CHAPTER 5

Satellite Signal Acquisition, Tracking, and Data Demodulation

Phillip W. Ward
NAVWARD GPS Consulting

John W. Betz and Christopher J. Hegarty
The MITRE Corporation

5.1 Overview

In practice, a GPS receiver must first replicate the PRN code that is transmitted by the SV being acquired by the receiver; then it must shift the phase of the replica code until it correlates with the SV PRN code. When cross-correlating the transmitted PRN code with a replica code, the same correlation properties occur that occurs for the mathematical autocorrelation process for a given PRN code. As will be seen in this chapter, the mechanics of the receiver correlation process are very different from the autocorrelation process because only selected points of the correlation envelope are found and examined by the receiver. When the phase of the GPS receiver replica code matches the phase of the incoming SV code, there is maximum correlation. When the phase of the replica code is offset by more than 1 chip on either side of the incoming SV code, there is minimum correlation. This is indeed the manner in which a GPS receiver detects the SV signal when acquiring or tracking the SV signal in the code phase dimension. It is important to understand that the GPS receiver must also detect the SV in the carrier phase dimension by replicating the carrier frequency plus Doppler (and usually eventually obtains carrier phase lock with the SV signal by this means). Thus, the GPS signal acquisition and tracking process is a two-dimensional (code and carrier) signal replication process.

In the code or range dimension, the GPS receiver accomplishes the cross-correlation process by first searching for the phase of the desired SV and then tracking the SV code state. This is done by adjusting the nominal spreading code chip rate of its replica code generator to compensate for the Doppler-induced effect on the SV PRN code due to LOS relative dynamics between the antenna phase centers of the receiver and the SV. There is also an apparent Doppler effect on the code tracking loop caused by the frequency offset in the receiver's reference oscillator with respect to its specified frequency. This common mode error effect, which is the time bias rate that is ultimately determined by the navigation solution, is quite small for the

code tracking loop and is usually neglected for code tracking and measurement purposes. The code correlation process is implemented as a real-time multiplication of the phase-shifted replica code with the incoming SV code, followed by an integration and dump process. The objective of the GPS receiver is to keep the prompt phase of its replica code generator at maximum correlation with the desired SV code phase. Typically, three correlators are used for tracking purposes, one at the prompt or on-time correlation position for carrier tracking and the other two located symmetrically early and late with respect to the prompt phase for code tracking. Modern receivers use multiple (even massively multiple) correlators to speed up the search process and some use multiple correlators for robust code tracking.

However, if the receiver has not simultaneously adjusted (tuned) its replica carrier signal so that it matches the frequency of the desired SV carrier, then the signal correlation process in the range dimension is severely attenuated by the resulting frequency response roll-off characteristic of the GPS receiver. This has the consequence that the receiver never acquires the SV. If the signal was successfully acquired because the SV code and frequency were successfully replicated during the search process, but the receiver subsequently loses track of the SV frequency, then the receiver subsequently loses code track as well. Thus, in the carrier Doppler frequency dimension, the GPS receiver accomplishes the carrier matching (wipeoff) process by first searching for the carrier Doppler frequency of the desired SV and then tracking the SV carrier Doppler state. It does this by adjusting the nominal carrier frequency of its replica carrier generator to compensate for the Doppler-induced effect on the SV carrier signal due to LOS relative dynamics between the receiver and the SV. There is also an apparent Doppler error effect on the carrier loop caused by the frequency offset in the receiver's reference oscillator with respect to its specified frequency. This error, which is common to all satellites being tracked by the receiver, is determined by the navigation filter as the time bias rate in units of seconds per second. This error in the carrier Doppler phase measurement is important to the search process (if known) and is an essential correction to the carrier Doppler phase measurement process.

The two-dimensional search and tracking process can best be explained and understood in progressive steps. The clearest explanation is in reverse sequence from the events that actually take place in a typical real world GPS receiver, namely signal search and acquisition followed by steady state tracking. The two-dimensional search and acquisition process is easier to understand if the two-dimensional steady state tracking process is explained first (Section 5.2). Once in steady state tracking, the two-dimensional code and carrier tracking process is easier to understand if the carrier tracking process is explained first even though both the code and carrier tracking processes are taking place simultaneously. This is the explanation sequence that will be used. The explanation will first be given in the context of a generic GPS receiver architecture with minimum use of equations. This high-level overview will then be followed by more detailed explanations of the carrier (Section 5.3) and code tracking loops (Section 5.4), including the most useful equations. Sections 5.5 through 5.7 cover additional topics regarding the tracking loops. Section 5.5 addresses the design of loop filters. Section 5.6 discusses measurement errors and tracking thresholds. Section 5.7 describes how the pseudorange, delta pseudorange, and integrated Doppler measurements are formed from the natural measurements of a GPS receiver.

The remainder of the chapter addresses acquisition (Section 5.8); other functions performed by the receiver including the sequence of initial operations (Section 5.9), data demodulation (Section 5.10), and special baseband functions (Section 5.11) such as SNR estimation and lock detection; and some special topics. The special topics include the use of digital processing (Section 5.12), considerations for indoor use (Section 5.13), and techniques to track the Y code without cryptographic access to this signal (Section 5.14). Throughout the chapter, extensive use of spreadsheet approximation equations and some experience-proven, rule-of-thumb, tracking threshold criteria are presented that will make it practical for the reader to not only understand but actually design the baseband portion of a GPS receiver.

5.2 GPS Receiver Code and Carrier Tracking

Most modern GPS receiver designs are digital receivers. These receiver designs have evolved rapidly toward higher and higher levels of digital component integration, and this trend is expected to continue. Also, microprocessors and their specialized cousin, DSPs, are becoming so powerful and cost effective that software defined receivers (SDRs) are being developed that use no custom digital components. For this reason, a high-level block diagram of a modern generic digital GPS receiver will be used to represent a generic GPS receiver architecture, as shown in Figure 5.1. The GPS RF signals of all SVs in view are received by a RHCP antenna with nearly hemispherical (i.e., above the local horizon) gain coverage. These RF signals are amplified by a low noise preamplifier (preamp), which effectively sets the noise figure of the receiver. There may be a passive bandpass prefilter between the antenna and preamp to minimize out-of-band RF interference. These amplified and signal condi-

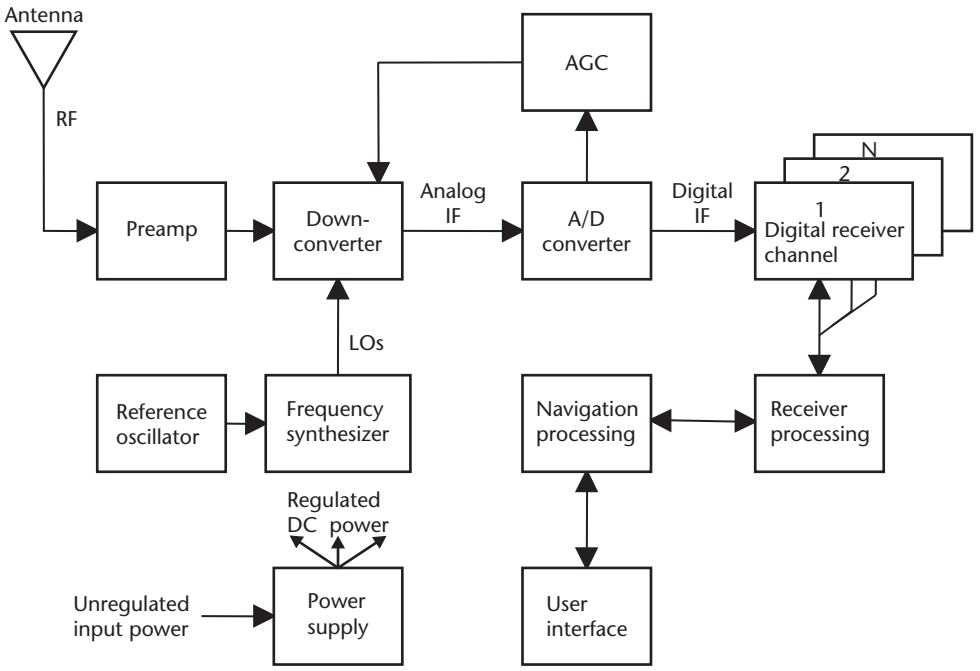


Figure 5.1 Generic digital GPS receiver block diagram.

tioned RF signals are then down-converted to an IF using signal mixing frequencies from local oscillators (LOs). The LOs are derived from the reference oscillator by the frequency synthesizer, based on the frequency plan of the receiver design. One LO per downconverter stage is required. Two-stage down-conversion to IF is typical, but one-stage down-conversion and even direct L-band digital sampling have also been used. However, since nearly 100 dB of signal gain is required prior to digitization, placing all of this gain at L-band is conducive to self-jamming in the receiver front end, so downconversion is assumed here. The LO signal mixing process generates both upper and lower sidebands of the SV signals, so the lower sidebands are selected and the upper sidebands and leak-through signals are rejected by a postmixer bandpass filter. The signal Dopplers and the PRN codes are preserved after the mixing process. Only the carrier frequency is lowered, but the Doppler remains referenced to the original L-band signal. The A/D conversion process and automatic gain control (AGC) functions take place at IF. Not shown in the block diagram are the baseband timing signals that are provided to the digital receiver channels by the frequency synthesizer phase locked to the reference oscillator's stable frequency. The IF must be high enough to provide a single-sided bandwidth that will support the PRN code chipping frequency. An antialiasing IF filter must suppress the stopband noise (unwanted out-of-band signals) to levels that are acceptably low when this noise is aliased into the GPS signal passband by the A/D conversion process. The signals from all GPS satellites in view are buried in thermal noise at IF.

At this point the digitized IF signals are ready to be processed by each of the N digital receiver channels. No demodulation has taken place, only signal gain and conditioning plus A/D conversion into the digital IF. Traditionally, these digital receiver channel functions are implemented in one or more application-specific integrated circuits (ASICs), but SDRs would use field programmable gate arrays (FPGAs) or even DSPs. This is why these functions are shown as separate from the receiver processing function in the block diagram of Figure 5.1. The name *digital receiver channel* is somewhat misleading since it is neither the ASIC nor FPGA but the receiver processing function that usually implements numerous essential but complex (and fortunately less throughput-demanding) baseband functions, such as the loop discriminators and filters, data demodulation, SNR meters, and phase lock indicators. The receiver processing function is usually a microprocessor. The microprocessor not only performs the baseband functions, but also the decision-making functions associated with controlling the signal preprocessing functions of each digital receiver channel. It is common that a single high-speed microprocessor supports the receiver, navigation, and user interface functions.

Figure 5.2 illustrates a high-level block diagram typical of one of the digital receiver channels where the digitized received IF signal is applied to the input. For simplification, only the functions associated with the code and carrier tracking loops are illustrated, and the receiver channel is assumed to be tracking the SV signal in steady state. Referring to Figure 5.2, first the digital IF is stripped of the carrier (plus carrier Doppler) by the replica carrier (plus carrier Doppler) signals to produce in-phase (I) and quadraphase (Q) sampled data. Note that the replica carrier signal is being mixed with all of the in-view GPS SV signals (plus noise) at the digital IF. The I and Q signals at the outputs of the mixers have the desired phase relationships

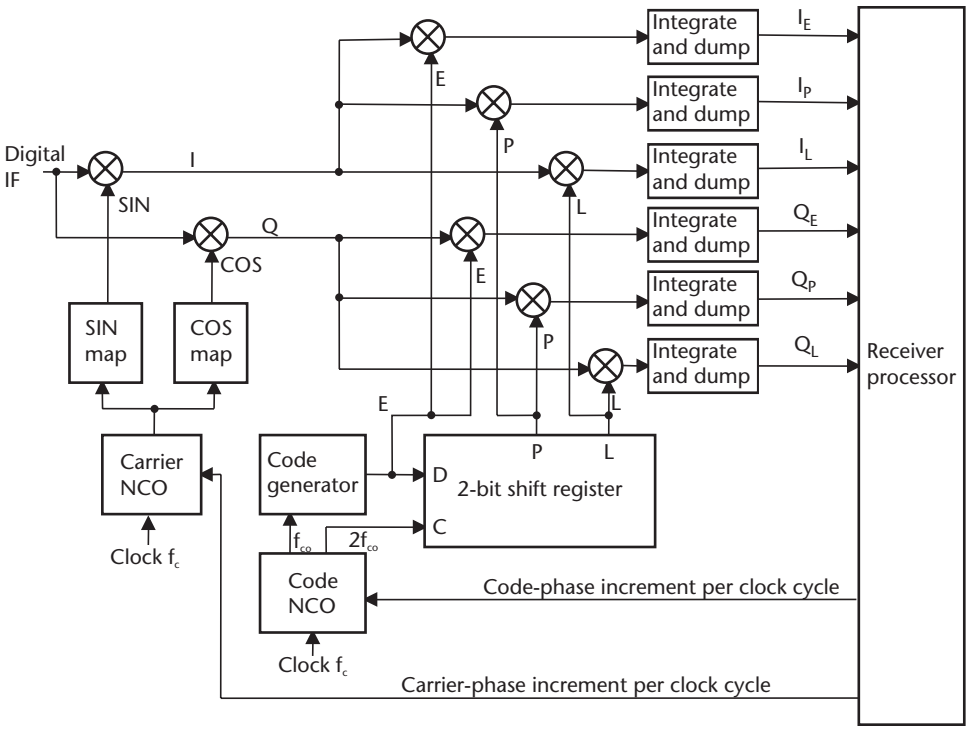


Figure 5.2 Generic digital receiver channel block diagram.

with respect to the detected carrier of the desired SV. However, the code stripping processes that collapse these signals to baseband have not yet been applied. Therefore, the *I* and *Q* signals at the output of the carrier mixers are dominated by noise. The desired SV signal remains buried in noise until the *I* and *Q* signals are collapsed to baseband by the code stripping process that follows. The replica carrier (including carrier Doppler) signals are synthesized by the carrier numerically controlled oscillator (NCO) and the discrete sine and cosine mapping functions.

The code wipeoff function could have been implemented before the carrier wipeoff function in this design, but this would increase the carrier wipeoff complexity with no improvement in receiver performance. The wipeoff sequence presented in Figure 5.2 is the least complex design.

Later, it will be shown that the NCO produces a staircase function whose period is the desired replica carrier plus Doppler period. The sine and cosine map functions convert each discrete amplitude of the staircase function to the corresponding discrete amplitude of the respective sine and cosine functions. By producing *I* and *Q* component phases 90° apart, the resultant signal amplitude can be computed from the vector sum of the *I* and *Q* components, and the phase angle with respect to the *I*-axis can be determined from the arctangent of *Q*/*I*. In closed loop operation, the carrier NCO is controlled by the carrier tracking loop in the receiver processor. In phase lock loop (PLL) operation, the objective of the carrier tracking loop is to keep the phase error between the replica carrier and the incoming SV carrier signals at zero. Any misalignment in the replica carrier phase with respect to the incoming SV signal carrier phase produces a nonzero phase angle of the prompt *I* and *Q* vector magnitude, so that the amount and direction of the phase change can

be detected and corrected by the carrier tracking loop. When the PLL is phase locked, the I signals are maximum (signal plus noise) and the Q signals are minimum (containing only noise).

In Figure 5.2, the I and Q signals are then correlated with early, prompt, and late replica codes (plus code Doppler) synthesized by the code generator, a 2-bit shift register, and the code NCO. In closed loop operation, the code NCO is controlled by the code tracking loop in the receiver processor. In this example, the code NCO produces twice the code generator clocking rate, $2f_{co}$, and this is fed to the clock input of the 2-bit shift register. The code generator clocking rate, f_{co}, that contains the nominal spreading code chip rate (plus code Doppler) is fed to the code generator. The NCO clock, f_c, should be a much higher frequency than the shift register clock, $2f_{co}$. With this combination, the shift register produces two phase-delayed versions of the code generator output. As a result, there are three replica code phases designated as early (E), prompt (P), and late (L). E and L are typically separated in phase by 1 chip and P is in the middle. Not shown are the controls to the code generator that permit the receiver processor to preset the initial code tracking phase states that are required during the code search and acquisition (or reacquisition) process.

The prompt replica code phase is aligned with the incoming SV code phase producing maximum correlation if it is tracking the incoming SV code phase. Under this circumstance, the early phase is aligned a fraction of a chip period early, and the late phase is aligned the same fraction of the chip period late with respect to the incoming SV code phase, and these correlators produce about half the maximum correlation. Any misalignment in the replica code phase with respect to the incoming SV code phase produces a difference in the vector magnitudes of the early and late correlated outputs so that the amount and direction of the phase change can be detected and corrected by the code tracking loop.

5.2.1 Predetection Integration

Predetection is the signal processing after the IF signal has been converted to baseband by the carrier and code stripping processes, but prior to being passed through a signal discriminator (i.e., prior to the nonlinear signal detection process). Extensive digital predetection integration and dump processes occur after the carrier and code stripping processes. This causes very large numbers to accumulate, even though the IF A/D conversion process is typically with only 1 to 3 bits of quantization resolution with the carrier wipeoff process involving a matching multiplication precision and the code wipeoff process that follows usually involving only 1-bit multiplication.

Figure 5.2 shows three complex correlators required to produce three in-phase components, which are integrated and dumped to produce I_E, I_P, I_L and three quadraphase components integrated and dumped to produce Q_E, Q_P, Q_L. The carrier wipeoff and code wipeoff processes must be performed at the digital IF sample rate, which is of the order of 50 MHz for a military P(Y) code receiver (that also operates with C/A code), 5 MHz for civil C/A code receivers that use 1-chip E-L correlator spacing, and up to 50 MHz for civil C/A code receivers that use narrow correlator spacing for improved multipath error performance. The integrate and dump accumulators provide filtering and resampling at the processor baseband

input rate, which can be at 1,000 Hz during search modes or as low as 50 Hz during track modes, depending on the desired dwell time during search or the desired predetection integration time during track. The 50- to 1,000-Hz rates are well within the servicing capability of modern high-speed microprocessors, but the 5- or 50-MHz rates are challenging even for modern DSPs. This further explains why the high-speed but simple processes are implemented in a custom digital ASIC or FPGA, while the low-speed but complex processes are implemented in a microprocessor.

The hardware integrate and dump process in combination with the baseband signal processing integrate and dump process (described next) defines the predetection integration time. Later, it will be shown that the predetection integration time is a compromise design. It must be as long as possible to operate under weak or RF interference signal conditions, and it must be as short as possible to operate under high dynamic stress signal conditions.

5.2.2 Baseband Signal Processing

Figure 5.3 illustrates typical baseband code and carrier tracking loops for one receiver channel in the closed loop mode of operation. The functions are typically performed by the receiver processor shown in Figure 5.2. The combination of these carrier and code tracking baseband signal processing functions and the digital receiver channel carrier and code wipeoff and predetection integration functions form the carrier and code tracking loops of one GPS receiver channel.

The baseband functions are usually implemented in firmware. Note that the firmware need only be written once, since the microprocessor runs all programs sequentially. This is contrasted to the usual parallel processing that takes place in the

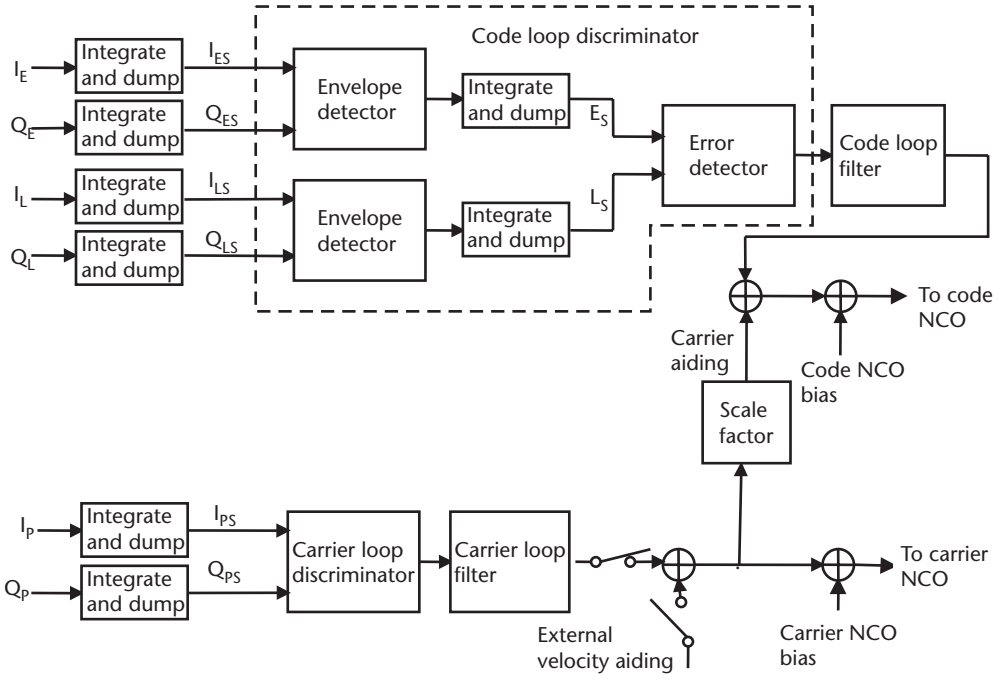


Figure 5.3 Generic baseband processor code and carrier tracking loops block diagram.

digital receiver ASIC(s) or FPGA(s), but even these devices can multiplex their digital processes sequentially in order to reduce gate count if they are capable of running faster than real time. Therefore, the ASIC, FPGA, and microprocessor programs can be designed to be reentrant with a unique variable area for each receiver channel so that only one copy of each algorithm is required to service all receiver channels. This reduces the gate count or program memory requirements and ensures that every receiver baseband processing function is identical. Digital multiplexing also eliminates interchannel bias in the ASIC or FPGA (hardware portion of the digital receiver) with no performance loss. (Section 7.2.7.2 further discusses interchannel biases.)

The three complex pairs of baseband *I* and *Q* signals from the digital receiver may be resampled again by the integrate and dump accumulators. The total combined duration of the receiver and processor integrate and dump functions establishes the predetection integration time for the signal. Normally, this cannot exceed 20 ms, which is the 50-Hz navigation message data bit period for the GPS C/A and P(Y) code signals. Figure 5.4 illustrates the phase alignment needed to prevent the predetection integrate and dump intervals from integrating across a SV data transition boundary. The start and stop boundaries for these integrate and dump functions should not straddle the data bit transition boundaries because each time the SV data bits change signs, the signs of the subsequent integrated *I* and *Q* data may

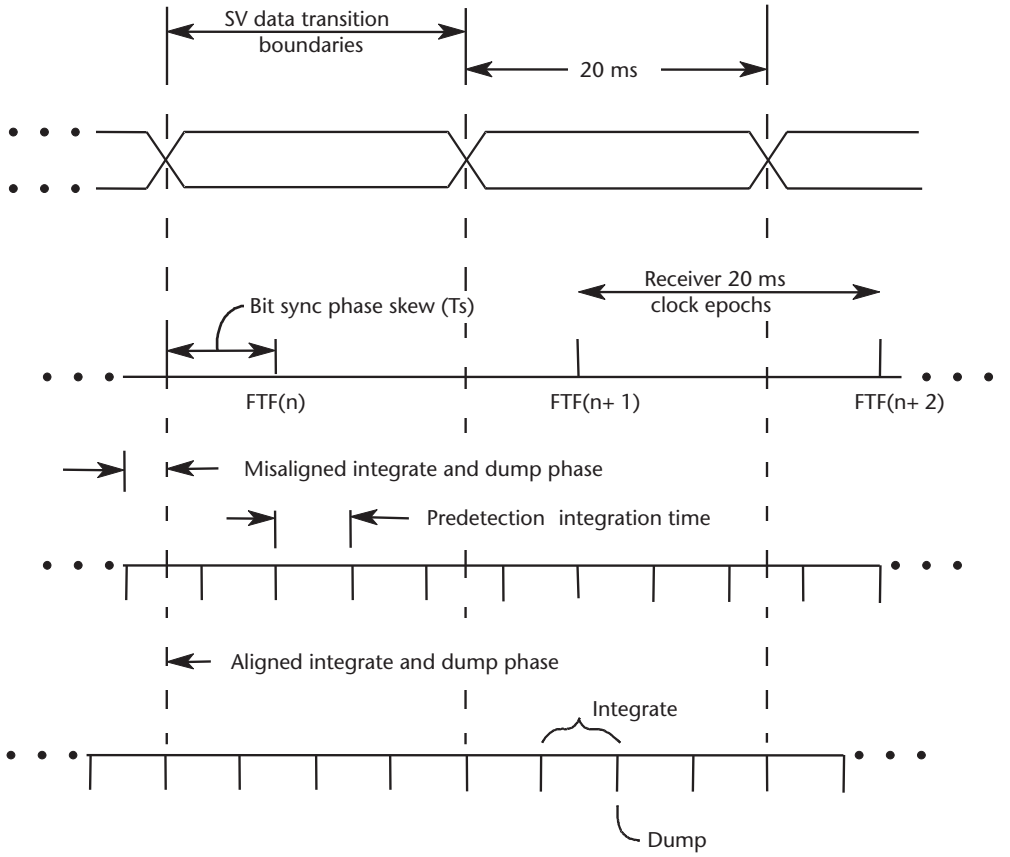


Figure 5.4 Phase alignment of predetection integrate and dump intervals with SV data transition boundaries.

change. If the boundary is straddled and there is a data transition, the integration and dump result for that interval will be degraded. In the worst case, if the data transition occurs at the halfway point, the signal will be totally canceled for that interval. Usually, during initial C/A code signal search, acquisition, and loop closure, the receiver does not know where the SV data bit transition boundaries are located because each C/A code epoch is only 1 ms in duration but the data bit is 20 ms in duration. Then, the performance degradation has to be accepted until the bit synchronization process locates the data bit transitions. During these times, short predetection integration times are used in order to ensure that most of the integrate and dump operations do not contain a data transition boundary. With signals that have spreading code periods that are as long or longer than the data bit period, receivers can choose longer predetection time intervals that are aligned with data bit edges.

As shown in Figure 5.4, the SV data transition boundary usually does not align with the receiver's 20-ms clock boundary, which will hereafter be called the fundamental time frame (FTF). The phase offset is shown as *bit sync phase skew*. A bit synchronization process determines this phase offset shortly after the signal has been acquired when the receiver does not know its position and precise GPS time. In general, the bit sync phase skew is different for every SV being tracked because even though the data transitions are well aligned at SV transmit time, the difference in range to the user causes them to be skewed at receive time. This range difference amounts to about a 20-ms variation from zenith to horizon. The receiver design must accommodate these data bit phase skews if an optimum predetection integration time is used. This optimization is assumed in the generic receiver design, but some receiver designs do not implement this added complexity. They use short (suboptimal) predetection integration times.

5.2.3 Digital Frequency Synthesis

In this generic design example, both the carrier and code tracking loops use an NCO for precision replica carrier and code generation. The NCO provides measurements that contain negligible quantization noise [1].

One replica carrier cycle and one replica code cycle are completed each time the NCO overflows. A block diagram of the carrier loop NCO and its sine and cosine mapping functions are shown in Figure 5.5 [1]. In Figure 5.3, note that there is a code NCO bias and a carrier NCO bias applied to their respective NCOs. These biases set the NCO frequency to the nominal code spreading code chip rate and IF carrier frequency, respectively, because they are constants. As an NCO bias computational example using the equation for output frequency in Figure 5.5, assume that the bias is set for the P(Y) code nominal spreading code chip rate of 10.23 MHz. Assume a 32-bit NCO with a clock $f_s = 200$ MHz, then the code NCO bias is $M = 10.23 \times 2^{32}/200 = 2.1969 \times 10^8$. This value of M sets the NCO output frequency to 10.23 MHz with a resolution of $200 \times 10^6/2^{32} = 0.046566$ Hz.

For the carrier NCO, the map functions convert the amplitude of the NCO staircase output [Figure 5.6(a)] into the appropriate trigonometric functions as shown in Figure 5.6(b, c). Figure 5.7 illustrates the basic idea of digital frequency synthesizer design.

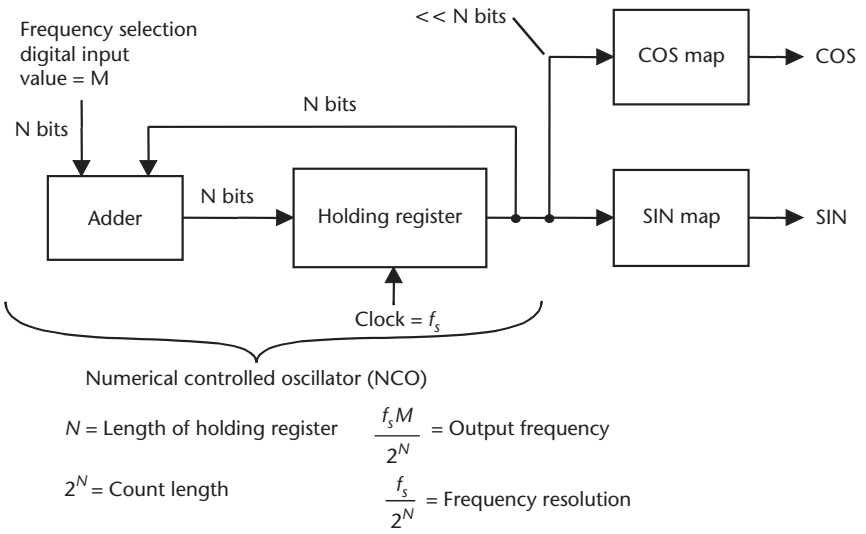


Figure 5.5 Digital frequency synthesizer block diagram.

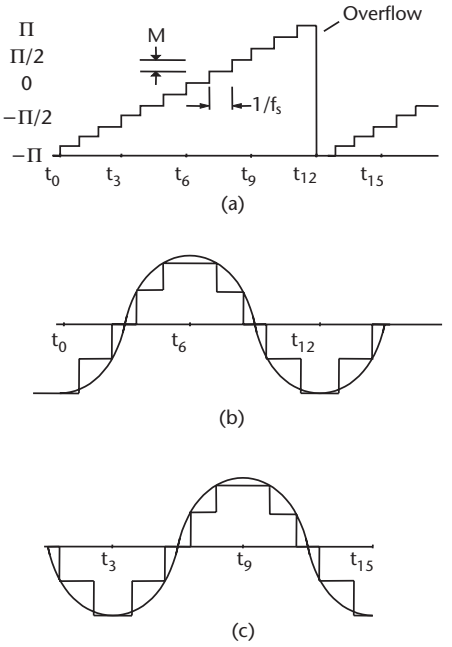
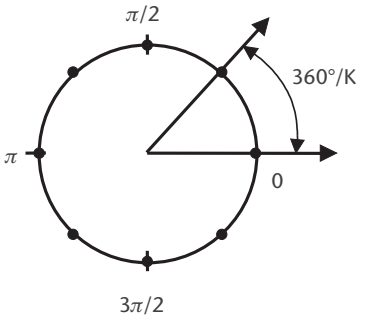


Figure 5.6 Digital frequency synthesizer waveforms: (a) NCO phase state, (b) COS map output, and (c) SIN map output.

5.2.4 Carrier Aiding of Code Loop

In Figure 5.3, the carrier loop filter output is adjusted by a scale factor and added to the code loop filter output as aiding. This is called a *carrier-aided* code loop. The scale factor is required because the Doppler effect on the signal is inversely proportional to the wavelength of the signal. Therefore, for the same relative velocity between the SV and the GPS receiver, the Doppler on the spreading code chip rate is much smaller than the Doppler on the L-band carrier. (Keep in mind that even



Degrees	Holding register (binary)	SIN map (sign magnitude)	COS map (sign magnitude)
0	1 0 0...	0 0 0	0 1 1
45	1 0 1...	0 1 0	0 1 0
90	1 1 0...	0 1 1	0 0 0
135	1 1 1...	0 1 0	1 1 0
180	0 0 0...	0 0 0	1 1 1
225	0 0 1...	1 1 0	1 1 0
270	0 1 0...	1 1 1	0 0 0
315	0 1 1...	1 1 0	0 1 0

Maps for J = 3, K = 2^J = 8

Notes:
1. The number of bits, J, is determined for the SIN and COS outputs. The phase plane of 360 degrees is subdivided into 2^J = K phase points.
2. K values are computed for each waveform, one value per phase point. Each value represents the amplitude of the waveform to be generated at that phase point. The upper J bits of the holding register are used to determine the address of the waveform amplitude.
3. Rate at which phase plane is traversed determines the frequency of the output waveform.
4. The upper bound of the amplitude error is 2ΠK.
5. The approximate amplitude error is: 2ΠK cos ϕ(t), where ϕ(t) is the phase angle.

