Contents

Chapter 6

Fundamentals of Synchronization

The analysis and developments of Chapters 1-4 presumed that the modulator and demodulator are synchronized. That is, both modulator and demodulator know the exact symbol rate and the exact symbol phase, and where appropriate, both also know the exact carrier frequency and phase. In practice, the common (receiver/transmitter) knowledge of the same timing and carrier clocks rarely occurs unless some means is provided for the receiver to synchronize with the transmitter. Such synchronization is often called **phase-locking.**

In general, phase locking uses three component operations as generically depicted in Figure 6.1:[1]

1. **Phase-error generation** - this operation, sometimes also called "phase detection," derives a phase difference between the received signal's phase $\theta(t)$ and the receiver estimate of this phase, $\hat{\theta}(t)$. The actual signals are $s(t) = cos\left(\omega_{lo}t + \theta(t)\right)$ and $\hat{s}(t) = cos\left(\omega_{lo}t + \hat{\theta}(t)\right)$, but only their phase difference is of interest in synchronization. This difference is often called the **phase error**, $\phi(t) = \theta(t) - \hat{\theta}(t)$. Various methods of phase-error generation are discussed in Section 6.1.

2. **Phase-error processing** - this operation, sometimes also called "loop filtering" extracts the essential phase difference trends from the phase error by averaging. Phase-error processing typically rejects random noise and other undesirable components of the phase error signal. Any gain in the phase detector is assumed to be absorbed into the loop filter. Both analog and digital phase-error processing, and the general operation of what is known as a "phase-lock loop," are discussed in Section 6.2.

3. **Local phase reconstruction** - this operation, which in some implementations is known as a "voltage-controlled oscillator" (VCO), regenerates the local phase from the processed phase error in an attempt to match the incoming phase, $\theta(t)$. That is, the phase reconstruction tries to force $\phi(t) = 0$ by generation of a local phase $\hat{\theta}(t)$ so that $\hat{s}(t)$ matches $s(t)$. Various types of voltage controlled oscillators and other methods of regenerating local clock are also discussed in Section 6.1

Any phase-locking mechanism will have some finite delay in practice so that the regenerated local phase will try to project the incoming phase and then measure how well that projection did in the form of a new phase error. The more quickly the phase-lock mechanism tracks deviations in phase, the more susceptible it will be to random noise and other imperfections. Thus, the communication engineer must trade these two competing effects appropriately when designing a synchronization system. The design of the transmitted signals can facilitate or complicate this trade-off. These design trade-offs will be generally examined in Section 6.2. In practice, unless pilot or control signals are embedded in the transmitted signal, it is necessary either to generate an estimate of transmitted signal's clocks or to generate an estimate of the phase error directly. Sections 6.3 and 6.4 specifically examine phase detectors for situations where such clock extraction is necessary for timing recovery (recovery of the symbol clock) and carrier recovery (recovery of the phase of the carrier in passband transmission), respectively. In

[1]These operations may implemented in a wide variety of ways.

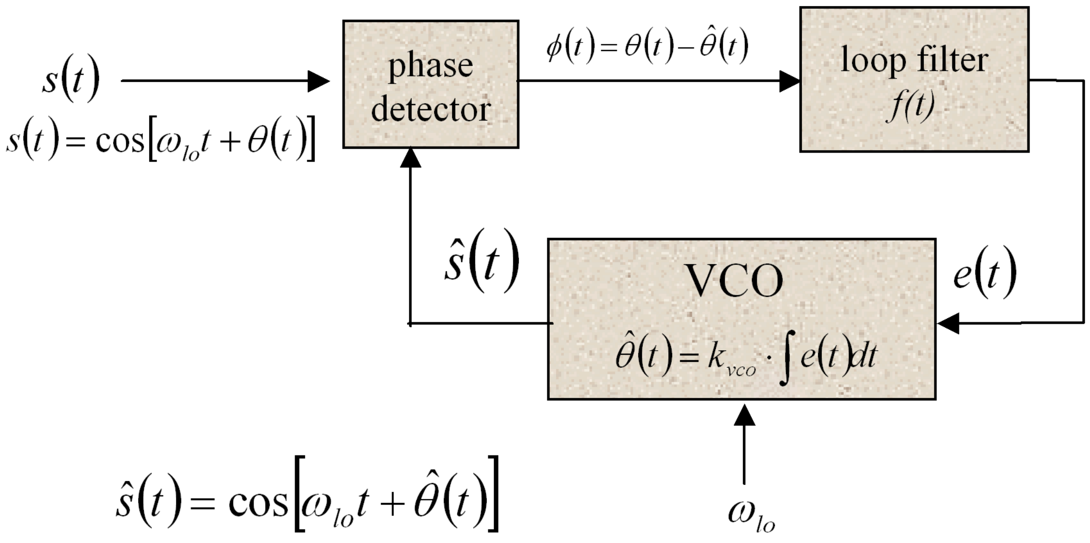

$s(t)$
$s(t)=\cos[\omega_{lo}t+\theta(t)]$
phase detector
$\phi(t)=\theta(t)-\hat{\theta}(t)$
loop filter
$f(t)$
$\hat{s}(t)$
VCO
$\hat{\theta}(t)=k_{vco}\cdot\int e(t)dt$
$e(t)$
$\hat{s}(t)=\cos\left[\omega_{lo}t+\hat{\theta}(t)\right]$
ω_{lo}

Figure 6.1: General phase-lock loop structure.

addition to symbol timing, data is often packetized or framed. Methods for recovering frame boundary are discussed in Section 6.5. Pointers and add/delete (bit robbing/stuffing) are specific mechanisms of local phase reconstruction that allow the use of asynchronous clocks in the transmitter and receiver. These methods find increasing use in modern VLSI implementations of receivers and are discussed in Section 6.6.

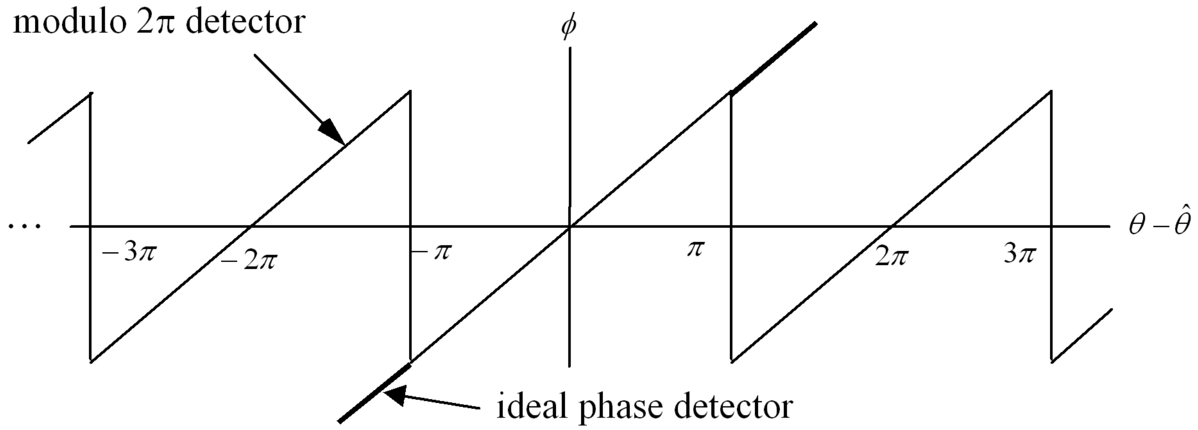


Figure 6.2: Comparison of ideal and mod-2π phase detectors.

6.1 Phase computation and regeneration

This section describes the basic operation of the phase detector and of the voltage-controlled oscillator (VCO) in Figure 6.1 in more detail.

6.1.1 Phase Error Generation

Phase-error generation can be implemented continuously or at specific sampling instants. The discussion in this subsection will therefore not use a time argument or sampling index on phase signals. That is $\theta(t) \to \theta$.

Ideally, the two phase angles, θ, the phase of the input sinusoid s, and $\hat{\theta}$, the estimated phase produced at the VCO output, would be available. Then, their difference ϕ could be computed directly.

Definition 6.1.1 (ideal phase detector) *A device that can compute exactly the difference $\phi = \theta - \hat{\theta}$ is called an* **ideal phase detector**.

Ideally, the receiver would have access to the sinusoids $s = \cos(\theta_{lo} + \theta)$ and $\hat{s} = \cos(\theta_{lo} + \hat{\theta})$ where θ_{lo} is the common phase reference that depends on the local oscillator frequency; θ_{lo} disappears from the ensuing arguments, but in practice the general frequency range of $\frac{d\theta}{dt} = \omega_{lo}$ will affect implementations. A seemingly straightforward method to compute the phase error would then be to compute θ, the phase of the input sinusoid s, and $\hat{\theta}$, the estimated phase, according to

$$\theta = \pm \arccos\left[s\right] - \theta_{lo} \;\; ; \;\; \hat{\theta} = \pm \arccos\left[\hat{s}\right] - \theta_{lo} \quad . \tag{6.1}$$

Then, $\phi = \theta - \hat{\theta}$, to which the value θ_{lo} is inconsequential in theory. However, a reasonable implementation of the arccos function can only produce angles between 0 and π, so that ϕ would then always lie between $-\pi$ and π. Any difference of magnitude greater than π would be thus effectively be computed modulo $(-\pi, \pi)$. The arccos function could then be implemented with a look-up table.

Definition 6.1.2 (mod-2π phase detector) *The* arccos *look-up table implementation of the phase detector is called a* **modulo-2π phase detector**[2].

The characteristics of the mod-2π phase detector and the ideal phase detector are compared in Figure 6.2. If the phase difference does exceed π in magnitude, the large difference will not be exhibited in the phase error ϕ - this phenomenon is known as a "cycle slip," meaning that the phase lock loop missed (or added) an entire period (or periods) of the input sinusoid. This is an undesirable phenomena in most

[2]Sometimes also called a "sawtooth" phase detector after the phase characteristic in Figure 6.2.

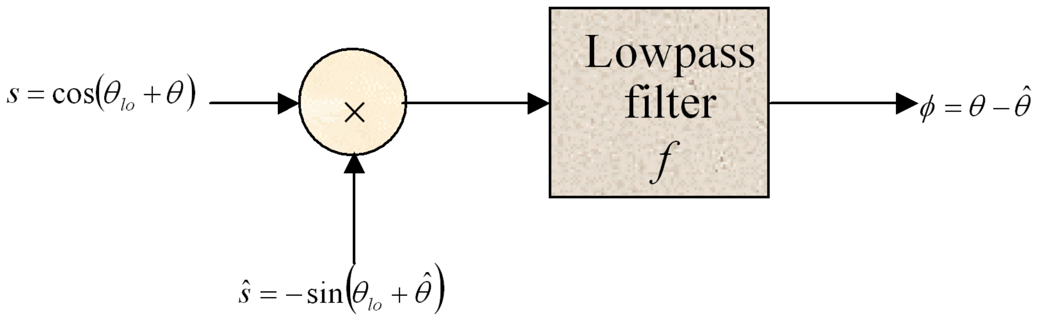

$s = \cos(\theta_{lo} + \theta)$
$\times$
Lowpass filter
f
$\phi = \theta - \hat{\theta}$
$\hat{s} = -\sin(\theta_{lo} + \hat{\theta})$

Figure 6.3: Demodulating phase detector.

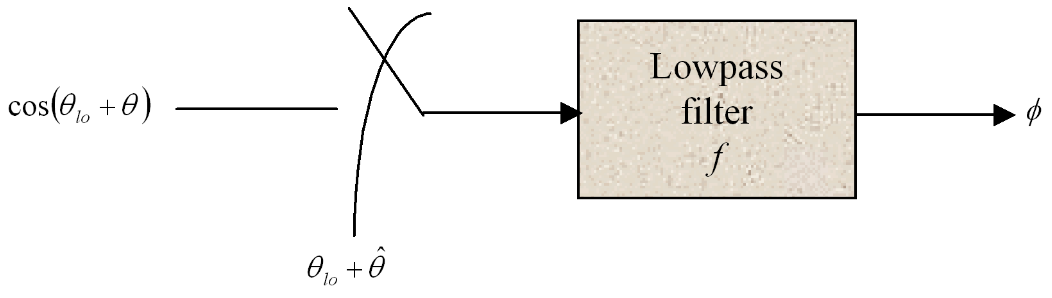


Figure 6.4: Sampling phase detector.

applications, so after a phase-lock loop with mod-2π phase detector has converged, one tries to ensure that $|\phi|$ cannot exceed π. In practice, a good phase lock loop should keep this phase error as close to zero as possible, so the condition of small phase error necessary for use of the mod-2π phase detector is met. The means for avoiding cycle slips is to ensure that the local-oscillator frequency is less than double the desired frequency and greater than 1/2 that same local-oscillator frequency.

