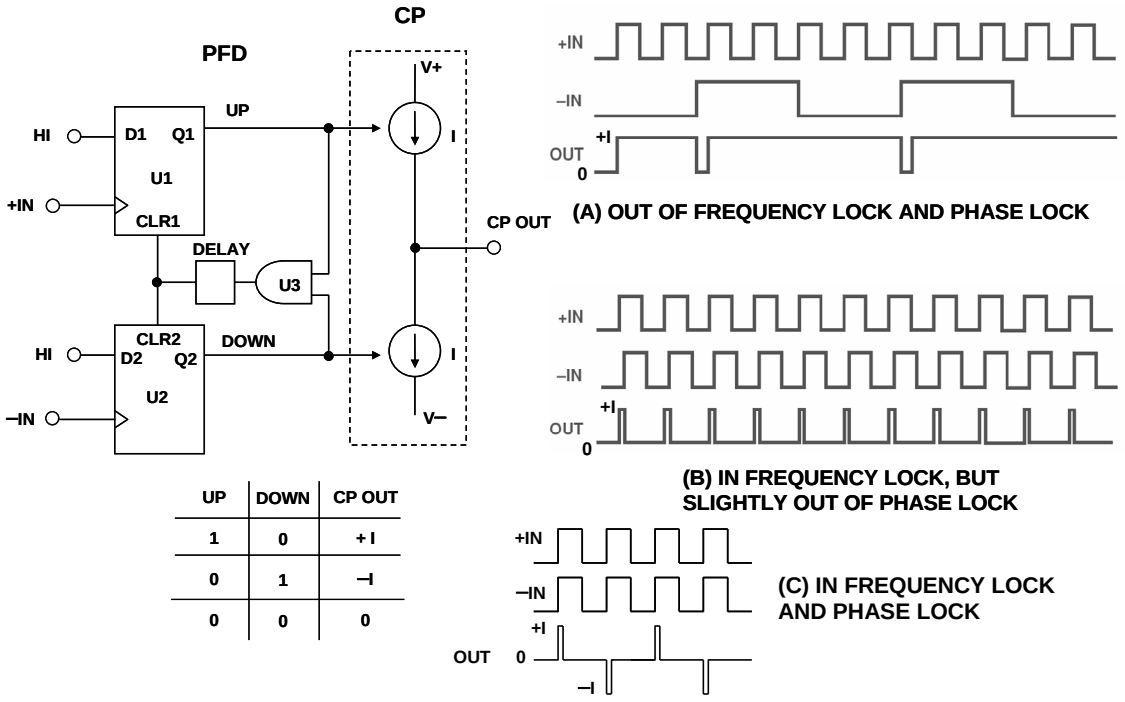
Figure 1: Basic Phase Locked Loop (PLL) Model

The basic blocks of the PLL are the *Error Detector* (composed of a *phase frequency detector* and a *charge pump*), *Loop Filter*, *VCO*, and a *Feedback Divider*. Negative feedback forces the error signal, e(s), to approach zero at which point the feedback divider output and the reference frequency are in phase and frequency lock, and $F_O = N_{FREF}$.

Referring to Figure 1, a system for using a PLL to generate higher frequencies than the input, the VCO oscillates at an angular frequency of ω_O. A portion of this signal is fed back to the error detector, via a frequency divider with a ratio 1/N. This divided down frequency is fed to one input of the error detector. The other input in this example is a fixed reference signal. The error detector compares the signals at both inputs. When the two signal inputs are equal in phase and frequency, the error will be constant and the loop is said to be in a "locked" condition.

PHASE FREQUENCY DETECTOR (PFD)

Figure 2 shows a popular implementation of a *Phase Frequency Detector* (PFD), basically consisting of two D-type flip flops. One Q output enables a positive current source; and the other Q output enables a negative current source. Assuming that, in this design, the D-type flip flop is positive-edge triggered, the possible states are shown in the logic table.



