fluence the suitability of a PLL for arbitrary clock synthesis, including VCO stability and loop-filter characteristics. VCO stability is improved by filtering its power supply. Typical core clock synthesis applications in a microprocessor or logic IC may require multiplication by 2 or 4, which is relatively straightforward. As the multiplication ratio increases, the control path between the phase detector and the VCO is degraded, because fewer feedback adjustment opportunities exist for every VCO clock period. If a PLL is multiplying a reference by 100, the phase detector will only be able to judge phase errors on 1 percent of the VCO edges. There is the potential for VCO wander during the long intervals between phase detector corrections.

Most digital PLLs contain a lowpass loop filter integrated on chip that is matched to the typical operating scenario of a 1:1 input/output ratio or some small multiplication factor. Larger multiplication ratios may require a more complex off-chip filter that is specifically designed for the application's requirements. This is where PLL design in a digital system can start to get complicated. Fortunately, semiconductor manufacturers have documentation and applications engineering staff available to assist with such tasks. One set of devices that are well suited for digital frequency synthesis is Texas Instruments' TLC29xx family. The TLC29xx devices have separately connected phase detector and VCO sections such that an external loop-filter can be customized to the application.

It is not very common for a digital system to require complex multiplication ratios in a PLL, but the requirement does exist in applications that have multiple interfaces that run at different frequencies and must be phase locked to prevent data loss. One example is a digital video processor. The primary clock frequency of many digital video standards is 27 MHz; however, some newer high-definition video standards operate at 74.25 MHz. These frequencies are related by a factor of 2.75. If a digital video processor must perform some conversion between these two interfaces and do so in a manner that precisely matches their data rates, a PLL is necessary to lock one of the interfaces to the other. Otherwise, each interface would run on its own oscillator, and small amounts of frequency drift would soon cause one to get a little faster or slower than the other. No matter how accurate an oscillator is, it has a finite deviation from its nominal frequency. When two oscillators are paired with the expectation that they run at constant rates, a problem will eventually develop. As soon as a rate mismatch develops, the input and output data rates no longer match, and data is corrupted.

For the sake of discussion, assume that the 27-MHz interface is the master, and the 74.25-MHz interface is the slave. Because a counter cannot directly divide by 2.75, a ratio of integers must be calculated. The smallest pair of integers that yields $M \div N = 2.75$ is $M = 11$ and $N = 4$. This means that the PLL is essentially performing an $11\times$ clock multiplication function, because the 27 MHz reference is divided by 4 to yield a 6.75-MHz PLL input.

The task of designing a PLL to implement large multiplication factors is decidedly non-trivial because of the problems of stability and jitter. Selecting an appropriate lowpass filter that keeps the loop from unstable oscillations and that adequately addresses VCO jitter can involve significant control systems theory and analog filter design skills.

16.5 DELAY-LOCKED LOOPS

PLLs have traditionally been considered the standard mechanism for implementing zero-delay buffering and clock multiplication. Their flexibility comes at a certain price for manufacturers of digital ICs, because PLLs are analog circuits that must be isolated from noisy digital switching power supplies for low-jitter operation. Although the problems of on-chip isolation have been addressed for a long time, the problem persists. Many large digital ICs are now manufactured with a purely digital

delay-locked loop, or DLL, that produces similar results to a PLL. Instead of controlling a VCO to vary the phase of the output clock, a DLL contains a many-tap digital delay line through which the reference clock propagates. According to the detected phase difference between the reference and feedback clocks, a specific propagation delay can be picked in real time to align the edges. A DLL may reduce problems in certain designs as a result of the reduced noise sensitivity of purely digital circuits and the lack of a VCO. Figure 16.15 shows a basic digital DLL where the delay is programmed by selecting one of many delay line taps with a multiplexer. DLLs can also be designed as analog circuits by employing a voltage controlled delay line. However, the purely digital implementation is preferable due to its improved noise immunity.

A DLL must have sufficient delay granularity at each tap to be effective in minimizing skew between its input and output. If the incremental delay is 100 ps, the DLL can offer skew no better than ±50 ps. The phase detector within a DLL is able to dynamically adjust the delay line to compensate for changing propagation delays through active and passive elements over time, temperature, and voltage. In this regard, it is very similar to a PLL's dynamic compensation.

It is worth noting that different companies and engineers take differing positions on whether a PLL or DLL is superior for clock management. There are those who trust the time-proven PLL methodology and those who believe that a purely digital circuit is the cure-all for noise-related problems. In most situations, there is no choice, because a company that manufactures FPGAs or microprocessors has already made the decision and fabricated one solution. Custom IC design processes sometimes allow the customer to choose one over the other. In reality, DLLs and PLLs have both been demonstrated to work well in millions of individual units shipped, and one shouldn't be overly concerned unless working with very high-speed designs where picoseconds of jitter and skew can become significant problems.

16.6 SOURCE-SYNCHRONOUS CLOCKING

Clock distribution becomes a more challenging task as the number of low-skew loads increases and the operating frequency increases. Despite the low-skew technologies discussed so far in this chapter, there are practical limitations on how little skew can be achieved across a high-fan-out clock tree. When a system requires more low-skew clocks than a single buffer can drive, multiple buffers are required, and part-to-part skew becomes the limiting factor rather than output-to-output skew on a single buffer. Truly high-fan-out clock trees may require multiple levels of buffering, and each level adds to the overall skew. Zero-delay clock buffers can be used in place of low-skew buffers, but PLLs introduce jitter and have finite output-to-output skew as well.

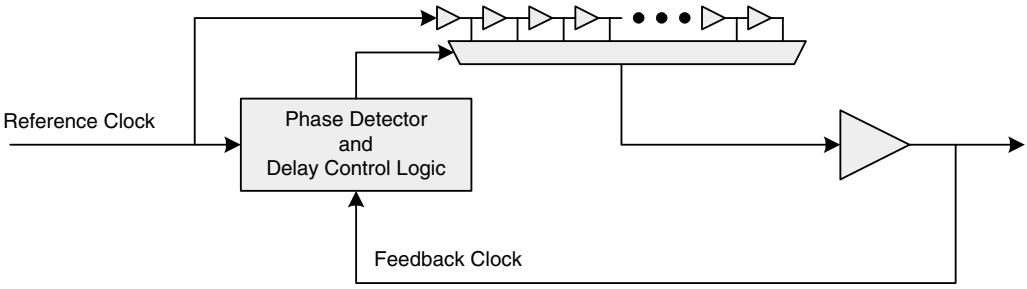

FIGURE 16.15 Generic delay-locked loop.