

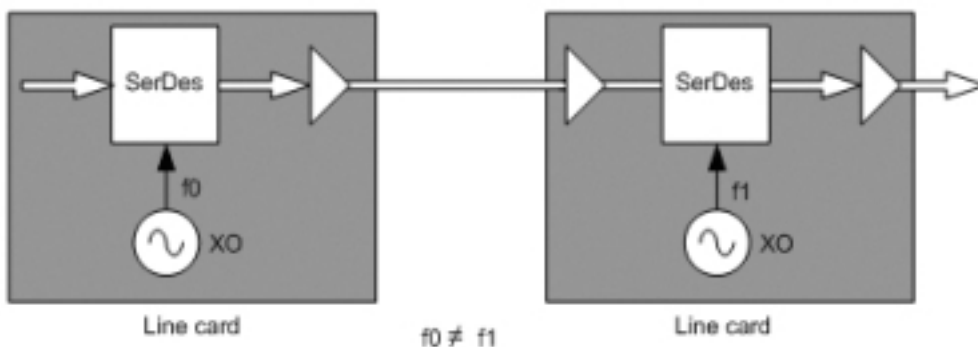
When to Use a Clock vs. an Oscillator

James Wilson, Marketing Manager

Hardware design in high performance applications such as networking, wireless/RF transmission, broadcast video and test and measurement is becoming increasingly complex as hardware designers grapple with the need to support a growing number of standards, protocols, and specifications within a single hardware design. A few examples illustrate this trend. The latest networking gear is being designed to support not only SONET/SDH and Ethernet, but also high-definition video transmission. Next generation wireless infrastructure equipment is being designed to support both WiMAX and LTE (Long-Term Evolution). Broadcast video equipment, which spans a wide variety of functions including image capture, encoding, decoding, processing and video transport, must support NTSC and PAL standards to ensure worldwide compatibility. As the lines between traditionally different types of equipment become increasingly blurred, a separate challenge arises. Designing the most efficient timing architecture is paramount in these applications to minimize design time and BOM cost.

A wide range of timing solutions are available, including crystal oscillators (XO), voltage-controlled crystal oscillators (VCXO), and clocks. No one size fits all strategy applies when it comes to component selection. Picking the right device for a particular application is dependent on a number of factors, including whether or not the clocks must be synchronized to an externally provided reference clock, the system architecture of the processor and high speed serial data transmission ICs, and the frequency and jitter requirements of the end application. In high performance applications, low jitter and low phase noise are critical given that they have a direct impact on the bit-error rate in high speed serial data transmission applications and the signal-to-noise ratio of analog to digital data converters.

Asynchronous Clocking



The simplest clock generation source is an oscillator (XO), which generates a single output frequency for a single component. XOs are oftentimes used in asynchronous applications, as shown in Figure 1. Each oscillator provides a local reference to maintain two independent clock domains. System operation requires XO frequencies be close but not identical. This architecture is ideal for burst-mode traffic applications.

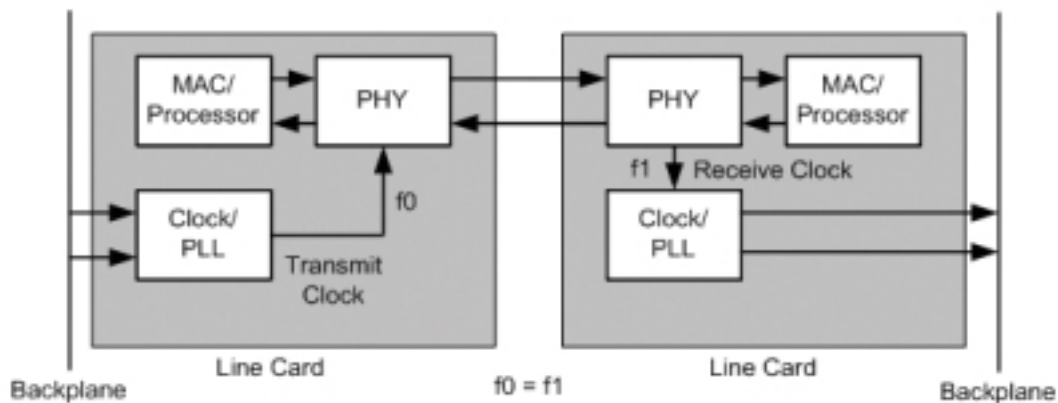
When to Use a Clock vs. an Oscillator

Published on Electronic Component News (<http://www.ecnmag.com>)

Continuous communication requires bit or packet stuffing and FIFO management to prevent overflow/underflow conditions. Video processing equipment and 10/100/1000BaseT Ethernet are examples of applications that use asynchronous clocking.

XO selection should be based on the frequency, jitter and stability requirements of the end application.