UP	DOWN	CP OUT
1	0	+ I
0	1	–I
0	0	0

Figure 2: Phase/Frequency Detector (PFD) Driving Charge Pump (CP)

Consider now how the circuit behaves if the system is out of lock and the frequency at +IN is much higher than the frequency at –IN, as shown in Figure 2A. Since the frequency at +IN is much higher than that at –IN, the UP output spends most of its time in the high state. The first rising edge on +IN sends the output high and this is maintained until the first rising edge occurs on –IN. In a practical system this means that the output, and thus the input to the VCO, is driven higher, resulting in an increase in frequency at –IN. This is exactly what is desired. If the frequency on +IN were much lower than on –IN, the opposite effect would occur. The output at OUT would spend most of its time in the low condition. This would have the effect of driving the VCO in the negative direction and again bring the frequency at –IN much closer to that at +IN, to approach the locked condition.

Figure 2B shows the waveforms when the inputs are frequency-locked and close to phase-lock. Since +IN is leading –IN, the output is a series of positive current pulses. These pulses will tend to drive the VCO so that the –IN signal become phase-aligned with that on +IN. When this occurs, if there were no delay element between U3 and the CLR inputs of U1 and U2, it would be possible for the output to be in high-impedance mode, producing neither positive nor negative current pulses. This would not be a good situation.

The VCO would drift until a significant phase error developed and started producing either positive or negative current pulses once again. Over a relatively long period of time, the effect of this cycling would be for the output of the charge pump to be modulated by a signal that is a sub-harmonic of the PFD input reference frequency. Since this could be a low frequency signal, it would not be attenuated by the loop filter and would result in very significant spurs in the VCO output spectrum, a phenomenon known as the "backlash" or "dead zone" effect.

The delay element between the output of U3 and the CLR inputs of U1 and U2 ensures that it does not happen. With the delay element, even when the +IN and –IN are perfectly phase-aligned, there will still be a current pulse generated at the charge pump output as shown in Figure 2C. The duration of this delay is equal to the delay inserted at the output of U3 and is known as the anti-backlash pulse width.

Note that if the +IN frequency is lower than the −IN frequency and/or the +IN phase lags the −IN phase, then the output of the charge pump will be a series of negative current pulses—the reverse of the condition shown in (A) and (B) in Figure 2.

PRESCALERS

In the classical Integer-N synthesizer, the resolution of the output frequency is determined by the reference frequency applied to the phase detector. So, for example, if 200 kHz spacing is required (as in GSM phones), then the reference frequency must be 200 kHz. However, getting a stable 200 kHz frequency source is not easy. A sensible approach is to take a good crystal-based high frequency source and divide it down. For example, the desired frequency spacing could be achieved by starting with a 10 MHz frequency reference and dividing it down by 50. This approach is shown in Figure 3A.