Definition 6.1.3 (demodulating phase detector) *Another commonly encountered phase detector, both in analog and digital form, is the* **demodulating phase detector** *shown in Figure 6.3, where*

$$\phi = f * \left[-\sin(\theta_{lo} + \hat{\theta}) \cdot \cos(\theta_{lo} + \theta) \right] \quad , \tag{6.2}$$

and f is a lowpass filter that is cascaded with the phase-error processing in the phase-locking mechanism.

The basic concept arises from the relation

$$-\cos(\omega_{lo} t + \theta) \cdot \sin\left(\omega_{lo} t + \hat{\theta}\right) = \frac{1}{2}\left\{ -\sin\left(2\omega_{lo} t + \theta + \hat{\theta}\right) + \sin\left(\theta - \hat{\theta}\right) \right\} \quad , \tag{6.3}$$

where the sum-phase term (first term on the right) can be eliminated by lowpass filtering; this lowpass filtering can be absorbed into the loop filter that follows the phase detector (see Section 6.2). The phase detector output is thus proportional to $\sin(\phi)$. The usual assumption with this type of phase detector is that ϕ is small ($\phi << \frac{\pi}{6}$), and thus

$$\sin(\phi) \approx \phi \quad . \tag{6.4}$$

When ϕ is small, generation of the phase error thus does not require the arcsin function.

Another way of generating the phase error signal is to use the local sinusoid to sample the incoming

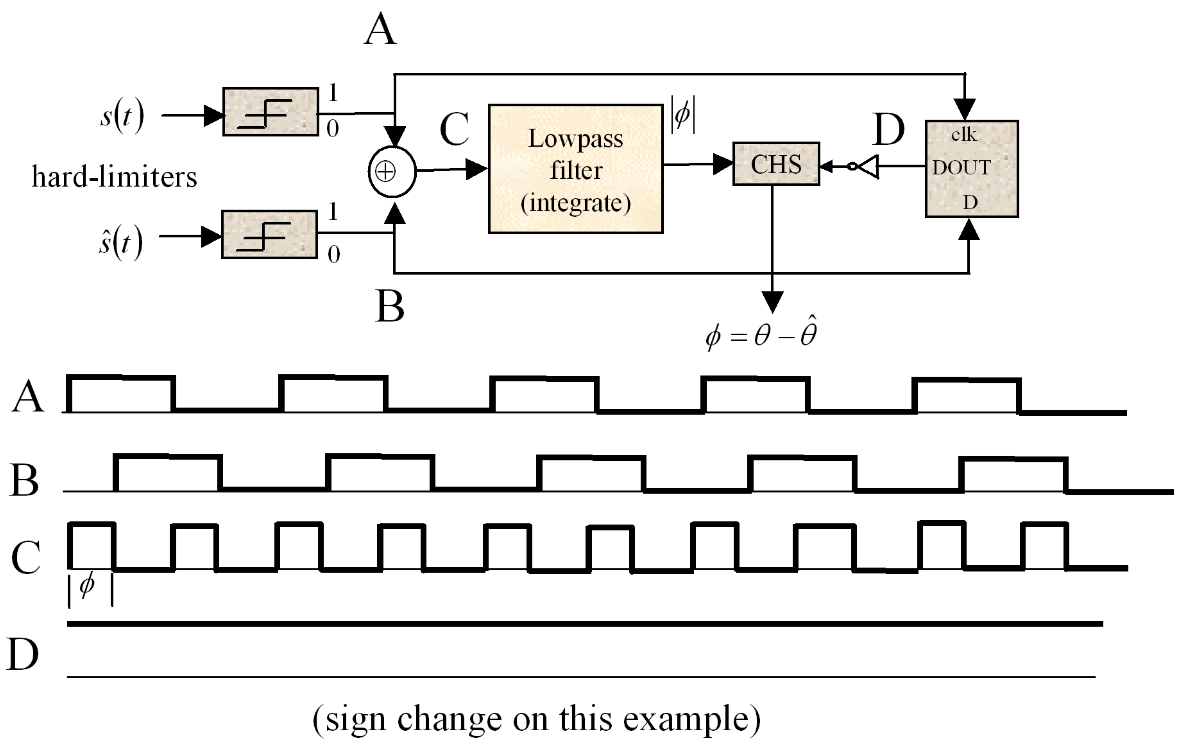

A
$s(t)$
1
0
hard-limiters
$\hat{s}(t)$
1
0
C
Lowpass filter (integrate)
$|\phi|$
CHS
D
clk
DOUT
D
B
$\phi = \theta - \hat{\theta}$
A
B
C
$|\phi|$
D
(sign change on this example)

Figure 6.5: Binary phase detector.

sinusoid as shown in Figure 6.4. If the rising edge of the local sinusoid is used for the sampling instant, then

$$\theta_{lo} + \hat{\theta} = -\frac{\pi}{2} \quad . \tag{6.5}$$

(At this phase, $\hat{s} = 0$.) Then, at the time of this rising edge, the sampled sinusoid s has phase

$$\theta_{lo} + \hat{\theta} + \theta - \hat{\theta} = -\frac{\pi}{2} + \phi \quad , \tag{6.6}$$

so that $s(t)$ at these times t_k is

$$s(t_k) = cos(-\frac{\pi}{2} + \phi) = \sin(\phi) \approx \phi \quad . \tag{6.7}$$

Such a phase detector is called a **sampling phase detector.** The lowpass filter "holds" the sample value of phase.

Another type of phase detector is the **binary phase detector** shown in Figure 6.5. In the binary phase detector, the phase difference between the two sinusoids is approximately the width of the high signals at point C in Figure 6.5. The hard limiters are used to covert the sinusoids into 1/0 square waves or binary signals. The adder is a binary adder. The lowpass filter just averages (integrates) the error signal, so that its output is proportional to the magnitude of the phase error. The important sign of the phase error is determined by "latching" the polarity of $\hat{s}$ when s goes high (leading-edge triggered D flip-flop).[3]

6.1.2 Voltage Controlled Oscillators

The voltage controlled oscillator basically generates a sinusoid with phase difference (or derivative) proportional to the input control voltage $e(t)$.

[3]When A and B are phase aligned, then $\phi = 0$, so that the race condition that exists in the clocking and data set-up on line B should not be of practical significance if sufficiently high-speed logic is used.

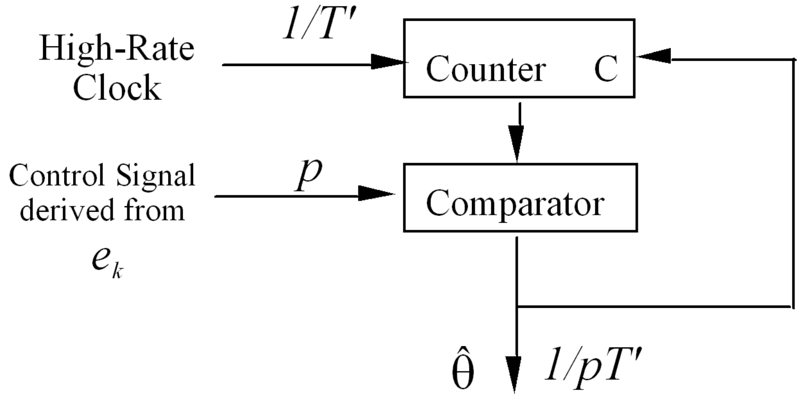


Figure 6.6: Discrete-time VCO with all-digital realization.

Definition 6.1.4 (Voltage Controlled Oscillator) *An ideal* **voltage controlled oscillator (VCO)** *has an output sinusoid with phase $\hat{\theta}(t)$ that is determined by an input* **error** *or* **control** *signal according to*

$$\frac{d\hat{\theta}}{dt} = k_{vco} \cdot e(t) \quad , \tag{6.8}$$

in continuous time, or approximated by

$$\hat{\theta}_{k+1} = \hat{\theta}_k + k_{vco} \cdot e_k \quad . \tag{6.9}$$

in discrete time.

Analog VCO physics are beyond the scope of this text, so it will suffice to just state that devices satisfying (6.8) are readily available in a variety of frequency ranges. When the input signal e_k in discrete time is digital, the VCO can also be implemented with a look-up table and adder according to (6.9) whose output is used to generate a continuous time sinusoidal equivalent with a DAC. This implementation is often called a **numerically controlled oscillator** or NCO.

Yet another implementation that can be implemented in digital logic on an integrated circuit is shown in Figure 6.6. The high-rate clock is divided by the value of the control signal (derived from e_k) by connecting the true output of the comparator to the clear input of a counter. The higher the clock rate with respect to the rates of interest, the finer the resolution on specifying $\hat{\theta}_k$. If the clock rate is $1/T'$, then the divider value is p or $p+1$, depending upon whether the desired clock phase is late or early, respectively. A maximum phase change with respect to a desired phase (without external smoothing) can thus be $T'/2$. For 1% clock accuracy, then the master clock would need to be 50 times the generated clock frequency. With an external analog smoothing of the clock signal (via bandpass filter centered around the nominal clock frequency), a lower frequency master clock can be used. Any inaccuracy in the high-rate clock is multiplied by p so that the local oscillator frequency must now lay in the interval $f_{lo} \cdot \left((1 - \frac{1}{2p}), f_{lo} \cdot (1+p)\right)$ to avoid cycle slipping. (When $p = 1$, this reduces to the interval mentioned earlier.) See also Section 6.2.3 on phase-locking at rational multiples of a clock frequency.

The Voltage-Controlled Crystal Oscillator (VCXO)

In many situations, the approximate clock frequency to be derived is known accurately. Conventional crystal oscillators usually have accuracies of 50 parts per million (ppm) or better. Thus, the VCO need only track over a small frequency/phase range. Additionally in practice, the derived clock may be used to sample a signal with an analog-to-digital converter (ADC). Such an ADC clock should not jitter about its nominal value (or significant signal distortion can be incurred). In this case, a VCXO normally replaces the VCO. The VCXO employs a crystal (X) of nominal frequency to stabilize the VCO close to the nominal value. Abrupt changes in phase are not possible because of the presence of the crystal. However, the high stability and slow variation of the clock can be of crucial importance in digital receiver designs. Thus, VCXO's are used instead of VCO's in designs where high stability sample clocking is necessary.

Basic Jitter effect

The effect of oscillator jitter is approximated for a waveform $x(t)$ according to

$$\delta x \approx \frac{dx}{dt}\delta t \quad , \tag{6.10}$$

so that

$$(\delta x)^2 \approx \left(\frac{dx}{dt}\right)^2 (\delta t)^2 \quad . \tag{6.11}$$

A signal-to-jitter-noise ratio can be defined by

$$\text{SNR} = \frac{x^2}{(\delta x)^2} = \frac{x^2}{(dx/dt)^2(\delta t)^2} \quad . \tag{6.12}$$

For the highest frequency component of $x(t)$ with frequency f_{max}, the SNR becomes

$$\text{SNR} = \frac{1}{4\pi^2(f_{max} \cdot \delta t)^2} \quad , \tag{6.13}$$

which illustrates a basic time/frequency uncertainty principle: If $(\delta t)^2$ represents jitter in squared seconds, then jitter must become smaller as the spectrum of the signal increases. An SNR of 20 dB (factor of 100 in jitter) with a signal with maximum frequency of 1 MHz would suggest that jitter be below 16 ns.

6.1.3 Maximum-Likelihood Phase Estimation

A number of detailed developments on phase lock loops attempt to estimate phase from a likelihood function:

$$\max_{x,\theta} p_{y/x,\theta} \quad . \tag{6.14}$$

Maximization of such a function can be complicated mathematically, often leading to a series of approximations for various trigonometric functions that ultimately lead to a quantity proportional to the phase error that is then used in a phase-lock loop. Such approaches are acknowledged here, but those interested in the ultimate limits of synchronization performance are referred elsewhere. Practical approximations under the category of "decision-directed" synchronization methods in Sections 6.3.2 and 6.4.2.