Figure 5.7 Digital frequency synthesizer design.

though the carrier has been downconverted to IF and the NCO carrier bias is set to the IF, the carrier Doppler effect remains referenced to L-band.) The scale factor that compensates for this difference in frequency is given by:

$$Scale\ factor = \frac{R_c}{f_L}\ (\text{dimensionless}) \tag{5.1}$$

where:

R_c = spreading code chip rate (Hz) plus Doppler effect
= R_0 for P(Y) code = 10.23 Mchip/s + P(Y) Doppler effect
= R_0/10 for C/A code = 1.023 Mchip/s + C/A Doppler effect
f_L = L-band carrier (Hz)
= 154 R_0 for L1
= 120 R_0 for L2

Table 5.1 shows the three practical combinations of this scale factor.

The carrier loop output should always provide Doppler aiding to the code loop because the carrier loop jitter is orders of magnitude less noisy than the code loop and thus much more accurate. The carrier loop aiding removes virtually all of the LOS dynamics from the code loop, so the code loop filter order can be made smaller, its update rate slower, and its bandwidth narrower than for the unaided case, thereby reducing the noise in the code loop measurements. In fact, the code loop only tracks the dynamics of the ionospheric delay plus noise. When both the

Table 5.1 Scale Factors for Carrier Aided Code

Carrier Frequency (Hz)	*Code Rate (chips/s)*	*Scale Factor*
L1 = 154 R_0	R_0/10 for C/A	1/1540 = 0.00064935
L1 = 154 R_0	R_0 for P(Y)	1/154 = 0.00649350
L2 = 120 R_0	R_0 for P(Y)	1/120 = 0.00833333

code and carrier loops must maintain track, nothing is lost in tracking performance by using carrier aiding for an unaided GPS receiver, even though the carrier loop is the weakest link.

5.2.5 External Aiding

As shown in Figure 5.3, external velocity aiding, say from an inertial measurement unit (IMU), can be provided to the receiver channel in closed carrier loop operation. The switch, shown in the unaided position, must be closed when external velocity aiding is applied. At the instant that external aiding is injected, the loop filter state must be set to the time bias rate if known; otherwise, it is zeroed. The external rate aiding must be converted into LOS velocity aiding with respect to the GPS satellite. The lever arm effects on the aiding must be computed with respect to the GPS antenna phase center, which requires knowledge of the vehicle attitude and the location of the antenna phase center with respect to the navigation center of the external source of velocity aiding. For closed carrier loop operation, the aiding must be very precise and have little or no latency or the tracking loop must be delay-compensated for the latency. If open carrier loop aiding is implemented, less precise external velocity aiding is required, but there are no meaningful delta range measurements available. Also, it is not likely that the SV navigation message data can be demodulated in this mode, so it is a short-term, weak signal hold-on strategy. In this open-loop weak signal hold-on case, the output of the carrier loop filter is not combined with the external velocity aiding to control the carrier NCO, but the open-loop output of the filter can be used to provide a SNR computation. (External aiding using IMU and other sensor measurements is discussed further in Chapter 9.)

5.3 Carrier Tracking Loops

Figure 5.8 presents a block diagram of a GPS receiver carrier tracking loop. The programmable designs of the carrier predetection integrators, the carrier loop discriminators, and the carrier loop filters characterize the receiver carrier tracking loop. These three functions determine the two most important performance characteristics of the receiver carrier loop design: the carrier loop thermal noise error and the maximum LOS dynamic stress threshold. Since the carrier tracking loop is always the weak link in a stand-alone GPS receiver, its threshold characterizes the unaided GPS receiver performance.

The carrier loop discriminator defines the type of tracking loop as a PLL, a Costas PLL (which is a PLL-type discriminator that tolerates the presence of data

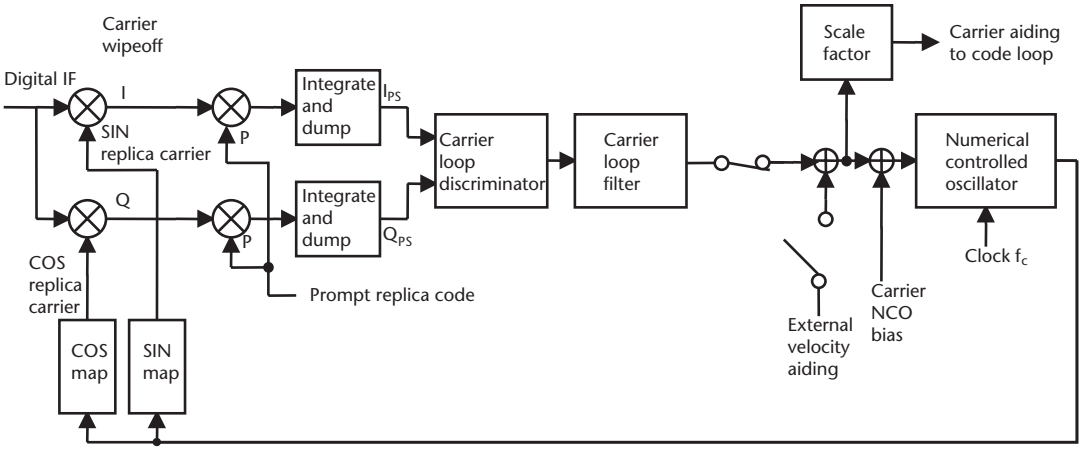


Figure 5.8 Generic GPS receiver carrier tracking loop block diagram.

modulation on the baseband signal), or a frequency lock loop (FLL). The PLL and the Costas loop are the most accurate, but they are more sensitive to dynamic stress than the FLL. The PLL and Costas loop discriminators produce phase error estimates at their outputs. The FLL discriminator produces a frequency error estimate. Because of this, there is also a difference in the architecture of the loop filter, described later.

There is a paradox that the GPS receiver designer must solve in the design of the predetection integration time and the discriminator and loop filter functions of the carrier tracking loop. To tolerate dynamic stress, the predetection integration time should be short, the discriminator should be an FLL, and the carrier loop filter bandwidth should be wide. However, for the carrier measurements to be accurate (have low noise), the predetection integration time should be long, the discriminator should be a PLL, and the carrier loop filter noise bandwidth should be narrow. In practice, some compromise must be made to resolve this paradox. A well-designed GPS receiver should close its carrier tracking loops with short predetection integration times, using an FLL and a wideband carrier loop filter. Assuming there is data modulation on the carrier, it should then systematically transition into a Costas PLL, gradually adjusting the predetection integration time equal to the period of the data transitions while also gradually adjusting the carrier tracking loop bandwidth as narrow as the maximum anticipated dynamics permits. Later, an FLL-assisted-PLL carrier tracking loop will be described that automatically adjusts to dynamic stress.

5.3.1 Phase Lock Loops

If there was no 50-Hz data modulation on the GPS signal, the carrier tracking loop discriminator could use a pure PLL discriminator. For example, a P(Y) code receiver could implement a pure PLL discriminator for use in the L2 carrier tracking mode if the control segment turns off data modulation. Although this mode is specified as a possibility, it is unlikely to be activated. This mode is specified in IS-GPS-200 [2] because pure PLL operation enables an improved signal tracking threshold by up to 6 dB. All modernized GPS signals make provisions for dataless carrier tracking in

addition to providing data, but the provision involves sharing the total signal power between a half-power component that contains the data and another half-power component that is dataless. The sharing technique loses 3 dB from the dataless component used for tracking, but there is a net gain of 3 dB when tracking the dataless signal with a pure PLL.

It is also possible to implement short-term pure PLL modes by a process called data wipeoff. The GPS receiver typically acquires a complete copy of the full navigation message after 25 iterations of the 5 subframes (12.5 minutes), or the current data can be provided by some external means. The receiver then can compute the navigation message sequence until the GPS control segment uploads a new message or until the SV changes the message. Until the message changes significantly, the GPS receiver can perform data wipeoff of each bit of the incoming 50-Hz navigation data message and use a pure PLL discriminator. The receiver baseband processing function does this by reversing the sign of the integrated prompt I and Q components in accordance with a consistent algorithm. For example, if I_{PS} and Q_{PS} have predetection integration times of 5 ms, then there are four samples of I_{PS} and Q_{PS} between each SV data bit transition that are assured to have the same sign. This sign will be the sign of the data bit known by the receiver a priori for that data interval. Each 5-ms sample may fluctuate in sign due to noise. If the known data bit for this interval is a "0," then the data wipeoff process does nothing to all four samples. If the known data bit for this interval is a "1," then the sign is reversed on all four samples.

Table 5.2 illustrates the four-quadrant arctangent discriminator algorithm and a simple approximation using Q normalized by a long-term average of the prompt envelope. Interestingly, the Q approximation has been proven experimentally to slightly outperform the theoretically optimal and more complex ATAN2 function. Figure 5.9(a) compares the phase error outputs of these PLL discriminators assuming no noise in the I and Q signals. Note that the ATAN2 discriminator is the only one that remains linear over the full input error range of ±180°. However, in the presence of noise, both of the discriminator outputs are linear only near the 0° region. These PLL discriminators will achieve the 6-dB improvement in signal tracking threshold (by comparison with the Costas discriminators described next) for the dataless carrier because they track the full four quadrant range of the input signal.

5.3.2 Costas Loops

Any carrier loop that is insensitive to the presence of data modulation is usually called a Costas loop since Costas was the original inventor. Table 5.3 summarizes several GPS receiver Costas PLL discriminators, their output phase errors, and their characteristics. Figure 5.9(b) compares the phase error outputs of these Costas PLL discriminators, assuming no noise in the I and Q signals. As shown, the two-quadrant ATAN Costas discriminator of Table 5.3 is the only Costas PLL discriminator that remains linear over half of the input error range (± 90°). In the presence of noise, all of the discriminator outputs are linear only near the 0° region.

Referring to the carrier tracking loop block diagram of Figure 5.8, the phase derotation (carrier wipeoff) process used in the generic receiver design requires only two multiples. Assuming that the carrier loop is in phase lock and that the replica

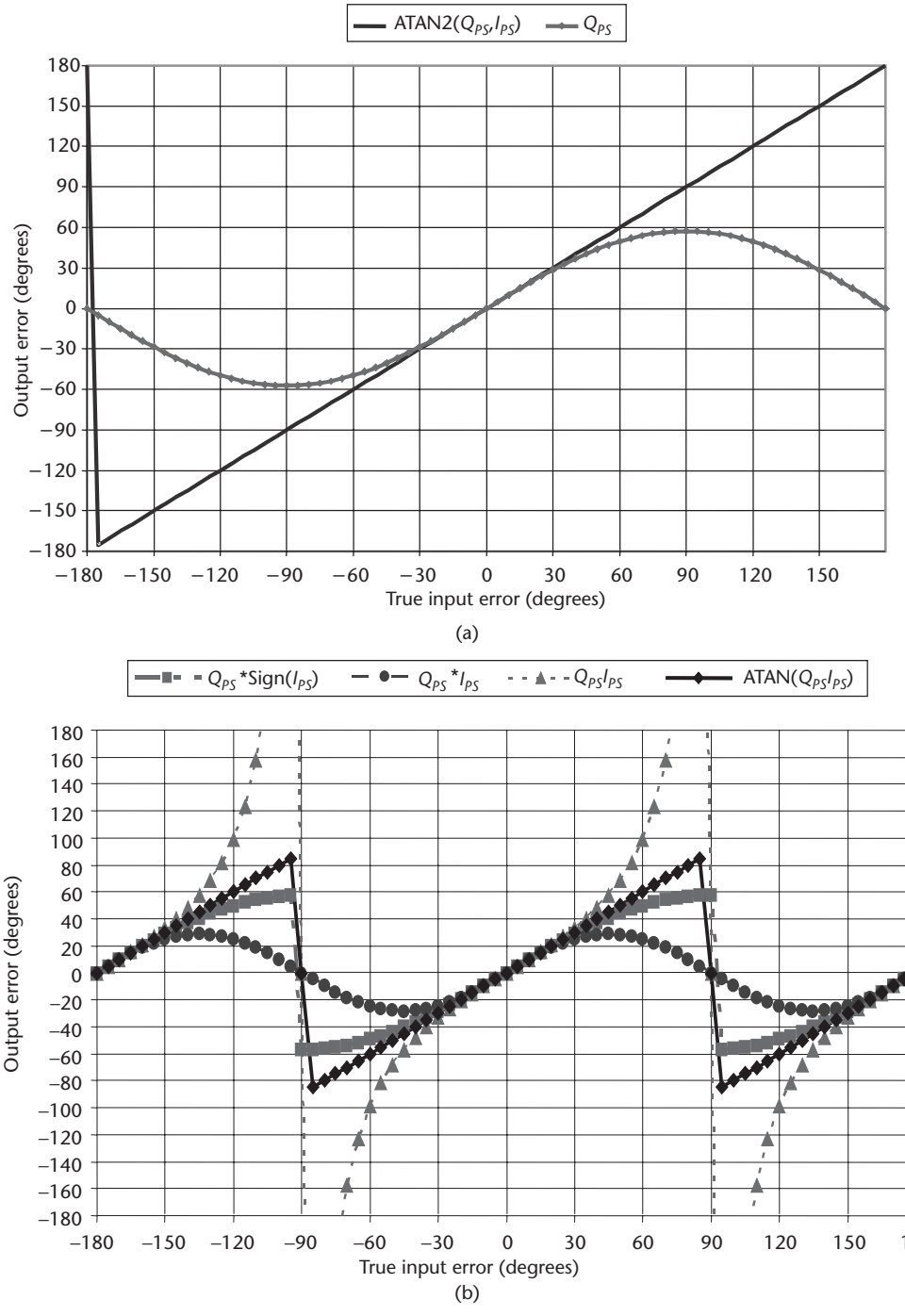


Figure 5.9 (a) Comparison of PLL discriminators, and (b) comparison of Costas PLL discriminators.

sine function is in-phase with the incoming SV carrier signal (converted to IF), this results in a sine squared product at the I output, which produces maximum I_{PS}

Table 5.2 PLL Discriminator

Discriminator Algorithm	*Output Phase Error*	*Characteristics*
$ATAN2(Q_{PS}, I_{PS})$	ϕ	Four-quadrant arctangent. Optimal (maximum likelihood estimator) at high and low SNR. Slope not signal amplitude dependent. High computational burden. Usually table lookup implementation.
$\frac{Q_{PS}}{Ave\sqrt{I_{PS}^2 + Q_{PS}^2}}$	$\sin\phi$	Q_{PS} normalized by averaged prompt envelope. Slightly outperforms four-quadrant arctangent. Q_{PS} approximates ϕ to $\pm 45^\circ$. Normalization provides insensitivity at high and low SNR. Also keeps slope not signal amplitude dependent. Low computational burden.

Table 5.3 Common Costas Loop Discriminators

Discriminator Algorithm	*Output Phase Error*	*Characteristics*
$Q_{PS} \times I_{PS}$	$\sin 2\phi$	Classic Costas analog discriminator. Near optimal at low SNR. Slope proportional to signal amplitude squared A^2. Moderate computational burden.
$Q_{PS} \times \text{Sign}\,(I_{PS})$	$\sin\phi$	Decision directed Costas. Near optimal at high SNR. Slope proportional to signal amplitude A. Least computational burden.
Q_{PS}/I_{PS}	$\tan\phi$	Suboptimal but good at high and low SNR. Slope not signal amplitude dependent. Higher computational burden. Divide by zero error at $\pm 90^\circ$.
$ATAN\,(Q_{PS}/I_{PS})$	ϕ	Two-quadrant arctangent. Optimal (maximum likelihood estimator) at high and low SNR. Slope not signal amplitude dependent. Highest computational burden. Usually table lookup implementation.

amplitude (signal plus noise) following the code wipeoff and integrate and dump process. The replica cosine function is 90° out of phase with the incoming SV carrier. This results in a cosine × sine product at the Q output, which produces minimum Q_{PS} amplitude (noise only). For this reason, I_{PS} will be near its maximum (and will flip 180° each time the data bit changes sign), and Q_{PS} will be near its minimum (and will also flip 180° each time the data bit changes sign).

Note that the classical complex pair carrier phase derotation scheme is not used in this generic receiver design because the natural GPS IF signal is real. The classical phase derotation scheme requires an I and Q input signal at IF. This, in turn, requires the real IF signal be phase shifted 90° to produce the quadrature component. This is a design penalty of added IF circuit complexity. If this is done on the

analog side, the Nyquist sample rate is half that of the generic A/D converter requirement. This is a design benefit if the A/D converter speed presents a design limitation, but this is not likely with today's technology. But two A/D converters are required to digitize the *I* and *Q* input signals. This is a design penalty that doubles the A/D components. A single A/D converter can be used to produce the *I* and *Q* signals if a technique referred to as *quadrature sampling* (also known as *pseudosampling* or *IF sampling*) is employed. However, the classical phase derotation process still requires four multiplies and two additions with no additional performance improvement. This is a phase derotation design penalty of two additional multiplies and two adds. The generic design is therefore the preferred carrier phase derotation scheme.

These PLL characteristics are illustrated in Figure 5.10, where the phasor, *A* (the vector sum of I_{PS} and Q_{PS}), tends to remain aligned with the *I*-axis and switches 180° during each data bit reversal.

It is straightforward to detect the bits in the SV data message stream using a Costas PLL. The I_{PS} samples are simply accumulated for one data bit interval, and the sign of the result is the data bit. Since there is a 180° phase ambiguity with a Costas PLL, the detected data bit stream may be normal or inverted. This ambiguity is resolved during the frame synchronization process by comparing the known preamble at the beginning of each subframe both ways (normal and inverted) with the bit stream. If a match is found with the preamble pattern inverted, the bit stream is inverted and the subframe synchronization is confirmed by parity checks on the TLM and HOW. Otherwise, the bit stream is normal. Once the phase ambiguity is resolved, it remains resolved until the PLL loses phase lock or slips cycles. If this

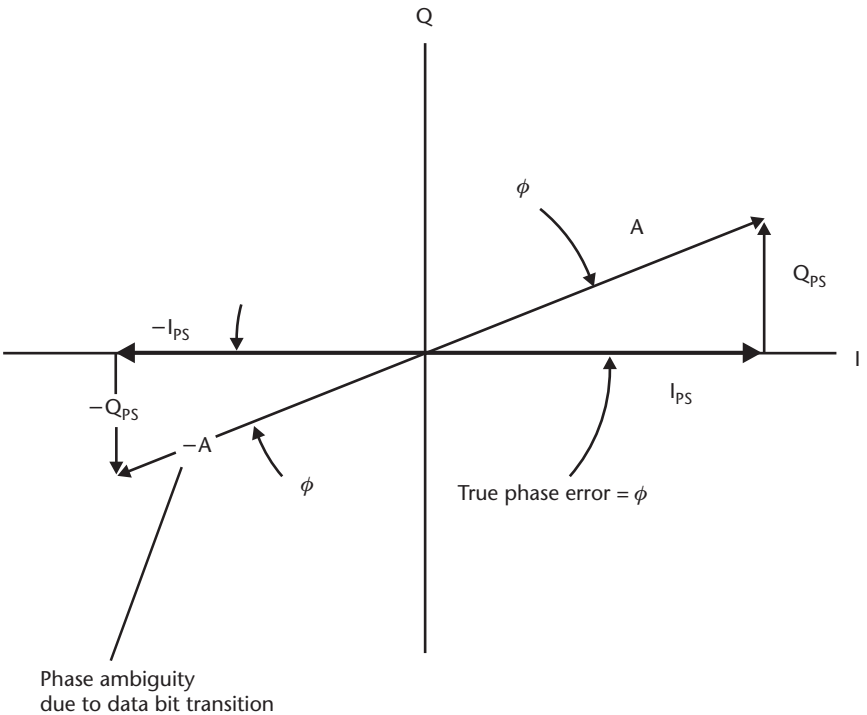


Figure 5.10 *I, Q* phasor diagram depicting true phase error between replica and incoming carrier phase.

happens, the ambiguity must be resolved again. The 180° ambiguity of the Costas PLL can be resolved by referring to the phase detection result of the data bit demodulation. If the data bit phase is normal, then the carrier Doppler phase indicated by the Costas PLL is correct. If the data bit phase is inverted, then the carrier Doppler phase indicated by the Costas PLL phase can be corrected by adding 180°.

Costas PLLs as well as conventional PLLs are sensitive to dynamic stress, but they produce the most accurate velocity measurements. For a given signal power level, Costas PLLs also provide the most error-free data demodulation in comparison to schemes used with FLLs. Therefore, this is the desired steady state tracking mode of the GPS receiver carrier tracking loop. It is possible for a PLL to close in a false phase lock mode if there is excess frequency error at the time of loop closure. Therefore, a well-designed GPS receiver carrier tracking loop will close the loop with a more dynamically robust FLL operated at wideband. Then it will gradually reduce the carrier tracking loop bandwidth and transition into a wideband PLL operation in order to systematically reduce the pull-in frequency error. Finally, it will narrow the PLL bandwidth to the steady state mode of operation. If dynamic stress causes the PLL to lose lock, the receiver will detect this with a sensitive phase lock detector and transition back to the FLL. The PLL closure process is then repeated.

5.3.3 Frequency Lock Loops

PLLs replicate the exact phase and frequency of the incoming SV (converted to IF) to perform the carrier wipeoff function. FLLs perform the carrier wipeoff process by replicating the approximate frequency, and they typically permit the phase to rotate with respect to the incoming carrier signal. For this reason, they are also called *automatic frequency control* (AFC) loops. The FLLs of GPS receivers must be insensitive to 180° reversals in the I and Q signals. Therefore, the sample times of the I and Q signals should not straddle the data bit transitions. During initial signal acquisition, when the receiver does not know where the data transition boundaries are, it is usually easier to maintain frequency lock than phase lock with the SV signal while performing bit synchronization. This is because the FLL discriminators are less sensitive to situations where some of the I and Q signals do straddle the data bit transitions. When the predetection integration times are small compared to the data bit transition intervals, fewer integrate and dump samples are corrupted, but the squaring loss is higher. Table 5.4 summarizes several GPS receiver FLL discriminators, their output frequency errors, and their characteristics.

Figure 5.11 compares the frequency error outputs of each of these discriminators assuming no noise in the I_{PS} and Q_{PS} samples. Figure 5.11(a) illustrates that the frequency pull-in range with a 5-ms predetection integration time (200-Hz bandwidth) has twice the pull-in range of Figure 5.11(b) with a 10-ms predetection integration time (100-Hz bandwidth). Note in both figures that the single-sided frequency pull-in ranges of the cross and ATAN2(dot, cross) FLL discriminators are equal to half the predetection bandwidths. The (cross) × sign(dot) FLL discriminator frequency pull-in ranges are only one-fourth of the predetection bandwidths. Also note that the (cross) × sign(dot) and the cross FLL discriminator outputs, whose outputs are sine functions divided by the sample time

Table 5.4 Common Frequency Lock Loop Discriminators

Discriminator Algorithm	*Output Frequency Error*	*Characteristics*
$\frac{cross}{(t_2 - t_1)}$ where: $cross = I_{PS1} \times Q_{PS2} - I_{PS2} \times Q_{PS1}$	$\frac{\sin[(\phi_2 - \phi_1)]}{t_2 - t_1}$	Near optimal at low SNR. Slope proportional to signal amplitude squared A^2. Least computational burden.
$\frac{(cross) \times sign(dot)}{(t_2 - t_1)}$ where: $dot = I_{PS1} \times I_{PS2} + Q_{PS1} \times Q_{PS2}$ $cross = I_{PS1} \times Q_{PS2} - I_{PS2} \times Q_{PS1}$	$\frac{\sin[2(\phi_2 - \phi_1)]}{t_2 - t_1}$	Decision directed. Near optimal at high SNR. Slope proportional to signal amplitude A. Moderate computational burden
$\frac{ATAN2(dot, cross)}{(t_2 - t_1)}$	$\frac{\phi_2 - \phi_1}{t_2 - t_1}$	Four-quadrant arctangent. Maximum likelihood estimator. Optimal at high and low SNR. Slope not signal amplitude dependent. Highest computational burden. Usually table lookup implementation.

Note: Integrated and dumped prompt samples I_{PS1} and Q_{PS1} are the samples taken at time t_1, just prior to the samples I_{PS2} and Q_{PS2} taken at a later time t_2. These two adjacent samples should be within the same data bit interval. The next pair of samples are taken starting $(t_2 - t_1)$ seconds after t_2 (i.e., no I and Q samples are reused in the next discriminator computation).

interval $(t_2 - t_1)$ in seconds, are also divided by 4 to more accurately approximate the true input frequency error. The ATAN2 (x, y) function returns the answer in radians, is converted to degrees, divided by the sample time interval $(t_2 - t_1)$ in seconds, and is also divided by 360 to produce at its output a true representation of the input frequency error within its pull-in range. The amplitudes of all of the discriminator outputs are reduced (their slopes tend to flatten), and they tend to start rounding off near the limits of their pull-in range as the noise levels increase.

The I, Q phasor diagram in Figure 5.12 depicts the change in phase, $\phi_2 - \phi_1$, between two adjacent samples of I_{PS} and Q_{PS}, at times t_1 and t_2. This phase change over a fixed time interval is proportional to the frequency error in the carrier tracking loop. The figure also illustrates that there is no frequency ambiguity in the GPS receiver FLL discriminator because of data transitions, provided that the adjacent I and Q samples are taken within the same data bit interval. However, it is possible for the FLL loop to close with a false frequency lock in a high dynamic environment. For this reason, very short predetection integration times (wider pull-in range) are important for initial FLL loop closure. For example, if the search dwell time was 1 ms or 2 ms, then the initial predetection integration time in FLL should be the same. Note that with a FLL, the phasor, A, which is the vector sum of I_{PS} and Q_{PS}, rotates at a rate directly proportional to the frequency error (between the replica carrier and the incoming carrier). When true frequency lock is actually achieved, the vector stops rotating, but it may stop at any angle with respect to the I-axis. For this reason, coherent code tracking, as will be discussed in the following section, is not possible while in FLL because it depends on the I components being maximum (signal plus noise) and the Q components to be minimum (noise only) (i.e., in phase lock). It is possible to demodulate the SV data bit stream in FLL by a technique called *differential demodulation*. Because the demodulation technique involves a differentia-

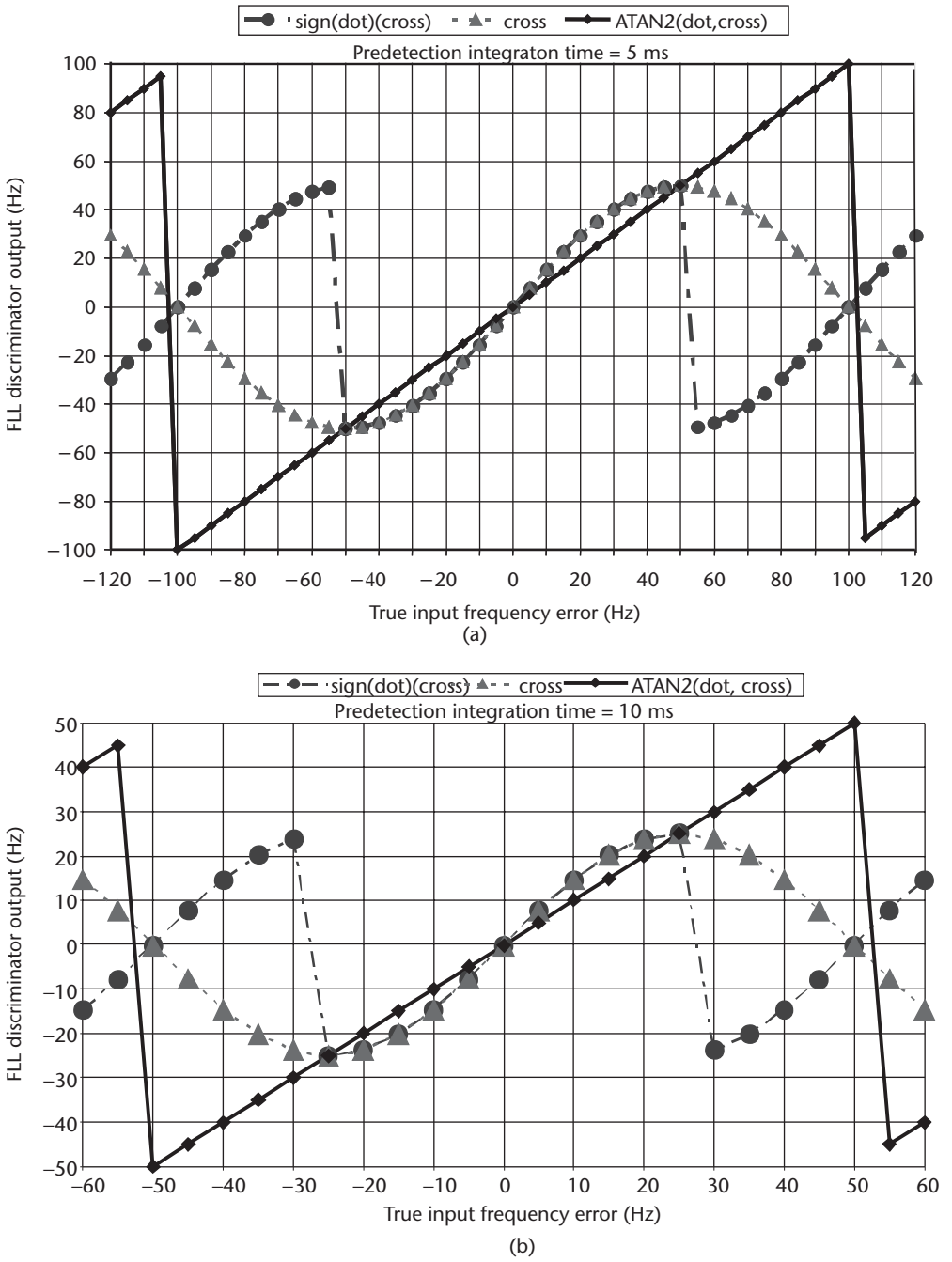


Figure 5.11 Comparison of frequency lock loop discriminators: (a) 5-ms predetection integration time, and (b) 10-ms predetection integration time.

tion (noisy) process, detecting the change in sign of the phasor in a FLL is noisier than detecting the sign of the integrated (lower noise) I_{PS} in a PLL. Therefore, for the same signal quality, FLL data detection has a much higher bit and word error rate than PLL data detection.

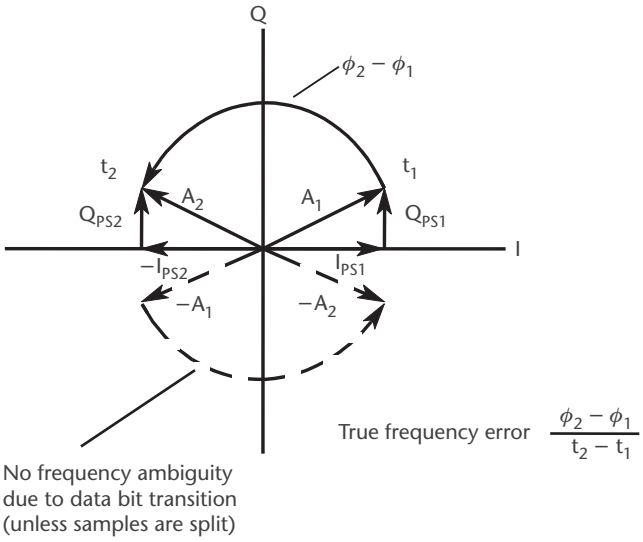


Figure 5.12 *I, Q* phasor diagram depicting true frequency error between replica and incoming carrier frequency.