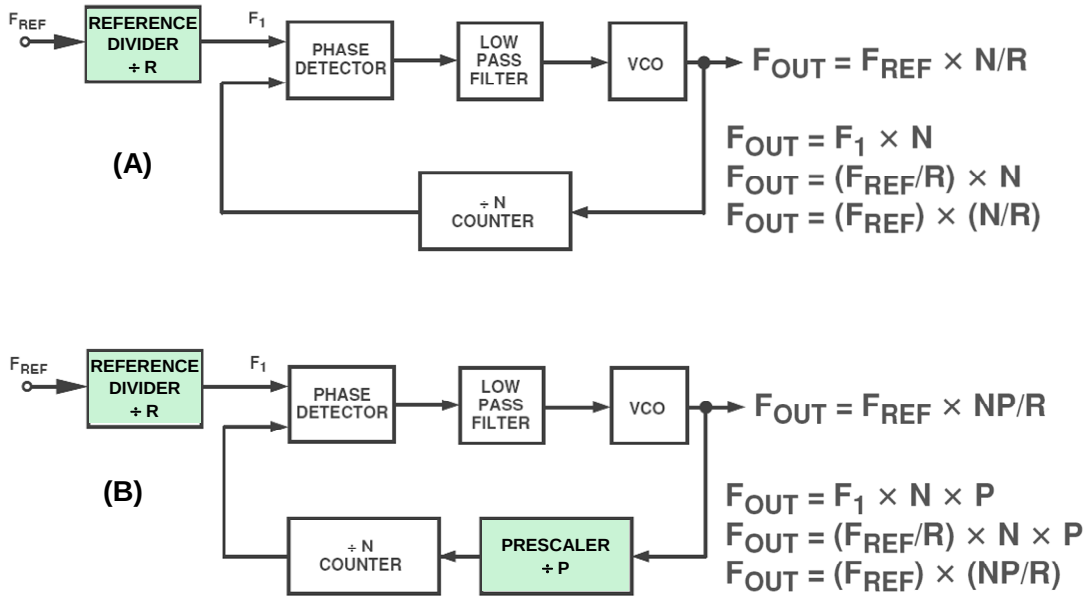


Figure 3: Adding an Input Reference Divider and a Prescaler to the Basic PLL

The "N counter," also known as the N divider, is the programmable element that sets the relationship between the input and output frequencies in the PLL. The complexity of the N counter has grown over the years. In addition to a straightforward N counter, it has evolved to include a *prescaler*, which can have a *dual modulus*. This structure has grown as a solution to the problems inherent in using the basic divide-by-N structure to feed back to the phase detector when very high-frequency outputs are required. For example, let's assume that a 900 MHz output is required with 10 Hz spacing. A 10 MHz reference frequency might be used, with the R-Divider set at 1000. Then, the N-value in the feedback would need to be of the order of 90,000. This would mean at least a 17-bit counter capable of dealing with an input frequency of 900 MHz. To handle this range, it makes sense to precede the programmable counter with a fixed counter element to bring the very high input frequency down to a range at which standard CMOS will operate. This counter, called a *prescaler*, is shown in Figure 3B.

However, note that using a standard prescaler as shown reduces the system resolution to F1×P. This issue can be addressed by using a dual-modulus prescaler which has the advantages of a standard prescaler, but without loss of resolution. A dual-modulus prescaler is a counter whose division ratio can be switched from one value to another by an external control signal. It's use is described shown in Figure 4.

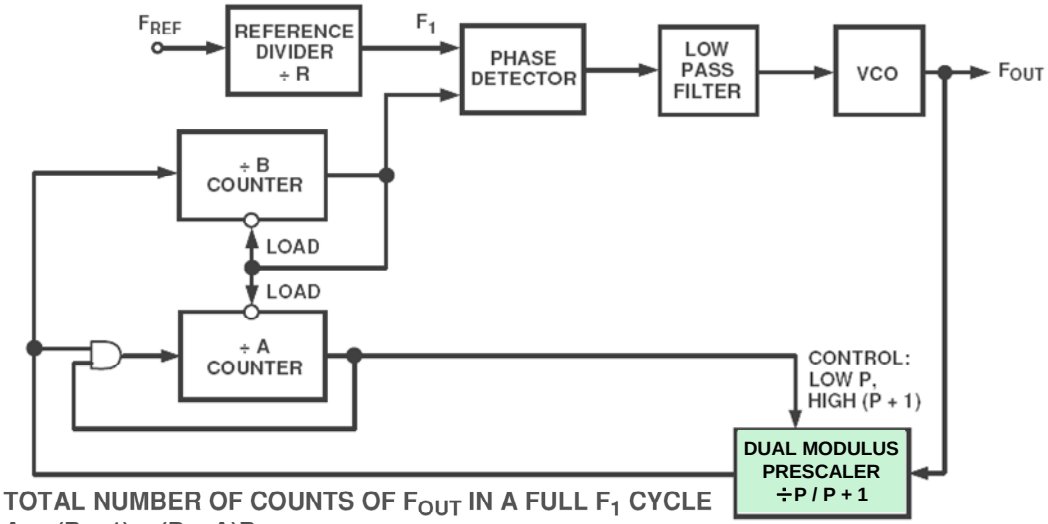


Figure 4: Adding a Dual Modulus Prescaler to the PLL

By using the dual-modulus prescaler with an A and B counter, one can still maintain output resolution of F1. However, the following conditions must be met:

1. The output signals of both counters are High if the counters have not timed out.
2. When the B counter times out, its output goes Low, and it immediately loads both counters to their preset values.

3. The value loaded to the B counter must always be greater than that loaded to the A counter.

Assume that the B counter has just timed out and both counters have been reloaded with the values A and B. Let's find the number of VCO cycles necessary to get to the same state again.

As long as the A counter has not timed out, the prescaler is dividing down by P + 1. So, both the A and B counters will count down by 1 every time the prescaler counts (P + 1) VCO cycles. This means the A counter will time out after ((P + 1) × A) VCO cycles.