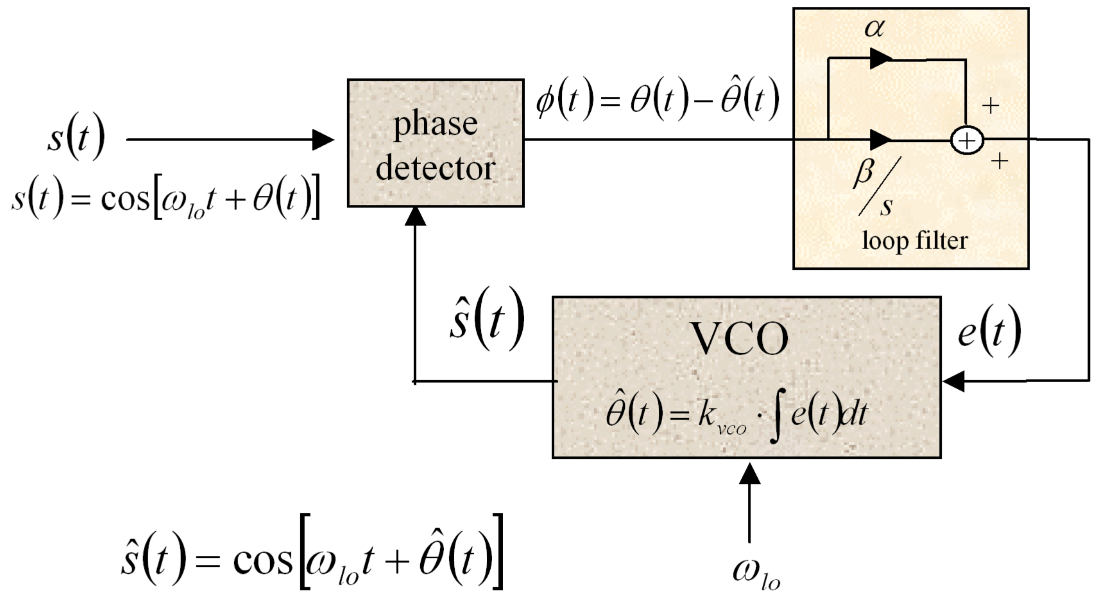

$s(t)$
$s(t) = \cos[\omega_{lo}t + \theta(t)]$
phase detector
$\phi(t) = \theta(t) - \hat{\theta}(t)$
α
$+$
$+$
β / s
loop filter
$\hat{s}(t)$
VCO
$\hat{\theta}(t) = k_{vco} \cdot \int e(t)dt$
$e(t)$
$\hat{s}(t) = \cos[\omega_{lo}t + \hat{\theta}(t)]$
ω_{lo}

Figure 6.7: Continuous-time PLL with loop filter.

6.2 Analysis of Phase Locking

Both continuous-time and discrete-time PLL's are analyzed in this section. In both cases, the loop-filter characteristic is specified for both first-order and second-order PLL's. First-order loops are found to track only constant phase offsets, while second-order loops can track both phase and/or frequency offsets.

6.2.1 Continuous Time

The continuous-time PLL has a phase estimate that follows the differential equation

$$\dot{\hat{\theta}}(t) = k_{vco} \cdot f(t) * \left(\theta(t) - \hat{\theta}(t)\right) \quad . \tag{6.15}$$

The transfer function between PLL output phase and input phase is

$$\frac{\hat{\theta}(s)}{\theta(s)} = \frac{k_{vco} \cdot F(s)}{s + k_{vco} \cdot F(s)} \tag{6.16}$$

with s the Laplace transform variable. The corresponding transfer function between phase error and input phase is

$$\frac{\phi(s)}{\theta(s)} = 1 - \frac{\hat{\theta}(s)}{\theta(s)} = \frac{s}{s + k_{vco} \cdot F(s)} \quad . \tag{6.17}$$

The cases of both first- and second-order PLL's are shown in Figure 6.7 with $\beta = 0$, reducing the diagram to a first-order loop.

First-Order PLL

The first order PLL has

$$k_{vco} \cdot F(s) = \alpha \tag{6.18}$$

(a constant) so that phase errors are simply integrated by the VCO in an attempt to set the phase $\hat{\theta}$ such that $\phi = 0$. When ϕ =0, there is zero input to the VCO and the VCO output is a sinusoid at frequency ω_{lo}. Convergence to zero phase error will only happen when $s(t)$ and $\hat{s}(t)$ have the same frequency (ω_{lo}) with initially a constant phase difference that can be driven to zero under the operation of the first-order PLL, as is subsequently shown.

The response of the first-order PLL to an initial phase offset (θ_0) with

$$\theta(s) = \frac{\theta_0}{s} \tag{6.19}$$

is

$$\hat{\theta}(s) = \frac{\alpha \cdot \theta_0}{s(s+\alpha)} \tag{6.20}$$

or

$$\hat{\theta}(t) = \left(\theta_0 - \theta_0 \cdot e^{-\alpha t}\right) \cdot u(t) \tag{6.21}$$

where $u(t)$ is the unit step function (=1, for $t > 0$, = 0 for $t < 0$). For stability, $\alpha > 0$. Clearly

$$\hat{\theta}(\infty) = \theta_0 \quad . \tag{6.22}$$

An easier analysis of the PLL final value is through the final value theorem:

$$\begin{aligned} \phi(\infty) &= \lim_{s \to 0} s \cdot \phi(s) && (6.23) \\ &= \lim_{s \to 0} s \cdot \frac{\theta_0}{s+\alpha} && (6.24) \\ &= 0 \quad , && (6.25) \end{aligned}$$

so that the final phase error is zero for a unit-step phase input. For a linearly varying phase (that is a constant frequency offset), $\theta(t) = (\theta_0 + \Delta t)u(t)$, or

$$\theta(s) = \frac{\theta_0}{s} + \frac{\Delta}{s^2} \quad , \tag{6.26}$$

where Δ is the frequency offset

$$\Delta = \frac{d\theta}{dt} - \omega_{lo} \quad . \tag{6.27}$$

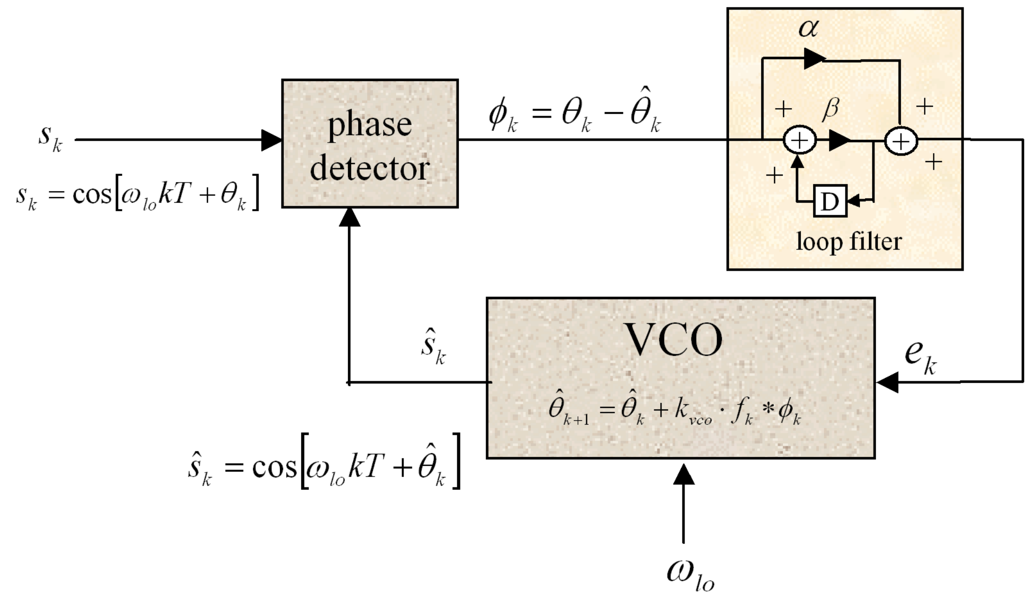
In this case, the final value theorem illustrates the first-order PLL's eventual phase error is

$$\phi(\infty) = \frac{\Delta}{\alpha} \quad . \tag{6.28}$$

A larger "loop gain" α causes a smaller the offset, but a first-order PLL cannot drive the phase error to zero. The steady-state phase instead lags the correct phase by Δ/α. However, larger α forces larger the bandwidth of the PLL. Any small noise in the phase error then will pass through the loop with less attenuation by the PLL, leading to a more noisy phase estimate. As long as $|\Delta/\alpha| < \pi$, then the modulo-2π phase detector functions with constant non-zero phase error. The range of Δ for which the phase error does not exceed π is known as the **pull range** of the PLL

$$\text{pull range} = |\Delta| < |\alpha| \cdot \pi \quad . \tag{6.29}$$

That is, any frequency deviation less than the pull range will result in a constant phase "lag" error. Such a non-zero phase error may or may not present a problem for the associated transmission system. A better method by which to track frequency offset is the second-order PLL.



s_k
$s_k = \cos[\omega_{lo} kT + \theta_k]$
phase detector
$\phi_k = \theta_k - \hat{\theta}_k$
α
β
D
loop filter
VCO
$\hat{\theta}_{k+1} = \hat{\theta}_k + k_{vco} \cdot f_k * \phi_k$
e_k
$\hat{s}_k$
$\hat{s}_k = \cos[\omega_{lo} kT + \hat{\theta}_k]$
ω_{lo}

Figure 6.8: Discrete-time PLL.

Second-Order PLL

The second order PLL uses the loop filter to integrate the incoming phase errors as well as pass the errors to the VCO. The integration of errors eventually supplies a constant signal to the VCO, which in turn forces the VCO output phase to vary linearly with time. That is, the frequency of the VCO can then be shifted away from ω_{lo} permanently, unlike the operation with first-order PLL.

In the second-order PLL,

$$k_{vco} \cdot F(s) = \alpha + \frac{\beta}{s} \quad . \tag{6.30}$$

Then

$$\frac{\hat{\theta}(s)}{\theta(s)} = \frac{\alpha s + \beta}{s^2 + \alpha s + \beta} \quad , \tag{6.31}$$

and

$$\frac{\phi(s)}{\theta(s)} = \frac{s^2}{s^2 + \alpha s + \beta} \quad . \tag{6.32}$$

One easily verifies with the final value theorem that for either constant phase or frequency offset (or both) that

$$\phi(\infty) = 0 \tag{6.33}$$

for the second-order PLL. For stability,

$$\alpha > 0 \tag{6.34}$$
$$\beta > 0 \quad . \tag{6.35}$$

6.2.2 Discrete Time

This subsection examines **discrete-time** first-order and second-order phase-lock loops. Figure 6.8 is the discrete-time equivalent of Figure 6.1. Again, the first-order PLL only recovers phase and not frequency. The second-order PLL recovers both phase and frequency. The discrete-time VCO follows

$$\hat{\theta}_{k+1} = \hat{\theta}_k + k_{vco} \cdot f_k * \phi_k \quad , \tag{6.36}$$

and generates $\cos\left(\omega_{lo}\cdot kT+\hat{\theta}_k\right)$ as the local phase at sampling time k. This expression assumes discrete "jumps" in the phase of the VCO. In practice, such transitions will be smoothed by any VCO that produces a sinusoidal output and the analysis here is only approximate for $kT \le t \le (k+1)T$, where T is the sampling period of the discrete-time PLL. Taking the D-Transforms of both sides of (6.36) equates to

$$D^{-1}\cdot\hat{\Theta}(D) = [1-k_{vco}\cdot F(D)]\cdot\hat{\Theta}(D)+k_{vco}\cdot F(D)\cdot\Theta(D) \quad . \tag{6.37}$$

The transfer function from input phase to output phase is

$$\frac{\hat{\Theta}(D)}{\Theta(D)} = \frac{D\cdot k_{vco}\cdot F(D)}{1-(1-k_{vco}\cdot F(D))\,D} \quad . \tag{6.38}$$

The transfer function between the error signal $\Phi(D)$ and the input phase is thus

$$\frac{\Phi(D)}{\Theta(D)} = \frac{\Theta(D)}{\Theta(D)}-\frac{\hat{\Theta}(D)}{\Theta(D)} = 1-\frac{D\cdot k_{vco}\cdot F(D)}{1-(1-k_{vco}\cdot F(D))\,D} = \frac{D-1}{D(1-k_{vco}\cdot F(D))-1} \quad . \tag{6.39}$$

$F(D)$ determines whether the PLL is first- or second-order.