5.4 Code Tracking Loops

Figure 5.13 shows a block diagram of a GPS receiver code tracking loop. The design of the programmable predetection integrators, the code loop discriminator, and the code loop filter characterizes the receiver code tracking loop. These three functions determine the most important two performance characteristics of the receiver code loop design: the code loop thermal noise error and the maximum LOS dynamic stress threshold. Even though the carrier tracking loop is the weak link in terms of the receiver's dynamic stress threshold, it would be disastrous to attempt to aid the

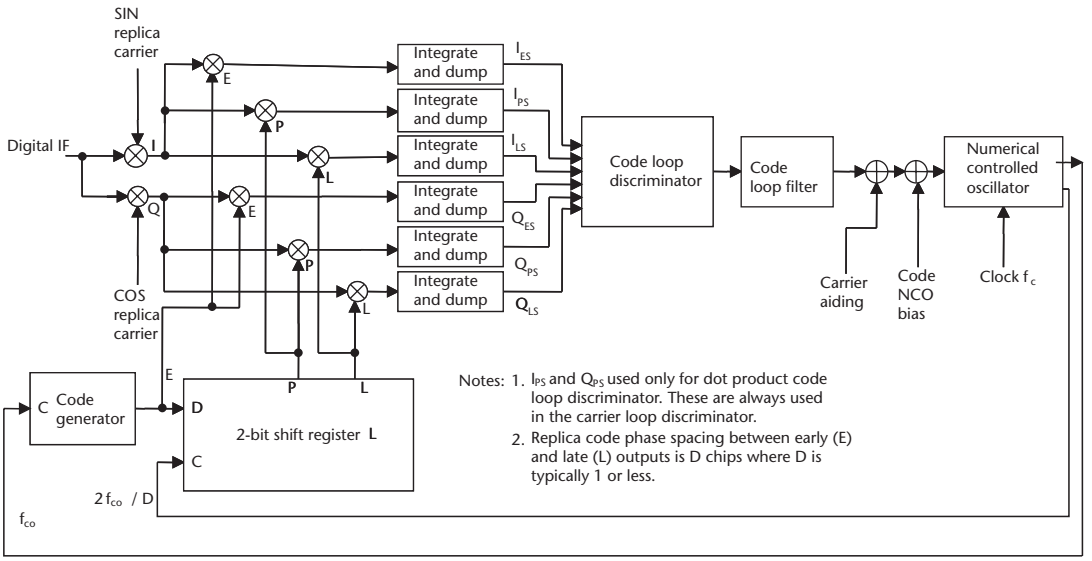


Figure 5.13 Generic GPS receiver code tracking loop block diagram.

carrier loop with the code loop output. This is because, unaided, the code loop thermal noise is orders of magnitude larger than the carrier loop thermal noise.

Table 5.5 summarizes four GPS receiver delay lock loop (DLL) discriminators and their characteristics. The fourth DLL discriminator is called a coherent dot product DLL. A more linear version can be implemented using only the E and L components, but the dot product slightly outperforms it. The coherent DLL provides superior performance when the carrier loop is in PLL. Under this condition, there is signal plus noise in the I components and mostly noise in the Q components. However, this high-precision DLL mode fails if there are frequent cycle slips or total loss of phase lock because the phasor rotates, causing the signal power to be shared in both the I and Q components, which consequently causes power loss in the coherent DLL. Successful operation requires a sensitive phase lock detector and rapid transition to the quasi-coherent DLL. All of the DLL discriminators can be normalized. Normalization removes the amplitude sensitivity, which improves performance under rapidly changing SNR conditions. Therefore, normalization helps the DLL tracking and threshold performance to be independent of AGC performance. However, normalization does not prevent reduction of the gain (slope) when SNR decreases. As SNR is reduced, the DLL slope approaches zero. Since loop bandwidth

Table 5.5 Common Delay Lock Loop Discriminators

Discriminator Algorithm	*Characteristics*
$\frac{1}{2}\frac{E-L}{E+L}$ where: $E=\sqrt{I_{ES}^2+Q_{ES}^2}, L=\sqrt{I_{LS}^2+Q_{LS}^2}$	Noncoherent early minus late envelope normalized by $E+L$ to remove amplitude sensitivity. High computational load. For 1-chip $E-L$ correlator spacing, produces true tracking error within ±0.5 chip of input error (in the absence of noise). Becomes unstable (divide by zero) at ±1.5-chip input error, but this is well beyond code tracking threshold in the presence of noise.
$\frac{1}{2}(E^2-L^2)$	Noncoherent early minus late power. Moderate computational load. For 1-chip $E-L$ correlator spacing, produces essentially the same error performance as 0.5 $(E-L)$ envelope within ±0.5 chip of input error (in the absence of noise). Can be normalized with E^2+L^2.
$\frac{1}{2}[(I_{ES}-I_{LS})I_{PS}+(Q_{ES}-Q_{LS})Q_{PS}]$ (dot product) $\frac{1}{4}[(I_{ES}-I_{LS})/I_{PS}+(Q_{ES}-Q_{LS})/Q_{PS}]$ (normalized with I_{PS}^2 and Q_{PS}^2)	Quasi-coherent dot product power. Uses all three correlators. Low computational load. For 1-chip $E-L$ correlator spacing, it produces nearly true error output within ±0.5 chip of input (in the absence of noise). Normalized version shown second using $I_{PS}{}^2$ and $Q_{PS}{}^2$, respectively.
$\frac{1}{2}(I_{ES}-I_{LS})I_{PS}$ (dot product) $\frac{1}{4}\frac{(I_{ES}-I_{LS})}{I_{PS}}$ (normalized with I_{PS}^2)	Coherent dot product. Can be used only when carrier loop is in phase lock. Low computational load. Most accurate code measurements. Normalized version shown second using $I_{PS}{}^2$.

Note: The code loop discriminator outputs may be summed to reduce the iteration rate of the code loop filter as compared to that of the carrier loop filter when the code loop is aided by the carrier loop. The rule-of-thumb limit is that total integration time must be less than one-fourth the DLL bandwidth. Note that this does not increase the predetection integration time for the code loop but does reduce noise. However the code loop NCO must be updated every time the carrier loop NCO is updated, even though the code loop filter output has not been updated. The last code loop filter output is combined with the current value of carrier aiding.

is roughly proportional to loop gain, loop bandwidth approaches zero at low SNR. This results in poor DLL response to dynamic stress and can result in instability if a third-order DLL filter is used (never used with carrier-aided code implementation). Carrier aiding (including externally provided carrier aiding) minimizes this problem, but the phenomena may produce unexpected DLL behavior at very low SNR.

Figure 5.14 compares the four DLL discriminator outputs. The plots assume 1-chip spacing between the early and late correlators. This means that the 2-bit shift register is shifted at twice the clock rate of the code generator. Also assumed is an ideal correlation triangle (infinite bandwidth) and that there is no noise on the I and Q measurements. For typical receiver bandwidths, the correlation peak tends to be rounded, the ramps on either side of the peak are nonlinear, and the correlation amplitudes at ± 0.5-chip from the correlation peak are slightly higher than for the infinite bandwidth case, while the prompt correlation amplitude is slightly lower.

The normalized early minus late envelope discriminator is very popular because its output error is linear over a 1-chip range, but the dot product power discriminator slightly outperforms it. Some GPS receiver designs synthesize the early minus late replica code as a combined replica signal. The benefit is that only one complex correlator is required to generate an early minus late output. This can be normalized with the prompt signal, but linear operation in the 1-chip range can only be achieved with $E + L$ normalization. This requires dedicated E and L correlators.

To reduce the computational burden of forming the GPS signal envelopes (the magnitude of the I and Q vectors), approximations are often used. Two of the most popular approximations (named after their originators) are the JPL approximation and the Robertson approximation.

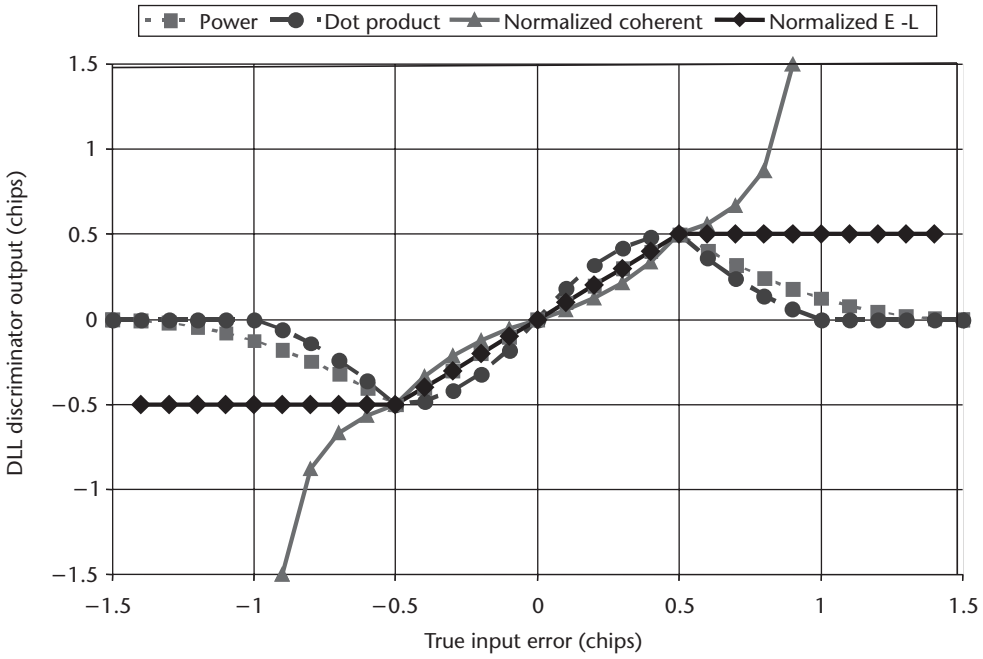


Figure 5.14 Comparison of delay lock loop discriminators.

The JPL approximation to $A = \sqrt{I^2 + Q^2}$ is defined by:

$$\begin{aligned} A_{ENV} &= X + 1/8Y \quad \text{if } X \geq 3Y \\ A_{ENV} &= 7/8\,X + 1/2Y \quad \text{if } X < 3Y \\ &\text{where} \\ X &= MAX(|I|,|Q|) \\ Y &= MIN(|I|,|Q|) \end{aligned} \tag{5.2}$$

The Robertson approximation is:

$$A_{ENV} = MAX(|I| + 1/2\,|Q|, |Q| + 1/2|I|) \tag{5.3}$$

The JPL approximation is more accurate but has a greater computational burden.

Figure 5.15 illustrates the envelopes that result for three different replica code phases being correlated simultaneously with the same incoming SV signal. For ease of visualization, the in-phase component of the incoming SV signal is shown without noise. The three replica phases are 1/2 chip apart and are representative of the early, prompt, and late replica codes that are synthesized in the code loop of Figure 5.13.

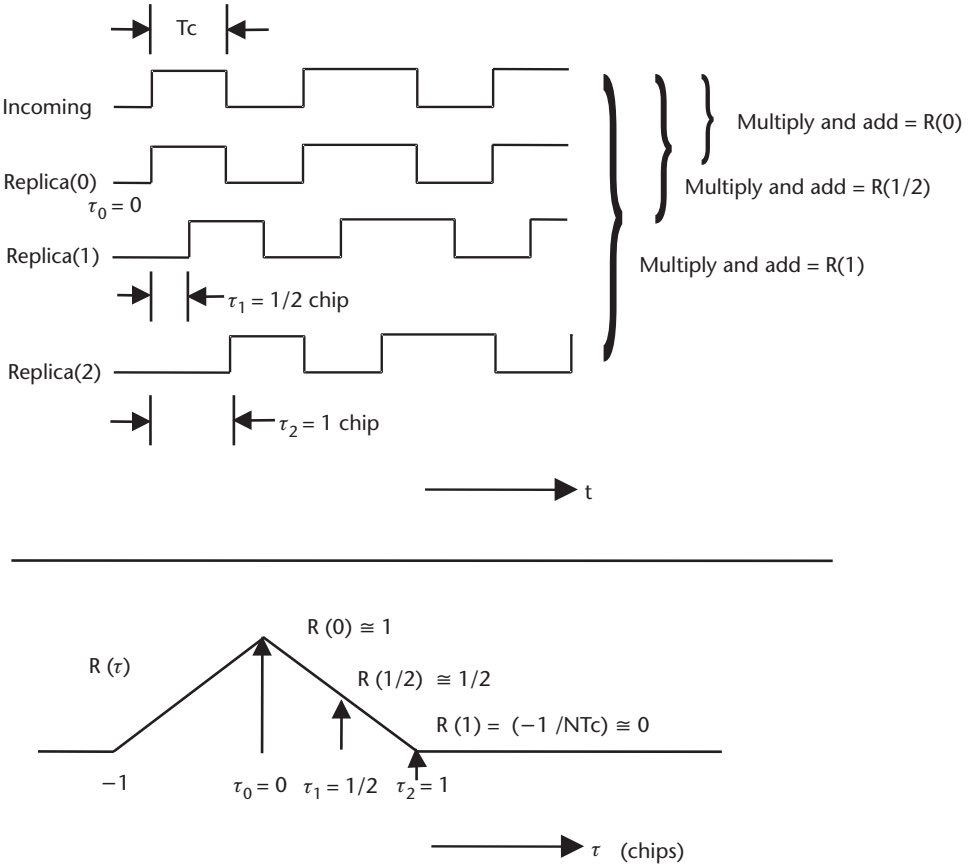


Figure 5.15 Code correlation process for three different replica code phases.

Figure 5.16 illustrates how the early, prompt, and late envelopes change as the phases of the replica code signals are advanced with respect to the incoming SV signal. For ease of visualization, only 1 chip of the continuous PRN signal is shown, and the incoming SV signal is shown without noise. Figure 5.17 illustrates the normalized early minus late envelope discriminator error output signals corresponding to the four replica code offsets in Figure 5.16. The closed code loop operation becomes apparent as a result of studying these replica code phase changes, the envelopes that they produce, and the resulting error output generated by the early minus late envelope code discriminator. If the replica code is aligned, then the early and late envelopes are equal in amplitude and no error is generated by the discriminator. If the replica code is misaligned, then the early and late envelopes are unequal by an amount that is proportional to the amount of code phase error between the replica and the incoming signal (within the limits of the correlation interval). The code discriminator senses the amount of error in the replica code and the direction (early or late) from the difference in the amplitudes of the early and late envelopes. This error is filtered and then applied to the code loop NCO, where the output frequency is increased or decreased as necessary to correct the replica code generator phase with respect to the incoming SV signal code phase.

The discriminator examples given thus far have assumed that each channel of the GPS receiver contains three complex code correlators to provide early, prompt, and late correlated outputs. In early generations of GPS receiver designs, analog correlators were used instead of digital correlators. There was strong emphasis on reducing the number of expensive and power-hungry analog correlators, so there were numerous code tracking loop design innovations that minimized the number of correlators. The *tau-dither* technique time shares the early and late replica code with one complex (I and Q) correlator. This suffers a 3-dB loss of tracking threshold in the code loop because only half the energy is available from the early and late signals. This loss of threshold is unimportant in an unaided GPS receiver design because there is usually more than a 3-dB difference between the conventional code

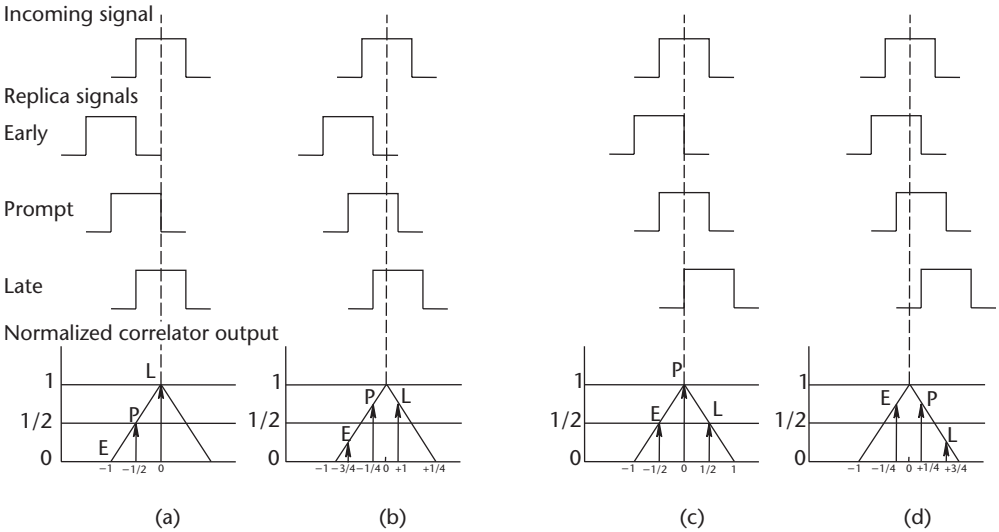


Figure 5.16 Code correlation phases: (a) replica code 1/2-chip early, (b) replica code 1/4-chip early, (c) replica code aligned, and (d) replica code 1/4-chip late.

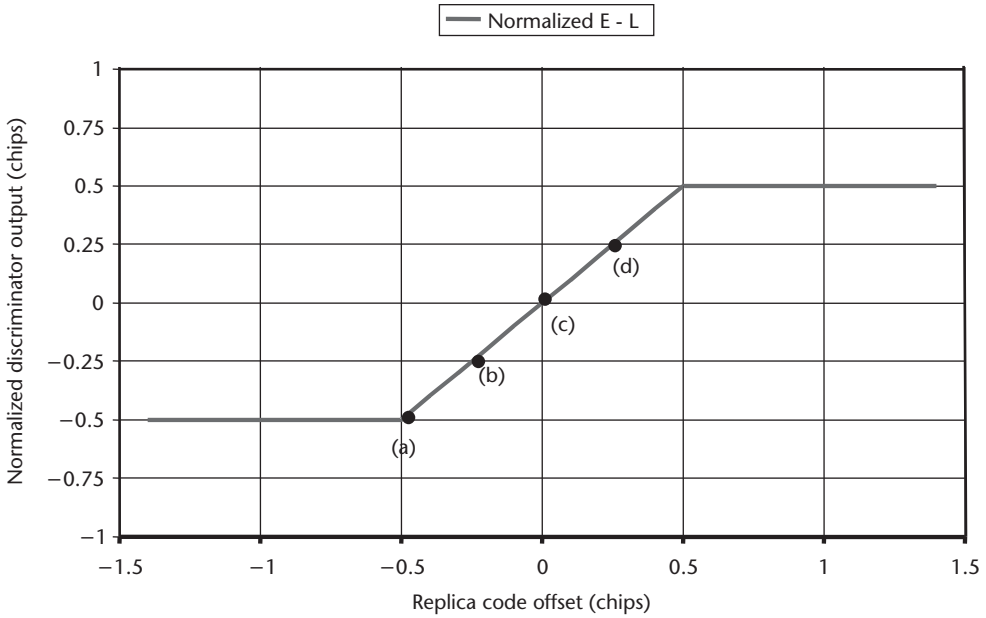


Figure 5.17 Code discriminator output versus replica code offset.

loop and carrier tracking loop thresholds. The extra margin in the code loop threshold only pays off for aided GPS receivers. The Texas Instruments TI 4100 GPS receiver [3] not only used the tau-dither technique, but also time shared only two analog correlators and the same replica code and carrier generators to simultaneously and continuously track (using 2.5-ms dwells) the L1 P code and L2 P code signals of four GPS satellites. It also simultaneously demodulated the 50-Hz navigation messages. Because the L2 tracking was accomplished by tracking L1-L2, this signal with nearly zero dynamics permitted very narrow bandwidth tracking loops and therefore suffered only a little more than 6 dB of tracking threshold losses instead of the expected 12 dB. Since the same circuits were time shared across all channels and frequencies, there was zero interchannel bias error in the TI 4100 measurements.

Modern digital GPS receivers often contain many more than three complex correlators because digital correlators are relatively inexpensive (e.g., only one exclusive-or circuit is required to perform the 1-bit multiply function). The innovations relating to improved performance through the use of more than three complex correlators include faster acquisition times [4], multipath mitigation (e.g., see [5], and also Section 6.3), and a wider discriminator correlation interval that provides jamming robustness when combined with external (IMU) aiding [6]. However, there is no improvement in tracking error due to thermal noise or improvement in tracking threshold using multiple correlators. Reducing parts count and power continue to be important, so multiplexing is back in vogue. The speed of digital circuits has increased to the point that correlators, NCOs, and other high-speed baseband functions can be digitally multiplexed without a significant power penalty because of the reduction in feature size of faster digital components. The multiplexing is faster than the real-time digital sampling of the GPS signals by a factor of N, where N is the number of channels sharing the same device. Since there is no loss of energy, there is

no loss of signal processing performance, as was the case with the TI 4100 analog multiplexing. There is also no interchannel bias error.

5.5 Loop Filters

The objective of the loop filter is to reduce noise in order to produce an accurate estimate of the original signal at its output. The loop filter order and noise bandwidth also determine the loop filter's response to signal dynamics. As shown in the receiver block diagrams, the loop filter's output signal is effectively subtracted from the original signal to produce an error signal, which is fed back into the filter's input in a closed loop process. There are many design approaches to digital filters. The design approach described here draws on existing knowledge of analog loop filters, then adapts these into digital implementations. Figure 5.18 shows block diagrams of first, second, and third-order analog filters.[1] Analog integrators are represented by $1/s$, the Laplace transform of the time domain integration function. The input signal is multiplied by the multiplier coefficients, then processed as shown in Figure 5.18. These multiplier coefficients and the number of integrators completely determine the loop filter's characteristics. Table 5.6 summarizes these filter characteristics and provides all of the information required to compute the filter coefficients for first, second, and third-order loop filters. Only the filter order and noise bandwidth must be chosen to complete the design.

Figure 5.19 depicts the block diagram representations of analog and digital integrators. The analog integrator of Figure 5.19(a) operates with a continuous time domain input, $x(t)$, and produces an integrated version of this input as a continuous time domain output, $y(t)$. Theoretically, $x(t)$ and $y(t)$ have infinite numerical resolution, and the integration process is perfect. In reality, the resolution is limited by noise, which significantly reduces the dynamic range of analog integrators. There are also problems with drift.

The boxcar digital integrator of Figure 5.19(b) operates with a sampled time domain input, $x(n)$, which is quantized to a finite resolution and produces a discrete integrated output, $y(n)$. The time interval between each sample, T, represents a unit delay, z^{-1}, in the digital integrator. The digital integrator performs discrete integration perfectly with a dynamic range limited only by the number of bits used in the accumulator, A. This provides a dynamic range capability much greater than can be achieved by its analog counterpart, and the digital integrator does not drift. The boxcar integrator performs the function $y(n) = T[x(n)] + A(n-1)$, where n is the discrete sampled sequence number.

Figure 5.19(c) depicts a digital integrator that linearly interpolates between input samples and more closely approximates the ideal analog integrator. This is called the bilinear z-transform integrator. It performs the function $y(n) = T/2[x(n)] + A(n-1) = 1/2[A(n) + A(n-1)]$. The digital filters depicted in Figure 5.20 result when the Laplace integrators of Figure 5.18 are each replaced with the digital

1. Jerry D. Holmes originally developed these analog and digital loop filter architectures and filter parameters. They were used in the first commercial GPS receiver design, the TI 4100 NAVSTAR Navigator, Texas Instruments, Inc., 1982.

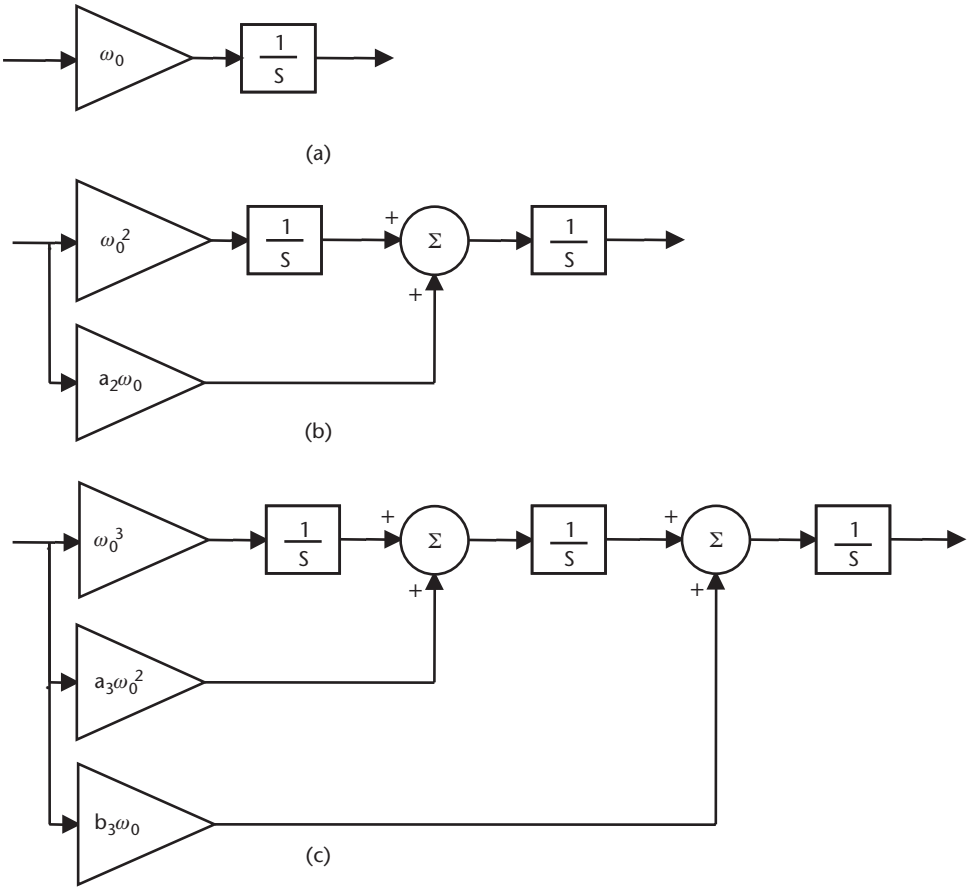


Figure 5.18 Block diagrams of: (a) first-, (b) second-, and (c) third-order analog loop filters.

Table 5.6 Loop Filter Characteristics

Loop Order	*Noise Bandwidth B_n (Hz)*	*Typical Filter Values*	*Steady State Error*	*Characteristics*
First	$\frac{\omega_0}{4}$	ω_0 $B_n = 0.25\ \omega_0$	$\frac{(dR/dt)}{\omega_0}$	Sensitive to velocity stress. Used in aided code loops and sometimes used in aided carrier loops. Unconditionally stable at all noise bandwidths.
Second	$\frac{\omega_0(1+a_2^2)}{4a_2}$	ω_0^2 $a_2\omega_0 = 1.414\omega_0$ $B_n = 0.53\ \omega_0$	$\frac{(d^2R/dt^2)}{\omega_0^2}$	Sensitive to acceleration stress. Used in aided and unaided carrier loops. Unconditionally stable at all noise bandwidths.
Third	$\frac{\omega_0(a_3b_3^2+a_3^2-b_3)}{4(a_3b_3-1)}$	ω_0^3 $a_3\omega_0^2 = 1.1\omega_0^2$ $b_3\omega_0 = 2.4\omega_0$ $B_n = 0.7845\ \omega_0$	$\frac{(d^3R/dt^3)}{\omega_0^3}$	Sensitive to jerk stress. Used in unaided carrier loops. Remains stable at $B_n \le 18$ Hz.

Source: [7].
Note: The loop filter natural radian frequency, ω_0, is computed from the value of the loop filter noise bandwidth, B_n, selected by the designer. R is the LOS range to the satellite. The steady state error is inversely proportional to the nth power of the tracking loop bandwidth and directly proportional to the nth derivative of range, where n is the loop filter order. Also see footnote 1.

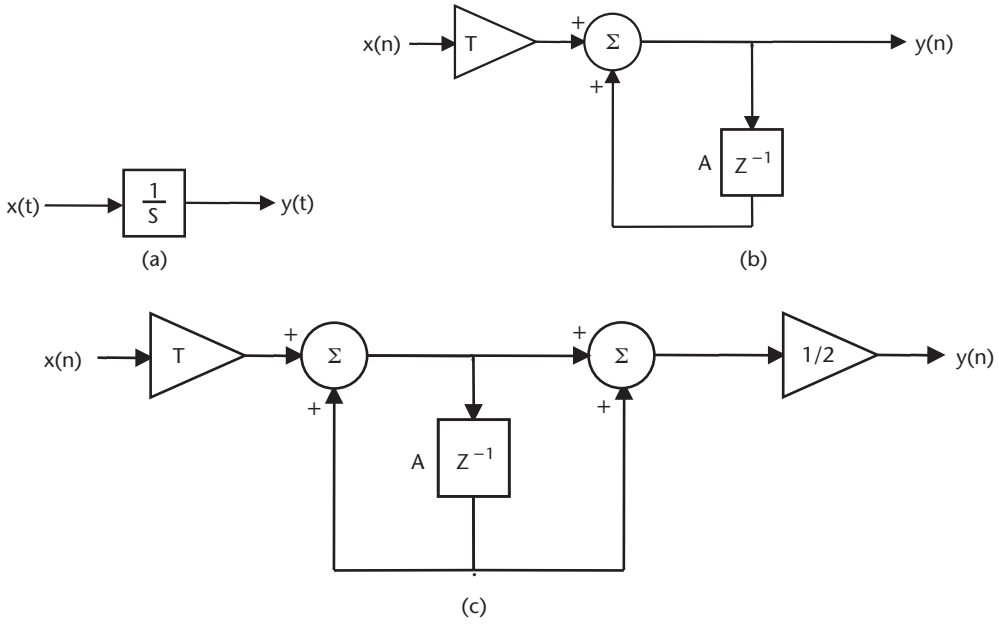


Figure 5.19 Block diagrams of: (a) analog, (b) digital boxcar, and (c) digital bilinear transform integrators.

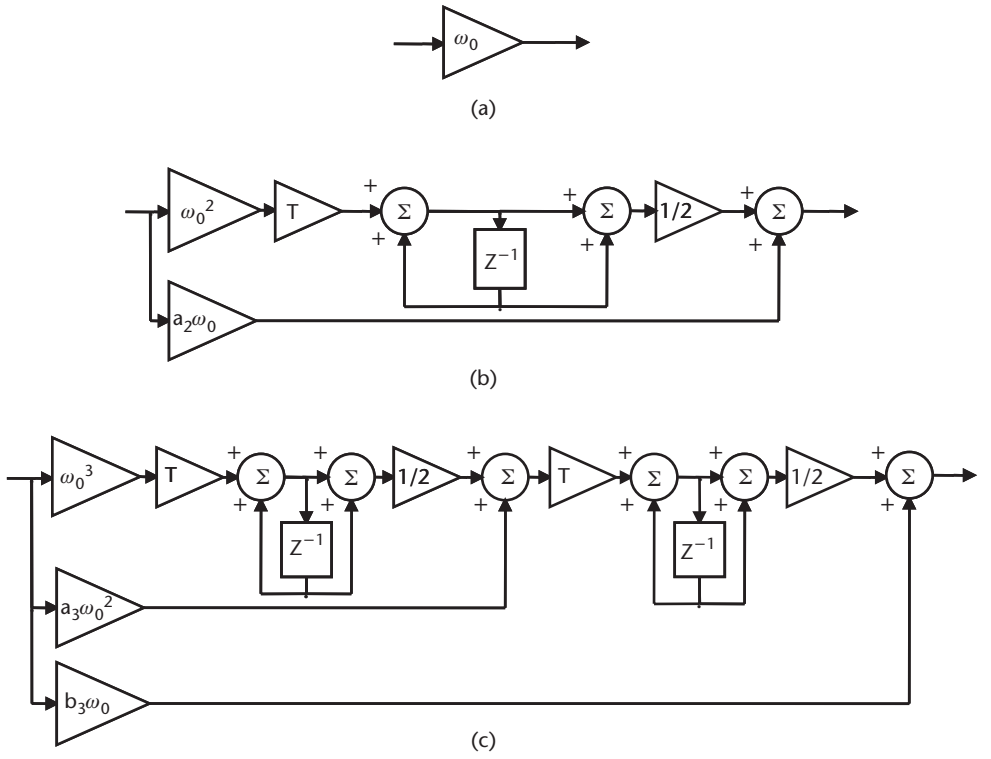


Figure 5.20 Block diagrams of (a) first-, (b) second-, and (c) third-order digital loop filters excluding last integrator (the NCO).

bilinear integrator shown in Figure 5.19(c). The last digital integrator is not included because this function is implemented by the NCO. The NCO is equivalent to the boxcar integrator of Figure 5.19(b).