At this point the prescaler is switched to divide-by-P. It is also possible to say that at this time the B counter still has (B – A) cycles to go before it times out. How long will it take to do this: ((B – A) × P).

The system is now back to the initial condition where we started.

The total number of VCO cycles needed for this to happen is :

$$\begin{aligned} N &= [A \times (P + 1)] + [(B - A) \times P] \\ &= AP + A + BP - AP \\ &= BP + A. \end{aligned}$$

Therefore, $F_{OUT} = (F_{REF}/R) \times (BP + A)$, as in Figure 4.

There are many specifications to consider when designing a PLL. The input RF frequency range and the channel spacing determine the value of the R and N counter and the prescaler parameters.

The loop bandwidth determines the frequency and phase lock time. Since the PLL is a negative feedback system, phase margin and stability issues must be considered.
Spectral purity of the PLL output is specified by the phase noise and the level of the reference-related spurs.

Many of these parameters are interactive; for instance, lower values of loop bandwidth lead to reduced levels of phase noise and reference spurs, but at the expense of longer lock times and less phase margin.

Because of the many tradeoffs involved, the use of a PLL design program such as the Analog Devices' ADIsimPLL™ allows these tradeoffs to be evaluated and the various parameters adjusted to fit the required specifications. The program not only assists in the theoretical design, but also aids in parts selection and determines component values.

OSCILLATOR/PLL PHASE NOISE

A PLL is a type of oscillator, and in any oscillator design, frequency stability is of critical importance. We are interested in both long-term and short-term stability. Long-term frequency

stability is concerned with how the output signal varies over a long period of time (hours, days, or months). It is usually specified as the ratio, Δf/f for a given period of time, expressed as a percentage or in dB.

Short-term stability, on the other hand, is concerned with variations that occur over a period of seconds or less. These variations can be random or periodic. A spectrum analyzer can be used to examine the short-term stability of a signal. Figure 5 shows a typical spectrum, with random and discrete frequency components causing a broad skirt and spurious peaks.

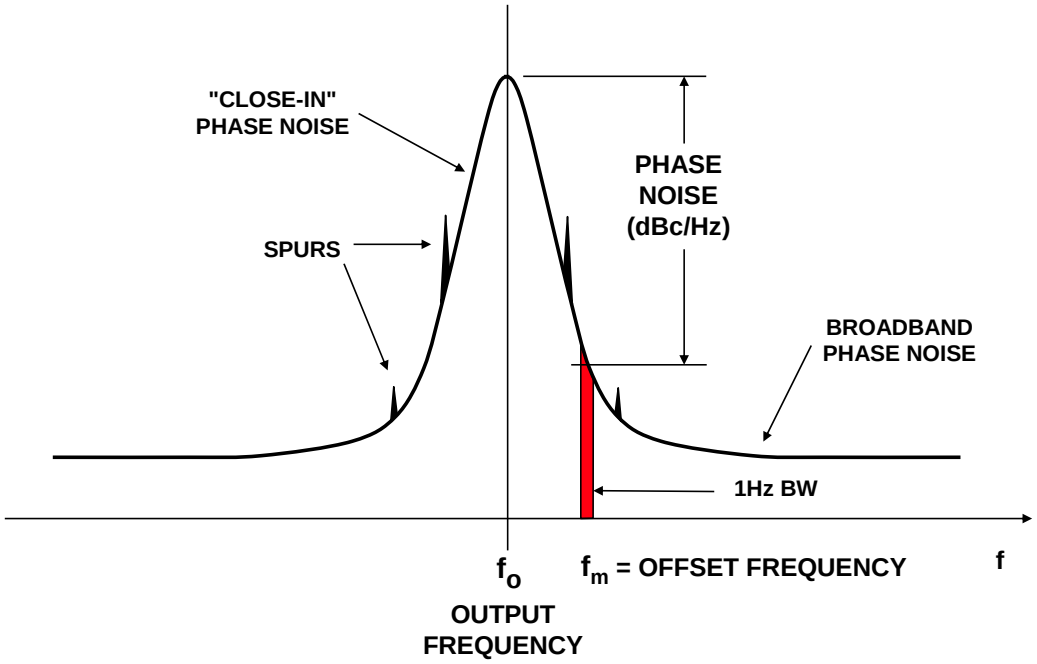


Figure 5: Oscillator Phase Noise and Spurs

The discrete spurious components could be caused by known clock frequencies in the signal source, power line interference, and mixer products. The broadening caused by random noise fluctuation is due to phase noise. It can be the result of thermal noise, shot noise, and/or flicker noise in active and passive devices.