First-Order Phase Lock Loop

In the first-order PLL, the loop filter is a (frequency-independent) gain α, so that

$$k_{vco}\cdot F(D) = \alpha \quad . \tag{6.40}$$

Then

$$\frac{\hat{\Theta}(D)}{\Theta(D)} = \frac{\alpha D}{1-(1-\alpha)D} \quad . \tag{6.41}$$

For stability, $|1-\alpha|<1$, or

$$0 \le \alpha < 2 \tag{6.42}$$

for stability. The closer α to 2, the wider the bandwidth of the overall filter from $\Theta(D)$ to $\hat{\Theta}(D)$, and the more (any) noise in the input sinusoid can distort the estimated phase. The first-order loop can track and drive to zero any phase difference between a constant θ_k and $\hat{\theta}_k$. To see this effect, the input phase is

$$\theta_k = \theta_0 \quad \forall \;\; k \ge 0 \quad , \tag{6.43}$$

which has D-Transform

$$\Theta(D) = \frac{\theta_0}{1-D} \quad . \tag{6.44}$$

The phase-error sequence then has transform

$$\begin{aligned} \Phi(D) &= \frac{D-1}{D\,(1-k_{vco}F(D))-1}\Theta(D) && (6.45)\\ &= \frac{(D-1)\theta_0}{(1-D)(D(1-\alpha)-1)} && (6.46)\\ &= \frac{\theta_0}{1-(1-\alpha)D} && (6.47) \end{aligned}$$

Thus

$$\phi_k = \begin{cases} \theta_0(1-\alpha)^k & k \ge 0 \\ 0 & k<0 \end{cases} \quad , \tag{6.48}$$

and $\phi_\infty \to 0$ if (6.42) is satisfied. This result can also be obtained by the final value theorem for one-sided D-Transforms

$$\lim_{k\to\infty}\;\phi_k \;=\lim_{D\to 1}\;(1-D)\cdot\Phi(D) = 0 \quad . \tag{6.49}$$

The first-order loop will exhibit a constant phase offset, at best, for any frequency deviation between θ_k and $\hat{\theta}_k$. To see this constant-lag effect, the input phase can be set to

$$\theta_k = \Delta \cdot k \quad \forall \; k \geq 0 \quad , \tag{6.50}$$

where $\Delta = \omega_{offset}T$, which has D-Transform[4]

$$\Theta(D) = \frac{\Delta D}{(1-D)^2} \quad . \tag{6.51}$$

The phase-error sequence then has transform

$$\begin{aligned}
\Phi(D) &= \frac{D-1}{D\left(1 - k_{vco} \cdot F(D)\right) - 1}\Theta(D) & (6.52)\\
&= \frac{D-1}{D\left(1 - k_{vco} \cdot F(D)\right) - 1} \cdot \frac{\Delta D}{(1-D)^2} & (6.53)\\
&= \frac{\Delta D}{(1-D)(1-D(1-\alpha))} & (6.54)
\end{aligned}$$

This steady-state phase error can also be computed by the final value theorem

$$\lim_{k\to\infty} \phi_k = \lim_{D\to 1} (1-D)\Phi(D) = \frac{\Delta}{\alpha} \quad . \tag{6.55}$$

This constant-lag phase error is analogous to the same effect in the continuous-time PLL. Equation (6.55) can be interpreted in several ways. The main result is that the first order loop cannot track a nonzero frequency offset $\Delta = \omega_{offset}T$, in that the phase error cannot be driven to zero. For very small frequency offsets, say a few parts per million of the sampling frequency or less, the first-order loop will incur only a very small penalty in terms of residual phase error (for reasonable α satisfying (6.42)). In this case, the first-order loop may be sufficient in terms of magnitude of phase error, in nearly estimating the frequency offset. In order to stay within the linear range of the modulo-2π phase detector (thereby avoiding cycle slips) after the loop has converged, the magnitude of the frequency offset $|\omega_{offset}|$ must be less than $\frac{\alpha\pi}{T}$. The reader is cautioned not to misinterpret this result by inserting the maximum α (=2) into this result and concluding than any frequency offset can be tracked with a first-order loop as long as the sampling rate is sufficiently high. Increasing the sampling rate $1/T$ at fixed α, or equivalently increasing α at fixed sampling rate, increase the bandwidth of the phase-lock loop filter. Any noise on the incoming phase will be thus less filtered or "less rejected" by the loop, resulting in a lower quality estimate of the phase. A better solution is to often increase the order of the loop filter, resulting in the following second-order phase lock loop.

As an example, let us consider a PLL attempting to track a 1 MHz clock with a local oscillator clock that may deviate by as much as 100 ppm, or equivalently 100 Hz in frequency. The designer may determine that a phase error of $\pi/20$ is sufficient for good performance of the receiver. Then,

$$\frac{\omega_{offset} \cdot T}{\alpha} \leq \frac{\pi}{20} \tag{6.56}$$

or

$$\alpha \geq 40 \cdot f_{offset} \cdot T \quad . \tag{6.57}$$

Then, since $f_{offset} = 100$ Hz, and if the loop samples at the clock speed of $1/T = 10^6$ MHz, then

$$\alpha > 4 \times 10^{-3} \quad . \tag{6.58}$$

Such a small value of α is within the stability bound of $0 < \alpha < 2$. If the phase error or phase estimate are relatively free of any "noise," then this value is probably acceptable. However, if either the accuracy of the clock is less or the sampling rate is less, then an unacceptably large value of α can occur.

[4]Using the relation that $(D)\frac{dF}{dD} \leftrightarrow kx_k$, with $x_k = \Delta \; \forall \; k \geq 0$.

Noise Analysis of the First-Order PLL If the phase input to the PLL (that is θ_k) has some zero-mean "noise" component with variance σ_θ^2, then the component of the phase error caused by the noise is

$$\Phi_n(D) = \frac{1-D}{1-[1-\alpha]\cdot D} \cdot N_\theta(D) \tag{6.59}$$

or equivalently

$$\phi_k = (1-\alpha)\cdot\phi_{k-1} + n_{\theta,k} - n_{\theta,k-1} \quad . \tag{6.60}$$

By squaring the above equation and finding the steady-state constant value $\sigma_{\phi,k}^2 = \sigma_{\phi,k-1}^2 = \sigma_\phi^2$ and setting $E\left[n_{\theta,k}\cdot n_{\theta,k-l}\right] = \sigma_\theta^2 \cdot \delta_l$ via algebra,

$$\sigma_\phi^2 = \frac{\sigma_\theta^2}{1-\alpha/2} \quad . \tag{6.61}$$

Larger values of α create larger response to the input phase noise. If $\alpha \to 2$, the phase error variance becomes infinite. Small values of α are thus attractive, limiting the ability to keep the constant phase error small. The solution is to use the second-order PLL of the next subsection.

Second-Order Phase Lock Loop

In the second-order PLL, the loop filter is an accumulator of phase errors, so that

$$k_{vco}\cdot F(D) = \alpha + \frac{\beta}{1-D} \quad . \tag{6.62}$$

This equation is perhaps better understood by rewriting it in terms of the second order difference equations for the phase estimate

$$\hat{\Delta}_k = \hat{\Delta}_{k-1} + \beta\phi_k \tag{6.63}$$
$$\hat{\theta}_{k+1} = \hat{\theta}_k + \alpha\phi_k + \hat{\Delta}_k \tag{6.64}$$

In other words, the PLL accumulates phase errors into a frequency offset (times T) estimate $\hat{\Delta}_k$, which is then added to the first-order phase update at each iteration. Then

$$\frac{\hat{\Theta}(D)}{\Theta(D)} = \frac{(\alpha+\beta)D - \alpha D^2}{1-(2-\alpha-\beta)D+(1-\alpha)D^2} \quad , \tag{6.65}$$

which has poles $\frac{1}{(1-\frac{\alpha+\beta}{2})\pm\sqrt{(\frac{\alpha+\beta}{2})^2-\beta}}$. For stability, α and β must satisfy,

$$0 \le \alpha < 2 \tag{6.66}$$
$$0 \le \beta < 1 - \frac{\alpha}{2} - \sqrt{\frac{\alpha^2}{2} - 1.5\alpha + 1} \quad . \tag{6.67}$$

Typically, $\beta < \left(\frac{\alpha+\beta}{2}\right)^2$ for real roots, which makes $\beta << \alpha$ since $\alpha+\beta<1$ in most designs.

The second-order loop will track for any frequency deviation between θ_k and $\hat{\theta}_k$. To see this effect, the input phase is again set to

$$\theta_k = \Delta k \quad \forall \;\; k \ge 0 \quad , \tag{6.68}$$

which has D-Transform

$$\Theta(D) = \frac{\Delta D}{(1-D)^2} \quad . \tag{6.69}$$

The phase-error sequence then has transform

$$\Phi(D) = \frac{D-1}{D\left(1-k_{vco}\cdot F(D)\right)-1}\Theta(D) \tag{6.70}$$
$$= \frac{(1-D)^2}{1-(2-\alpha-\beta)D+(1-\alpha)D^2}\cdot\frac{\Delta D}{(1-D)^2} \tag{6.71}$$
$$= \frac{\Delta D}{1-(2-\alpha-\beta)D+(1-\alpha)D^2} \tag{6.72}$$

This steady-state phase error can also be by the final value theorem

$$\lim_{k\to\infty} \phi_k = \lim_{D\to 1} (1-D)\Phi(D) = 0 \quad . \tag{6.73}$$

Thus, as long as the designer chooses α and β within the stability limits, a second-order loop should be able to track any constant phase or frequency offset. One, however, must be careful in choosing α and β to reject noise, equivalently making the second-order loop too sharp or narrow in bandwidth, can also make its initial convergence to steady-state very slow. The trade-offs are left to the designer for any particular application.

Noise Analysis of the Second-Order PLL The power transfer function from any phase noise component at the input to the second-order PLL to the output phase error is found to be:

$$\frac{|\ 1 - e^{-jmath\omega}\ |^4}{|\ 1 - (2-\alpha-\beta)e^{-jmath\omega} + (1-\alpha)e^{2\jmath\omega}\ |^2} \quad . \tag{6.74}$$

This transfer-function can be calculated and multiplied by any input phase-noise power spectral density to get the phase noise at the output of the PLL. Various stable values for α and β (i.e., that satisfy Equations (6.66) and (6.67)) may be evaluated in terms of noise at the output and tracking speed of any phase or frequency offset changes. Such analysis becomes highly situation dependent.

Phase-Jitter Noise Specifically following the noise analysis, a phase noise may have a strong component at a specific frequency (which is often called "phase jitter." Phase jitter often occurs at either the power-line frequency (50-60 Hz) or twice the power line frequency (100-120 Hz) in many systems.[5] In other situations, other radio-frequency or ringing components can be generated by a variety of devices operating within the vicinity of the receiver. In such a situation, the choices of α and β may be such as to try to cause a notch at the specific frequency. A higher-order loop filter might also be used to introduce a specific notch, but overall loop stability should be checked as well as the transfer function.

6.2.3 Phase-locking at rational multiples of the provided frequency

Clock frequencies or phase errors may not always be computed at the frequency of interest. Figure 6.9 illustrates the translation from a phase error measured at frequency $\frac{1}{T}$ to a frequency $\frac{p}{q} \cdot \frac{1}{T} = \frac{1}{T'}$ where p and q are any positive co-prime integers. The output frequency to be used from the PLL is $1/T'$. A high frequency clock with period

$$T'' = \frac{T}{p} = \frac{T'}{q} \tag{6.75}$$

is used so that

$$pT'' = T \tag{6.76}$$

$$qT'' = T' \quad . \tag{6.77}$$

The divides in Figure 6.9 use the counter circuits in Figure 6.6. To avoid cycle slips, the local oscillator should be between $f_{lo} \cdot \big((1-\frac{p}{2}), (1+p)\big)$.

[5]This is because power supplies in the receiver may have a transformer from AC to DC internal voltages in chips or components and it is difficult to completely eliminate leakage of the energy going through the transformer via parasitic paths into other components.

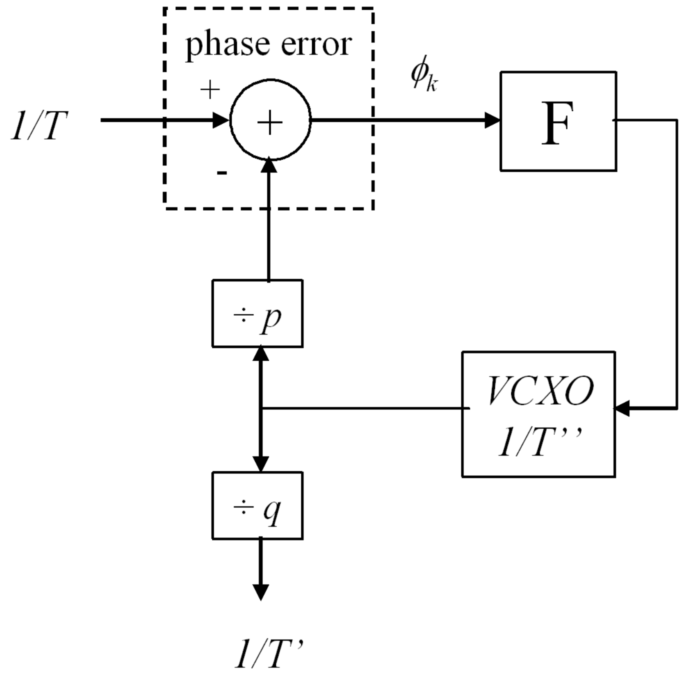


Figure 6.9: Phase-locking at rational fractions of a clock frequency.