Synchronous Clocking

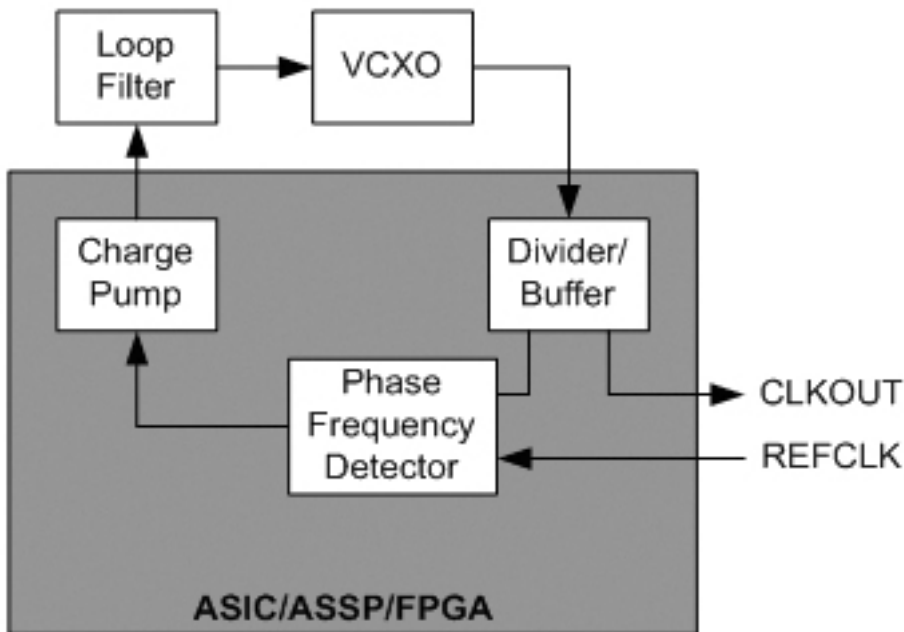


Synchronous clocking is most often used in applications that require continuous communication. Network latency and variability in latency must be minimized. To accomplish this, applications including SONET/SDH, Synchronous Ethernet (SyncE), wireless backhaul, and video transport require that the source and destination operate at the same frequency. On the transmit side, the clocks that provide timing for the transmit path of the high speed SerDes are locked to a highly accurate reference clock. Both primary and secondary reference clocks are supplied from a centralized timing source (e.g. GPS). A PLL is used to lock to this backplane reference, attenuate jitter on the clock signal to remove unwanted noise, and provide a low jitter output clock to the PHY. On the receive side, a Clock and Data Recovery (CDR) unit is used to recover the system clock. This CDR may be an external component or integrated within the PHY depending on the application. A XO may be used to center the CDR to provide fast acquisition and lock. The recovered clock passes through another PLL to divide the clock frequency down to a lower rate. Local timing can either be synchronized to this clock or to another local clock that is synchronized to a centralized timing source. Synchronization to the source ensures clock synchronization across all nodes in the network. Depending on system requirements, the PLLs used in this application may require a low loop bandwidth to filter unwanted jitter from the clock signal.

The PLL can be implemented discretely using an integrated clock IC or a Voltage-Controlled Crystal Oscillator (VCXO), phase detector, and loop filter. A discrete solution is preferable when the lowest possible jitter and best possible phase noise are required. In some applications, processors or ASICs integrate the phase detector and charge pump within the IC so that only a VCXO and external loop filter are required as shown in Figure 3 below.

When to Use a Clock vs. an Oscillator

Published on Electronic Component News (<http://www.ecnmag.com>)



However, there are multiple disadvantages with a discrete PLL solution. A discrete PLL requires analog design expertise and is sensitive to board-level noise, so special care must be taken in the design and PCB layout. In addition, a discrete PLL typically provides a single output frequency. If the design's frequency requirements change, a separate VCXO must be sourced. In some applications, multiple VCXOs are required to generate all of the required frequencies in the application, increasing BOM complexity. To address these shortcomings, dual, quad, and even any-rate I2C-programmable VCXOs like Silicon Lab's Si571 are available now to address multi-rate applications by replacing multiple discrete VCXOs with a single device.

An alternate approach is to use a jitter attenuating clock multiplier IC, which integrates PLL circuitry on-chip. The clock multiplier maintains lock to the reference clock, filters unwanted jitter and generates a multiplied frequency output clock for the transmitter. Special care must be taken in clock multiplier IC selection, as all clock multipliers are not created equal. For high speed serial data transmission applications, only the highest performance clock multiplier ICs provide the jitter performance necessary to meet the end application requirements. The key specification is maximum jitter generation as opposed to typical jitter generation. A clock multiplier that specifies maximum jitter best enables the hardware designer to allocate a jitter budget among the data path and timing components in the board and ensure there is sufficient margin under all conditions. Also, check the loop bandwidth options available on the clock multiplier IC. If jitter attenuation of the reference clock is required, then oftentimes the loop bandwidth must be 1 kHz or less. Lastly, confirm the clock multiplier IC supports all required frequency plans. An example of a frequency-agile clock multiplier that provides jitter attenuation and any-rate frequency synthesis is the Silicon Labs Si5319.

Lastly, a clock solution is preferable to a discrete solution when system-level clock functions are required. An example of this is hitless switching between input clocks, in which the clock monitors the quality of a primary reference clock and switches to a secondary reference upon detection of an alarm condition on the primary clock. Another popular system-level clock requirement is holdover, in which the clock

When to Use a Clock vs. an Oscillator

Published on Electronic Component News (<http://www.ecnmag.com>)

continues to generate a stable output clock in the absence of a valid reference clock. Clocks are available from multiple suppliers that address these system-level requirements.

Source URL (retrieved on 02/01/2015 - 4:34pm):

<http://www.ecnmag.com/articles/2009/07/when-use-clock-vs-oscillator>