The phase noise spectrum of an oscillator shows the noise power in a 1 Hz bandwidth as a function of frequency. Phase noise is defined as the ratio of the noise in a 1 Hz bandwidth at a specified frequency offset, fm, to the oscillator signal amplitude at frequency f_o.

It is customary to characterize an oscillator in terms of its single-sideband phase noise as shown in Figure 6, where the phase noise in dBc/Hz is plotted as a function of frequency offset, f_m, with the frequency axis on a log scale. Note the actual curve is approximated by a number of regions, each having a slope of $1/f^x$, where x = 0 corresponds to the "white" phase noise region (slope = 0 dB/decade), and x = 1, corresponds to the "flicker" phase noise region (slope = –20 dB/decade). There are also regions where x = 2, 3, 4, and these regions occur progressively closer to the carrier frequency.

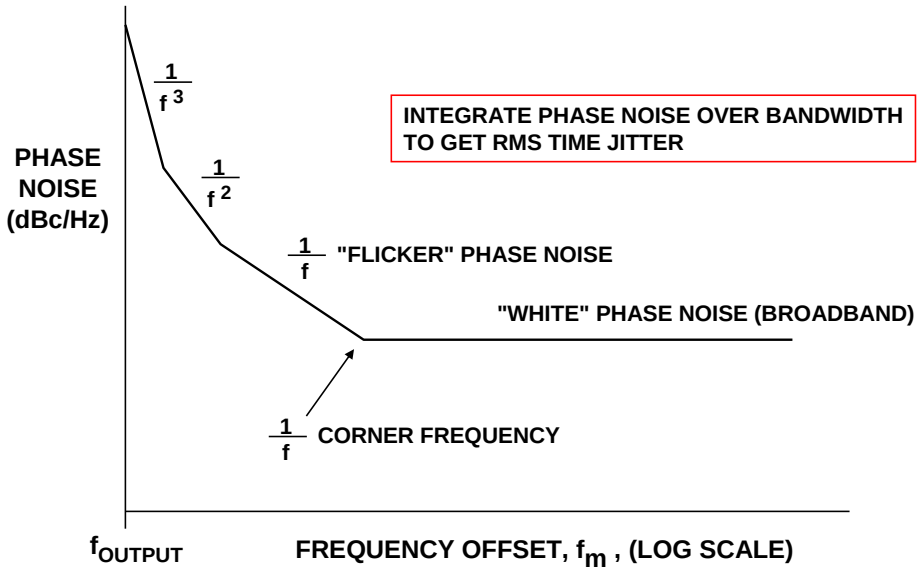


Figure 6: Phase Noise in dBc/Hz Versus Frequency Offset from Output Frequency

Note that the phase noise curve is somewhat analogous to the input voltage noise spectral density of an amplifier. Like amplifier voltage noise, low 1/f corner frequencies are highly desirable in an oscillator.

In some cases, it is useful to convert phase noise into time jitter. This can be done by basically integrating the phase noise plot over the desired frequency range. (See Tutorial MT-008, Converting Oscillator Phase Noise to Time Jitter). The ability to perform this conversion between phase noise and time jitter is especially useful when using the PLL output to drive an ADC sampling clock. Once the time jitter is known, its effect on the overall ADC SNR can be evaluated. The ADIsimPLL™ program (to be discussed shortly) performs the conversion between phase noise and time jitter.

FRACTIONAL-N PHASE LOCKED LOOPS

Fractional-N PLLs have been utilized since the 1970s. As has been discussed, the resolution at the output of an integer-N PLL is limited to steps of the PFD input frequency as shown in Figure 7A, where the PFD input is 0.2 MHz.