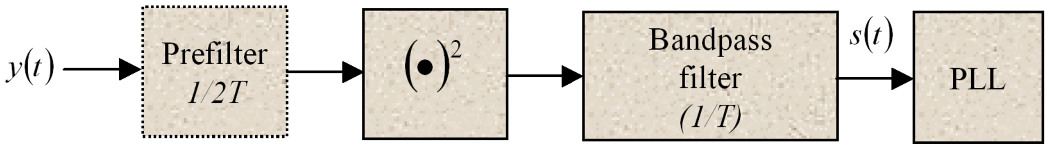


Figure 6.10: Square-law timing recovery.

6.3 Symbol Timing Synchronization

Generally in data transmission, a sinusoid synchronized to the symbol rate is not supplied to the receiver. The receiver derives this sinusoid from the received data. Thus, the unabetted PLL's studied so far would not be sufficient for recovering the symbol rate. The recovery of this symbol-rate sinusoid from the received channel signal, in combination with the PLL, is called **timing recovery**. There are two types of timing recovery. The first type is called **open loop** timing recovery and does not use the receiver's decisions. The second type is called **decision-directed** or **decision-aided** and uses the receiver's decisions. Such metods are an approximation to the ML synchronization in (6.14). Since the recovered symbol rate is used to sample the incoming waveform in most systems, care must be exerted in the higher-performance decision-directed methods that not too much delay appears between the sampling device and the decision device. Such delay can seriously degrade the performance of the receiver or even render the phase-lock loop unstable.

Subsection 6.3.1 begins with the simpler open-loop methods and Subsection 6.3.2 then progresses to decision-directed methods.

6.3.1 Open-Loop Timing Recovery

Probably the simplest and most widely used timing-recovery method is the **square-law timing-recovery** method of Figure 6.10. The present analysis ignores the optional prefilter momentarily, in which case the nonlinear squaring device produces at its output

$$y^2(t) = \left[\sum_m x_m p(t-mT) + n(t)\right]^2 \quad . \tag{6.78}$$

The expected value of the square output (assuming, as usual, that the successively transmitted data symbols x_m are independent of one another) is

$$\begin{aligned} E\left\{y^2(t)\right\} &= \sum_m \sum_n \mathcal{E}_{\boldsymbol{x}} \cdot \delta_{mn} \cdot p(t-mT) \cdot p(t-nT) + \sigma_n^2 & (6.79)\\ &= \mathcal{E}_{\boldsymbol{x}} \cdot \sum_m p^2(t-mT) + \sigma_n^2 \quad , & (6.80)\end{aligned}$$

which is periodic, with period T. The bandpass filter attempts to replace the statistical average in (6.80) by time-averaging. Equivalently, one can think of the output of the square device as the sum of its mean value and a zero-mean noise fluctuation about that mean,

$$y^2(t) = E\left\{y^2(t)\right\} + \left(y^2(t) - E\left\{y^2(t)\right\}\right) \quad . \tag{6.81}$$

The second term can be thought of as noise as far as the recovery of the timing information from the first term. The bandpass filter tries to reduce this noise. The instantaneous values for this noise depend upon the transmitted data pattern and can exhibit significant variation, leading to what is sometimes called "data-dependent" timing jitter. The bandpass filter and PLL try to minimize this jitter. Sometimes "line codes" (or basis functions) are designed to insure that the underlying transmitted data pattern results in significantly less data-dependent jitter effects. Line codes are seqential encoders (see Chapter

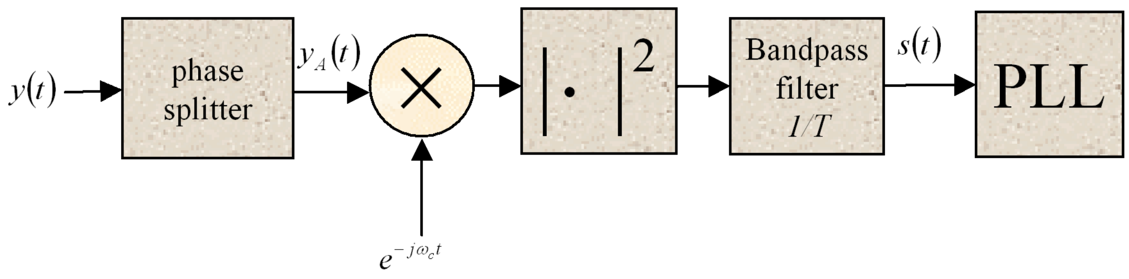


Figure 6.11: Envelope timing recovery.

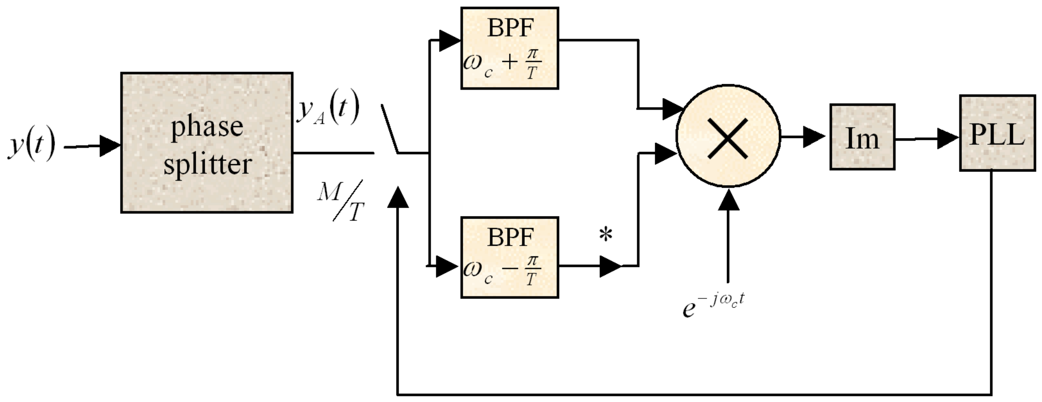


Figure 6.12: Bandedge timing recovery.

8) that use the system state to amplify clock-components in any particular stream of data symbols. Well-designed systems typically do not need such codes, and they are thus not addressed in this text.

While the signal in (6.80) is periodic, its amplitude may be small or even zero, depending on $p(t)$. This amplitude is essentially the value of the frequency-domain convolution $P(f) * P(f)$ evaluated at $f = 1/T$ – when $P(f)$ has small energy at $f = 1/(2T)$, then the output of this convolution typically has small energy at $f = 1/T$. In this case, other "even" nonlinear functions can replace the squaring circuit. Examples of even functions include a fourth-power circuit, and absolute value. Use of these functions may be harder to analyze, but basically their use attempts to provide the desired sinusoidal output. The effectiveness of any particular choice depends upon the channel pulse response. The prefilter can be used to eliminate signal components that are not near $1/2T$ so as to reduce noise further. The PLL at the end of sqare-law timing recovery works best when the sinusoidal component of $1/2T$ at the prefilter output has maximum amplitude relative to noise amplitude. This maximum amplitude occurs for a specific symbol-clock phase. The frequency $1/2T$ is the "bandedge". Symbol-timing recovery systems often try to select a timing phase (in addition to recovering the correct clock frequency) so that the bandedge component is maximized. Square-law timing recovery by itself does not guarantee a maximum bandedge component.

For QAM data transmission, the equivalent of the square-law timing recovery is known as **envelope timing recovery,** and is illustrated in Figure 6.11. The analysis is basically the same as the real baseband case, and the nonlinear element could be replaced by some other nonlinearity with real output, for instance the equivalent of absolute value would be the magnitude (square root of the sum of squares of the two real and imaginary inputs). A problem with envelope timing is the à prior need for the carrier.

One widely used method for avoiding the carrier-frequency dependency is the so-called "bandedge" timing recovery in Figure 6.12. The two narrowband bandpass filters are identical. Recall from the earlier discussion about the fractionally spaced equalizer in Chapter 3 that timing-phase errors could lead to aliased nulls within the Nyquist band. The correct choice of timing phase should lead (approximately) to a maximum of the energy within the two bandedges. This timing phase would also have maximized the energy into a square-law timing PLL. At this band-edge maximizing timing phase, the output of the multiplier following the two bandpass filters in Figure 6.12 should be maximum, meaning that quantity should be real. The PLL forces this timing phase by using the samples of the imaginary part of the multiplier output as the error signal for the phase-lock loop. While the carrier frequency is presumed to be known in the design of the bandpass filters, knowledge of the exact frequency and phase is not necessary in filter design and is otherwise absent in "bandedge" timing recovery.

6.3.2 Decision-Directed Timing Recovery

Decisions can also be used in timing recovery. A common decision-directed timing recovery method minimizes the mean-square error, over the sampling time phase, between the equalizer (if any) output and the decision, as in Figure 6.13. That is, the receiver chooses τ to minimize

$$J(\tau) = E\left\{|\hat{x}_k - z(kT+\tau)|^2\right\} \quad , \tag{6.82}$$

where $z(kT+\tau)$ is the equalizer output (LE or DFE) at sampling time k corresponding to sampling phase τ. The update uses a stochastic-gradient estimate of τ in the opposite direction of the unaveraged derivative of $J(\tau)$ with respect to τ. This derivative is (letting $\epsilon_k \stackrel{\Delta}{=} \hat{x}_k - z(kT+\tau)$)

$$\frac{dJ}{d\tau} = \Re\left[E\left\{2\epsilon_k^* \cdot (-\frac{dz}{d\tau})\right\}\right] \quad . \tag{6.83}$$

The (second-order) update is then

$$\begin{aligned} \tau_{k+1} &= \tau_k + \alpha \cdot \Re\{\epsilon_k^* \cdot \dot{z}\} + T_k & (6.84) \\ T_k &= T_{k-1} + \beta \cdot \Re\{\epsilon_k^* \cdot \dot{z}\} \quad . & (6.85) \end{aligned}$$

This type of decision-directed phase-lock loop is illustrated in Figure 6.13. There is one problem in the implementation of the decision-directed loop that may not be obvious upon initial inspection of Figure 6.13: the implementation of the differentiator. The problem of implementing the differentiator is facilitated if the sampling rate of the system is significantly higher than the symbol rate. Then the differentiation can be approximated by simple differences between adjacent decision inputs. However, a higher sampling rate can significantly increase system costs.

Another symbol-rate sampling approach is to assume that z_k corresponds to a bandlimited waveform within the Nyquist band:

$$z(t+\tau) = \sum_m z_m \cdot \text{sinc}(\frac{t+\tau-mT}{T}) \quad . \tag{6.86}$$

Then the derivative is

$$\begin{aligned} \frac{d}{dt}z(t+\tau) &\stackrel{\Delta}{=} \dot{z}(t+\tau) \\ &= \sum_m z_m \cdot \frac{d}{dt}\text{sinc}(\frac{t+\tau-mT}{T}) & (6.87) \\ &= \sum_m z_m \cdot \frac{1}{T} \cdot \left[\frac{\cos\left(\frac{\pi(t+\tau-mT)}{T}\right)}{\frac{t+\tau-mT}{T}} - \frac{\sin\left(\frac{\pi(t+\tau-mT)}{T}\right)}{\pi\left(\frac{t+\tau-mT}{T}\right)^2}\right] & (6.88) \end{aligned}$$

which if evaluated at sampling times $t = kT - \tau$ simplifies to

$$\dot{z}(kT) = \sum_m z_m \cdot \left(\frac{(-1)^{k-m}}{(k-m)T}\right) \tag{6.89}$$

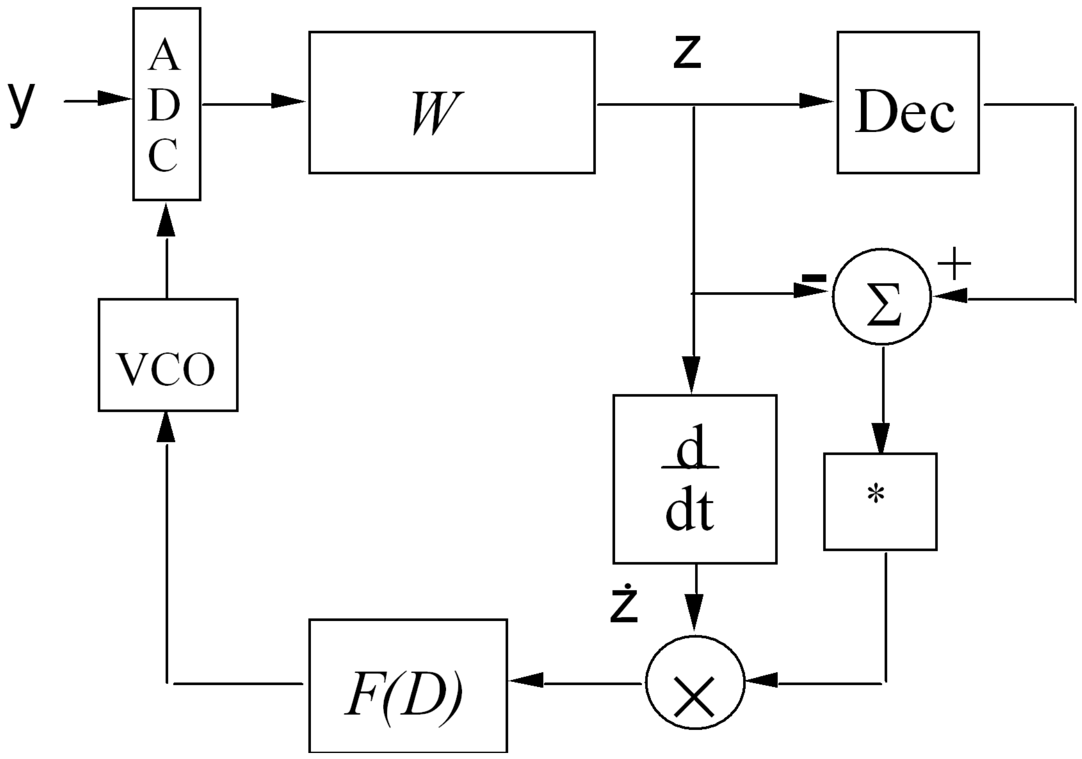


Figure 6.13: Decision-directed timing recovery.