Figure 5.21 illustrates two FLL-assisted PLL loop filter designs (see footnote 1). Figure 5.21(a) depicts a second-order PLL filter with a first-order FLL assist. Figure 5.21(b) depicts a third-order PLL filter with a second-order FLL assist. If the PLL error input is zeroed in either of these filters, the filter becomes a pure FLL. Similarly, if the FLL error input is zeroed, the filter becomes a pure PLL. The lowest noise loop closure process is to close in pure FLL, then apply the error inputs from both discriminators as an FLL-assisted PLL until phase lock is achieved, then convert to pure PLL until phase lock is lost. However, if the noise bandwidth parameters are chosen correctly, there is very little loss in the ideal carrier tracking threshold performance when both discriminators are continuously operated [7]. In general, the natural radian frequency of the FLL, ω_{0f}, is different from the natural radian frequency of the PLL, ω_{0p}. These natural radian frequencies are determined from the desired loop filter noise bandwidths, B_{nf} and B_{np}, respectively. The values for the second-order coefficient a_2 and third-order coefficients a_3 and b_3 can be determined from Table 5.6. These coefficients are the same for FLL, PLL, or DLL applications if the loop order and the noise bandwidth, B_n, are the same. Note that the FLL coefficient insertion point into the filter is one integrator back from the PLL and DLL insertion points. This is because the FLL error is in units of hertz (change in range per unit of time), whereas the PLL and DLL errors are in units of phase (range).

A loop filter parameter design example will clarify the use of the equations in Table 5.6. Suppose that the receiver carrier tracking loop will be subjected to high

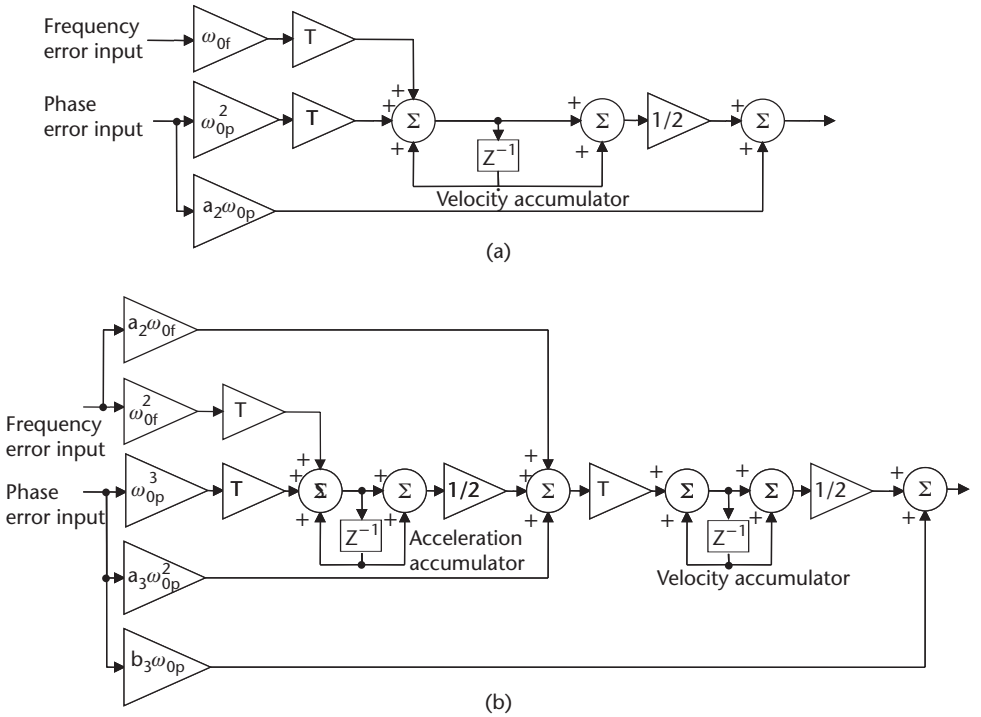


Figure 5.21 Block diagrams of FLL-assisted PLL filters: (a) second-order PLL with first-order FLL assist, and (b) third-order PLL with second-order FLL assist.

acceleration dynamics and will not be aided by an external navigation system, but must maintain PLL operation. A third-order loop is selected because it is insensitive to acceleration stress. To minimize its sensitivity to jerk stress, the noise bandwidth, B_n, is chosen to be the widest possible consistent with stability. Table 5.6 indicates that $B_n \leq 18$ Hz is safe. This limitation has been determined through extensive Monte Carlo simulations and is related to the maximum predetection integration time (which is typically the same as the reciprocal of the carrier loop iteration rate) plus extremes of noise and dynamic range. If $B_n = 18$ Hz, then $\omega_0 = B_n/0.7845 = 22.94455$ rad/s. The three multipliers shown in Figure 5.20(c) are computed as follows:

$$\omega_0^3 = 12{,}079.21$$
$$a_3\omega_0^2 = 1.1\omega_0^2 = 579.10$$
$$b_3\omega_0 = 2.4\omega_0 = 55.07$$

If the carrier loop is updated at a 200-Hz rate, then $T = 0.005$ second for use in the digital integrators. This completes the third-order filter parameter design. The remainder of the loop filter design is the implementation of the digital integrator accumulators to ensure that they will never overflow (i.e., that they have adequate dynamic range). The use of floating point arithmetic in modern microprocessors with built-in floating point hardware greatly simplifies this part of the design process. Note that in Figure 5.21(b), the velocity accumulator contains the loop filter estimate of LOS velocity between the antenna phase center and the SV. This estimate includes a self-adjusting bias component that compensates the carrier tracking loop for the reference oscillator frequency error (i.e., the time bias rate error that is in common with all tracking channels). Similarly, the acceleration accumulator contains the loop filter estimate of LOS acceleration that includes a self-adjusting bias component, which compensates the carrier tracking loop for the time rate of change of the reference oscillator frequency error. These accumulators should be initialized to zero just before initial loop closure unless good estimates of the correct values are known a priori. Also, they should be reset to their bias components (as learned by the navigation process) or to zero if unknown at the exact instance of injecting external carrier velocity aiding into the closed loop.

It should be noted that the loop filters described in this section, and in general any loop filters that are based on an adaptation of analog designs, only achieve the design noise bandwidth, B_n, when the product B_nT is very small (well below unity). As this product increases, the true noise bandwidth tends to be larger than the target value, and eventually the loop becomes unstable. An alternative loop formulation described in [8] overcomes some of these limitations. However, instability for extremely large values of the product B_nT is inevitable for any loop filter.

5.6 Measurement Errors and Tracking Thresholds

The GPS measurement errors and tracking thresholds are closely related because the receiver loses lock when the measurement errors exceed a certain boundary. Because the code and carrier tracking loops are nonlinear, especially near the

threshold regions, only Monte Carlo simulations of the GPS receiver under the combined dynamic and SNR conditions will determine the true tracking performance [9]. However, general rules that approximate the measurement errors of the tracking loops can be used based on closed form equations. Numerous sources of measurement errors are in each type of tracking loop. However, it is sufficient for rule-of-thumb tracking thresholds to analyze only the dominant error sources.

5.6.1 PLL Tracking Loop Measurement Errors

The dominant sources of phase error in a GPS receiver PLL are phase jitter and dynamic stress error. A conservative rule of thumb for tracking threshold is that the 3-sigma jitter must not exceed one-fourth of the phase pull-in range of the PLL discriminator. Only arctangent carrier phase discriminators are considered for the generic receiver design. In the case of a dataless PLL four-quadrant arctangent discriminator whose phase pull-in range is 360°, the 3-sigma rule threshold is therefore 90°. For the case where there is data modulation, the PLL two-quadrant arctangent discriminator must be used and has a phase pull-in range of 180°. Therefore the 3-sigma rule threshold is 45°. Therefore, the PLL rule thresholds are stated as follows:

$$\begin{aligned} 3\sigma_{PLL} &= 3\sigma_j + \theta_e \le 90^\circ \text{ (dataless)} \\ 3\sigma_{PLL} &= 3\sigma_j + \theta_e \le 45^\circ \text{ (data present)} \end{aligned} \tag{5.4}$$

where:

σ_j = 1-sigma phase jitter from all sources except dynamic stress error

θ_e = dynamic stress error in the PLL tracking loop

Equation (5.4) implies that dynamic stress error is a 3-sigma effect and is additive to the phase jitter. The phase jitter is the RSS of every source of uncorrelated phase error, such as thermal noise and oscillator noise. Oscillator noise includes vibration-induced jitter and Allan deviation–induced jitter. It also includes satellite oscillator phase noise. Even though IS-GPS-200 [2] specifies that this is no greater than 0.1 rad (5.7°) 1-sigma tracking error in a 10-Hz PLL, the operational SVs exhibit about an order of magnitude lower error than this to date. This external source of noise jitter is not included in the foregoing analysis but should be considered in very narrowband PLL applications.

In the P(Y) code and C/A code examples to follow, the presence of data modulation is assumed. Expanding on (5.4), the 1-sigma rule threshold for the PLL tracking loop for the two-quadrant arctangent discriminator is therefore:

$$\sigma_{PLL} = \sqrt{\sigma_{tPLL}^2 + \sigma_v^2 + \theta_A^2} + \frac{\theta_e}{3} \le 15^\circ \qquad \text{(data present)} \tag{5.5}$$

where:

σ_{tPLL} = 1-sigma thermal noise in degrees

σ_v = 1-sigma vibration-induced oscillator jitter in degrees

θ_A = Allan variance–induced oscillator jitter in degrees

5.6.1.1 PLL Thermal Noise

Often the PLL thermal noise is treated as the only source of carrier tracking error, since the other sources of PLL jitter may be either transient or negligible. The thermal noise jitter for an arctangent PLL is computed as follows:

$$\sigma_{PLLt} = \frac{360}{2\pi}\sqrt{\frac{B_n}{C/N_0}\left(1+\frac{1}{2TC/N_0}\right)} \quad \text{(degrees)} \tag{5.6}$$

$$\sigma_{PLLt} = \frac{\lambda_L}{2\pi}\sqrt{\frac{B_n}{C/N_0}\left(1+\frac{1}{2TC/N_0}\right)} \quad (\text{m}) \tag{5.7}$$

where:

B_n = carrier loop noise bandwidth (Hz)

C/N_0 = carrier to noise power expressed as a ratio (Hz)

$= 10^{\frac{(C/N_0)_{dB}}{10}}$ for $(C/N_0)_{dB}$ expressed in dB-Hz

T = predetection integration time (seconds)

λ_L = GPS L-band carrier wavelength (m)

= (299,792,458 m/s)/(1,575.42 MHz) = 0.1903 m/cycle for L1

= (299,792,458 m/s)/(1,227.6 MHz) = 0.2442 m/cycle for L2

Note that (5.6) and (5.7) do not include factors relating to C/A code or P(Y) code or the loop filter order. Also note that (5.6) is independent of carrier frequency when the error is expressed in units of degrees. The carrier thermal noise error standard deviation is strictly dependent on the carrier–to-noise power ratio, C/N_0, the noise bandwidth, B_n, and the predetection integration time, T. The carrier-to-noise power ratio, C/N_0, is an important factor in many GPS receiver performance measures. It is computed as the ratio of recovered power, C, (in W) from the desired signal to the noise density N_0 (in W/Hz). Methods for determining C/N_0 with and without external interference are described in Section 6.2.2. If C/N_0 increases (e.g., the recovered signal power is increased or the noise level is decreased), the standard deviation decreases. Decreasing the noise bandwidth reduces the standard deviation. The part of the equation involving the predetection integration time, T, is called the squaring loss. Increasing the predetection integration time reduces the squaring loss, which in turn decreases the standard deviation.

It is a common misconception that the GPS P(Y) (precision) codes always produce more accurate measurements than the C/A (coarse) codes. While the P(Y) code signal provides the potential for better code tracking accuracy, the PLL thermal noise error is the same for either code for the same C/N_0, since PLL processing uses quantities after the spreading code has been stripped off. Further, since the C/A code is 3 dB stronger than P(Y) code, the carrier thermal noise error is smaller for C/A

code receivers than for P(Y) code receivers, assuming no signal jamming and that all other things in the receiver designs are equal. It is another common misconception that the carrier loop measurement is a velocity measurement, when actually it is a range measurement. The PLL thermal noise error is in units of range because it is part of the carrier Doppler phase measurement (i.e., a measurement of some fraction of the carrier wavelength). The velocity is approximated using the change in carrier Doppler phase between two carrier range measurements over a short time. Any single PLL measurement is a precise phase measurement of the carrier Doppler phase within one cycle. The differential measurement must also account for the integer number of carrier Doppler phase cycles that occur between measurements.

Figure 5.22 illustrates the PLL thermal noise jitter plotted as a function of $(C/N_0)_{dB}$ for three different noise bandwidths assuming the maximum 20-ms predetection integration time. Even though the thermal noise jitter is not directly dependent on the loop order, there is an indirect relationship. The loop order is sensitive to the same order of dynamics (first order to velocity stress, second order to acceleration stress, and third order to jerk stress), and the loop bandwidth must be wide enough to accommodate these higher-order dynamics. In general, when the loop order is made higher, there is an improvement in dynamic stress performance. Thus, the thermal noise can be reduced for the same minimum C/N_0 by increasing the loop order and reducing the noise bandwidth while also improving the dynamic performance.

5.6.1.2 Vibration-Induced Oscillator Phase Noise

Vibration-induced oscillator phase noise is a complex analysis problem. In some cases, the expected vibration environment is so severe that the reference oscillator

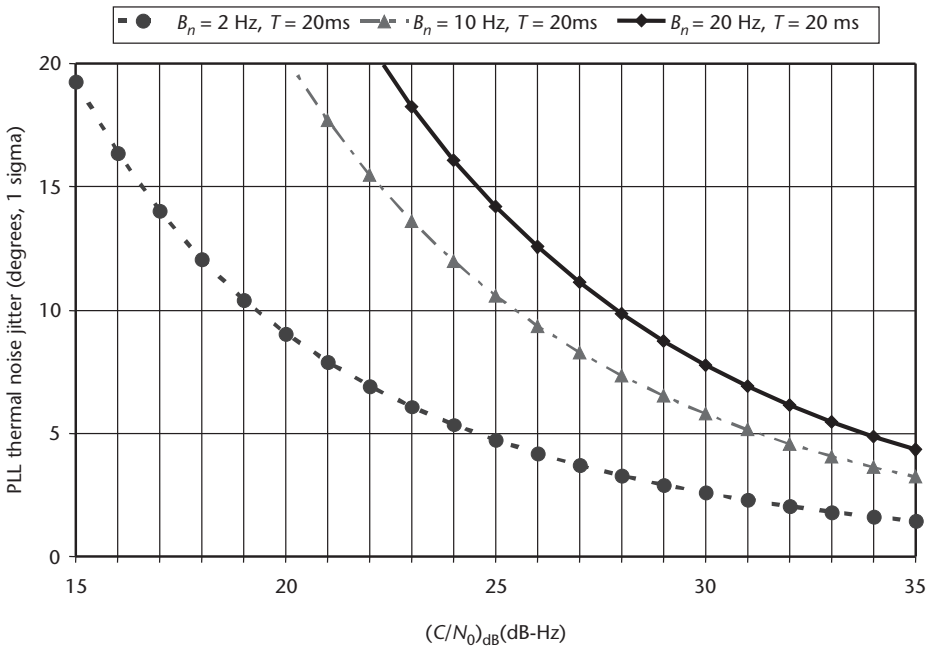


Figure 5.22 PLL thermal noise jitter.

must be mounted using vibration isolators in order for the GPS receiver to successfully operate in PLL. The equation for vibration induced oscillator jitter is:

$$\sigma_v = \frac{360 f_L}{2\pi} \sqrt{\int_{f\,\min}^{f\,\max} S_v^2(f_m) \frac{P(f_m)}{f_m^2} df_m} \quad \text{(degrees)} \tag{5.8}$$

where:

f_L = L-band input frequency in Hz

$S_v(f_m)$ = oscillator vibration sensitivity of $\Delta f/f_L$ per g as a function of f_m

f_m = random vibration modulation frequency in Hz

$P(f_m)$ = power curve of the random vibration in g^2/Hz as a function of f_m

g = the acceleration due to gravity ≈ 9.8 m/s^2

If the oscillator vibration sensitivity, $S_v(f_m)$, is not variable over the range of the random vibration modulation frequency, f_m, then (5.8) can be simplified to:

$$\sigma_v = \frac{360 f_L S_v}{2\pi} \sqrt{\int_{f\,\min}^{f\,\max} \frac{P(f_m)}{f_m^2} df_m} \quad \text{(degrees)} \tag{5.9}$$

As a simple computational example, assume that the random vibration power curve is flat from 20 Hz to 2,000 Hz with an amplitude of 0.005 g^2/Hz. If $S_v = 1 \times 10^{-9}$ parts/g and f_L = L1 = 1,575.42 MHz, then the vibration-induced phase jitter using (5.9) is:

$$\sigma_v = 90.265 \sqrt{0.005 \int_{20}^{2000} \frac{df_m}{f_m^2}} = 90.265 \sqrt{0.005 \left(\frac{1}{20} - \frac{1}{200} \right)} = 1.42^\circ$$

5.6.1.3 Allan Deviation Oscillator Phase Noise

The equations used to determine Allan deviation phase noise are empirical. They are stated in terms of what the requirements are for the short-term stability of the reference oscillator as determined by the Allan variance method of stability measurement. (Appendix B contains a description of the Allan variance.) The equation for short-term Allan deviation for a second-order PLL is [10]:

$$\sigma_A(\tau) = 2.5 \frac{\Delta\theta}{\omega_L \tau} \quad \text{(dimensionless units of } \Delta f/f) \tag{5.10}$$

where:

$\Delta\theta$ = rms jitter into phase discriminator due to the oscillator (rad)

ω_L = L-band input frequency = $2\pi f_L$ (rad/s)

τ = short-term stability gate time for Allan variance measurement (seconds).

The equation for a third-order PLL is similar [10]:

$$\sigma_A(\tau) = 2.25\frac{\Delta\theta}{\omega_L\tau}(\text{dimensionless units of } \Delta f/f) \tag{5.11}$$

If the Allan variance, $\sigma_A^2(\tau)$, has already been determined for the oscillator for the short-term gate time, τ, then the Allan deviation–induced jitter in degrees, $\theta_A = 360\Delta\theta/2\pi$, can be computed from the previous equations. Usually $\sigma_A^2(\tau)$ changes very little for the short-term gate times involved. These gate times must include the reciprocal of the range of noise bandwidths used in the carrier loop filters, $\tau = 1/B_n$. A short-term gate-time range of 5 ms to 1,000 ms should suffice for all PLL applications. Rearranging (5.10) using these assumptions, the equation for the second-order loop is:

$$\theta_{A2} = 144\frac{\sigma_A(\tau)f_L}{B_n} \quad \text{(degrees)} \tag{5.12}$$

Rearranging (5.11) using these assumptions, the equation for the third-order loop is:

$$\theta_{A3} = 160\frac{\sigma_A(\tau)f_L}{B_n} \quad \text{(degrees)} \tag{5.13}$$

For example, assume that the loop filter is third-order with a noise bandwidth, $B_n = 18$ Hz, tracking the L1 signal, and the Allan deviation is specified to be $\sigma_A(\tau) = 1 \times 10^{-10}$ or better for gate times that include $\tau = 1/B_n = 56$ ms. The phase jitter contribution due to this error is $\theta_{A3} = 1.40°$ or less. Obviously, a reference oscillator with a short-term Allan deviation characteristic that is more than an order of magnitude worse than this example will cause PLL tracking problems.

Figure 5.23 graphically portrays the sensitivity of a third-order PLL to changes in short-term Allan deviation performance of the reference oscillator, especially as the noise bandwidth, B_n, is narrowed. The objective of narrowing the bandwidth is to reduce the thermal noise error to improve the tracking threshold. However, as Figure 5.23 illustrates, the Allan deviation effects begin to dominate at the narrower noise bandwidths. This effect is usually the primary source of aided GPS receiver narrowband PLL tracking problems, assuming that the external velocity aiding accuracy is not the limiting factor. However, even for an unaided GPS receiver, a reference oscillator with a poor Allan deviation characteristic, say a $\Delta f/f$ of less than 1×10^{-9}, will prevent reliable PLL operation. Therefore, the oscillator specification for Allan deviation is important for all GPS receiver designs.

5.6.1.4 Dynamic Stress Error

The dynamic stress error is obtained from the steady state error formulas shown in Table 5.6. This error depends on the loop bandwidth and order. The maximum dynamic stress error may be slightly larger than the steady state error if the loop filter response to a step function has overshot, but the steady state error formula will

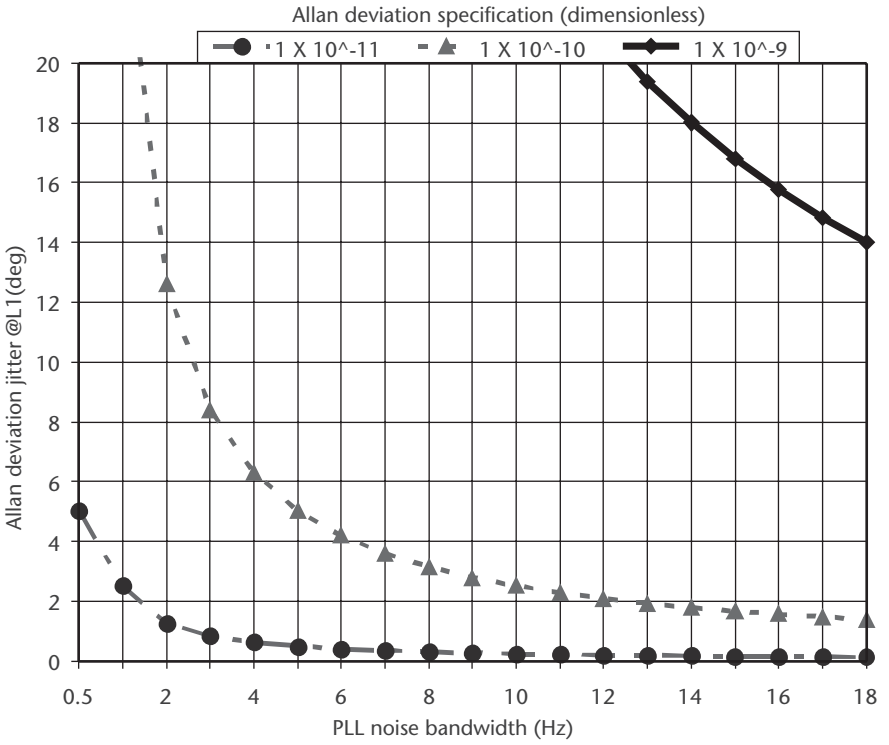


Figure 5.23 Allan deviation jitter in third-order PLL at L1.

suffice. There should be no more than about a 7% overshoot if the filter is designed for minimum mean square error, which is the case for the typical loop filter coefficients shown in the table. From Table 5.6, a second-order loop with minimum mean square error, the dynamic stress error is:

$$\theta_{e2} = \frac{d^2R/dt^2}{\omega_0^2} = \frac{d^2R/dt^2}{\left(\frac{B_n}{0.53}\right)^2} = 0.2809\frac{d^2R/dt^2}{B_n^2} \quad \text{(degrees)} \tag{5.14}$$

where d^2R/dt^2 = maximum LOS acceleration dynamics (°/s^2).

From Table 5.6, a third-order loop with minimum mean square error, the dynamic stress error is defined as follows:

$$\theta_{e3} = \frac{d^3R/dt^3}{\omega_0^3} = \frac{d^3R/dt^3}{\left(\frac{B_n}{0.7845}\right)^3} = 0.4828\frac{d^3R/dt^3}{B_n^3} \quad \text{(degrees)} \tag{5.15}$$

where d^3R/dt^3 = maximum LOS jerk dynamics (°/s^3).

Note that (5.15) is a 3-sigma error. As an example of how this error is computed, suppose the third-order loop noise bandwidth is 18 Hz and the maximum LOS jerk dynamic stress to the SV is 10 g/s = 98 m/s^3. To convert this to °/s^3, multiply the jerk dynamics by the number of carrier wavelengths contained in 1m in units of

°/m. For L1, d^3R/dt^3 = (98 m/s^3) × (360°/cycle) × (1,575.42 × 10^6 cycles/s)/c = 185,398°/s^3 where c = 299,792,458 m/s is the speed of light. For L2, $d^3R/dt^3 = 98 \times 360 \times 1{,}227.60 \times 10^6/c$ = 144,666°/s^3.

Using (5.15), the 3-sigma stress error for an 18-Hz third-order PLL is 15.35° for L1 and 11.96° for L2. These are well below the 45° 3-sigma rule-of-thumb levels.

5.6.1.5 Reference Oscillator Acceleration Stress Error

The PLL cannot tell the difference between the dynamic stress induced by real dynamics and the dynamic stress caused by changes in frequency in the reference oscillator due to acceleration sensitivity of the oscillator. The oscillator change in frequency due to dynamic stress is:

$$\Delta f_g = 360\, S_g f_L G(t) \quad (°/s) \tag{5.16}$$

where:

S_g = g-sensitivity of the oscillator ($\Delta f/f$ per g)
f_L = L-band input frequency (Hz)
= 1,575.42 MHz for L1
= 1,227.76 MHz for L2
$G(t)$ = acceleration stress in g as a function of time

For the components of $G(t)$ due to acceleration (g), the units of Δf_g are °/s, a velocity error as sensed by the loop filter. For an unaided second-order carrier tracking loop, this acceleration-induced oscillator error can be ignored because it is insensitive to velocity stress. For the components of $G(t)$ due to jerk stress (g/s), the units of Δf_g are °/s^2, an acceleration error as sensed by the loop filter. For an unaided third-order carrier tracking loop, this jerk-induced oscillator error can be ignored because it is insensitive to acceleration stress. In reality, there will always be some level of dynamic stress that will adversely affect tracking loop regardless of the loop filter order because there are always higher order components of dynamic stress when the host vehicle is subjected to dynamics. Nothing can be done about this for an unaided tracking loop except to align the least sensitive S_g axis of the reference oscillator along the direction of the anticipated maximum dynamic stress, but this is often impractical. For an externally aided tracking loop where the LOS dynamic stress can be measured and S_g is known, it is prudent to model this acceleration stress sensitivity and apply the correction to the aiding. Note that, like all oscillator-induced errors, the error is common mode to all receiver tracking channels, so one correction applies to all aided channels.

5.6.1.6 Total PLL Tracking Loop Measurement Errors and Thresholds

Figure 5.24 illustrates the total PLL jitter as a function of $(C/N_0)_{dB}$ for a third-order PLL, including all effects described in (5.5), (5.6), (5.9), (5.13), and (5.15). Equation (5.5) can be rearranged to solve for the dynamic stress error, and this can be solved

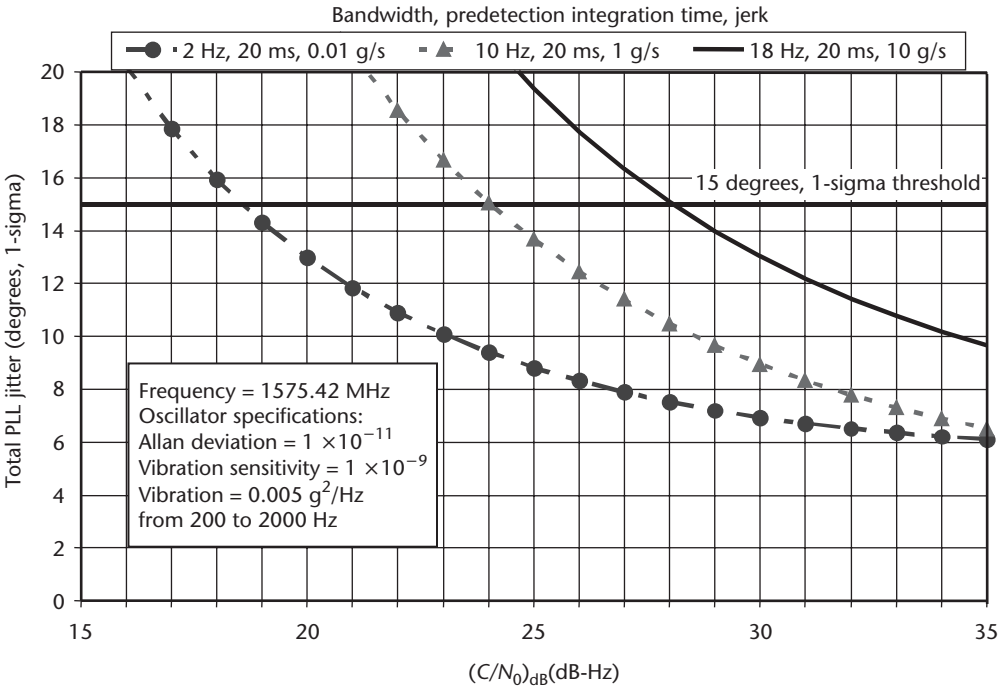


Figure 5.24 Total PLL jitter for third-order carrier loop.

for the dynamic stress at threshold. Figure 5.25 illustrates the dynamic stress at threshold as a function of noise bandwidth for a third-order PLL.

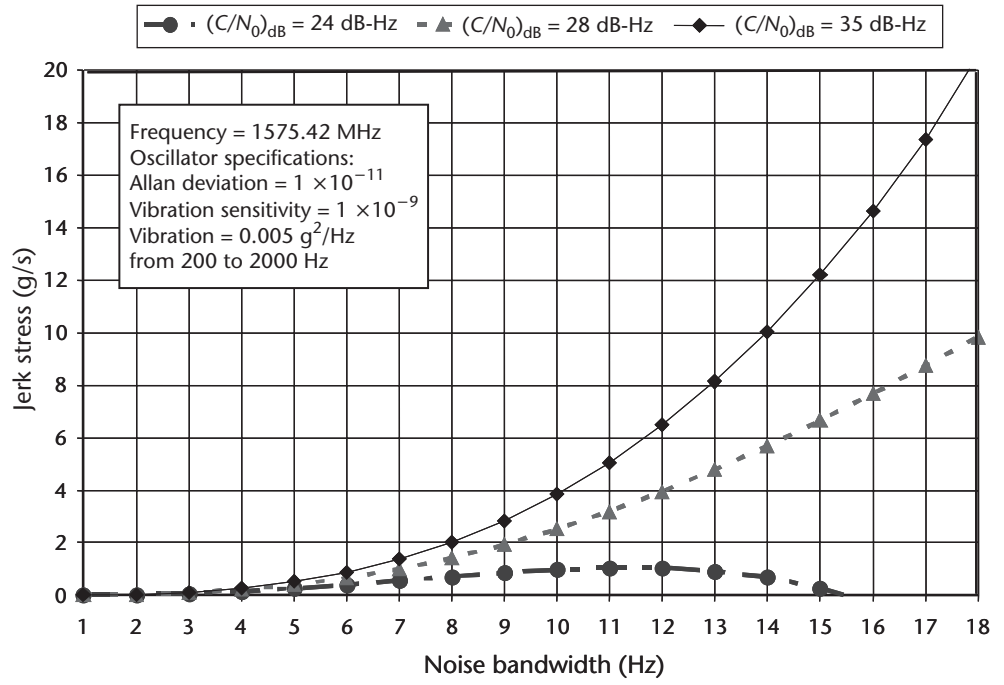


Figure 5.25 Jerk stress thresholds for third-order PLL.

5.6.2 FLL Tracking Loop Measurement Errors

The dominant sources of frequency error in a GPS receiver FLL are frequency jitter due to thermal noise and dynamic stress error. The rule of thumb for tracking threshold is that the 3-sigma jitter must not exceed one-fourth of the frequency pull-in range of the FLL discriminator. As observed in Figure 5.11, the FLL discriminator pull-in range is $1/T$ Hz. Therefore, the general rule of thumb for FLL tracking threshold is:

$$3\sigma_{FLL} = 3\sigma_{tFLL} + f_e \le 1/4T \quad \text{(Hz)} \tag{5.17}$$

where:

$3\sigma_{tFLL}$ = 3-sigma thermal noise frequency jitter

f_e = dynamic stress error in the FLL tracking loop

Equation (5.17) shows that the dynamic stress frequency error is a 3-sigma effect and is additive to the thermal noise frequency jitter. The reference oscillator vibration and Allan deviation–induced frequency jitter are small-order effects on the FLL and are considered negligible. The 1-sigma frequency jitter threshold would be $1/(12T) = 0.0833/T$ Hz.

The FLL tracking loop jitter due to thermal noise is:

$$\sigma_{tFLL} = \frac{1}{2\pi T}\sqrt{\frac{4FB_n}{C/N_0}\left[1+\frac{1}{TC/N_0}\right]} \quad \text{(Hz)} \tag{5.18}$$

$$\sigma_{tFLL} = \frac{\lambda_L}{2\pi T}\sqrt{\frac{4FB_n}{C/N_0}\left[1+\frac{1}{TC/N_0}\right]} \quad \text{(m/s)} \tag{5.19}$$

where:

$F = 1$ at high C/N_0

$= 2$ near threshold

Note that (5.17) is independent of modulation design and loop order. It is independent of L-band carrier frequency if the error units are expressed in hertz.