Fractional-N allows the resolution at the PLL output to be reduced to small fractions of the PFD frequency as shown in Figure 7B, where the PFD input frequency is 1 MHz. It is possible to generate output frequencies with resolutions of 100s of Hz, while maintaining a high PFD frequency. As a result the N-value is significantly less than for integer-N.

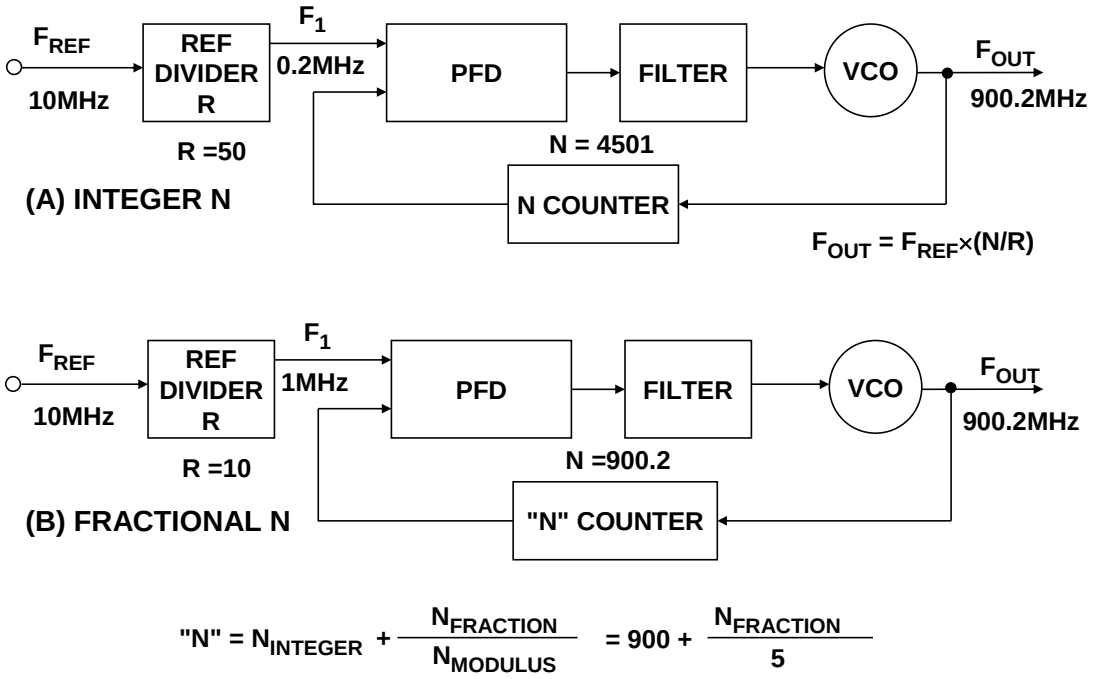


Figure 7: Integer-N Compared to Fractional-N Synthesizer

Since noise at the charge pump is multiplied up to the output at a rate of 20logN, significant improvements in phase noise are possible. For a GSM900 system, the fractional-N ADF4252 offers phase noise performance of –103 dBc/Hz, compared with –93 dBc/Hz for the ADF4106 integer-N PLL.

Also offering a significant advantage is the lock-time improvement made possible by fractional-N. The PFD frequency set to 20 MHz and loop bandwidth of 150 kHz will allow the synthesizer to jump 30 MHz in less than 30 µs. Current base stations require two PLL blocks to ensure that LOs can meet the timing requirements for transmissions. With the super-fast lock times of fractional-N, future synthesizers will have lock time specs that allow the two "ping-pong" PLLs to be replaced with a single fractional-N PLL block.

The downside of fractional-N PLLs is higher spurious levels. A fractional-N divide by 900.2 (See Figure 7B) consists of the N-divider dividing by 900 80% of the time, and by 901 20% of the time. The average division is correct, but the instantaneous division is incorrect. Because of this, the PFD and charge pump are constantly trying to correct for instantaneous phase errors. The heavy digital activity of the sigma-delta modulator, which provides the averaging function, creates spurious components at the output. The digital noise, combined with inaccuracies in matching the hard-working charge pump, results in spurious levels greater than those allowable by most communications standards. Only recently have fractional-N parts, such as the ADF4252, made the necessary improvements in spurious performance to allow designers to consider their use in traditional integer-N markets.