or

$$\dot{z}_k = z_k * g_k \quad , \tag{6.90}$$

where

$$g_k \triangleq \begin{cases} 0 & k = 0 \\ \frac{(-1)^k}{kT} & k \neq 0 \end{cases} \quad . \tag{6.91}$$

The problem with such a realization is the length of the filter g_k. The delay in realizing such a filter could seriously degrade the overall PLL performance. Sometimes, the derivative can be approximated using the two terms g_{-1} and g_1 by

$$\dot{z}_k = \frac{z_{k+1} - z_{k-1}}{2T} \quad . \tag{6.92}$$

The Timing Function and Baud-Rate Timing Recovery The timing function is defined as the expected value of the error signal supplied to the loop filter in the PLL. For instance, in the case of the decision-directed loop using $\dot{z}_k$ in (6.92) the timing function is

$$E\left\{u(\tau)\right\} = \Re\left[E\left\{\epsilon_k^* \cdot \frac{z_{k+1} - z_{k-1}}{T}\right\}\right] = \Re\frac{1}{T}\left[E\left\{\hat{x}_k^* \cdot z_{k+1} - \hat{x}_k^* \cdot z_{k-1}\right\} + E\left\{z_k^* \cdot z_{k-1} - z_k^* \cdot z_{k+1}\right\}\right] \tag{6.93}$$

the real parts of the last two terms of (6.93) cancel, assuming that the variation in sampling phase, τ, is small from sample to sample. Then, (6.93) simplifies to

$$E\{u(\tau)\} = \mathcal{E}_{\boldsymbol{x}} \cdot [p(\tau - T) - p(\tau + T)] \quad , \tag{6.94}$$

essentially meaning the phase error is zero at a symmetry point of the pulse. The expectation in (6.94) is also the mean value of

$$u(\tau) = \hat{x}_k^* \cdot z_{k-1} - \hat{x}_{k-1}^* \cdot z_k \quad , \tag{6.95}$$

which can be computed without delay.

In general, the timing function can be expressed as

$$u(\tau) = G(\hat{\boldsymbol{x}}_k) \bullet \boldsymbol{z}_k \tag{6.96}$$

so that $E[u(\tau)]$ is as in (6.94). The choice of the vector function G for various applications can depend on the type of channel and equalizer (if any) used, and $\boldsymbol{z}_k$ is a vector of current and past channel outputs.

6.3.3 Pilot Timing Recovery

In pilot timing recovery, the transmitter inserts a sinusoid of frequency equal to q/p times the the desired symbol rate. The PLL of Figure 6.9 can be used to recover the symbol rate at the receiver. Typically, pilots are inserted at unused frequencies in transmission. For instance, with OFDM and DMT systems in Chapter 4, a single tone may be used for a pilot. For instance, in 802.11(a) WiFi systems, 5 pilot tones are inserted (in case up to 4 of them do not pass through the unknown channel). The effect of jitter and noise are largely eliminated because the PLL sees no data-dependent jitter at the pilot frequency if the receiver filter preceding the PLL is sufficiently narrow.

In some transmission systems, the pilot is added at the Nyquist frequency exactly 90 degrees out of phase with the nominal +,-,+,- sequence that would be obtained by sampling in phase at the sampling (symbol in QAM) clock. Insertion 90 degrees out of phase means that the band-edge component is maximized when then the samples see zero energy at this frequency.

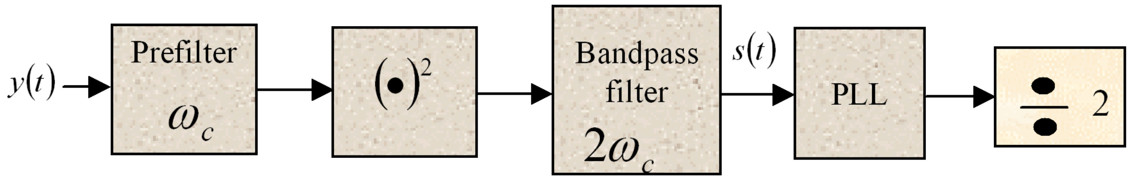


Figure 6.14: Open-loop carrier recovery.

6.4 Carrier Recovery

Again in the case of carrier recovery, there are again two basic types of carrier recovery in data transmission, open-loop and decision directed, as in Sections 6.4.1 and 6.4.2.

6.4.1 Open-Loop Carrier Recovery

A model for the modulated signal with some unknown offset, θ, in the phase of the carrier signal is

$$y(t) = \Re\left\{\sum_m x_m \cdot p(t - mT) \cdot e^{\jmath(\omega_c t+\theta)}\right\} \quad , \tag{6.97}$$

or equivalently

$$y(t) = \frac{y_A(t) + y_A^*(t)}{2} \quad , \tag{6.98}$$

where

$$y_A(t) = \sum_m x_m \cdot p(t - mT) \cdot e^{\jmath(\omega_c t+\theta)} \quad . \tag{6.99}$$

The autocorrelation function for y is

$$\begin{aligned} r_y(\tau) &= E\left\{y(t)y(t+\tau)\right\} & (6.100)\\ &= E\left\{\left(\frac{y_A(t)+y_A^*(t)}{2}\right)\left(\frac{y_A(t+\tau)+y_A^*(t+\tau)}{2}\right)\right\} & (6.101)\\ &= \frac{1}{4}E\left[y_A(t)\cdot y_A^*(t+\tau) + y_A^*(t)\cdot y_A(t+\tau)\right] & \\ &+ \frac{1}{4}E\left[y_A(t)\cdot y_A(t+\tau) + y_A^*(t)\cdot y_A^*(t+\tau)\right] & (6.102)\\ &= \frac{1}{2}\Re\left[r_{y_A}(\tau)\right] + \frac{1}{2}\Re\left[r_{y_A}(\tau)\cdot e^{2\jmath(\omega_c t+\theta)}\right] \quad . & (6.103) \end{aligned}$$

The average output of a squaring device applied to $y(t)$ is thus

$$E\left\{y^2(t)\right\} = \frac{1}{2}\Re\left[r_{y_A}(0)\right] + \frac{1}{2}\Re\left[r_{y_A}(0)e^{2\jmath(\omega_c t+\theta)}\right] \quad , \tag{6.104}$$

which is a sinusoid at twice the carrier frequency. The square-law carrier-recovery circuit is illustrated in Figure 6.14. Note the difference between this circuitry and the envelope timing recovery, where the latter squares the magnitude of the complex baseband equivalent for $y(t)$, whereas the circuit in Figure 6.14 squares the channel output directly (after possible prefiltering). The bandpass filter again tries to average any data-dependent jitter components (and/or noise) from the squared signal, and the PLL can be used to further tune the accuracy of the sinusoid. The output frequency is double the desired carrier frequency and is divided by 2 to achieve the carrier frequency. The division is easily implemented with a single "flip-flop" in digital circuitry. The pre-filter in Figure 6.14 can be used to reduce noise entering the PLL.

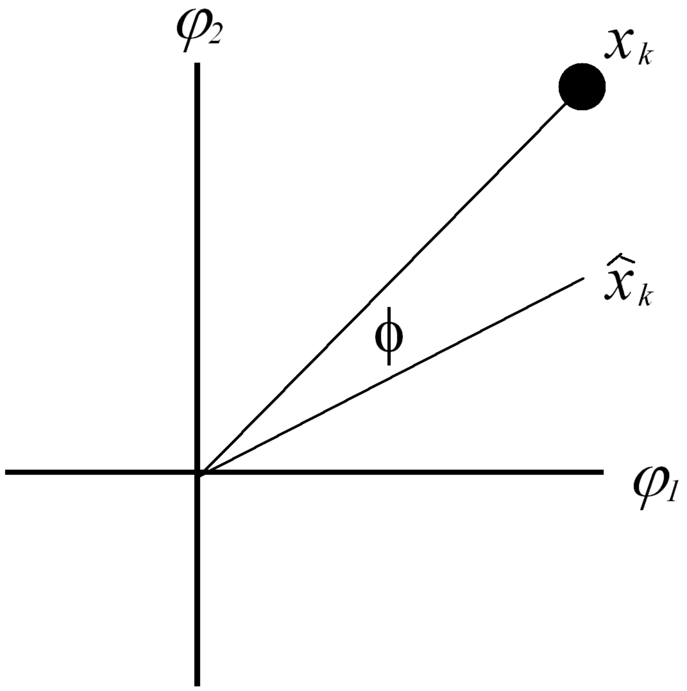


Figure 6.15: Decision-directed phase error.

6.4.2 Decision-Directed Carrier Recovery

Decision-directed carrier recovery is more commonly encountered in practice than open loop carrier recovery. The basic concept used to derive the error signal for phase locking is illustrated in Figure 6.15. The decision-device input $\hat{x}_k$ is not exactly equal to the decision-device output x_k. Since

$$x_k = a_k + \jmath b_k \;\; ; \;\; \hat{x}_k = \hat{a}_k + \jmath \hat{b}_k \quad , \tag{6.105}$$

then

$$\begin{aligned} \frac{x_k}{\hat{x}_k} &= \frac{|x_k|}{|\hat{x}_k|} \cdot e^{\jmath \phi_k} && (6.106) \\ &= \frac{a_k + \jmath b_k}{\hat{a}_k + \jmath \hat{b}_k} && (6.107) \\ &= \frac{\left(a_k \hat{a}_k + b_k \hat{b}_k\right) + \jmath \left(\hat{a}_k b_k - a_k \hat{b}_k\right)}{\hat{a}_k^2 + \hat{b}_k^2} \quad , && (6.108) \end{aligned}$$

which leads to the result

$$\begin{aligned} \phi_k &= \arctan \frac{\hat{a}_k b_k - a_k \hat{b}_k}{a_k \hat{a}_k + b_k \hat{b}_k} && (6.109) \\ &\approx \arctan \frac{1}{\mathcal{E}_{\boldsymbol{x}}} \left(\hat{a}_k b_k - a_k \hat{b}_k\right) && (6.110) \\ &\approx \frac{1}{\mathcal{E}_{\boldsymbol{x}}} \left(\hat{a}_k b_k - a_k \hat{b}_k\right) && (6.111) \\ &\propto \left(\hat{a}_k b_k - a_k \hat{b}_k\right) \quad , && (6.112) \end{aligned}$$

with the approximation being increasingly accurate for small phase offsets. Alternatively, a look-up table for arcsin or for arctan could be used to get a more accurate phase-error signal ϕ_k. For large constellations, the phase error must be smaller than that which would cause erroneous decisions most of the time. Often, a training data pattern is used initially in place of the decisions to converge the carrier recovery before switching to unknown data and decisions. Using known training patterns, the phase error should not exceed π in magnitude since a modulo-2π detector is implied.

6.4.3 Pilot Carrier Recovery

In pilot carrier recovery, the transmitter inserts a sinusoid of frequency equal to q/p times the the desired cariier frequency. The PLL of Figure 6.9 can be used to recover the carrier rate at the receiver. Typically, pilots are inserted at unused frequencies in transmission. For instance, with OFDM and DMT systems in Chapter 4, a single tone may be used for a pilot. In this case, if the timing reference and the carrier are locked in some known rational relationship, recover of one pilot supplies both signals using dividers as in Figure 6.9. In some situations, the carrier frequencies may apply outside of the modem box itself (for instance a baseband feed of a QAM signal for digital TV) to a broadcaster who elsewhere decides the carrier frequency (or channel) and there provides the carrier translation of frequency band). The 5 pilors 802.11(a) WiFi systems also provide for the situation where the symbol clock and carrier frequency may not appear the same even if locked because of movement in a wireless environment (leading to what is called "Doppler Shift" of the carrier caused by relative motion). Then at least two pilots would be of value (and actually 5 are used to provide against loss of a few). The effect of jitter and noise are largely eliminated because the PLL sees no data-dependent jitter at the pilot frequency if the receiver filter preceding the PLL is sufficiently narrow.