Because the FLL tracking loop involves one more integrator than the PLL tracking loop of the same order, the dynamic stress error is:

$$f_e = \frac{d}{dt}\left(\frac{1}{360\omega_0^n}\frac{d^nR}{dt^n}\right) = \frac{1}{360\omega_0^n}\frac{d^{n+1}R}{dt^{n+1}} \quad \text{(Hz)} \tag{5.20}$$

As an example of how the dynamic stress error is computed from (5.20), assume a second-order FLL with a noise bandwidth of 2 Hz and a predetection integration time of 5 ms. From Table 5.6, for a second-order loop $B_n = 0.53\omega_0$, so $\omega_0^2 = (2/0.53)^2$

= 14.24 Hz^2. If the maximum LOS jerk dynamics is 10 *g*/s = 98 m/s^3, then this translates into $d^3R/dt^3 = 98 \times 360 \times 1{,}575.42 \times 10^6/c = 185{,}398°/s^3$ for L1. Substituting these numbers into (5.20) results in a maximum dynamic stress error of f_e = 185,398/(14.24 × 360) = 36 Hz. Since the 3-sigma threshold is 1/(4 × 0.005) = 50 Hz, the FLL noise bandwidth is acceptable. Figure 5.26 illustrates the FLL thermal noise tracking jitter and tracking thresholds, assuming a second-order loop under 10 *g*/s jerk dynamics with typical noise bandwidths and predetection integration times.

Figure 5.27 illustrates the jerk stress thresholds for a second-order FLL as a function of noise bandwidth with $(C/N_0)_{dB}$ as a running parameter. Comparing the thresholds in Figure 5.27 for the second-order FLL with those of Figure 5.25 for a third-order PLL, notice that the FLL has much better dynamic stress performance than the PLL at the same noise bandwidths and C/N_0. For example, at 10 Hz and 35 dB-Hz, the FLL can tolerate up to 240 *g*/s while the PLL can only tolerate up to 4 *g*/s. The spread is much smaller for weaker C/N_0. The PLL would have performed moderately better under dynamic stress if the predetection integration time had been reduced from 20 ms to 5 ms, as was the case for the FLL. This comparison reinforces the earlier statements that a robust GPS receiver design will use an FLL as a backup to the PLL during initial loop closure and during high dynamic stress with loss of phase lock but will revert to pure PLL for the steady state low to moderate dynamics in order to produce the highest accuracy carrier Doppler phase measurements. Also note that the maximum predetection integration time for FLL with C/A code or P(Y) code is 10 ms, since two samples within one data transition interval are required for the FLL discriminator.

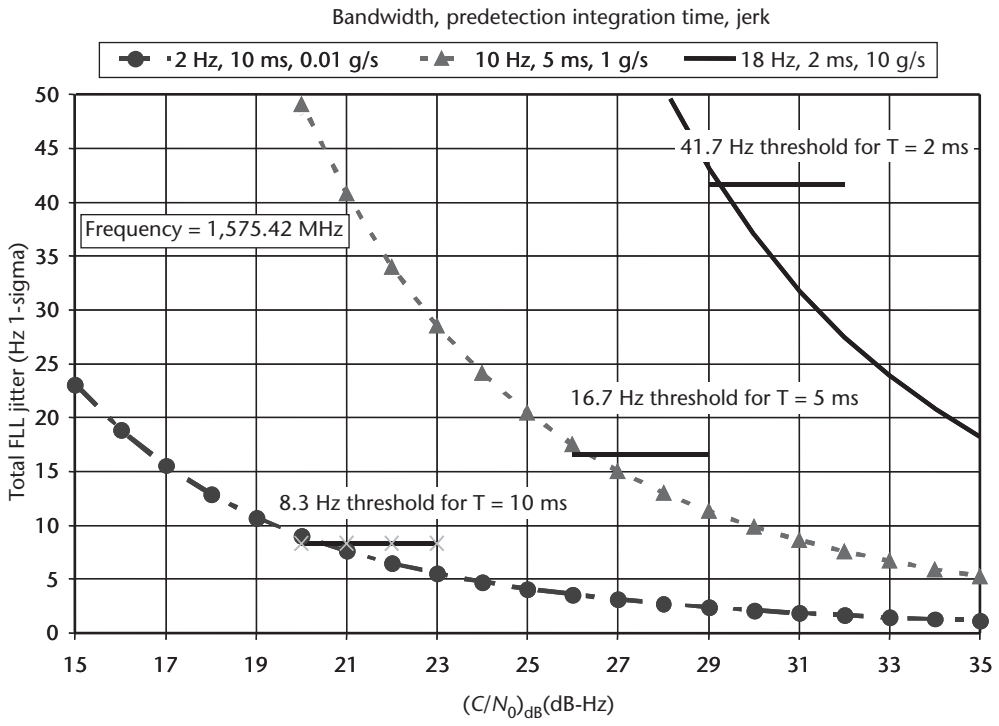


Figure 5.26 Total FLL jitter for second-order carrier loop.

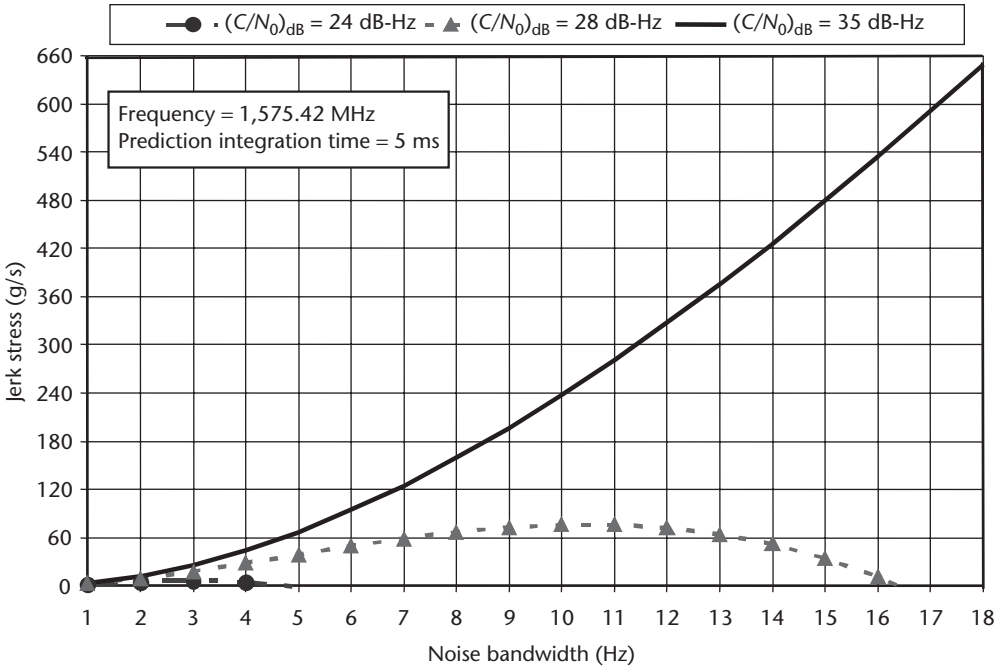


Figure 5.27 Jerk stress thresholds for second-order FLL.

5.6.3 C/A and P(Y) Code Tracking Loop Measurement Errors

When there is no multipath or other distortion of the received signal and no interference, the dominant sources of range error in a GPS receiver code tracking loop (DLL) are thermal noise range error jitter and dynamic stress error. The rule-of-thumb tracking threshold for the DLL is that the 3-sigma value of the jitter due to all sources of loop stress must not exceed half of the linear pull-in range of the discriminator. Therefore, the rule-of-thumb tracking threshold is:

$$3\sigma_{DLL} = 3\sigma_{tDLL} + R_e \leq D/2 \tag{5.21}$$

where:

σ_{tDLL} = 1-sigma thermal noise code tracking jitter (chips)

R_e = dynamic stress error in the DLL tracking loop (chips)

D = early-to-late correlator spacing (chips)

A general expression for thermal noise code tracking jitter for a noncoherent DLL discriminator is [11]:

$$\sigma_{tDLL} \cong \frac{1}{T_c}\sqrt{\frac{B_n \int_{-B_{fe}/2}^{B_{fe}/2} S_s(f)\sin^2(\pi f D T_c)\,df}{(2\pi)^2\, C/N_0 \left(\int_{-B_{fe}/2}^{B_{fe}/2} f S_s(f)\sin(\pi f D T_c)\,df\right)^2}} \times \sqrt{\left[1+\frac{\int_{-B_{fe}/2}^{B_{fe}/2} S_s(f)\cos^2(\pi f D T_c)\,df}{TC/N_0\left(\int_{-B_{fe}/2}^{B_{fe}/2} S_s(f)\cos(\pi f D T_c)\,df\right)^2}\right]} \quad \text{(chips)} \tag{5.22}$$

where:

B_n = code loop noise bandwidth (Hz)

$S_S(f)$ = power spectral density of the signal, normalized to unit area over infinite bandwidth

B_{fe} = double-sided front-end bandwidth (Hz)

T_c = chip period (seconds) = $1/R_c$ where R_c is the chipping rate

For BPSK-R(n) modulations (see Section 4.2.3) such as P(Y) code (n = 10) and C/A code (n = 1), and when using a noncoherent early-late power DLL discriminator, the thermal noise code tracking jitter can be found by substituting (4.14) into (5.22). The result can be approximated by [12]:

$$\sigma_{tDLL} \cong \begin{cases} \sqrt{\dfrac{B_n}{2C/N_0} D\left[1+\dfrac{2}{TC/N_0(2-D)}\right]}, & D \geq \dfrac{\pi R_c}{B_{fe}} \\ \sqrt{\dfrac{B_n}{2C/N_0}\left(\dfrac{1}{B_{fe}T_c}+\dfrac{B_{fe}T_c}{\pi-1}\left(D-\dfrac{1}{B_{fe}T_c}\right)^2\right)} \times\left[1+\dfrac{2}{TC/N_0(2-D)}\right], & \dfrac{R_c}{B_{fe}} < D < \dfrac{\pi R_c}{B_{fe}} \\ \sqrt{\dfrac{B_n}{2C/N_0}\left(\dfrac{1}{B_{fe}T_c}\right)\left[1+\dfrac{1}{TC/N_0}\right]}, & D \leq \dfrac{R_c}{B_{fe}} \end{cases} \quad \text{(chips)} \tag{5.23}$$

The part of the right-hand side of (5.22) and (5.23) in brackets involving the predetection integration time, T, is called the squaring loss. Hence, increasing T reduces the squaring loss in noncoherent DLLs. When using a coherent DLL discriminator, the bracketed term on the right is equal to unity (no squaring loss) [12]. As seen in (5.23), the DLL jitter is directly proportional to the square root of the filter noise bandwidth (lower B_n results in a lower jitter, which, in turn, results in a lower C/N_0 threshold). Also, increasing the predetection integration time, T,

results in a lower C/N_0 threshold, but with less effect than reducing B_n. Reducing the correlator spacing, D, also reduces the DLL jitter at the expense of increased code tracking sensitivity to dynamics. Narrowing D should be accompanied by increasing the front-end bandwidth B_{fe} to avoid "flattening" of the DLL correlation peak in the region where the narrow correlators are being operated. In fact, (5.23) shows that there is no benefit to reducing D to less than the spreading code rate divided by the front-end bandwidth.

As D becomes vanishingly small, the trigonometric functions in (5.22) can be replaced by their first-order Taylor Series expansions about zero, and this equation becomes

$$\sigma_{tDLL} \cong \frac{1}{T_c}\sqrt{\frac{B_n}{(2\pi)^2 (C/N_0)\int_{-B_{fe}/2}^{B_{fe}/2} f^2 S_s(f)df}\left[1+\frac{1}{T(C/N_0)\int_{-B_{fe}/2}^{B_{fe}/2} S_s(f)df}\right]} \quad \text{(chips)} \quad (5.24)$$

The term $\sqrt{\int_{-B_{fe}/2}^{B_{fe}/2} f^2 S_S(f)df}$ is called the root-mean-squared (RMS) bandwidth of the signal, and it is a measure of the "sharpness" of the correlation peak. Clearly, signals with larger RMS bandwidths offer the potential for more accurate code tracking. In fact, the frequency-squared term in the RMS bandwidth indicates that even very small amounts of high frequency content in the signal can enable more accurate code tracking. Intuitively, these high frequency components produce sharper edges and more distinct zero crossings in the waveform, enabling more accurate code tracking.

The use of carrier-aided code (practically a universal design practice) effectively removes the code dynamics, so the use of narrow correlators is an excellent design tradeoff for C/A code receivers. For C/A code receivers where the correlation interval in units of time is an order of magnitude greater than for P(Y) code, reducing the correlator spacing reduces the effects of thermal noise and multipath (see Section 6.3) on C/A, but this also requires increasing the front-end bandwidth, which increases the vulnerability to in-band RF interference.

Note that the thermal noise is independent of tracking loop order in (5.23) Also note that the thermal noise is the same for either C/A code or P(Y) code when expressed in units of chips. However, all other things being equal, the thermal noise is ten times larger for the C/A code than the P(Y) code because the chip wavelength of the C/A code is ten times longer than for P(Y) code. For example, from (5.23), this is readily observed if the measurement is converted to meters. To convert to meters, multiply (5.23) by $c \cdot T_c$ (e.g., by 293.05 m/chip for C/A code or by 29.305 m/chip for P(Y) code).

Figure 5.28(a) uses (5.23) to compare C/A code and P(Y) code accuracy in units of meters. (Figure 5.28 also includes results for the M code that will be explained in Section 5.6.4.) A one-chip correlator E-L spacing (e.g., D = 1) and normalized receiver bandwidth $b \equiv B_{fe}/R_c = 2$ (i.e., the front-end receiver bandwidth is equal to twice the chip rate) is used for each BPSK-R result. Figure 5.28(b) uses (5.23) to

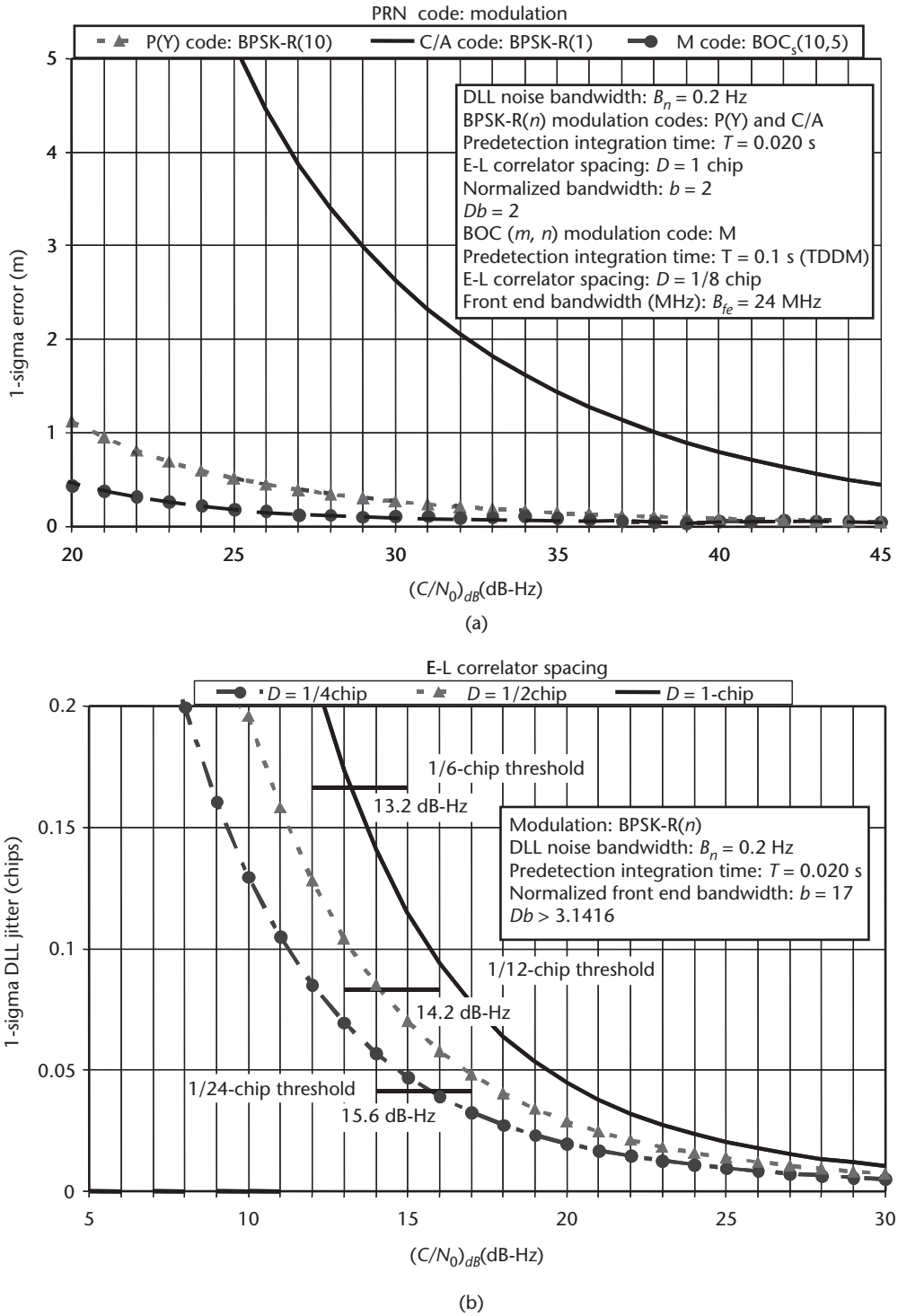
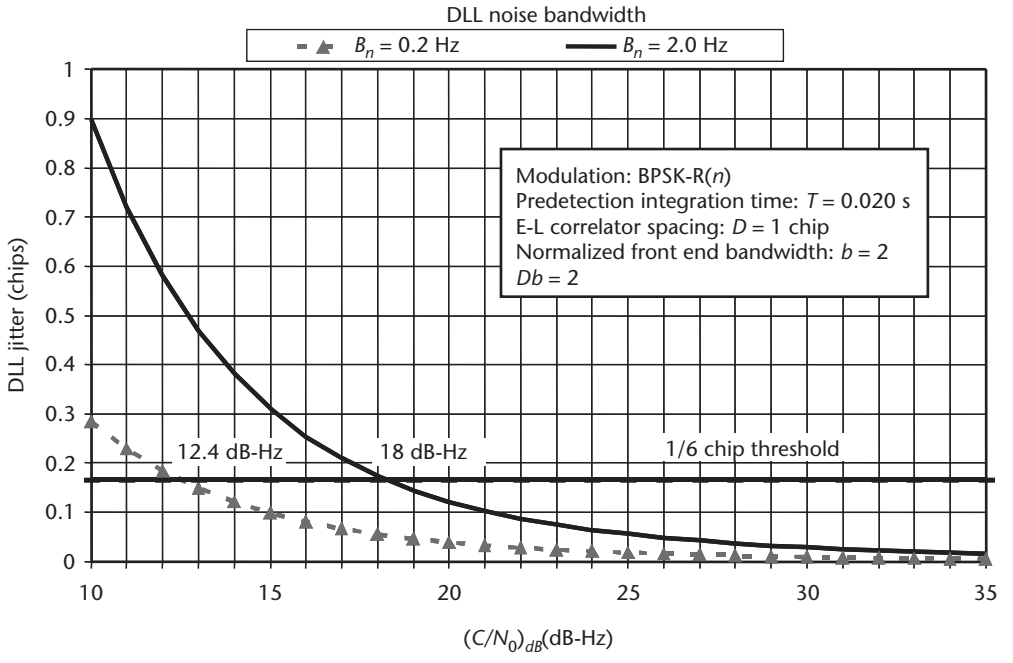
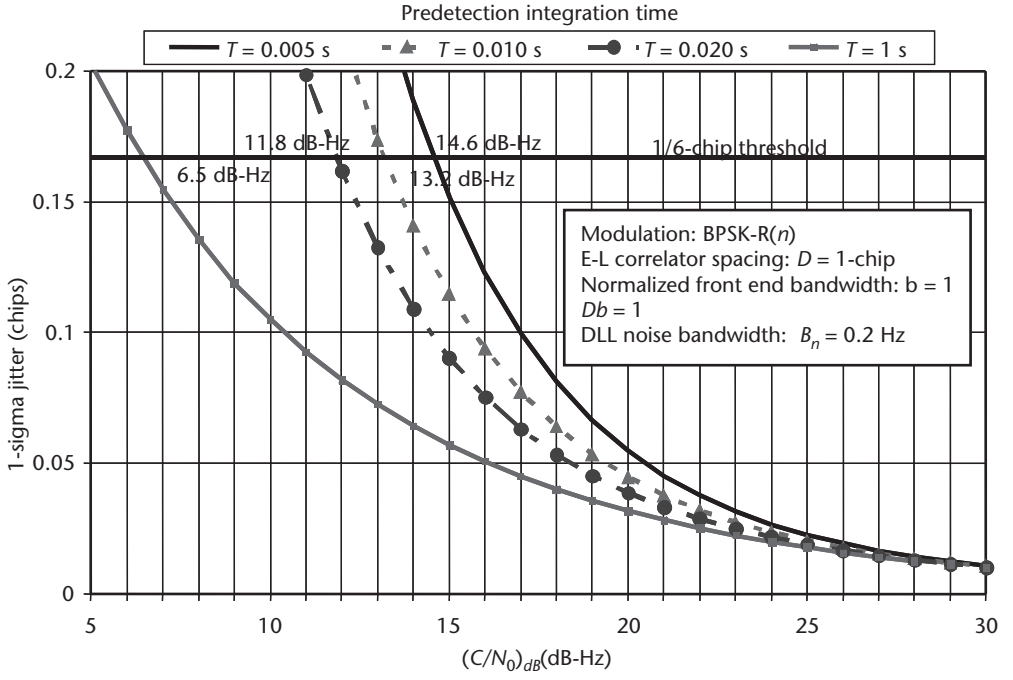


Figure 5.28 Delay lock loop jitter versus $(C/N_0)_{dB}$: (a) comparison of P(Y) code, C/A code, and M code DLL accuracy, (b) comparison of DLL jitter for different correlator spacing, (c) effect of noise bandwidth on DLL jitter, and (d) effect of predetection integration time on DLL jitter.



(c)



(d)

Figure 5.28 (Continued.)

compare the DLL performance for BPSK-R modulations with different correlator spacings and an extremely wide receiver bandwidth ($b = 17$). Note that both the thermal noise jitter and the rule-of-thumb thresholds are reduced when the

correlator spacing is reduced, but that there is only a slight loss of tracking threshold as the correlator spacing is reduced (provided that the receiver front end bandwidth is increased appropriately).

Figure 5.28(c) uses (5.23) with $Db = 2$ to demonstrate the improved DLL tracking threshold by reducing the noise bandwidth. Figure 5.28(d) uses (5.23) with $Db = 2$ to illustrate the improved DLL tracking threshold by increasing the predetection integration time. The 1-second predetection integration time provides the lowest code tracking threshold. To support a predetection integration time greater than 20 ms (the navigation message data bit interval), the data wipeoff process must be implemented for C/A code and the normal mode of P(Y) code. Recall that this technique uses the GPS receiver's a priori knowledge of the navigation message data bit stream to remove the 180° data transitions. This data wipeoff technique permits longer than 20-ms predetection integration times and, if properly implemented, can achieve nearly 6 dB of additional $(C/N_0)_{dB}$ threshold improvement [see Figure 5.28(d) at threshold crossings]. This is a short-term "desperation" DLL weak signal hold-on strategy for an externally aided GPS receiver when the carrier is aided open loop. Data wipeoff also improves the PLL tracking threshold when the carrier loop is closed-loop aided, but not to the extent that the code loop tracking threshold is improved. Changes in any part of the SV navigation message data stream by a GPS control segment upload or autonomously by the SV will cause errors in the data wipeoff, which, in turn, will cause deterioration in the tracking threshold.

The DLL tracking loop dynamic stress error is determined by:

$$R_e = \frac{d^n R/dt^n}{\omega_0^n} \quad \text{(chips)} \tag{5.25}$$

where d^nR/dt^n is expressed in chips/s^n.

As an example of how the dynamic stress error is computed from (5.25), assume that the code loop is an unaided third-order C/A code DLL with $B_n = 2$ Hz and $D = 1$ chip. From Table 5.6, for a third-order loop $\omega_0 = B_n/0.7845$. If the maximum LOS jerk stress is 10 g/s, then this is equivalent to $d^3R/dt^3 = 98$ m/s^3/293.05 m/chip = 0.3344 chips/s^3. Substituting these numbers into (5.25) results in a maximum dynamic stress error of $R_e = 0.02$ chip, a 3-sigma effect. Since the 3-sigma threshold is 1/2-chip, this would indicate that the DLL noise bandwidth is more than adequate for C/A code. If the receiver was P(Y) code, then $R_e = 0.2$ chip, which is still adequate. Note that carrier-aided code techniques removes virtually all the dynamic stress from the code tracking loop. Therefore, as long as the carrier loop remains stable, the code loop experiences negligible dynamic stress, and this effect is not included in the code loop tracking threshold analysis.

5.6.4 Modernized GPS M Code Tracking Loop Measurement Errors

The modernized GPS M code uses a BOC$_s$(10,5) modulation technique (see Sections 4.2.3 and 4.5.3). By substituting (4.17) into (5.22), the following approximation for M code DLL jitter in the presence of thermal noise can be determined [13]:

$$\sigma_{tM} \cong \begin{cases} \dfrac{1}{T_c}\sqrt{\dfrac{\dfrac{B_n}{(2\pi)^2\dfrac{C}{N_0}\left(0.66B_{fe}-7.7E6\right)^2}}{\left[1+\dfrac{1}{\left(B_{fe}\cdot 7.3E-8-0.96\right)T\dfrac{C}{N_0}}\right]}}, & 16\text{ MHz} \le B_{fe} \le 24.5\text{ MHz} \\ \dfrac{1}{T_c}\sqrt{\dfrac{\dfrac{B_n}{(2\pi)^2\dfrac{C}{N_0}\left(0.007B_{fe}+8.4E6\right)^2}}{\left[1+\dfrac{1}{0.837T\dfrac{C}{N_0}}\right]}}, & 24.5\text{ MHz} < B_{fe} \le 30\text{ MHz} \end{cases} \quad \text{(chips)} \quad (5.26)$$

where $\dfrac{1}{T_c} = R_c = 5.115$ Mchips/s.

To obtain the 1-sigma jitter in meters, multiply (5.26) by 58.610 m/chip. Figure 5.28(a) uses (5.26) in units of meters to compare the modernized M code accuracy with the P(Y) and C/A codes. Note that the correlator spacing, D (in M chips), does not appear in (5.26), but this approximation is restricted to an E–L spacing of 1/4 M code chips or less. The rule of thumb for M code DLL tracking threshold is identical to (5.21). M code has a provision for dataless tracking that permits extended predetection integration times, but this mode also loses 3 dB because every other code bit is dataless. Reduce $(C/N_0)_{dB}$ by 3 dB before converting to C/N_0 both places in (5.26) to account for this loss in signal (increase in jitter) when using the M code time division data multiplexing (TDDM) mode.

5.7 Formation of Pseudorange, Delta Pseudorange, and Integrated Doppler

Contrary to popular belief, the natural measurements of a GPS receiver are not pseudorange or delta pseudorange [14]. This section describes the natural measurements of a GPS receiver and describes how they may be converted into pseudorange, delta pseudorange, and integrated carrier Doppler phase measurements. The natural measurements are replica code phase and replica carrier Doppler phase (if the GPS receiver is in phase lock with the satellite carrier signal) or replica carrier Doppler frequency (if the receiver is in frequency lock with the satellite carrier signal). The replica code phase can be converted into satellite transmit time, which can be used to compute the pseudorange measurement. The replica carrier Doppler phase or frequency can be converted into delta pseudorange. The replica carrier Doppler phase measurements can also be converted into integrated carrier Doppler phase measurements used for ultraprecise (differential) static and kinematic surveying or positioning.

The most important concept presented in this section is the measurement relationship between the replica code phase state in the GPS receiver and the satellite transmit time. This relationship is unambiguous for P(Y) code, but can be ambiguous for C/A code. Every C/A code GPS receiver is vulnerable to this ambiguity problem and, under weak signal acquisition conditions, the ambiguity will occur. When the ambiguity does occur in a C/A code GPS receiver, it causes serious range measurement errors, which, in turn, result in severe navigation position errors.

5.7.1 Pseudorange

The definition of pseudorange to SV_i, where i is the PRN number is as follows:

$$\rho_i(n) = c\left[T_R(n) - T_{Ti}(n)\right] \quad (\text{m}) \tag{5.27}$$

where:

c = speed of light = 299,792,458 (m/s)

$T_R(n)$ = receive time corresponding to epoch n of the GPS receiver's clock (seconds)

$T_{Ti}(n)$ = transmit time based on the SV_i clock (seconds)

Figure 5.29 depicts the GPS satellite SV_i transmitting its PRN code PRN_i starting at the end of the GPS week. Corresponding to each chip of the PRN_i code is a linear SV_i clock time. When this signal reaches the GPS receiver, the transmit time, $T_{Ti}(n)$, is the SV_i time corresponding to the PRN code state that is being replicated at receiver epoch n. The pseudorange derived from this measurement corresponds to a

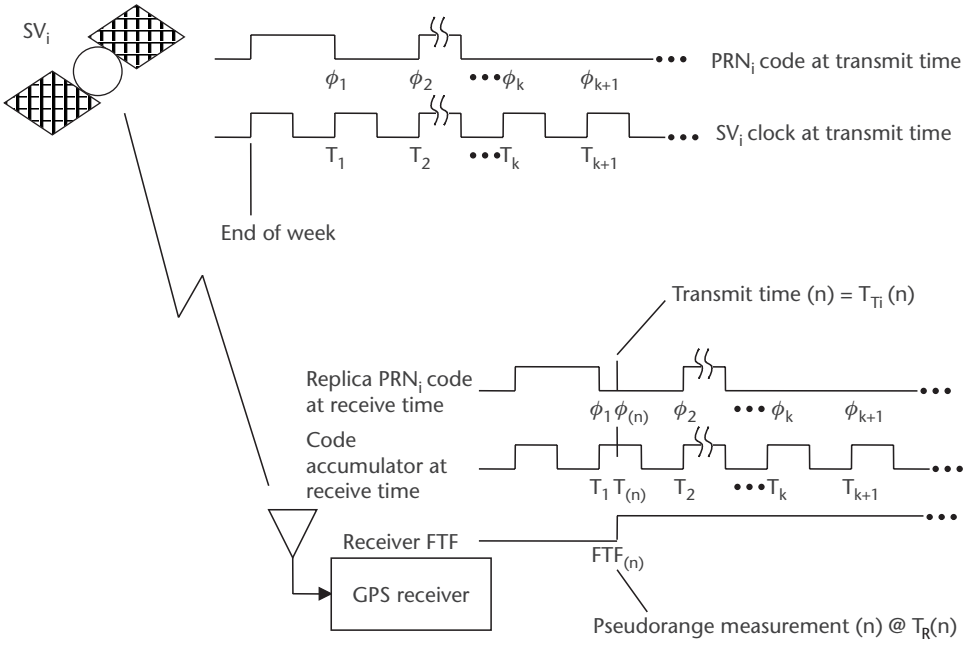


Figure 5.29 Relationship of satellite transmit time to pseudorange measurements.

particular receive time epoch (epoch n) in the GPS receiver. Every epoch in the PRN code that is transmitted by SV_i is precisely aligned to the GPS time of week as maintained inside SV_i's time-keeping hardware. When this transmitted PRN code reaches the user GPS receiver, which is successfully correlating a replica PRN code with it, the phase offset of the replica code with respect to the beginning of the GPS week represents the transmit time of SV_i. The first thing that the navigation process needs for position measurement incorporation is the SV_i transmit time corrected to true GPS time. It is therefore logical that the natural measurement (SV_i transmit time) should be sent to the navigation process (along with the receive time) and not the pseudorange measurement (along with the receive time). This is because the SV_i transmit time is lost after this artificial computation is performed. This forces the navigation process into a wasteful iterative process of computing the SV_i transmit time. Highly sophisticated GPS receivers implement vector tracking of the SVs instead of scalar tracking described herein. This overcomes this problem because either the raw I and Q measurements or the discriminator outputs are sent to the navigation process as measurements for Kalman filtering by the navigation process. Thus, the navigation process dynamically changes the noise bandwidth of the tracking loops in an optimal manner.