SIMPLIFYING PLL DESIGN USING ADIsimPLL™

The ADIsimPLL™ software is a complete PLL design package which can be downloaded from the Analog Devices' website. The software has a user-friendly graphical interface, and a complete comprehensive tutorial for first-time users.

Traditionally, PLL Synthesizer design relied on published application notes to assist in the design of the PLL loop filter. It was necessary to build prototype circuits to determine key performance parameters such as lock time, phase noise, and reference spurious levels. Optimization was accomplished by "tweaking" component values on the bench and repeating lengthy measurements.

ADIsimPLL both streamlines and improves the traditional design process. Starting with the "new PLL wizard," a designer constructs a PLL by specifying the frequency requirements of the PLL, selecting an integer-N or fractional-N implementation, and then choosing from a library of PLL chips, library or custom VCO, and a loop filter from a range of topologies. The program designs a loop filter and displays key parameters including phase noise, reference spurs, lock time, lock detect performance, and others.

ADIsimPLL operates with spreadsheet-like simplicity and interactivity. The full range of design parameters such as loop bandwidth, phase margin, VCO sensitivity, and component values can be altered with real-time updates of the simulation results. This allows the user to easily optimize the design for specific requirements. Varying the bandwidth, for example, enables the user to observe the tradeoff between lock time and phase noise in real-time, and with bench-measurement accuracy.

ADIsimPLL includes accurate models for phase noise, enabling reliable prediction of the synthesizer closed-loop phase noise. Users report excellent correlation between simulation and measurement. If required, the designer can work directly at the component level and observe the effects of varying individual component values.

The basic design process using ADIsimPLL can be summarized as follows:

1. Choose reference frequency, output frequency range, and channel spacing
2. Select PLL chip from list
3. Select VCO
4. Select loop filter configuration
5. Select loop filter bandwidth and phase margin
6. Run simulation
7. Evaluate time and frequency domain results
8. Optimize

ADIsimPLL works for integer-N or fractional-N PLLs, but does not simulate fractional-N spurs. Phase noise prediction for fractional-N devices assumes the device is operating in the "lowest phase noise" mode.

REFERENCES

1. Mike Curtin and Paul O'Brien, "Phase-Locked Loops for High-Frequency Receivers and Transmitters"

   Part 1, *Analog Dialogue*, 33-3, Analog Devices, 1999
   Part 2, *Analog Dialogue*, 33-5, Analog Devices, 1999
   Part 3, *Analog Dialogue*, 33-7, Analog Devices, 1999

2. Roland E. Best, *Phase Locked Loops, 5th Edition*, McGraw-Hill, 2003, ISBN: 0071412018.

3. Floyd M. Gardner, *Phaselock Techniques, 2nd Edition*, John Wiley, 1979, ISBN: 0471042943.

4. Dean Banerjee, *PLL Performance, Simulation and Design, 3rd Edition*, Dean Banerjee Publications, 2003, ISBN: 0970820712 .

5. Bar-Giora Goldberg, *Digital Frequency Synthesis Demystified*, Newnes, 1999, ISBN: 1878707477.

6. Brendan Daly, "Comparing Integer-N and Fractional-N Synthesizers," *Microwaves and RF*, September 2001, pp. 210-215.

7. Adrian Fox, "Ask The Applications Engineer-30 (Discussion of PLLs)," *Analog Dialogue*, 36-3, 2002.

8. Hank Zumbahlen, *Basic Linear Design*, Analog Devices, 2006, ISBN: 0-915550-28-1. Also available as *Linear Circuit Design Handbook*, Elsevier-Newnes, 2008, ISBN-10: 0750687037, ISBN-13: 978-0750687034. Chapter 4.

9. Walt Kester, *Analog-Digital Conversion*, Analog Devices, 2004, ISBN 0-916550-27-3, Chapter 6. Also available as *The Data Conversion Handbook*, Elsevier/Newnes, 2005, ISBN 0-7506-7841-0, Chapter 6.

10. Walt Kester, "Converting Oscillator Phase Noise to Time Jitter," Tutorial MT-008, Analog Devices

11. Design Tool: ADIsimPLL, Analog Devices, Inc.

12. Analog Devices PLL Product Portfolio: http://www.analog.com/pll