6.5 Frame Synchronization in Data Transmission

While carrier and symbol clock may have been well established in a data transmission system, the boundary of long symbols (say in the use of the DMT systems of Chapter 4 or the GDFE systems of Chapter 5) or packets may not be known to a receiver. Such synchronization requires searching for some known pattern or known characteristic of the transmitted waveform to derive a phase error (which is typically measured in the number of sample periods or dimensions in error). Such synchronization, once established, is only lost if some kind of catastrophic failure of other mechanisms has occured (or dramatic sudden change in the channel) because it essentially involves counting the number of samples from the recovered timing clock.

6.5.1 Autocorrelation Methods

Autocorrelation methods form the discrete channel output autocorrelation

$$\boldsymbol{R_{yy}}(l) = E\left[y_k y_{k-l}^*\right] = h_l * h_{-l}^* * \boldsymbol{R_{xx}}(l) + \boldsymbol{R_{nn}}(l) \tag{6.113}$$

from channel output samples by time averaging over some interval of M samples

$$\hat{\boldsymbol{R_{yy}}}(l) = \frac{1}{M}\sum_{m=1}^{M}\sum_{m=1}^{M} y_m y_{m-l}^* \quad . \tag{6.114}$$

In forming such a sum, the receiver implies a knowledge of sampling times $1, ..., M$. If those times are not coincident with the corresponding (including channel delay) positions of the packet in the transmitter, then the autocorrelation function will look shifted. for instance, if a peak was expected at position l and occurs instead at position l', then the difference is an indication of a packet timing error. The phase error used to drive a PLL is that difference. Typically, the VCO in this case supplies only discrete integer-number-of-sample steps. If noise is relatively small and h_l is known, then one error may be sufficient to find the packet boundary. If not, many repeated estimates of $\hat{\boldsymbol{R_{yy}}}$ at different phase adjustments may be tried.

There are 3 fundamental issues that affect the performance of such a scheme:

1. the knowledge of or choice of the input sequence x_k,
2. the channel response's (h_k's) effect upon the autocorrelation of the input,
3. the noise autocorrelation.

Synchronization Patterns

a synchronization patter or "synch sequence" is some known transmitted signal, typically with properties that create a large peak in the autocorrelation function at the channel output at a known lag l. Typically with small (or eliminated) ISI, such a sequence corresponds to a white input. Simple random data selection may often not be sufficient to guarantee a high autocorrelation peak for a short averaging period M. Thus, special sequences are often selected that have a such a short-term peaked time-averaged autocorrelation. A variety of sequences exist.

The most common types are between 1 and 2 cycles of a known period pattern. Pseudorandom binary sequences of degree p have period of $2^p - 1$ samples are those specified by their basic property that cyclic shifts of the sequence have each length-p binary patter (except all 0's) once and only once. These sequences are sometimes used in the spreading patterns from CDMA methods in the appendix of Chapter 5 also. A simple linear-feedback register implementation appears there. Here in Chapter 6, the important property is that the autocorrelation (with binary antipodal modulation) of such sequences is

$$\hat{\boldsymbol{R_{xx}}}(l) = \begin{cases} 1 & l = 0 \\ -\frac{1}{2^p-1} & l = 1, ..., 2^p - 1 \end{cases} \quad . \tag{6.115}$$

For reasonably long p, the peak is easily recognizable if the channel output has little or no distortion. In fact, the entire recognition of the lag can be positioned as a basic Chapter 1 detection problem, where each of the possible input shifts of the same known training sequence is to be detected. Because all these patterns have the same energy and they're almost orthogonal, a simple largest matched filter ouptut (the matched filter outputs are the computation of the autocorrelation at different time lags) with the probability of error well understood in Chapter 1.

Another synchronization pattern that can have appeal is the so-called "chirp" sequence

$$x_k = e^{\jmath 2\pi k^2/M} \tag{6.116}$$

which also has period M. It also has a single peak at time zero of size 1 and zero autocorrelation. It is harder to generate and requires a complex QAM baseband system, but in some sense is a perfect synch pattern.

A third alternative are the so-called "Barker codes" that are very short and not repeated and designed to still have peakiness in the face of unknown preceding and succeeding data surrounding the pattern. Such patterns may be periodically inserted in a transmission stream so that if the receiver for any reason lost packet synchronization (or a new received came on after a first had already acquired the signal if more than receiver scan the same or related channels), then the calculation of autocorrelation would immediatley recommence until the peak was found. An example of a good short synchronization pattern is the 7-symbol Barker code, which is forinstance used with 2B1Q transmission in some symmetric DSL systems that employ PAM modulation. This code transmits at maximum amplitude the binary pattern

$$+++--+- \tag{6.117}$$

(or its time reverse on loop 2 of HDSL systems) so $M = 7$ and only binary PAM is used when this sequence is inserted (which means the dmin is increase by 7 dB for a detection problem, simply because 2 levels replace 4 already). This pattern has a maximum $M \cdot \hat{\boldsymbol{R}}_{\boldsymbol{xx}}$ value of 7 units when aligned. The 13 place autocorrelation function is

$$1010107010101 \quad . \tag{6.118}$$

Because there are adjacent transmissions, the minimum distance of 7 units (may be reduced to 6 or 5, as long as the searching receiver has some idea when to look) multiplies a distance that is already 74 dB better than 4 level transmission. Thus, this pattern can be recoved quickly, even in a lot of noise.

Creative engineers can create patterns that will be easy to find in any given application.

The channel response

Severe ISI can cause a loss of the desired autocorrelation properties at the channel output. The usual situation is that ISI is not so severe because other systems are already in use to reduce it by the time that synchronization occurs. An exception would be when a receiver is looking for a training sequence to adapt its equalizers (see Chapter 7) initially and thus, time-zero for that training is necessary. In such a case, the cross-correlation of the channel output with the various delayed versions of the known training pattern can produce an estimate of h_l,

$$\frac{1}{M}\sum_{m=1}^{M} y_k x^*_{k-l} \approx h_l \tag{6.119}$$

so that the channel can be identified. Then the resultant channel can be used in a maximum likelihood search over the delayed patterns in the synchronization symbol to estimate best initial packet boundary.

The noise

Large or strongly correlated noise can cause the estimated autocorreleation to be poor. The entire channel output yk should be whitened (as in Chapter 5) with a linear predictor. An adaptive whitening (linear predictor) filter may be necessary as in Chapter 7. In this case, again the cross-correlation estimate of Equation (6.119) can be used to estimate the euqivalent channel after oveall whitening and the procedure of the previous section for ISI followed.

6.5.2 Searcing for the guard period or pilots

Large noise essentially requires the cross-correlation of channel output and input as in the previous section. Repeated versions of the synch patter may be required to get acceptable acquisition of packet boundary.

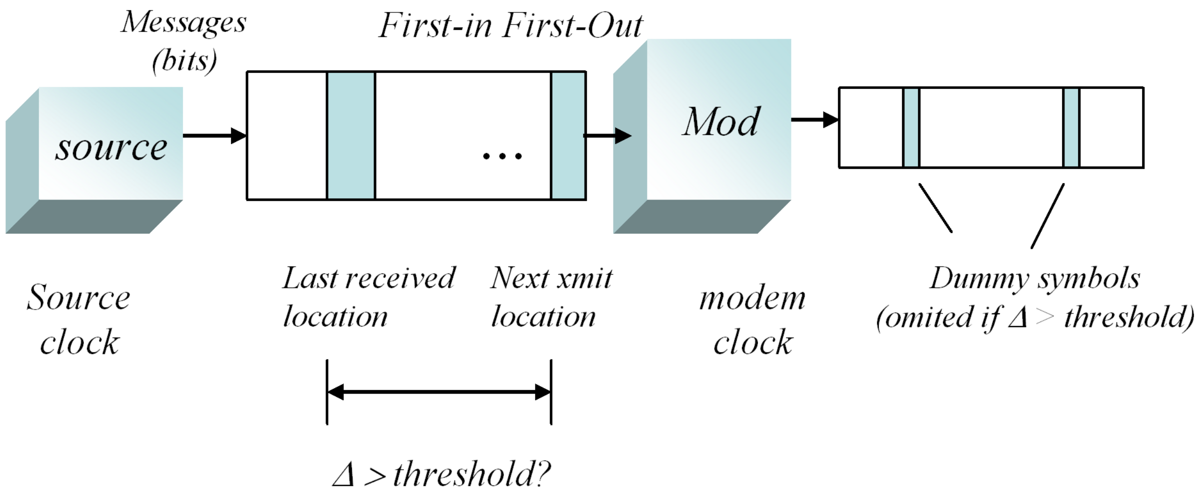


Figure 6.16: Rob-Stuff Timing.

6.6 Pointers and Add/Delete Methods

This final section of Chapter 6 addresses the question, "well fine, but how did the source message clock get synchronized to the symbol clock?" The very question itself recognizes and presumes that a bit stream (or message stream) from some source probably does not also have a clock associated with it (some do, some don't). The receiver of the message would also in some cases like to have that source clock and not the receiver's estimates of the transmitter's symbol and carrier clocks. Typically, the source cannot be relied upon to supply a clock, and even if it does, the stability of that clock may not match the requiremens for the heavily modulated and coded transmission system of Chapters 1-5.

Generally, the source clock can be monitored and essentially passed through the monitoring of interface buffers that collect (or distribute) messages. Figure 6.16 illustrates the situation where certain tolerances on the input clock of the source may be known but the symbol clock is not synchronized and only knows the approximate bit rate it must transmit. For such situations, the transmit symbol clock must always be the higher clock, sufficiently so that even if at its lowest end of the range of speed, it still exceeds the transmit clock. To match the two clocks, regularly inserted "dummy" messages or bits are sometimes not sent to reduce the queue depth awaiting transfer. These methods are known as "rob/stuff" or "add/delete" timing and sychronization methods.

The principle is simple. The buffer depth is monitored by subtracting the addresses of the next-to-transmit and last-in bits. If this depth is reducing or steady while above a safety threshold, then dummy bits are used and transmitted. If the depth starts to increase, then dummy bits are not transmitted until the queue depth exceeds a threshold.

The same process occurs at the receiver interface to the "sink" of the messages. The sink must be able to receive at least the data rate of the link and typically robs or stuffs are used to accelerate or slow the synch clock. A buffer is used in the same way. The sink's lowest clock frequency must be within the range of the highest output clock of the receiver in the transmission system.

The size of the queue in messages (or bits) needs to be at least as large as the maximum expected difference in source and transmit bit-rate (or message-rate) speeds times the the time period between potential dummy deletions, which are known as "robs." ('Stuffs are the usual dummies transmitted). The maximum speed of the transmission system with all robs must exceed the maximum source supply of messages or bits.

Add/drop or Add/delete methods are essentially the same and typically used in what are known as asynchronous networks (like ATM for Asynchronous Transfer Mode, which is really not asynchronous at all at the physically layer but does have gaps between packets that carry dummy data that is ignored and not passed through the entire system. ATM networks typically do pass an 8 kHz network clock reference throughout the entire system. This is done with a "pointer." The pointer typically is a position in a high-speed stream of ATM packets that says "the difference in time between this pointer and the last one you received is exactly 125 μs of the desired network clock. Various points along the entire path can then use the eventual received clock time at which that pointer was received to synchronize to the network clock. If the pointer arrives sooner than expected, the locally generated network reference is too slow and the phase error of the PLL is negative; if arrival is late, the phase error is positive.

Exercises - Chapter 5

6.1 *Phase locked loop error induction.*

a. (1 pt)

Use the D-transform equation

$$\frac{\hat{\Theta}(D)}{\Theta(D)} = \frac{\alpha D}{1-(1-\alpha)D}$$

to conclude that

$$\hat{\theta}_{k+1} = \hat{\theta}_k + \alpha\phi_k$$

where $\phi_k = \theta_k - \hat{\theta}_k$.

b. (3 pts)

Use your result above to show by induction that when $\theta_k = \theta_0$ and $\hat{\theta}_0 = 0$, we have:

$$\phi_k = (1-\alpha_k)^k\theta_0.$$

Note that this was shown another way on page 18 of Chapter 5.

c. (3 pts)

Use the result of part (a) to show (again by induction) that when $\theta_k = \theta_0 + k\Delta$ and $\hat{\theta}_0 = 0$, we have:

$$\phi_k = (1-\alpha_k)^k\theta_0 + \Delta\sum_{n=0}^{k-1}(1-\alpha)^n.$$

This confirms the result of equation 5.49 that ϕ_k converges to Δ/α.