Typically, the GPS receiver will take a set of measurements at the same receive time epoch. This is why the receive time is not identified with any particular SV PRN number in (5.27). When the GPS receiver schedules a set of measurements, it does this based on its own internal clock, which contains a bias error with respect to true GPS time. Eventually the navigation process learns this bias error as a byproduct of the GPS navigation solution. The SV transmit time also contains a bias error with respect to true GPS time, although the control segment ensures that this is maintained at less than 1 ms. This correction is transmitted to the receiver by SV_i as clock correction parameters via the navigation message. However, neither of these corrections is included in the pseudorange measurement of (5.27). These corrections and others are determined and applied by the navigation process.

5.7.1.1 Pseudorange Measurement

From (5.27), it can be concluded that if the receiver baseband process can extract the SV transmit time from the code tracking loop, then it can provide a pseudorange measurement. The precise transmit time measurement for SV_i is equivalent to its code phase offset with respect to the beginning of the GPS week. There is a one-to-one relationship between the SV_i replica code phase and the GPS time. Thus, for every fractional and integer chip advancement in the code phase of the PRN code generator since the initial (reset) state at the beginning of the week, there is a corresponding fractional and integer chip advancement in the GPS time. The fractional and integer chip codephase will hereafter be called the code state. The receiver baseband process time keeper, which contains the GPS time corresponding to this code state, will hereafter be called the code accumulator.

The replica code state corresponds to the receiver's best estimate of the SV transmit time. The receiver baseband process knows the code state because it sets the initial states during the search process and keeps track of the changes in the code state thereafter. The receiver baseband code tracking loop process keeps track of the GPS

transmit time corresponding to the phase state of the code NCO and the replica PRN code generator state after each code NCO update. It does this by discrete integration of every code phase increment over the interval of time since the last NCO update and adds this number to the code accumulator. The combination of the replica code generator state (integer code state) and the code NCO state (fraction code state) is the replica code state. Since the code phase states of the PRN code generator are pseudorandom, it would be impractical to read the code phase state of the PRN code generator and then attempt to convert this nonlinear code state into a linear GPS time state, say, by a table lookup.

There are too many possible code states, especially for the P code generator. A very practical way to maintain the GPS time in a GPS receiver is to use a separate code accumulator in the GPS receiver baseband process and to synchronize this accumulator to the replica PRN code generator phase state. Figure 5.30 (derived from the code tracking loop of Figure 5.13) illustrates the high-level block diagram relationship between the replica PRN code generator and the code accumulator (which is not included in Figure 5.13) in the code tracking loop of one GPS receiver channel.

A typical GPS navigation measurement incorporation rate is once per second. A typical GPS receiver FTF for scheduling measurements is 20 ms, which is the same as the 50-Hz navigation message data period. The receiver baseband process schedule for updating the code and carrier NCOs is usually some integer subset of the

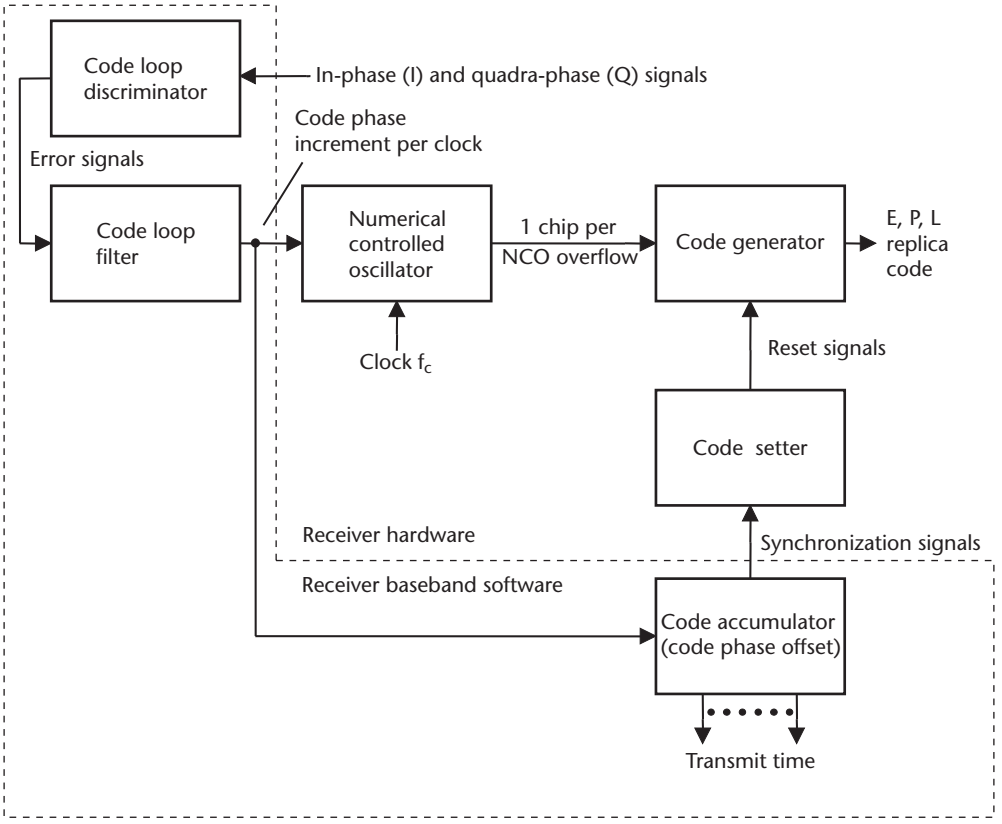


Figure 5.30 Relationship between PRN code generator and code accumulator.

FTF, such as 20, 10, 5, 2, or 1 ms. Assuming that the FTF is 20 ms, the receiver measurement process maintains a monotone counter (call it the FTF counter) in 20-ms increments derived from the receiver's reference oscillator. The FTF counter is set to zero at power up, counts up, rolls over, counts up, and so on. Assuming that the navigation measurement incorporation rate is 1 Hz, the navigation process will schedule measurements to be extracted from the code and carrier tracking loops every fiftieth FTF (i.e., based on the receiver's time epochs). When the receiver baseband process extracts the measurements from the code and carrier tracking loops, it time tags the measurements with the FTF count. The navigation process assigns and maintains a GPS receive time corresponding to the FTF count. The receive time initialization can be the first SV's transmit time plus a nominal propagation time of, say, 76 ms if the navigation process does not know the GPS time accurately. This will set the initial receive time accurate to within 20 ms.

When a pseudorange measurement is scheduled on FTF(n), the receiver baseband code tracking loop process extracts the SV_i transmit time from its code accumulator and propagates this time forward to FTF(n). The result is the SV_i transmit time with a measurement resolution of 2^{-N} of a code chip, where N is the number of bits in the code NCO adder. If the code NCO adder uses a 32-bit register, this measurement resolution is less than a quarter of a nanochip, which makes the code measurement quantization noise negligible. The receiver baseband process can compute the pseudorange measurement from the SV_i transmit time using (5.27) and time tag the measurement with FTF(n) before sending the result to the navigation process. However, the navigation process needs the (corrected) SV_i transmit time to compute the location of SV_i when it transmitted the measurement. Hence, the best measurement to send to the GPS navigation process is the (uncorrected) SV transmit time, along with the FTF time tag. The navigation process applies the clock correction (including relativity correction), uses the corrected SV_i transmit time to compute the SV_i position, then computes the pseudorange plus other corrections before incorporation of the measurement. (Satellite clock and relativistic corrections are discussed in Sections 7.2.1 and 7.2.3, respectively.)

5.7.1.2 Measurement Time Skew

Figure 5.4 illustrates the bit sync phase skew, T_s, which exists between the SV data transition boundaries and the receiver 20-ms clock epochs (i.e., the FTFs). The control segment ensures that every SV transmits every epoch within 1 ms of true GPS time (i.e., the SV clocks are aligned to within 1 ms of true GPS time). Therefore, all of the SV data transition boundaries are approximately aligned to true GPS time at transmit time. However, at the GPS receiver the SV data transition boundaries are, in general, skewed with respect to each other and with respect to the receiver's FTF boundary. This is because the SVs are at different ranges with respect to the user GPS receiver antenna phase center. The user GPS receiver must adjust the phases of its integrate and dump boundaries in order to avoid integrating across the SV data bit transition boundaries. The time skew, T_s, is different for each SV being tracked, and it changes with time because the range to the SVs change with time. Therefore, the epochs from each replica code generator, such as the C/A code 1-ms epochs, are skewed with respect to each other and to the FTF. As a result, the integrate and

dump times and the updates to the code and carrier NCOs are performed on a changing skewed time phase with respect to the FTF time phase, but the receiver baseband process learns and controls this time skew in discrete phase increments. The code accumulator is normally updated on the skewed time schedule that matches the code NCO update schedule. Therefore, if all of the GPS receiver measurements of a multiple channel GPS receiver are to be made on the same FTF, the contents of the code accumulator, when extracted for purposes of obtaining a measurement, must be propagated forward by the amount of the time skew between the code NCO update events and the FTF.

5.7.1.3 Maintaining the Code Accumulator

Although there are many code accumulator time-keeping conventions that would work, the following convention is convenient for setting the initial code generator and NCO phase states [1]. Three counters, Z, X1, and P, are maintained as the code accumulator. The Z counter (19 bits) accumulates in GPS time increments of 1.5 seconds, then is reset one count short of the maximum Z count of 1 week = 403,200. Hence, the maximum Z count is 403,199. The X1 counter (24 bits) accumulates in GPS time increments of integer P chips, then is reset one count short of the maximum X1 count of 1.5 seconds = 15,345,000. Hence, the maximum X1 count is 15,344,999. The P counter accumulates in GPS time increments of fractions of a P chip, then rolls over one count short of one P chip. The P counter is the same length as the code NCO adder. A typical length is 32 bits.

Note in Figure 5.13 that the code NCO synthesizes a code clock rate that is an integer multiple, $2/D$, faster than the code generator spreading code chip rate, where D is the E-L code correlator spacing in chips (often $D = 1$). This is required in order to generate phase shifted replica codes, which are necessary for error detection in the code discriminator. The P counter tracks the fractional part of the code phase state, which is the code NCO state. Using this convention and assuming that the code NCO and code accumulator are updated every T seconds, the algorithm for maintaining the code accumulator is as follows. Note that the equals sign means "is replaced with."

$$
\begin{aligned}
&P_{temp} = P + f_c \Delta\phi_{co} T \\
&P = \text{fractional part of } P_{temp} \quad \text{(chips)} \\
&X_{temp} = (X1 + \text{whole part of } P_{temp})/15{,}345{,}000 \\
&X1 = \text{remainder of } X_{temp} \quad \text{(chips)} \\
&Z = \text{remainder of } [(Z + \text{whole part of } X_{temp})/403{,}200] \quad \text{(1.5 seconds)}
\end{aligned} \tag{5.28}
$$

where:

P_{temp} = temporary P register

f_C = code NCO clock frequency (Hz)

$\Delta\phi_{CO}$ = code NCO phase increment per clock cycle

= code NCO bias + code loop filter velocity correction

T = time between code NCO updates (seconds)

The earlier definition of $\Delta\phi_{CO}$ contains two components, the code NCO bias and the code loop filter velocity correction. The code NCO bias (see Figure 5.13) is the phase increment per clock that accounts for the marching of time in the P code replica code generator. When applied to the P replica code generator, this is 10.23 Mchip/s. When applied to the C/A replica code generator, this is 1.023 Mchip/s. The code *E-L* correlator spacing, *D*, requires a second output from the NCO to be 2/*D* faster than the spreading code chip rate to clock the 2-bit shift register. Algorithm (5.29) assumes that the NCO produces the code generator clock. If the code generator clock is produced by dividing the NCO shift register clock by 2/*D*, then algorithm (5.29) and the NCO bias must be scaled accordingly. The code loop filter velocity correction is the combination of carrier aiding and code loop filter output. This combined output corrects the P replica code generator for Doppler (and a small order effect due to changes in ionospheric delay) referenced to the P code chip rate. Usually the code generator provides the divide-by-10 function required for the C/A code generator if both P and C/A codes are generated. This is the correct factor for the spreading code chip rate and the code Doppler/ionospheric delay components.

5.7.1.4 Obtaining a Measurement from the Code Accumulator

To obtain a measurement, the code accumulator must be propagated to the nearest FTF(n). This results in the set of measurements $P_i(n)$, $X1_i(n)$, and $Z_i(n)$ for SV_i. When converted to time units of seconds, the result is $T_{Ti}(n)$, the transmit time of SV_i at the receiver time epoch n. This is done very much like (5.28), except the time T is replaced with the skew time, T_s, and the code accumulator is not updated.

$$
\begin{aligned}
&P_{temp} = P + f_c \Delta\phi_{co} T_s \\
&P_i(n) = \text{fractional part of } P_{temp} \quad \text{(chips)} \\
&X_{temp} = (X1 + \text{whole part of } P_{temp})/15{,}345{,}000 \\
&X1_i(n) = \text{remainder of } X_{temp} \quad \text{(chips)} \\
&Z_i(n) = \text{remainder of } [(Z + \text{whole part of } X_{temp})/403{,}200] \quad \text{(1.5 seconds)}
\end{aligned} \tag{5.29}
$$

Note that (5.29) produces no error due to the measurement propagation process for the code accumulator measurements because the code NCO is running at a constant rate, $\Delta\phi_{CO}$ per clock, during the propagation interval. The following equation converts the code accumulator measurements to the SV_i transmit time, $T_{Ti}(n)$. Double precision floating point computations are assumed.

$$T_{Ti}(n) = \left[P_i(n) + X1_i(n)\right] \Big/ \left(10.23 \times 10^6\right) + Z_i(n) \times 1.5 \quad \text{(seconds)} \tag{5.30}$$

5.7.1.5 Synchronizing the Code Accumulator to the C/A code and P Code Generators

Synchronizing the code accumulator to the C/A code and P code generators is the most complicated part of the pseudorange measurement process. This is because the count sequences taking place in the code generator shift registers are PRN sequences, while the count sequence taking place in the code accumulator is a linear sequence. Fortunately, predictable reset timing events in the PRN shift registers per-

mit them to be synchronized to the code accumulator. The first thought might be to design the code generator shift registers such that they contain the linear counters that are synchronized by the hardware and read by the receiver baseband process. However, the phase states of the code generators must be controlled by the receiver baseband process. So a better design is to use code setters in the hardware and maintain the code accumulator in the receiver baseband processor. By far, the simplest case is the C/A code generator setup, described first.

C/A Code Setup

Figure 5.31 illustrates a high-level block diagram of the P and C/A code generators, including their code setters. (Details of code generation were discussed in Chapter 4.) Recall from Section 4.3.1.1 that the C/A code generator consists of two 10-bit linear feedback shift registers called the G1 and G2 registers [2]. The C/A code setup requires one 10-bit code setter to initialize both the G1 and G2 registers in the C/A code generator. The phase states of the G1 and G2 registers are the same for every SV for the same GPS time of week. It is the tap combination on the G2 register (or equivalently, the delay added to the G2 register) in combination with the G1 register that determines the PRN number. A typical C/A code setter is capable of setting the G1 and G2 registers to their initial states and to their midpoint states. Since it requires only 1 ms for the C/A code generator to cycle through its complete state, this code setter design example holds the maximum delay to 1/2 ms until the C/A code generator is synchronized to the code accumulator after initialization of the code setter. This code setter design counts from 0 to 511 (1/2 C/A code epoch) and then rolls over.

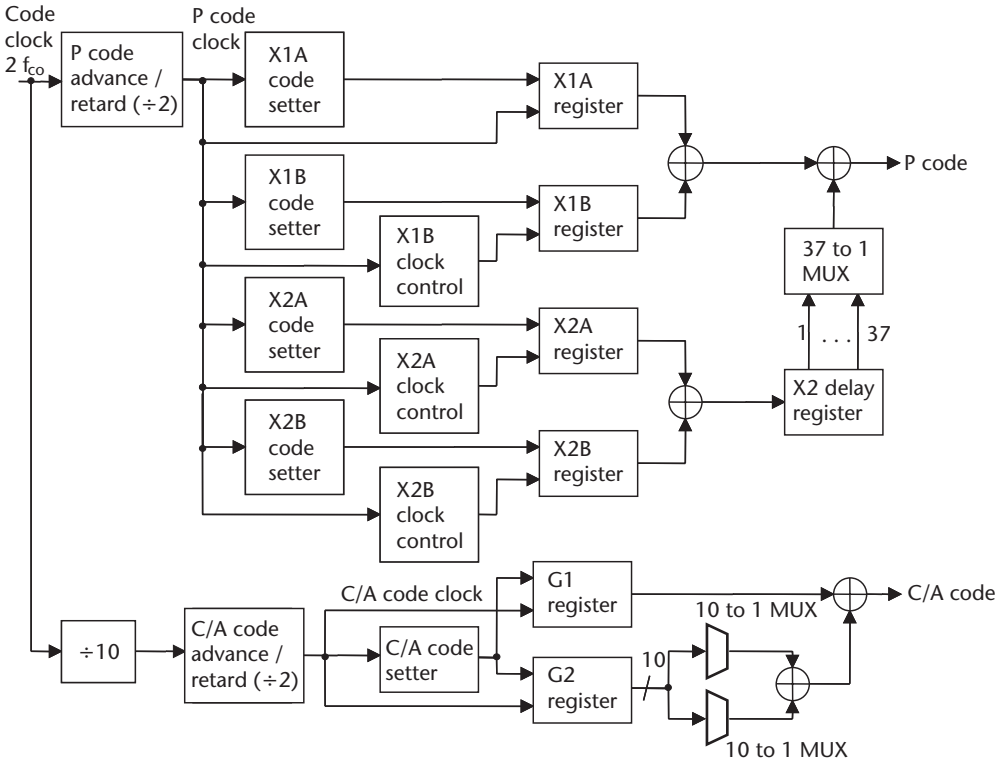


Figure 5.31 GPS code setter and code generator block diagram.

With the code setter hardware described earlier, the C/A code setup process works as follows. In accordance with a future time delay equal to a fixed number of code NCO reference clock cycles later, the code accumulator value for that future time is loaded into the code setter. This value matches the desired C/A code time after the scheduled time delay. The value for the code setter is computed just as if the 1,023 state C/A code generator had the same linear counting properties as a 1,023 bit counter. The code setter begins counting after the scheduled time delay, starting with the loaded count value. The code setter sets the G1 and G2 registers when the count rolls over. If the value sent from the code accumulator was greater than 511, the C/A code setter resets the G1 and G2 registers to their initial states. If the value sent from the code accumulator was less than or equal to 511, the code setter sets the G1 and G2 registers to their halfway points. As a result, the C/A replica code generator phase state matches the code accumulator GPS time state when the code setter rolls over and is synchronized to the code accumulator thereafter. When the receiver is tracking the SV after initialization, the code setter process can be repeated as often as desired without altering the C/A replica code generator phase state, because both the code accumulator and the code generator are ultimately synchronized by the same reference clock, the code NCO clock. If the receiver is in the search process, the C/A code advance/retard feature shown in Figure 5.31 provides the capability to add or remove clock cycles in half-chip increments. The code accumulator must keep track of these commanded changes. If the receiver can predict the satellite transmit time to within a few chips during the search process, it can use the code setter to perform a direct C/A code search. This condition is satisfied if the receiver has previously acquired four or more satellites and its navigation solution has converged. Ordinarily, all 1,023 C/A code chips are searched.

Some commercial C/A code receiver designs do not use a code NCO, but instead propagate the code generator at the nominal spreading code chip rate between code loop updates, tolerating the error build up due to code Doppler and ionospheric delay changes. Instead of the code NCO, a counter with a fractional chip advance/retard capability is used to adjust the phase of the C/A replica code generator in coarse phase increments. This results in a very low-resolution code measurement (large quantization noise) and a noisy pseudorange measurement in comparison to the code NCO technique.

The algorithm for the code accumulator output to the C/A code setter is as follows:

$$G = \text{remainder of}\left[\left\{\text{whole part of}\left[(X1/10)\right]\right\}/1{,}023\right] \tag{5.31}$$

where:

G = future scheduled C/A code time value sent to the code setter

$X1$ = future scheduled GPS time of week in P chips ($0 \le X1 \le 15{,}344{,}999$)

An alternative design to the code setter timing technique is to precompute and store all 1,023 10-bit PRN states for G1 and G2 in two tables, then use the value G from (5.31) as the index to these tables. This would result in two 10-bit words being transferred into a G1 buffer register and a G2 buffer register. At the instance of the

future code clock that corresponds to the computation of G, the buffer contents would be parallel transferred into the C/A code generator. This instantly aligns the replica code generator to the C/A code portion of the code accumulator. The ambiguity in the code accumulator must be removed by reading the handover word and placing it into the code accumulator at the correct epoch. This is described later.

P Code Setup

Figure 5.31 illustrates the high-level block diagram for the P code setup. (Details of P code generation were discussed in Chapter 4.) Recall from Chapter 4 that the P code generator contains four 12-bit linear feedback shift registers, called X1A, X1B, X2A, and X2B [2]. The PRN phase states of these four registers are the same for every SV PRN number at the same GPS time of the week. The unique PRN code is determined by the delay of the X2 output. Each of the four shift registers must have a corresponding 12-bit code setter. The P code setup is similar to the C/A code setup, but involves a much more complex code setter process, since, in general, all four shift registers must be reset at different time phases. There are two unusual code setter timing patterns that involve additional delays in the X1B, X2A, and X2B shift registers. The first occurs at the end of every X1 and X2 cycle. The second occurs at the end of the X2 cycle at the end of the GPS week.

Table 5.7 shows the count states for the first cycle of the GPS week for the four registers, starting with the 3,749th and 3,750th cycles of the X1A and X2A registers. After each cycle of X1A and X1B, the X1B epoch advances 1 chip ahead of X1A. The same pattern occurs between X2A and X2B. For the P code phases shown in Table 5.7, the X1 and X2 counts match during the first X1 cycle of the week (i.e., the X1A and X2A registers and the X1B and X2B registers are aligned to each other

Table 5.7 Count States for 3,749th and 3,750th Cycles of X1A in First X1 Cycle of Week

Z	*X1*	*X1A*	*X1B*	*D1B*	*X2*	*X2A*	*D2A*	*X2B*	*D2B*
0	15340563	3747	4092	0	15340563	3747	0	4092	0
0	15340564	3748	0	343	15340564	3748	0	0	380
.	.	.	.	.	.	.	.	.	.
0	15340907	4091	343	343	15340907	4091	0	343	380
0	15340908	0	344	343	15340908	0	37	344	380
.	.	.	.	.	.	.	.	.	.
0	15344655	3747	4091	343	15344655	3747	37	4091	380
0	15344656	3748	4092	343	15344656	3748	37	4092	380
0	15344657	3749	4092	342	15344657	3749	37	4092	379
.	.	.	.	.	.	.	.	.	.
0	15344998	4090	4092	1	15344998	4090	37	4092	38
0	15344999	4091	4092	0	15344999	4091	37	4092	37
1	0	0	0	0	15345000	4091	36	4092	36
.	.	.	.	.	.	.	.	.	.
1	35	35	35	0	15345035	4091	1	4092	1
1	36	36	36	0	15345036	4091	0	4092	0
1	37	37	37	0	0	0	0	0	0

Note: These count states are not the PRN code states contained in the shift registers.

in phase). This is the only time during the GPS week that they are aligned. As can be observed in Table 5.7, they become misaligned immediately after the first X1 cycle of the week completes. This is because following each X1 cycle, the X2 cycle continues for an additional 37 chips. These last two cycles are chosen for the most representative timing illustration because the X1 period is defined as 3,750 X1A cycles, which equals 1.5 seconds or one Z count. When the X1B code reaches its last chip in the last X1A cycle of an X1 cycle, the X1B register is held in its final state (4092) for 344 chips until the X1A register reaches its final state. Then X1A and X1B are reset. The Z count is incremented by one, and the X1 cycle starts over.

The X2 period is defined by 3,750 X2A cycles plus 37 chips. In the last X2A cycle of the X2 cycle, the X2B register is held in its final state (4092) until X2A reaches its final state, and then the X2A register is held in its final state (4091) and the X2B register continues to be held in its final state (4092) for an additional 37 chips. During this last X2A cycle, X2B is held in it final state (4092) for a total of 381 chips. Then X2A and X2B are reset and their cycles start over. Thus, the X2 epochs are delayed by 37 chips per Z count, with respect to the X1 epochs, until the end of the GPS week.

Note in Table 5.7 that the values for the X1A, X1B, X2A, and X2B registers are their count states, not their PRN code states. The PRN code states corresponding to the last two count states and the reset states are shown in Table 5.8. The only PRN code states that are important to the P code setters are the reset states, but the last two PRN code states prior to reset are useful for code generator verification purposes.

The previous description for X2 is correct except for the last X1A cycle of the GPS week shown in Table 5.9. During this last cycle, the X2A register holds in its final state (4091) and then the X2B register holds in its final state (4092) until the end of the last X1 cycle of the week. The X1B final state holding count is the same as for the rest of the week (compare with Table 5.7).

The same future scheduling must be performed to accomplish the code setup process for the P code generator as was used in the C/A code setup. The X1A and X2A code setters count P code clock cycles from 0 to 4,091 and the X1B and X2B code setters count from 0 to 4,092. This simplified code setter design example uses three countdown delay counters, D1B for X1B, D2A for X2A, and D2B for X2B, which are set by flags from the receiver baseband process at the end of an X1 or X2 cycle. Note that X1A is never delayed. It is possible to design the code setter so that the receiver baseband process does not have to set flags, but this simplified design example suffices to explain the principles involved. As observed in Tables 5.7 and

Table 5.8 PRN Code States Corresponding to Final Two and Reset Count States

Code Setter States (Decimal)	*X1A Code (Hexadecimal)*	*X1B Code (Hexadecimal)*	*X2A Code (Hexadecimal)*	*X2B Code (Hexadecimal)*
4090	892	·	E49	·
4091	124	955	C92	155
4092	·	2AA	·	2AA
0 (reset)	248	554	925	554

Table 5.9 Count States for 3,749th and 3,750th Cycles of X1A in Last X1 Cycle of Week

Z	*X1*	*X1A*	*X1B*	*D1B*	*X2*	*X2A*	*D2A*	*X2B*	*D2B*
403199	15339838	3022	3367	0	421475	4091	0	3989	0
403199	15339839	3023	3368	0	421476	0	1069	3990	0
.	.	.	.	.	.	.	.	.	.
403199	15339941	3125	3470	0	421578	102	1069	4092	0
403199	15339942	3126	3471	0	421579	103	1069	0	965
.	.	.	.	.	.	.	.	.	.
403199	15340563	3747	4092	0	422200	724	1069	621	965
403199	15340564	3748	0	343	422201	725	1069	622	965
.	.	.	.	.	.	.	.	.	.
403199	15340907	4091	343	343	422544	1068	1069	965	965
403199	15340908	0	344	343	422545	1069	1069	966	965
.	.	.	.	.	.	.	.	.	.
403199	15343929	3021	3365	343	425566	4090	1069	3987	965
403199	15343930	3022	3366	343	425567	4091	1069	3988	965
403199	15343931	3023	3367	343	425568	4091	1068	3989	965
.	.	.	.	.	.	.	.	.	.
403199	15344033	3125	3469	343	425670	4091	966	4091	965
403199	15344034	3126	3470	343	425671	4091	965	4092	965
403199	15344035	3127	3471	343	425672	4091	964	4092	964
.	.	.	.	.	.	.	.	.	.
403199	15344655	3747	4091	343	426292	4091	344	4092	344
403199	15344656	3748	4092	343	426293	4091	343	4092	343
403199	15344657	3749	4092	342	426294	4091	342	4092	342
.	.	.	.	.	.	.	.	.	.
403199	15344998	4090	4092	1	426635	4091	1	4092	1
403199	15344999	4091	4092	0	426636	4091	0	4092	0
0	0	0	0	0	0	0	0	0	0

Note: These count states are not the PRN code states contained in the shift registers.

5.9, D1B is always set to a count of 343 chips. D2A is set to 37 and D2B is set to 380 unless it is at the end of the last X2 cycle of the week, in which case they are set to 1,069 and 965, respectively. The rule followed by the code setters is that if their delay counter is nonzero, they hold their final states until the delay counter counts down to zero, then they rollover to the zero (reset) states. The delay counters rule is that if their code setters are not in the final state, they hold their delay counts until the code setter reaches its final state, then they begin counting down. The receiver baseband rule is that if the code setter counts to the reset state can be reached without using the delay counter, the receiver baseband process adjusts the code setter to

the appropriate value and does not set the delay counter flag. Otherwise, it sets the appropriate delay counter flags (which instructs the code setter hardware to put the appropriate maximum delays into the counters). This rule avoids the need for the code setter hardware to set any other count state into the delay counters other than their maximum delay counts.

The code setters are loaded with the timing state valid at the first epoch that the counting process begins. As each code setter rolls over to zero, it resets its corresponding code register to that register's initial code state. In addition, the X1A divide-by-3,750 counter and the Z counter must be set to their correct states when the X1A register is reset. Similarly, the X1B divide-by-3,749 counter, the X2A divide-by-3,750 counter, and the X2B divide-by-3,749 counter must be set to their correct states when their respective registers are reset by their code setters. Thus, when all four code setters have rolled over, the code generator is synchronized to the code accumulator. This requires approximately 500 μs in the worst case.

The final step in the explanation of the P code setter operation is the algorithm for converting the code accumulator into P code setter states and the setting of the three flags, D1B, D2A, and D2B. The code setter timing and the rules have already been explained. Figure 5.32 depicts the logical flow diagram that covers all P code setter timing conditions. This diagram should be compared to the count states in Tables 5.7 and 5.9. Note that if the P code generator is already synchronized to the code accumulator, the action of the P code setter does not alter the replica code state, since the reset pulses occur at exactly the same times that they naturally occur in the code generator. However, to ensure that the receiver baseband software code accumulator always matches the P code generator code state, it is prudent for the software to periodically repeat the code setup process.

Alternatively, all 4,092 PRN states of the 12-bit X1A and X1B shift registers and all 4,093 PRN states of the 12-bit X2A and X2B shift registers can be precomputed and stored as lookup tables that are indexed by the delays computed in Figure 5.32 (without the need for the flag setting logic). These PRN values can then be transferred to buffer code setter registers and their contents transferred into their respective registers at the appropriate epoch.

With Y code operation, the same P code processes are implemented in the same manner and then encrypted by a specialized hardware design. In the original military receivers, this was called the AOC. The AOC output, when combined with P code, synthesizes the Y code function before correlation with the incoming SV Y code signals. The AOC implements a classified encryption algorithm, and each receiver channel requires one AOC. The component that synchronizes all of the AOCs to their respective replica P code generators is the PPSSM. In newer military receivers, the AOC and the PPSSM functions are integrated into the receiver design. The new military GPS receiver engine is the SAASM. The SAASM is a more secure advanced military GPS receiver design. Only keyed PPS receivers can replicate the Y code.

This code setter design example supports direct P(Y) code acquisition. Direct P(Y) code acquisition is used if the receiver can accurately predict the satellite transmit time so that less time is required to acquire the P(Y) code by direct sequence than to perform a C/A code search and handover. The direct P(Y) code acquisition condition is satisfied if the receiver has previously acquired four or more satellites and its

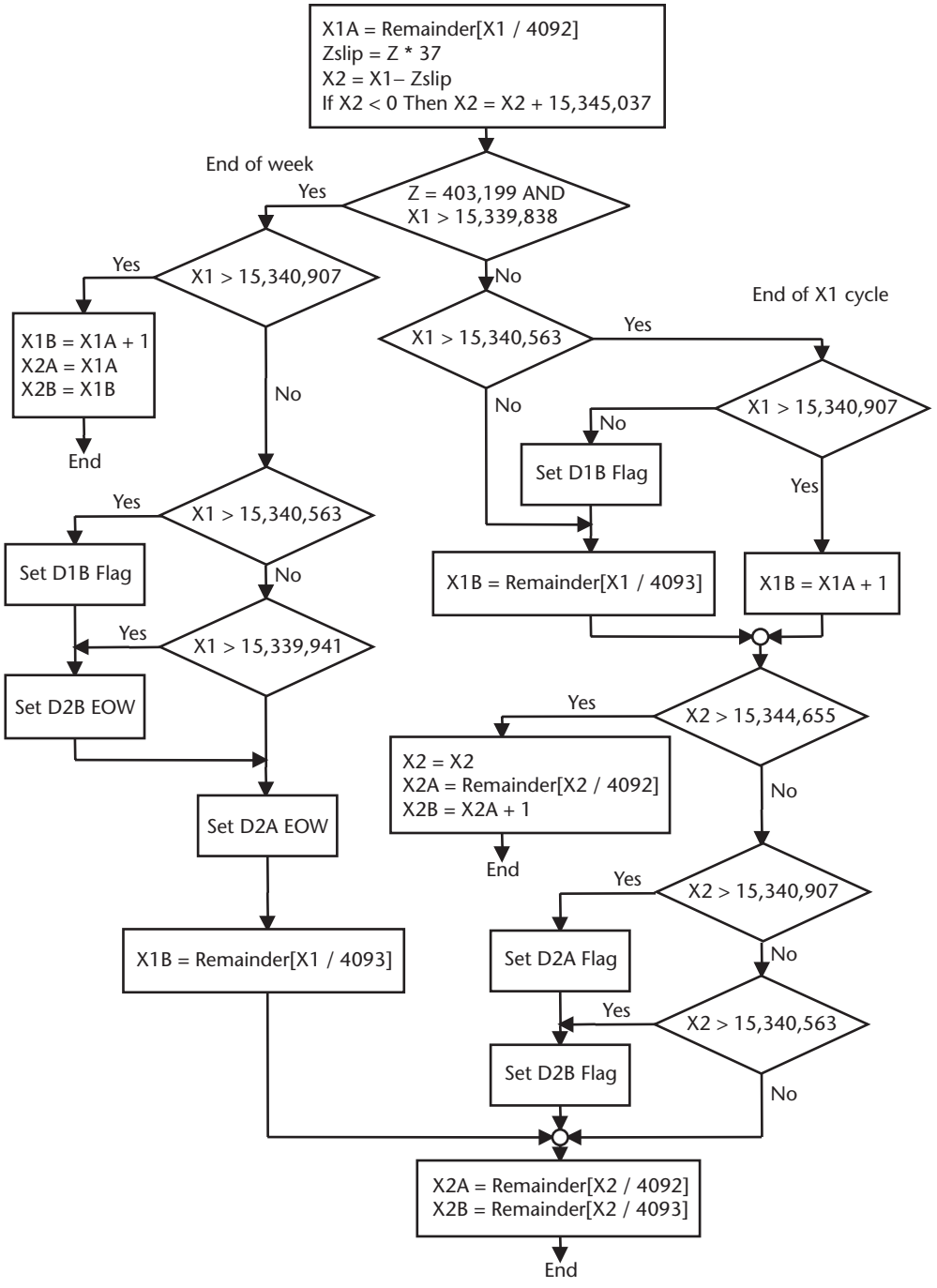


Figure 5.32 Flowchart of P code setter algorithm.

navigation solution has converged. Under certain jamming conditions, it may be impossible to acquire the C/A code but possible to acquire P(Y) code. Then, direct P(Y) code acquisition is essential. This is possible only if the navigation state has sufficient position, velocity, and time precision, plus the ephemeris data for all SVs to be acquired, or this information is transferred to the receiver by an operating host

GPS receiver. The most sensitive navigation state parameter is usually precise time because there is 1m of range uncertainty to each SV that must be searched for every 3 ns of time uncertainty. Massively parallel correlators help to minimize the sensitivity to time and position uncertainty. Reference [6] describes a multiple correlator/search detector architecture that supports rapid direct P(Y) code acquisition. If the receiver is engaged in the direct P(Y) code search process, the P code advance/retard function shown in Figure 5.31 provides the capability to add clock cycles or remove clock cycles in 1/2-chip increments. The code accumulator keeps track of these changes.

Obtaining Transmit Time from the C/A Code

Figure 5.33[2] illustrates the GPS timing relationships that enable a C/A code receiver to determine the true GPS transmit time. The C/A code repeats every 1 ms and is therefore ambiguous every 1 ms of GPS time (about every 300 km of range). There is a HOW at the beginning of every one of the five subframes of the satellite navigation message. The HOW contains the Z count of the first data bit transition boundary at the beginning of the next subframe. This is the first data bit of the TLM that precedes every HOW. The beginning of this 20-ms data bit is synchronized with the beginning of one of the satellite's C/A code 1-ms epochs, but there are 20 C/A code epochs in every data bit period. At this subframe epoch, the X1 register has just produced a carry to the Z-count, so the X1 count is zero. The C/A code ambiguity is resolved by setting the Z count to the HOW value and the X1 count to zero at the beginning of the next subframe. In practice, the actual values of the Z count and X1 count are computed for a near-term C/A code epoch without waiting for the next subframe.

The Z-count and X1-count will be correct if the GPS receiver has determined its bit synchronization to within 1 ms or better accuracy. This level of accuracy will perfectly align the 1-ms C/A code epoch with the 20-ms data bit transition point of the first bit in the following TLM word. Therefore, the C/A code transmit time will be unambiguous and correct. If the bit synchronization process makes an error in the alignment of the 1-ms replica C/A code epoch with the 20-ms data bit epoch, then the X1 count will be off by some integer multiple of 1 ms. If the receiver attempts a handover to P code and fails, the typical strategy is to try the handover again with 1-ms changes in the value used for X1, then 2-ms changes, and so on, before attempting to redetermine bit synchronization, which can take 6 seconds or longer and prevents processing of GPS measurements until complete.

A successful handover to P code verifies the bit synchronization process. However, if the GPS receiver is a C/A code receiver, the verification for correct bit synchronization is more difficult. This verification task must be performed by the navigation process. Since 1 ms of GPS time error is equivalent to about 300 km of pseudorange error, the navigation error can be quite serious. In the unlikely case that every channel makes the identical bit sync error, the navigation position error washes out of the position solution into the time bias solution, and the GPS time is in error by 1 ms. The typical bit sync error manifestations in the navigation solution are unrealistic local level velocity and elevation computations. The latitude and lon-

2. Jerry D. Holmes originally developed a similar GPS timing chart for use by the GPS Systems Engineering staff at Texas Instruments, Inc., during the GPS phase I development program, circa 1976.

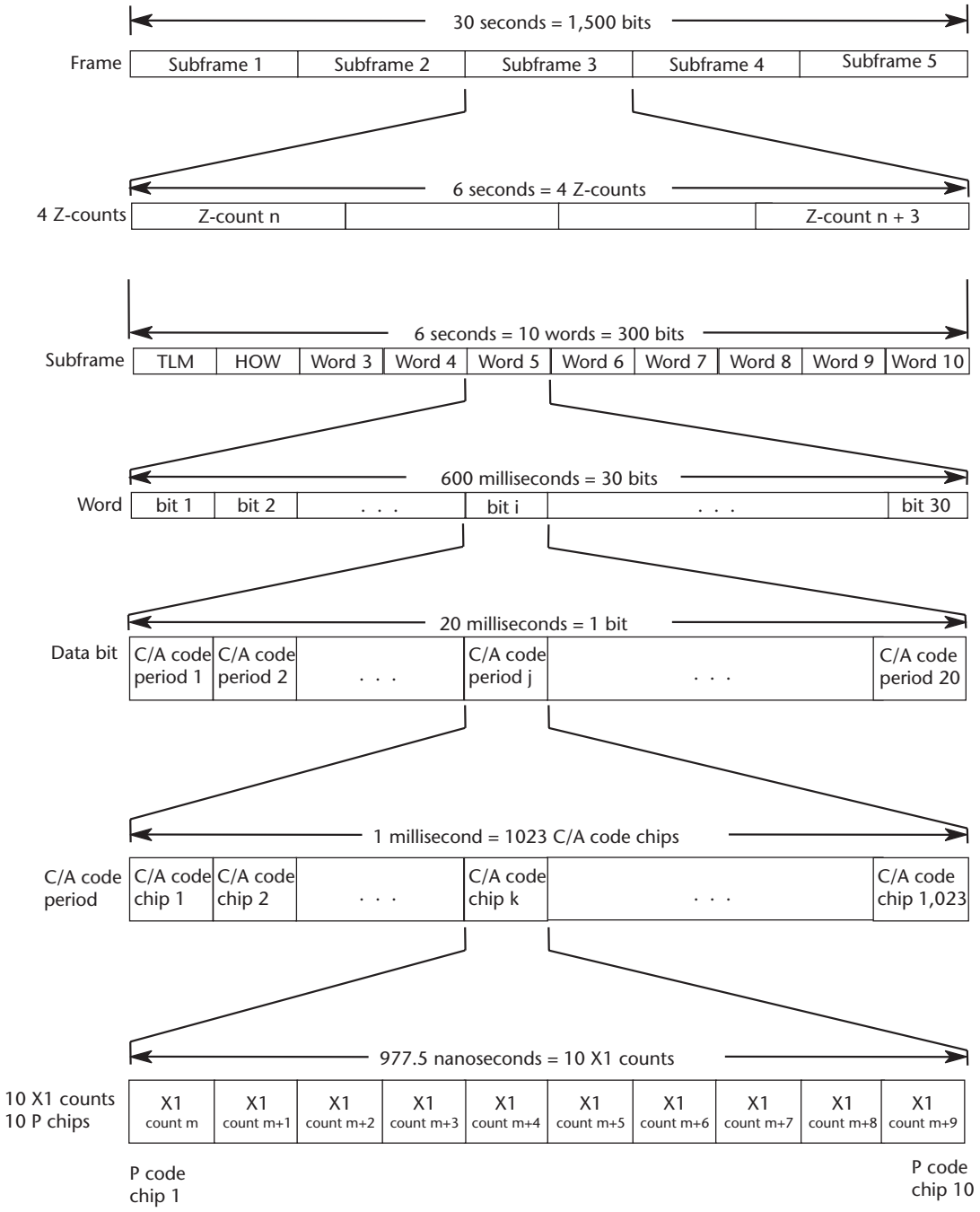


Figure 5.33 GPS C/A code timing relationships.

gitude computations are also unrealistic, but there is usually no boundary condition for comparison. However, the velocity and elevation computations can be compared to acceptable boundary conditions.

The bit synchronization process is a statistical process that is dependent on C/N_0. It will occasionally be incorrect. It will be incorrect almost every time the

C/N_0 drops below the bit synchronization design threshold. This causes serious navigation integrity problems for C/A code receivers under conditions of signal attenuation or RF interference. This problem is compounded if there is no design provision to adapt the bit synchronization process for poor C/N_0 conditions or for the navigation process to check for bit synchronization errors.

5.7.2 Delta Pseudorange

The definition of delta pseudorange to SV_i is as follows.

$$\Delta\rho_i(n) = \rho_i(n+J) - \rho_i(n-K) \quad (\text{m}) \tag{5.32}$$

where:

$\rho_i(n+J)$ = pseudorange at J FTF epochs later than FTF(n) (m)
$\rho_i(n-K)$ = pseudorange at K FTF epochs earlier than FTF(n) (m)
$J = 0$ or K depending on design preferences (dimensionless)

Even though (5.32) implies that delta pseudorange could be derived from the code tracking loop, the result would be a very noisy measurement. This differential measurement will be more than two orders of magnitude less noisy if taken from the carrier tracking loop operated as a PLL. If the carrier loop is operated as a FLL, the measurement is taken the same way, but the FLL is only about 100 times less noisy than code measurements.

The carrier Doppler phase measurements are extracted by the receiver baseband process from the carrier tracking loop using a carrier accumulator. It is more simple but similar to using the code accumulator to extract transmit time measurements from the code tracking loop. The carrier accumulator consists of the integer cycle count, N_{CA}, and the fractional cycle count, Φ_{CA}, of the carrier Doppler phase measurement.

The carrier accumulator is updated after each carrier loop output to the carrier NCO using the following algorithm (remembering that the equal sign means that the right-hand side replaces the value on the left-hand side):

$$\begin{aligned} \Phi_{temp} &= \Phi_{CA} + f_c \Delta\Phi_{CA} T \\ \Phi_{CA} &= \text{fractional part of } \Phi_{temp} \quad (\text{cycles}) \\ N_{CA} &= N_{CA}(\text{last value}) + \text{integer part of } \Phi_{temp} \quad (\text{cycles}) \end{aligned} \tag{5.33}$$

where:

Φ_{temp} = temporary Φ_{CA} register
f_c = carrier NCO clock frequency (Hz)
$\Delta\Phi_{CA}$ = carrier NCO carrier Doppler phase increment per clock cycle
= carrier loop filter velocity correction + velocity aiding (if any)
T = time between carrier NCO updates (seconds)

N_{CA} = integer carrier Doppler phase cycles since some starting point

The fractional part of the carrier accumulator, Φ_{CA}, is initialized to the same state as the carrier NCO at the beginning of the search process, which is typically zero. The integer number of carrier Doppler phase cycles, N_{CA}, is ambiguous. Since only differential measurements are taken from this register, the ambiguity does not matter. The carrier integer accumulator is usually set to zero when the carrier loop is first closed following a successful search operation. Note that the marching of time carrier NCO bias (see Figure 5.2) is not included in the carrier accumulator because it is simply a bias term to match the carrier at IF. Since only differential measurements are extracted from the carrier accumulator, this bias term, if included, would cancel out. The counter rolls over when the Doppler cycle count exceeds the count capacity or underflows if the Doppler count is in the reverse direction and drops below the zero count. The differential measurement comes out correct if the counter capacity is large enough to ensure that this happens no more than once between any set of differential measurements extracted from the carrier accumulator.

To extract a carrier Doppler phase measurement, $N_{CAi}(n)$, $\Phi_{CAi}(n)$, for SV_i corresponding to the carrier accumulator, it must be propagated forward to the nearest FTF(n) by the skew time, T_S, similar to the technique used in the code tracking loop.

$$\begin{aligned} \Phi_{temp} &= \Phi_{CA} + f_c \Delta\Phi_{CA} T_S \\ \Phi_{CAi} &= \text{fractional part of } \Phi_{temp} \qquad (\text{cycles}) \\ N_{CAi} &= \text{integer part of } \Phi_{temp} \qquad (\text{cycles}) \end{aligned} \tag{5.34}$$

Note that there is no error due to the measurement propagation process for the carrier Doppler phase measurement because the carrier NCO is running at a constant rate, $\Delta\Phi_{CA}$ per clock, during the propagation interval.

The precise delta pseudorange is simply the change in phase in the carrier accumulator during a specified time. The formula for extracting the delta pseudorange from the carrier accumulator is as follows.

$$\Delta\rho_i(n) = \left\{ \begin{matrix} \left[N_{CAi}(n+J) - N_{CAi}(n-K) \right] + \\ \left[\Phi_{CAi}(n+J) - \Phi_{CAi}(n-K) \right] \end{matrix} \right\} \lambda_L \quad (\text{m}) \tag{5.35}$$

where:

λ_L = wavelength of the L-band carrier frequency
= 0.1903 m/cycle for L1
= 0.2442 m/cycle for L2

As a design example, suppose the navigation measurement incorporation rate is 1 Hz (a very typical rate), then the delta pseudorange measurement should begin and end with each range measurement. To accommodate this for an FTF period of 20 ms, $J = 0$ and $K = -50$. Alternatively, if the navigation throughput permits, the

delta pseudorange measurement could be incorporated at a 50-Hz rate with $J = 0$ and $K = 1$. In either case, the delta range measurement should be modeled by the navigation process as a change in range over the previous second, not as an average velocity over the interval.

5.7.3 Integrated Doppler

The definition of integrated Doppler is obtained from (5.34). The integrated Doppler measurement for SV_i at FTF(n) can be converted to units of meters as follows.

$$ID_i(n) = \left[N_{CAi}(n) + \Phi_{CAi}(n)\right]\lambda_L \quad (\text{m}) \tag{5.36}$$

This measurement, when derived from a PLL, is used for ultraprecise differential interferometric GPS applications, such as static and kinematic surveying or for attitude determination. Note that when the integer cycle count ambiguity is resolved by the interferometric process, this measurement is equivalent to a pseudorange measurement with more than two orders of magnitude less noise than the transmit time (pseudorange) measurements obtained from the code loop. The integrated Doppler noise for a high-quality GPS receiver designed for interferometric applications typically is about 1 mm (1 sigma) under good signal conditions. A transmit time (pseudorange) measurement typically will have about 1m of noise (1 sigma). Once the integer cycle ambiguity is resolved, as long as the PLL does not slip cycles, the ambiguity remains resolved thereafter. (Further information on differential interferometric processing and ambiguity resolution is provided in Section 8.4.)

Two GPS receivers that are making transmit time and carrier Doppler phase measurements on their respective receiver epochs will in general be time skewed with respect to one another. For ultraprecise differential applications, it is possible to remove virtually all of the effects of time-variable bias by eliminating this time skew between GPS receivers (i.e., spatially separated GPS receivers can make synchronous measurements). This is accomplished by precisely aligning the measurements to GPS time epochs instead of to (asynchronous) receiver FTF epochs. Initially, of course, the measurements must be obtained with respect to the receiver FTF epochs. After the navigation process determines the time bias between its FTF epochs and true GPS time, each navigation request for a set of receiver measurements should include the current estimate of the time bias with respect to the FTF (a very slowly changing value if the reference oscillator is stable). The receiver measurement process then propagates the measurements to the FTF plus the time bias as nearly perfect (within nanoseconds) of true GPS time. These measurements are typically on the GPS 1-second time of week epoch. This is important for precision differential operation since as little as 1 second of time skew between receivers corresponds to satellite position changes of nearly 4,000m. Of course, the differential measurements can be propagated to align to the same time epoch if the GPS receiver's measurements are time skewed, but not with the accuracy that can be obtained if they are aligned to a common GPS time epoch within each GPS receiver during the original measurement process. The carrier Doppler measurement must be corrected for the frequency error in the satellite's atomic standard (i.e., reference oscillator) before measurement incorporation. This correction is broadcast in the

satellite's navigation message as the a_{f1} term (see Sections 4.4 and 7.2.1). The measurement also includes the receiver's reference oscillator frequency error. This error is determined as a time bias rate correction by the navigation solution. For some applications, it is also corrected for the differential ionospheric delay, but this is usually a negligible error.

5.8 Signal Acquisition

There is a large amount of literature on PRN code acquisition in direct sequence receivers. For an extensive historical survey and descriptions on time domain search detectors, [15] is recommended. Reference [16] describes the use of modern frequency domain search techniques for rapid acquisition. The following GPS signal acquisition material is based on traditional time domain search techniques.

GPS signal acquisition is a search process. This search process, like the tracking process, requires replication of both the code and the carrier of the SV to acquire the SV signal (i.e., the signal match for success is two dimensional). The range dimension is associated with the replica code. The Doppler dimension is associated with the replica carrier. The initial search process is always a C/A code search for C/A code receivers and usually begins with a C/A code search for P(Y) code receivers. The initial C/A code search usually involves replicating all 1,023 C/A code phase states in the range dimension. The criteria for direct C/A code and direct P(Y) code acquisitions were discussed in the previous section. If the range and Doppler uncertainty are known, then the search pattern should cover the 3-sigma values of the uncertainty. If the uncertainty is large in either or both dimensions, the search pattern is correspondingly large, and the expected search time increases. Some criteria must be established to determine when to terminate the search process for a given SV and select another candidate SV. Fortunately, the range dimension for C/A code search is bounded by the ambiguity of C/A code to only 1,023 chips total range uncertainty, but it is essentially unbounded for direct P(Y) code search.

The following example assumes that a C/A code search is being performed and that all 1,023 C/A code phases are being examined. The code phase is typically searched in increments of 1/2 chip. Each code phase search increment is a code bin. Each Doppler bin is roughly $2/(3T)$ Hz, where T is the search dwell time (the longer the dwell time, the smaller the Doppler bin). The combination of one code bin and one Doppler bin is a cell. Figure 5.34 illustrates the two-dimensional search process. If the Doppler uncertainty is unknown and the SV Doppler cannot be computed from a knowledge of the user position and time plus the SV orbit data, then the maximum user velocity plus just less than 800 m/s maximum SV Doppler (for worst case, see Section 2.5) for a stationary user must be searched in both directions about zero Doppler.

As stated earlier, one Doppler bin is defined as approximately $2/(3T)$, where T = signal integration time per cell or dwell time per cell. Dwell times can vary from less than 1 ms (Doppler bins of about 667 Hz) for strong signals up to 10 ms (67-Hz Doppler bins) for weak signals. The poorer the expected C/N_0, then the longer the dwell time (and overall search time) must be in order to have reasonable success of signal acquisition. Unfortunately, the actual C/N_0 is unknown until after the SV sig-

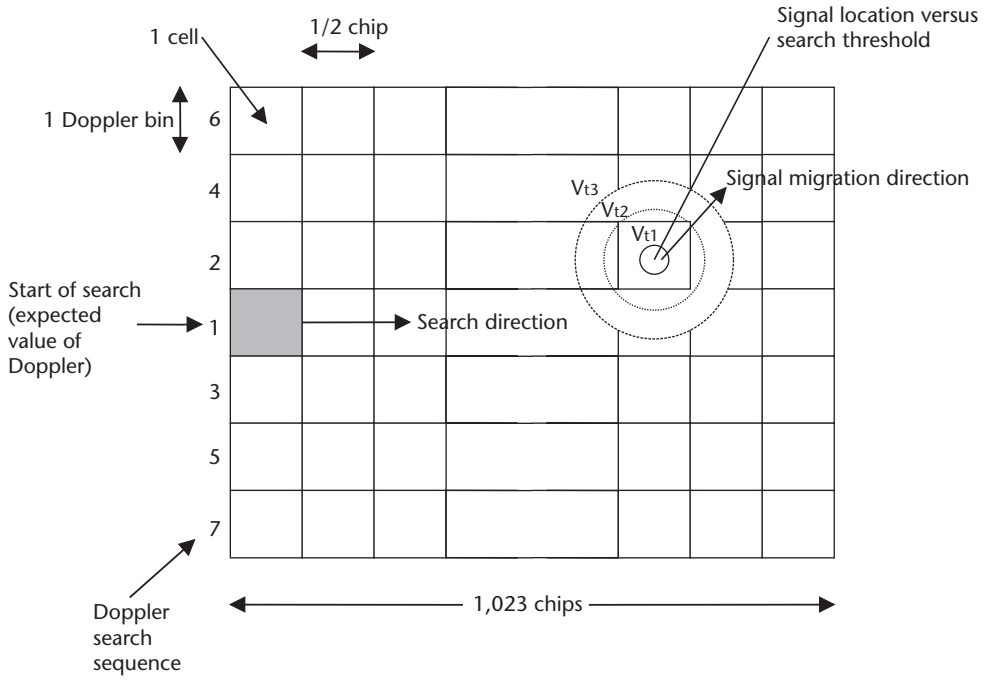


Figure 5.34 Two-dimensional C/A code search pattern.

nal is acquired. Signal obscuration (trees, buildings, snow or ice on the antenna, and so forth), RF interference, ionospheric scintillation, and antenna gain roll-off can all significantly reduce C/N_0 (see Chapter 6).

Referring to Figure 5.34, the search pattern usually follows the range direction from early to late in order to avoid multipath with Doppler held constant until all range bins are searched for each Doppler value. The direct arrival of a signal subject to multipath is always ahead in time of the reflected arrivals. In the Doppler bin direction, the search pattern typically starts from the mean value of the Doppler uncertainty (zero Doppler if the actual LOS velocity estimate is unknown) and then goes symmetrically one Doppler bin at a time on either side of this value until the 3-sigma Doppler uncertainty has been searched. Then the search pattern is repeated, typically with a reduction in the search threshold scale factor. It is important to recognize that the C/A code autocorrelation and crosscorrelation sidelobes can cause false signal detections if these sidelobes are strong enough. The sidelobes tend to increase as the search dwell time is decreased. To counter this problem, a combination of both increased dwell time (to minimize sidelobes) and a high detector threshold setting (to reject sidelobes) can be used for the initial search pass. On subsequent search passes, the dwell time and threshold can be decreased. The penalty for this scheme is increased search time when the C/N_0 is low.

During the dwell time, T, in each cell, the I and Q signals are integrated and dumped and the envelope $\sqrt{I^2 + Q^2}$ is computed or estimated. Each envelope is compared to a threshold to determine the presence or absence of the SV signal. The detection of the signal is a statistical process because each cell either contains noise with the signal absent or noise with the signal present. Each case has its own probability density function (pdf). Figure 5.35 illustrates a single trial (binary) decision

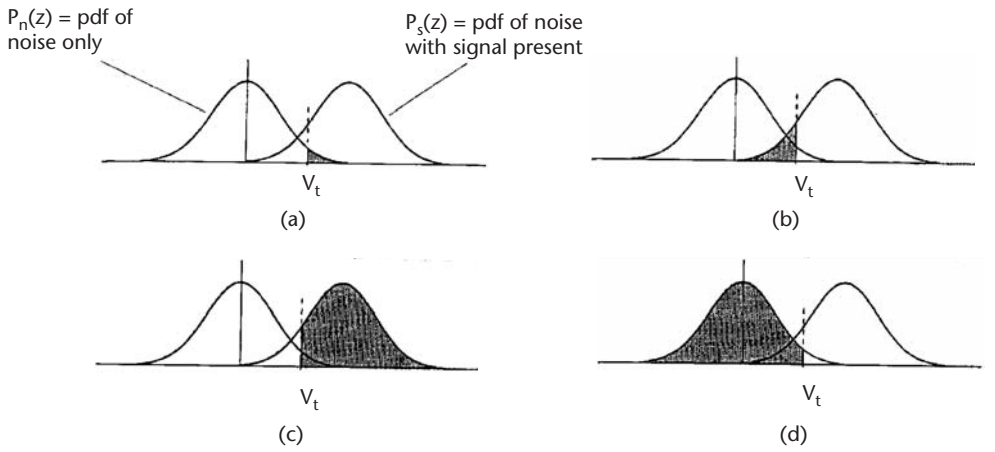


Figure 5.35 Pdfs for a binary decision: (a) shaded area represents probability of false alarm, (b) shaded area represents probability of false dismissal, (c) shaded area represents probability of detection, and (d) shaded area represents probability of correct dismissal.

example where both pdfs are shown. The pdf for noise with no signal present, $p_n(z)$, has a zero mean. The pdf for noise with the signal present, $p_s(z)$, has a nonzero mean. The single trial threshold is usually based on an acceptable single trial probability of false alarm, P_{fa}. For the chosen threshold, V_t, any cell envelope that is at or above the threshold is detected as the presence of the signal. Any cell envelope that is below the threshold is detected as noise. There are four outcomes of the single trial (binary) decision processes illustrated in Figure 5.35, two wrong and two right. By knowing the pdfs of the envelopes, the single trial probability can be computed by an appropriate integration with the threshold as one limit and infinity as the other. These integrations are shown as the shaded areas in Figure 5.35. The two statistics that are of most interest for the signal detection process are the single trial probability of detection, P_d, and the single trial probability of false alarm, P_{fa}. These are determined as follows:

$$P_d = \int_{V_t}^{\infty} p_s \, dz \tag{5.37}$$

$$P_{fa} = \int_{V_t}^{\infty} p_n \, dz \tag{5.38}$$

where:

$p_s(z)$ = pdf of the envelope in the presence of the signal
$p_n(z)$ = pdf of the envelope with the signal absent

To determine these pdfs, assume that I and Q have a Gaussian distribution. Assuming that the envelope is formed by $\sqrt{I^2 + Q^2}$, then $p_s(z)$ is a Ricean distribution [17] defined by:

$$p_s(z) = \begin{cases} \dfrac{z}{\sigma_n^2} e^{-\left(\frac{z^2+A^2}{2\sigma_n^2}\right)} I_0\left(\dfrac{zA}{\sigma_n^2}\right), & z \geq 0 \\ 0, & z < 0 \end{cases} \tag{5.39}$$

where:

z = value of the random variable

σ^2 = RMS noise power

A = RMS signal amplitude

$I_0\left(\dfrac{zA}{\sigma_n}\right)$ = modified Bessel function of zero order

$I_0(x) \approx \dfrac{e^x}{\sqrt{2\pi x}}$ for $x >> 1$

Equation (5.39) for $z \geq 0$ can be expressed in terms of the predetection SNR as presented to the envelope detector, C/N (dimensionless), as follows:

$$p_s(z) = \frac{z}{\sigma_n^2} e^{-\left(\frac{z^2}{2\sigma_n^2} + C/N\right)} I_0\left(\frac{z\sqrt{2C/N}}{\sigma_n}\right) \tag{5.40}$$

where:

C/N = predetection signal to noise ratio

$C/N = A^2 / 2\sigma_n^2$

$= (C/N_0)T$

T = search dwell time

For the case where there is no signal present, then evaluating (5.39) for $A = 0$ yields a Rayleigh distribution for $p_n(z)$, which is defined by:

$$p_n(z) = \frac{z}{\sigma_n^2} e^{-\left(\frac{z^2}{2\sigma_n^2}\right)} \tag{5.41}$$

The result of integrating (5.38) using the pdf of (5.41) is:

$$p_{fa} = e^{-\left(\frac{V_t^2}{2\sigma_n^2}\right)} \tag{5.42}$$

Rearranging (5.42) yields the threshold in terms of the desired single trial probability of false alarm and the measured 1-sigma noise power:

$$V_t = \sigma_n \sqrt{-2 \ln P_{fa}} = X\sigma_n \tag{5.43}$$

For example, if it is desired that $P_{fa} = 16\%$, then $V_t = X\sigma_n = 1.9144615\ \sigma_n$. Using this result, the single trial probability of detection, P_d, is computed for the expected C/N_0 and dwell time, T, using (5.37) and (5.40) with $\sigma_n = 1$ (normalized). Some examples of the single trial probability of detection are shown in Table 5.10 for various SNRs.

Figure 5.36, taken from [18], illustrates the structure of search detectors used for signal acquisition. Referring to Figure 5.36, two types of search detectors used in GPS receiver designs will be described. A variable dwell time detector makes a "yes" or "no" decision in a variable interval of time if "maybe" conditions are present. A fixed dwell time detector makes a "yes" or "no" decision in a fixed interval of time. The probability of detection and especially the poor false alarm rate from single dwell time detectors (single trial decisions as shown in Table 5.10) are usually unsatisfactory for GPS applications. So, single dwell time search detector schemes are seldom used. All other things being equal, a properly tuned variable dwell time (sequential) multiple trial detector will search faster than a fixed dwell time multiple trial detector.

5.8.1 Tong Search Detector

The first example of a search algorithm is a sequential variable dwell time search detector called the Tong detector. Figure 5.37 illustrates the block diagram of the Tong detector. Search algorithms are typically implemented as a receiver baseband process. Because of its simplicity, the Tong detector can be implemented as part of the receiver correlation and preprocessing hardware, with its search parameters programmed by the baseband process. The Tong detector has a reasonable computational burden and is excellent for detecting signals with an expected $(C/N_0)_{dB}$ of 25 dB-Hz or higher. If acquisition is to be performed under heavy jamming conditions where the $(C/N_0)_{dB}$ will be less than this, then a hybrid maximum-likelihood search detector should be used. A pure maximum-likelihood search detector would require the receiver hardware to produce the results of all of the search dwells in parallel, which is usually impractical. The Tong detector is a suboptimal search algorithm that requires an average factor of only 1.58 longer to make a decision than a maximum-likelihood (optimum) search algorithm [19].