6.2 *Second order PLL update equations.*

Starting from the transfer function for the phase prediction, we will derive the second order phase locked loop update equations.

a. (2 pts) Using only the transfer function

$$\frac{\hat{\Theta}(D)}{\Theta(D)} = \frac{(\alpha+\beta)D - \alpha D^2}{1-(2-\alpha-\beta)D + (1-\alpha)D^2}$$

show that

$$\frac{\Phi(D)}{\hat{\Theta}(D)} = \frac{(1-D)^2}{(\alpha+\beta)D - \alpha D^2}.$$

Recall that $\phi_k = \theta_k - \hat{\theta}_k$.

b. (2 pts) Use the above result to show that

$$\hat{\Theta}(D) = D\hat{\Theta}(D) + \alpha D\Phi(D) + \frac{D\beta\Phi(D)}{1-D}.$$

c. (3 pts)

Now, defining (as in the notes)

$$\hat{\Delta}(D) = \frac{\beta\Phi(D)}{1-D}$$

show that

$$\begin{aligned} \hat{\theta}_k &= \hat{\theta}_{k-1} + \alpha\phi_{k-1} + \hat{\Delta}_{k-1} && (6.120) \\ \hat{\Delta}_k &= \hat{\Delta}_{k-1} + \beta\phi_k. && (6.121) \end{aligned}$$

6.3 *PLL Frequency Offset - Final 1996*

A discrete-time PLL updates at 10 MHz and has a local oscillator frequency of 1 MHz. This PLL must try to phase lock to an almost 1 MHz sinusoid that may differ by 100 ppm (parts per million - so equivlaently a .01% difference in frequency). A first-order filter is used with α as given in this chapter. The phase error in steady-state operation of this PLL should be no more than .5% (.01π).

a. Find the value of α that ensures $|\phi| < .01(\pi)$. (2 pts)

b. Is this PLL stable if the 1 MHz is changed to 10 GHz? (1 pt)

c. Suppose this PLL were unstable - how could you fix part b without using to a 2^{nd}-order filter? (2 pts)

6.4 *First-order PLL - Final 1995*

A receiver samples channel output $y(t)$ at rate $2/T$, where T is the transmitter symbol period. This receiver only needs the clock frequency (not phase) exactly equal to $2/T$, so a first-order PLL with constant phase error would suffice for timing recovery. (update rate of the PLL is 400 kHz)

The transmit symbol rate is 400 kHz $\pm$50 ppm. The receiver VCXO is 800 kHz $\pm$50 ppm.

a. Find the greatest frequency difference of the transmit $2/T$ and the receiver $2/T$.

b. Find the smallest value of the FOPLL α that prevents cycle slipping.

c. Repeat part b if the transmit clock accuracy and VCO accuracy are 1%, instead of 50 ppm. Suggest a potential problem with the PLL in this case.

6.5 *Phase Hits*

A phase hit is a sudden change in the phase of a modulated signal, which can be modeled as a step-phase input into a phase-lock loop, without change in the frequency before or after the step in phase. Let a first-order discrete-time PLL operate with a sampling rate of 1 kHz (i.e., correct phase errors are correctly supplied somehow 1000 times per second) on a sinusoid at frequency approximately 640 Hz, which may have phase hits. Quick reaction to a phase hit unfortunately also forces the PLL to magnify random noise, so there is a trade-off between speed of reaction and noise immunity.

a. What is the maximum phase-hit magnitude (in radians) with which the designer must be concerned? (1 pt)

b. What value of the PLL parameter α will allow convergence to a 1% phase error (this means the phase offset on the 640 Hz signal is less than .02π) in just less than 1 s? Assume the value you determine is just enough to keep over-reaction to noise under control with this PLL. (2 pts)

c. For the α in part b, find the steady-state maximum phase error if the both the local oscillator and the transmitted clock each individually have 50 ppm accuracy (hint: the analysis for this part is independent of the phase hit)? What percent of the 640 Hz sinusoid's period is this steady-state phase error? (2 pts)

d. How might you improve the overall design of the phase lock loop in this problem so that the 1% phase error (or better) is maintained for both hits (after 1s) and for steady-state phase error, without increasing noise over-reaction? (1 pt)

6.6 *PLL - Final 2003 - 8 pts*

A square-law timing recovery system with symbol rate $1/T = 1$ produces a sinousoid with a jitery or noisy phase of $\theta = \Delta_0 \cdot k + \theta_0 + n_k$ where n_k is zero-mean Gaussian with variance σ_θ^2.

a. If a FO PLL with parameter α is used, what is a formula for $E\left[\hat{\theta}_{infty}\right]$? (1 pt)

b. What is a formula for the mean-square steady-state value of this FO PLL phase error? (1 pt)

c. If the symbol-time sampling error ϵ is small compared to a symbol interval so that $e^{\jmath 2\pi f \epsilon} \approx 1 + \jmath 2\pi f \epsilon$ for all frequencies of interest, what largest value of ϵ would insure less than .1 dB degradation in an equalizer/receiver output error of 13.5 dB? (2 pts)

d. With $\Delta = .1$ and $\sigma_\theta^2 = (.005 \cdot 2\pi)^2$, is there a value of the FO PLL for which the SNR is satisfied in part b? If so, what is it? (1 pt)

e. Propose a better design. Try to estimate the new phase-error variance of your design. (3 pts)

The next 4 problems are courtesy of 2006 EE379A student Jason Allen.

6.7 *Frequency offset study- 10 pts*

A data signal is multiplied by a sinusoidal carrier with a frequency f_c to be transmitted over a wireless communications channel. At the receiver this multiplication must be reversed to recover the baseband data signal. Ideally such recovery can be accomplished by multiplying the received data signal by a locally generated sinusoid whose frequency is also f_c Hz, and whose phase is identical to that of the carrier signal used at the transmitter. After passing the resultant signal through a lowpass filter, the original signal is recovered. In reality, the locally generated sinusoid will contain both a frequency and a phase offset. Let

$$f_{LO,Rx} = f_{LO,Tx} + \Delta f$$

represent the Local Oscillator (LO) at the receiver where Δf represents the frequency offset between $f_{LO,Rx}$ and $f_{LO,Tx}$. Ignoring the phase offset and using your basic knowledge of frequency mixing, find:

a. f_{sum}, the frequency of the sum-frequency term (sometimes called the "sum image"). (1 pt)

b. f_{diff}, the frequency of the difference image. (1 pt)

c. What are the results after lowpass filtering? (2 pts)

d. Is the downconverted data signal still at baseband? (1 pt)

e. Denoting the transmitted signal as $x(t)$ with an equivalent Fourier transform of $X(f)$, the "near-baseband" signal can be denoted as $X(f - \Delta f)$. What is the time domain representation of the "near-baseband" signal? (2 pts)

f. What effect will this have on the data signal's constellation? (1 pt)

g. Will this effect on the data signal constellation also effect the best symbol-by-symbol decision device (slicer)? (2 pts)

6.8 *Quadrature Detector- 5 pts*

a. Let $\hat{x}(t)$ be a quadrature input sinusoid given by

$$x(t) = cos(\theta_x(t)) + jsin(\theta_x(t))$$

and y(t) be the locally generated sinusoid given by

$$y(t) = cos(\theta_y(t)) - jsin(\theta_y(t))$$

A quadrature phase detector generates its error by multiplying the two signals together and taking the imaginary part of the output. Find the error signal for the given signals.

6.9 *3rd Order Loops - 12 pts*

Using the transfer function given in Equation 6.17 for the phase error and input phase, explore the outcomes of the following situations:

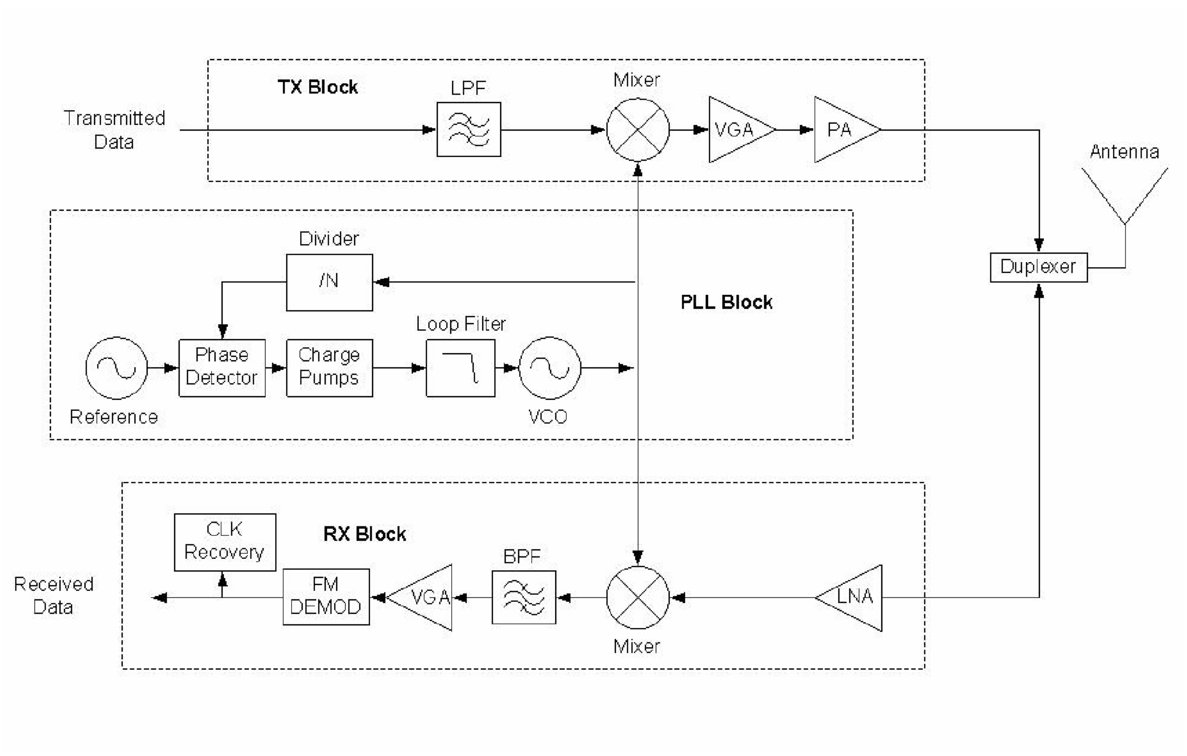


Figure 6.17: Block Diagram for Problem 6.9

a. Given a phase lock loop that has converged, how will this PLL respond to a phase step? From how large of a phase step can a PLL recover? (2 pts)

b. What is the effect on a locked PLL system when there is a jump in the input signal's frequency? (2 pts)

c. For the result of part b, show how a 1^{st}- and 2^{nd}-order PLL would effect the response of the system by substituting the values for $k_{vco} \cdot F(s)$ given in Figure 6.7. (2 pts)

d. In situations where the transmitter or receiver (or both) are moving with respect to each other, a Doppler Effect is seen by the PLL as a changing frequency per time, or as a frequency ramp. What is the effect of this change to the input signal on a converged PLL? (2 pts)

e. Design a Loop Filter for a 3^{rd}-Order PLL and show what effect it has in mitigating the error found in part d. (4 pts)

6.10 *Blue Tooth PLL - 8 pts*

"Bluetooth" is a wireless communication standard that was developed to support short-range voice and data transfer between portable devices. Figure **??** shows a simplified system diagram of a Bluetooth radio. In the United States, Bluetooth utilizes the Industrial Scientific Medicine (ISM) band between 2.4 - 2.4835 GHz split into channels 0 - 79.

a. If channel 0 is at 2.402 GHz and channel 79 is at 2.481 GHz, what is the channel spacing in the Bluetooth standard? (1 pt)

b. Assuming the VCO has an operating voltage of 0 - 3.3 volts, what tuning slope is required to cover the stated ISM band range? (2 pts)

c. Referring to the divider in Figure **??**, explain what effect a change in the value N would have on this system. (2 pts)

d. Since the ISM band is a generally unlicensed band, the FCC limits the permissible power output to 0 dBm. In order to extend the range of Bluetooth transmissions (i.e., increase the output power) as well as to compensate for interference from other signals in the band, Bluetooth utilizes a frequency hopping spread spectrum signal (FHSS) technique with a hop speed of 1,600 hops per second. Since the power is now spread over the entire spectrum, the power output can be as high as 20 dBm thus extending the range of communications up to 100m as well as providing a source of security against interference.

   From part b, assume that a microcontroller is used to store the hop sequence and implements the frequency hopping by changing the value N of the divider. Why is the settling time of the PLL important? Assuming that one VCO does in fact cover the entire needed band, when would the worst-case settling time occur? (2 pts)

e. Extra Credit: Where does the name Bluetooth come from? (1 pt)