Table 5.10 Single Trial Probability of Detection

C/N	P_d	$(C/N_0)_{dB}=(C/N)_{dB} - 10\log_{10}T$ (dB-Hz)			
(ratio)	*(dimensionless)*	*T = 1 ms*	*T = 2.5 ms*	*T = 5 ms*	*T = 10 ms*
1.0	0.431051970	30.00	26.02	23.01	20.00
2.0	0.638525844	33.01	29.03	26.02	23.01
3.0	0.780846119	34.77	30.79	27.78	24.77
4.0	0.871855378	36.02	32.04	29.03	26.02
5.0	0.927218854	36.99	33.01	30.00	26.99
6.0	0.959645510	37.78	33.80	30.79	27.78
7.0	0.978075147	38.45	34.47	31.46	28.45
8.0	0.988294542	39.03	35.05	32.04	29.03
9.0	0.993845105	39.54	35.56	32.55	29.54

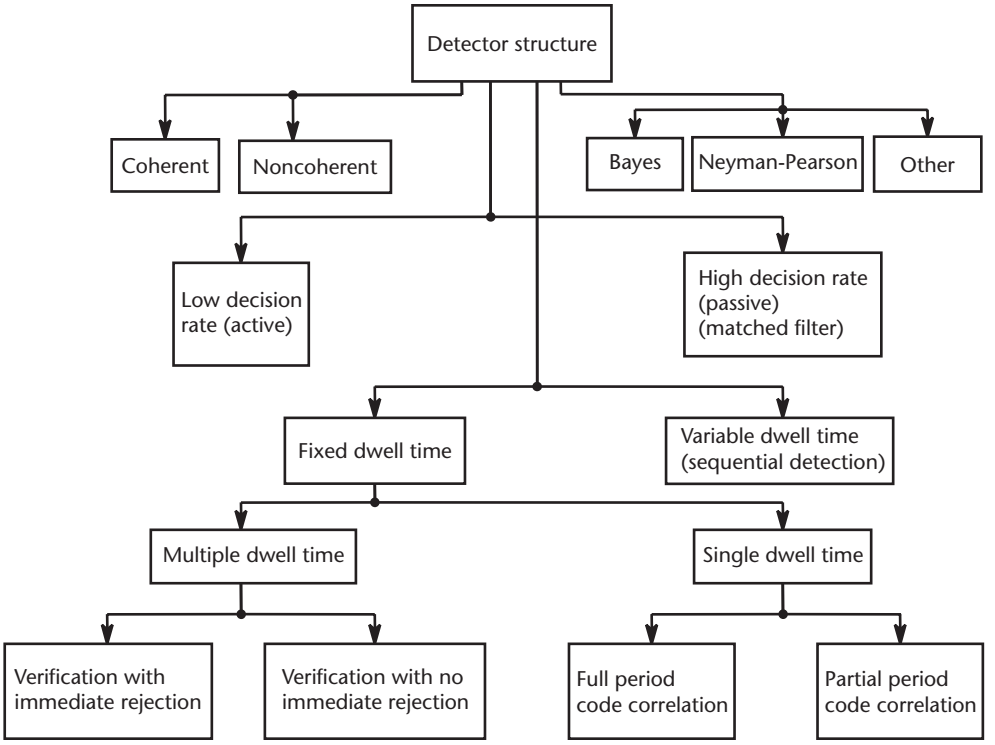


Figure 5.36 Structure of search detectors. (*Source:* [18].)

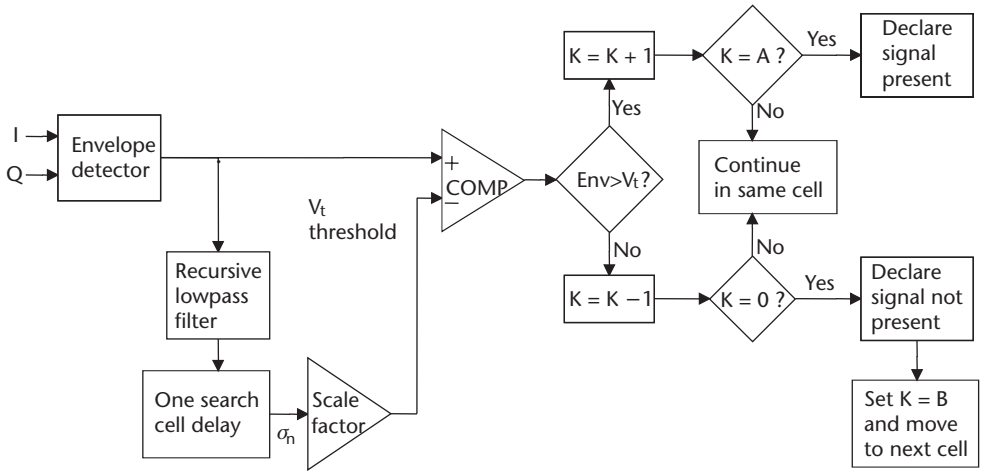


Figure 5.37 Tong (variable dwell) sequential code search algorithm.

The operation of the Tong detector is as follows:

1. Every T seconds an integrated correlation envelope is formed as $\sqrt{I^2 + Q^2}$ (or its approximation). If there are three correlators in the receiver, each correlator is typically spaced 1/2-chip apart and three Tong detectors are used. In this manner, three code bins at a time are searched, and the search

process speeds up by (almost) a factor of three. It is not quite a factor of three because all three search cells must be dismissed before the search process can proceed to the next three cells. In order to generate the I and Q signals, the receiver baseband search process has synthesized the correct Doppler (the center frequency of the respective search process Doppler bin in the search pattern) and the correct replica C/A code phase with the corresponding spreading code chip rate plus code Doppler. For example, the C/A code phase corresponding to the first of its 1,023 states is where the C/A code setter resets the G1 and G2 registers at the beginning of the cell integrate and dump time. The marching of time for the C/A code generator plus the code Doppler keeps its phase aligned to this cell. If the cell is dismissed, then the C/A code generator phase is advanced by 1/2 chip (times the number of correlators), and the search process is continued for that Doppler bin until the last C/A code phase state has been reached. Then the Doppler center frequency is shifted to the next bin in the search pattern and the process is repeated.

2. At each cell the up/down counter (K) is initialized to $K = B = 1$. Where a higher probability of detection and lower probability of false alarm are desired at the expense of search speed, then $B = 2$. If the envelope sample exceeds the threshold, V_t, then the up/down counter is incremented by one. If the sample does not exceed the threshold, then the up/down counter is decremented by one. As shown in Figure 5.37, one technique for obtaining the RMS noise, σ_n, which is used to set the threshold, is to pass the correlation envelopes into a recursive lowpass filter with a delay of one search cell. A better technique is for the receiver to synthesize the RMS noise by correlating the input signal with an unused PRN code (e.g., the G1 register output for C/A code search). The RMS output is multiplied by a scale factor, X, to obtain the threshold, V_t. Assuming that the envelope is formed by $\sqrt{I^2 + Q^2}$, then the scale factor is determined from (5.43), $X = \sqrt{-2\ln(P_{fa})}$, where P_{fa} is the single trial probability of false alarm. Typically, the envelope is determined from the Robertson approximation. In this case, it has been determined [17] that a multiplier factor of 1.08677793 must be used for the scale factor, so that: $X_R = 1.08677793\ X = \sqrt{-2.3621724\ln(P_{fa})}$.

   The determination of the most suitable single trial probability of false alarm, the overall false alarm, and the overall probability of detection is a tuning process. The final determination must be obtained by simulation. Assuming the Robertson envelope approximation, the scale factor range is typically from 1.8 ($P_{fa} = 25\%$) for low expected $(C/N_0)_{dB}$ (≥ 25 dB-Hz) to 2.1 ($P_{fa} = 16\%$) for high expected $(C/N_0)_{dB}$ (≥ 39 dB-Hz).

3. If the counter contents reach the maximum value, A, then the signal is declared present, and the Tong search is terminated. This is typically followed by additional vernier search processes designed to find the code phase and Doppler combination that produces the peak detection of the signal before the code/carrier loop closure process is begun. If the counter reaches 0, then the signal is declared absent and the search process is

advanced to the next cell. The determination of A must be by simulation. Its selection is a tradeoff between search speed and probability of detection, but a typical range is $A = 12$ for low expected C/N_0 to $A = 8$ for high expected C/N_0. Note that it is possible for the Tong detector to get trapped into an extended dwell in the same cell under certain poor signal conditions. For this reason, a *mush* counter should be used that counts every test within the same cell then declares the signal not present when the mush count exceeds A by some reasonable amount.

The mean number of dwell times to dismiss a cell containing noise only is determined as follows:

$$N_n = \frac{1}{1-2P_{fa}} \tag{5.44}$$

Since most of the time is spent searching cells that contain only noise, the Tong detector search speed can be estimated from:

$$R_S = \frac{d}{N_n T} = \frac{d\left(1-2P_{fa}\right)}{T} \qquad (\text{chips/s}) \tag{5.45}$$

where d = chips per cell (typically 1/2 chip per cell)

For example, for P_{fa} = 16%, a dwell time of 5 ms, and 1/2 chip per cell, the code search rate is = 68 chips/s. Note that the search speed increases when the probability of false alarm decreases.

The overall probability of false alarm for the Tong detector is [19]:

$$P_{FA} = \frac{\left(\frac{1-P_{fa}}{P_{fa}}\right)^B - 1}{\left(\frac{1-P_{fa}}{P_{fa}}\right)^{A+B-1} - 1} \tag{5.46}$$

The overall probability of detection for the Tong detector is [19]:

$$P_D = \frac{\left(\frac{1-P_d}{P_d}\right)^B - 1}{\left(\frac{1-P_d}{P_d}\right)^{A+B-1} - 1} \tag{5.47}$$

Figure 5.38 is a plot of (5.47) as a function of the Tong detector input SNR, $(C/N)_{dB}$, = $10\log_{10}(C/N)$ expressed in units of decibels, with $B = 1$ and A as a running parameter ranging from 2 to 12, and with the overall probability of false alarm set equal to 1×10^{-6} for every case [17]. Figure 5.38 illustrates the excellent search detector performance of the Tong detector and the increased sensitivity of the detector

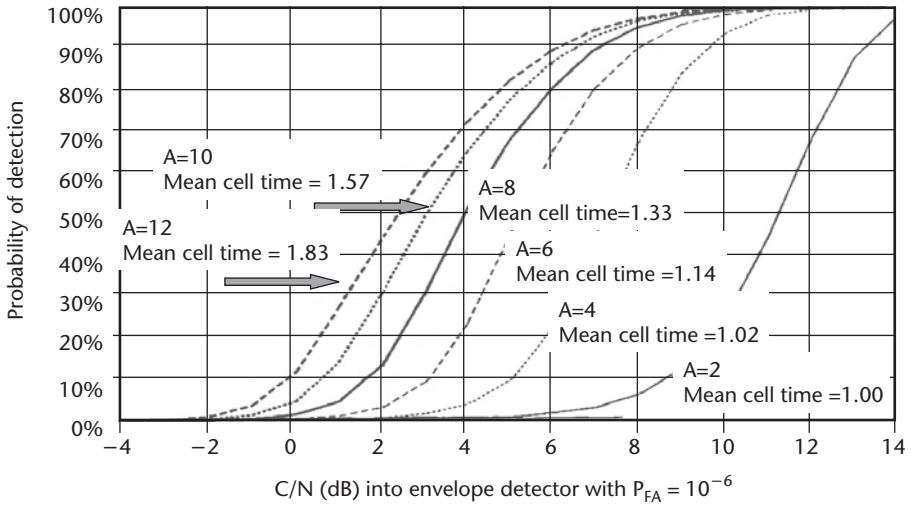


Figure 5.38 Probability of detection for Tong search detector.

with the increase of A. The cost of increasing A is shown as a decrease in the search rate.

5.8.2 *M* of *N* Search Detector

The second example of a search algorithm is a fixed interval detector called the M of N search detector. Figure 5.39 depicts the M of N search algorithm. The M of N search detector takes N envelopes and compares them to the threshold for each cell. If M or more of them exceed the threshold, then the signal is declared present. If not, the signal is declared absent, and the process is repeated for the next cell in the search pattern. These are treated as Bernoulli trials, and the number of envelopes, n, that exceed the threshold has a binomial distribution. The same threshold-setting technique is used and the same formula applies for the single trial probability of

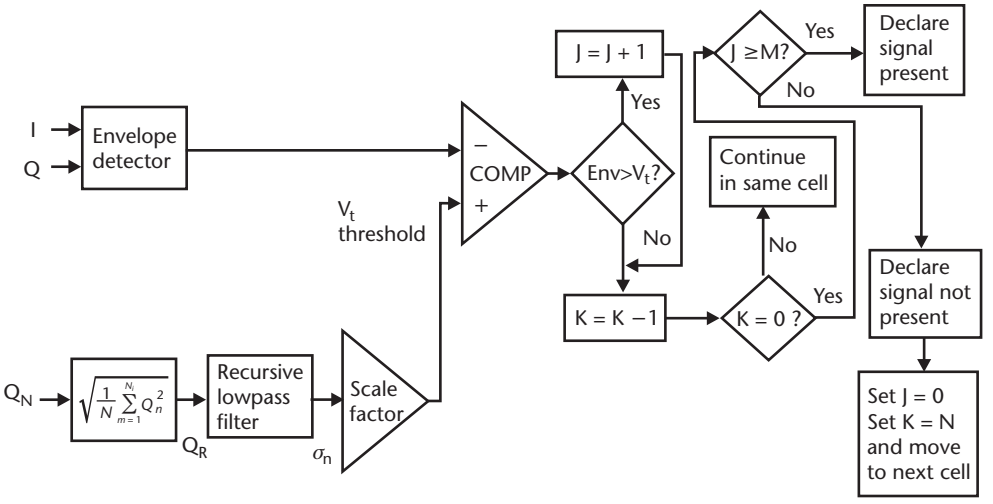


Figure 5.39 *M* of *N* (fixed interval) sequential code search algorithm.

false alarm, P_{fa}, as was described for the Tong detector. Note that the *M* of *N* algorithm in Figure 5.39 contains a superior reference noise source technique compared to that of the Tong algorithm in Figure 5.37. This requires a preplanned receiver design that provides the same PRN code for each channel that is not used by (does not correlate with) any SV in the constellation—such as G1(t) for C/A code and PRN 38 for P(Y) code—and one component of a spare complex correlator (shown as Q_N component in Figure 5.39).

The overall probability of false alarm in *N* trials is [17]:

$$P_{FA} = \sum_{n=M}^{N} \binom{N}{n} P_{fa}^{n} \left(1 - P_{fa}\right)^{N-n} = 1 - \sum_{n=0}^{M-1} \binom{N}{n} P_{fa}^{n} \left(1 - P_{fa}\right)^{N-n} = 1 - B\left(M - 1; N, P_{fa}\right) \tag{5.48}$$

where $B(k;N,p)$ is the cumulative pdf.

The overall probability of detection in *N* trials is [17]:

$$P_{D} = \sum_{n=M}^{N} \binom{N}{n} P_{d}^{n} \left(1 - P_{d}\right)^{N-n} = 1 - B\left(M - 1; N, P_{d}\right) \tag{5.49}$$

Figure 5.40 illustrates the *M* of *N* probability of detection versus *C/N* into the detector for $N = 8$ and $M = 3$, 4, 5, and 6 when $P_{FA} = 1 \times 10^{-6}$. By inspection of Figure 5.40, it is clear that $M = 5$ is the optimum value. The data were generated by computing P_{fa} given *M*, *N*, and P_{FA} using the following equation [17]:

$$P_{fa} = B^{-1}\left(M - 1; N, 1 - P_{FA}\right) \tag{5.50}$$

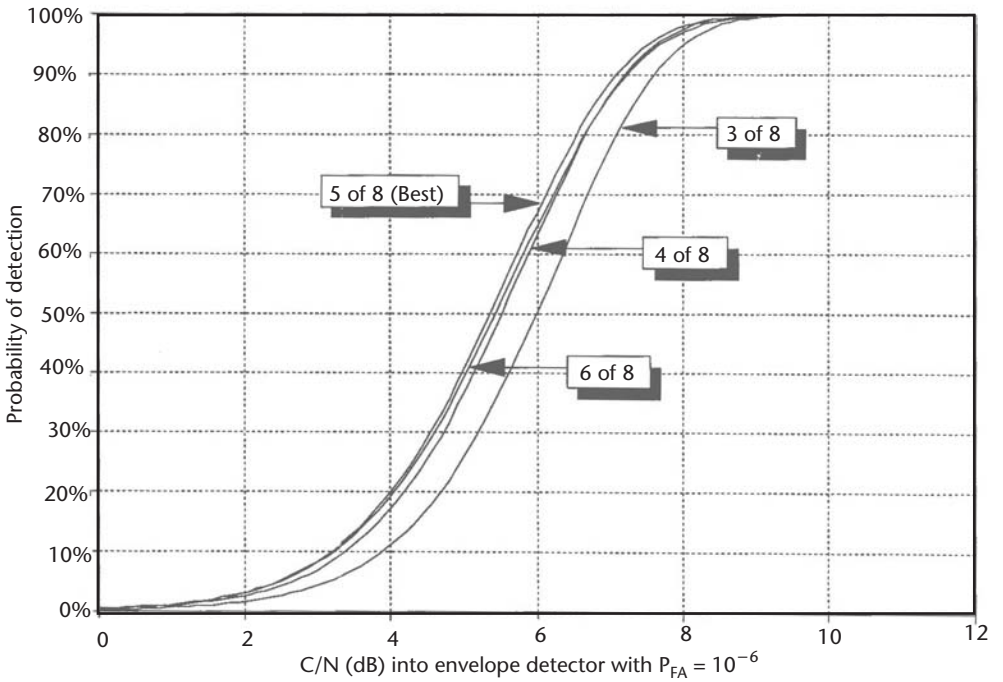


Figure 5.40 Probability of detection for *M* of *N* search detector.

The value for P_{fa} is then substituted into (5.43) to compute the threshold, V_t (assuming the signal is absent). V_t sets the lower limit of the integration when (5.40) is substituted into (5.37) as follows:

$$P_d = \int_{V_t}^{\infty} \frac{z}{\sigma_n^2} e^{-\left(\frac{z^2}{2\sigma_n^2} + C/N\right)} I_0\left(\frac{z\sqrt{2\,C/N}}{\sigma_n}\right) dz \tag{5.51}$$

Using (5.51), P_d is evaluated for each C/N with σ_n normalized to unity. Finally, P_D is computed for the M and N values using (5.49). The search speed for this example, assuming a dwell time of 5 ms, is $R_S = 1/(N\,2T) = 12.5$ chips/s, which is more than five times slower than the Tong search speed.

5.8.3 Direct Acquisition of GPS Military Signals

While P(Y) code was designed for acquisition through C/A code, efforts have been made to develop direct P(Y) acquisition capabilities, and such capabilities are provided in some modern P(Y) code receivers. Direct P(Y) code acquisition is used if the receiver can accurately predict the satellite transmit time so that less time is required to acquire the P(Y) code by direct sequence than to perform a C/A code search and handover. The direct P(Y) code acquisition condition is satisfied if the receiver has previously acquired four or more satellites and its navigation solution has converged. Under certain jamming conditions, it may be impossible to acquire the C/A code but possible to acquire P(Y) code. Then, direct P(Y) code acquisition is essential and can be supported if the navigation state has been transferred to the receiver with sufficient precision and the ephemeris data for all SVs to be acquired are present. The most sensitive navigation state parameter is precise time. Reference [6] describes a multiple correlator/search detector architecture that supports rapid direct P(Y) code acquisition.

In contrast to the P(Y) code signal, the M code signal was designed so that direct acquisition would be the primary means of acquisition, drawing on advances in acquisition algorithms and integrated circuit technology. The $\text{BOC}_s(10,5)$ modulation allows separate acquisition processing on upper and lower sidebands (with processing at the 5.115-MHz spreading code chip rate) and noncoherent integration of the results from the two sidebands, as illustrated in Figure 5.41 [20]. Note that the two sidebands may be selected and processed at the digital IF part of the receiver rather than by two L-band downconverters. Also note that the replica M code does not contain the square wave component. This approach suffers only a fraction of a decibel in performance compared to coherent processing of both sidebands.

Interestingly, when the sideband acquisition processing approach is used, M code signal direct acquisition processing uses approximately half the arithmetic operations and half the storage of Y code signal direct acquisition processing [20]. An integrated circuit based on this processing approach has been designed, built, and tested, displaying the ability to acquire even with relatively large initial time uncertainties in significant levels of jamming [21]. The processing architecture is based on computation of short-time correlations, followed by FFT backend processing for parallel search of multiple frequency values.

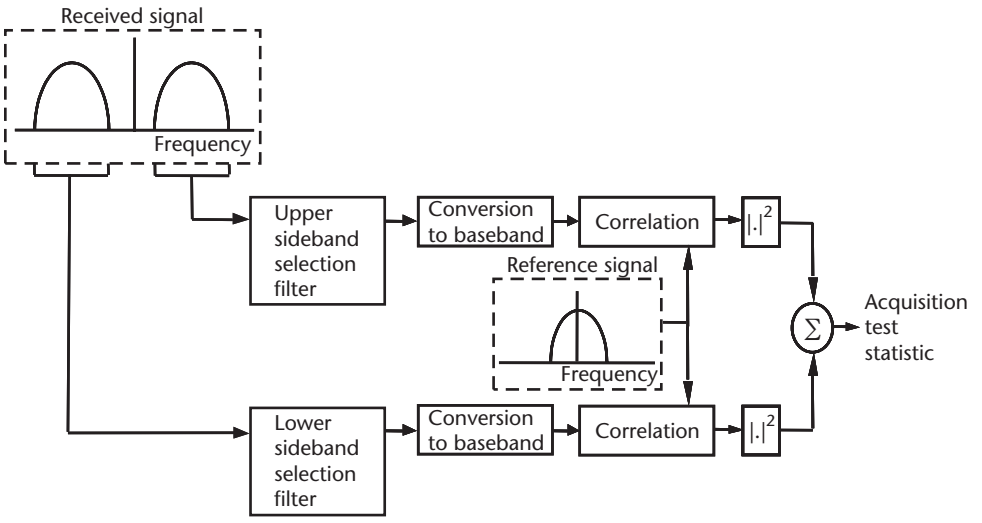


Figure 5.41 Sideband acquisition processing of the M code signal.

Acquisition in jamming requires long integration times. Coherent integration times are limited by data bit boundaries, oscillator stability, and dynamics. They also lead to narrow Doppler bins. Consequently, a large number of noncoherent integrations is employed in jamming. Detection performance is readily predicted using standard theory. The output signal-to-noise plus interference ratio (SNIR) after correlation is given by

$$\rho_o = \frac{T}{L}\frac{0.25C}{N_0 + J_0} \tag{5.52}$$

where T is the coherent integration time used in the correlations, L is the implementation loss expressed as a number greater than or equal to unity, the factor of 0.25 accounts for splitting the received signal power into four distinct segments (upper and lower sidebands, even and odd spreading symbols) in each coherent integration time, and J_0 is the effective power spectral density of the received jamming signal (see Section 6.2).

The detection probability is found using the generalized Marcum Q function. Using the notation $P_N(X, Y)$ [22] as the probability that the random variable with $2N$ degrees of freedom and SNIR of X exceeds threshold value of Y allows the detection probability to be expressed

$$P_d = P_{4N_n}\left(\rho_o, V_t\right) \tag{5.53}$$

where N_n is the number of coherent integrations times used, and V_t is the detection threshold calculated to provide the needed false alarm probability for the given number of noncoherent integrations. The factor of four in the subscript in (5.53) accounts for the fact that the number of complex quantities being noncoherently combined is four times the number of coherent integration times used, reflecting the combination of upper and lower sidebands, and even and odd spreading symbols.

The expressions (5.52) and (5.53) can be used to determine the number of coherent integration times needed to achieve a specific detection probability at a given false alarm probability.

The time (in seconds) to search the initial time uncertainty of $\pm\Delta$ seconds and an initial frequency uncertainty of $\pm\Phi$ Hz is then

$$T_{\text{search}} = N_n \left\lceil \frac{\Delta}{T} \right\rceil \left\lceil \frac{\Phi T}{N_{STC}} \right\rceil \tag{5.54}$$

where T is the coherent integration time, N_{STC} is the number of short-time correlations within the coherent integration time, and $\lceil x \rceil$ is the smallest integer greater than x.

5.9 Sequence of Initial Receiver Operations

The sequence of initial GPS receiver operations depends on the design of the receiver and the past history of the receiver operation. Obviously, the first operation is to select the satellites and then to conduct a search for the selected satellites. There is usually strong emphasis on how fast the receiver will acquire the selected satellites. To determine which satellites are visible and which constellation of visible satellites is the most suitable, three things are needed: (1) an up-to-date almanac; (2) rough estimates of user position and velocity; and (3) an estimate of user GPS time. If any of these parameters are missing or obsolete, the receiver has no choice but to perform what is called a *sky search*, described later. If all are available, then using the user position, the GPS time estimate, and the almanac, the SV positions and LOS Doppler can be computed. Using the estimated user position and the SV positions, the visible SVs can be determined. From the list of visible SVs and the user position, typically the best constellation geometry for good dilution of precision is determined. The best constellation might be selected based on some criteria other than dilution of precision depending on the application. When the constellation has been selected, the search process begins. From the user velocity and the SV LOS Doppler, the total LOS Doppler can be determined. This is used in the Doppler search pattern for the SV. If the approximate time and position are known and the ephemeris data has been obtained during a recent previous operation, the time to first fix can be around 30 seconds for a typical multichannel GPS receiver if the signals are unobstructed. It can require up to 30 seconds just to read the ephemeris data for the SV following signal acquisition. If the ephemeris is not available for the first fix, the almanac data is ordinarily used until the more precise data become available. Reading the almanac data following signal acquisition takes 12.5 minutes. The almanac data, used for SV selection and acquisition, is valid for several days, whereas the ephemeris data, used for navigation, begins to deteriorate after about 3 hours. For the best navigation accuracy, the ephemeris data should be updated each time newer data is available from the space segment.

A critical piece of information is time. Most modern GPS receivers have a built-in timepiece that continues to run even when the set is powered down. They also have nonvolatile memory that stores the last user position, velocity, and time

when the set was powered down, plus the ephemeris data for the last SVs tracked and the most recent almanac. These memory features support fast initial acquisition the next time the GPS receiver is powered up, assuming that the receiver has not been transported hundreds of miles to a new location while powered down or that several days elapse between operation. The stored ephemeris can be used to compute the first fix if it has been 3 hours or less since the receiver was last powered down.

The sky search is actually a bootstrap mode of operation to get the GPS receiver into operation when one or more of the almanac, position/velocity, and time parameters are missing or obsolete. Sky search is a remarkable feature made possible by the GPS C/A code design that permits the receiver to enter into the navigation mode without any a priori knowledge or any external help from the operator. Bootstrapping is virtually impossible for P(Y) code or M code without help from C/A code. The sky search mode requires the receiver to search the sky for all possible PRN codes, in all possible Doppler bins, and for all 1,023 code states of each PRN code until at least four SVs are acquired. This cold start process can require many minutes for the receiver to find visible SVs. The first four SVs found by sky search are unlikely to provide the best geometric performance, but after the almanac, position/velocity, and time information has been restored by using the first four SVs, the navigation process can then determine which SVs are visible and what is the best subset for navigation. For *all-in-view* GPS receivers that track up to 12 SVs simultaneously, good geometry is assured if all SVs in view have been acquired and their measurements incorporated into the navigation solution. This multiple channel feature significantly improves the time to first fix for the sky search mode, since there is significant parallelism in the search process and current almanac data is not required to determine the best geometry.

5.10 Data Demodulation

As noted in Section 5.3.2, when a Costas loop is operating, demodulating the navigation data bits is accomplished by accumulating the I_{PS} samples across one data bit interval and seeing if the result is positive or negative. The resultant bit error rate for the C/A code and P(Y) code signals is:

$$P_b = \frac{1}{2} erfc\left(\sqrt{(C/N_0)/R_b}\right) \tag{5.55}$$

where R_b is the data rate (in bits per second) and

$$erfc(x) = \frac{2}{\sqrt{\pi}} \int_{t=x}^{\infty} e^{-t^2}\, dt \tag{5.56}$$

is the complementary error function.

For the modernized GPS signals (L2C, L5, and the M code) and SBAS signals (discussed in Section 8.6.1.2), a rate half constraint length 7 convolutional code is employed for more robust data demodulation. With soft decision Viterbi decoding, the bit error rate (BER) may be tightly upper bounded for values of interest using [15]:

$$P_b \le \frac{1}{2}\left(36D^{10} + 211D^{12} + 1404D^{14} + 11633D^{16}\right)$$
$$D = \exp\left(-\frac{1}{2R_b}\frac{C}{N_0}\right) \tag{5.57}$$

It should be noted that both of these BER expressions presume that the PLL is perfectly tracking carrier phase without any slips. At low C/N_0's, phase tracking errors degrade the BER. If the C/N_0 is too low or if the signal dynamics are too severe, then as discussed in Section 5.6.1 the PLL is unable to track carrier phase, and data demodulation can no longer be performed.

Before detection of the data bits can commence, bit synchronization must be performed. This function may be performed by accumulating I_{PS} outputs over all possible start/end points and then comparing the power seen in each possibility. For instance, with the C/A code and P(Y) code signals, the I_{PS} outputs are normally based upon 1-ms integrations. There are 20 I_{PS} outputs per data bit and 20 possible bit edge timing possibilities to explore. It should be noted that if the receiver makes an error of 1 ms or more in estimating the location of a data bit edge, the results could be catastrophic. As noted in Section 5.7.1.5, a 300-km pseudorange error would be seen for a 1-ms error. Fortunately, robust detection of the correct bit edge location is not difficult via the previously mentioned technique if the power in the 20-ms accumulated I_{PS} outputs (i.e., the square of the 20-ms accumulations) for each of the 20 bit edge possibilities are compared over many bit durations.

5.11 Special Baseband Functions

Numerous special baseband functions must be implemented in a GPS receiver design, but the following three design examples are among the most important.

5.11.1 Signal-to-Noise Power Ratio Meter

An accurate measure of C/N_0 in each receiver tracking channel is probably the most important mode and quality control parameter in the receiver baseband area. The basic C/N_0 meter design in Figure 5.42 shows that prompt I and Q signals and noise samples from the same channel are integrated and dumped using K samples (typically over the maximum 20-ms predetection integration time). A power envelope is formed from the averaged prompt I and Q signals. This is passed through a lowpass filter (LPF) and output as an estimate of the carrier power for the numerator (even though it also contains noise). The averaged quadraphase components of the noise samples is scaled, then squared and passed through a LPF to form an estimate of the noise power for the denominator. The C/N_0 (ratio-Hz) estimate is formed by dividing the numerator (C) by the denominator (N_0).

5.11.2 Phase Lock Detector with Optimistic and Pessimistic Decisions

Many receiver control decisions are made based on the phase lock detector. Some applications require more certainty of phase lock than others. Figure 5.43 illustrates

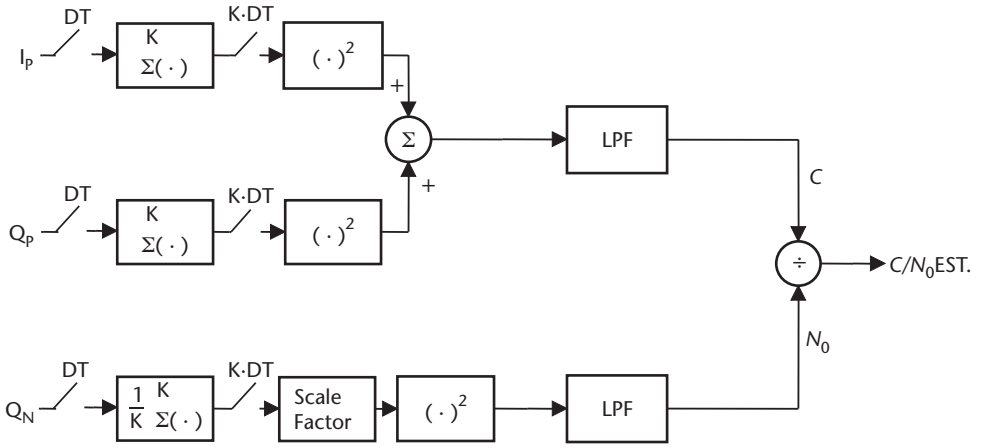


Figure 5.42 Basic carrier-to-noise power ratio meter.

a basic phase lock indicator design, which provides an *optimistic* phase lock indicator that decides quickly and changes its mind slowly but is not as reliable as the *pessimistic* phase lock indicator, which decides slowly and changes its mind quickly.

The phase lock indicator concept is simple: if the loop is in phase lock, then I_{PS} will be maximum and Q_{PS} will be minimum. The phase of the envelope tends to stay near the *I*-axis. As the phase jitter increases in the loop due to noise, dynamic stress, and so forth, the phase of the envelope jitters around the *I*-axis. Eventually, the jitter will reach a level where a cycle will be slipped or complete loss of phase lock occurs.

In Figure 5.43 the absolute values of the prompt *I* and *Q* signals and passes these through a LPF. The in-phase filtered result, I_P, is divided by a scale factor, K_2, and this result is compared to the unscaled quadraphase filtered result, Q_P. The decision is made that phase lock has been achieved if the scaled average absolute amplitude of I_P is greater than the average absolute amplitude of Q_P. This decision is based on the averaged phase being less than about 15° one sigma.

Observe in Figure 5.43 that the first positive decision results in an optimistic decision. The pessimistic decision that phase lock has been achieved does not occur until several optimistic decisions in a row are made. The first time that the threshold criteria is not met, the pessimistic phase lock indicator is set false. The optimistic

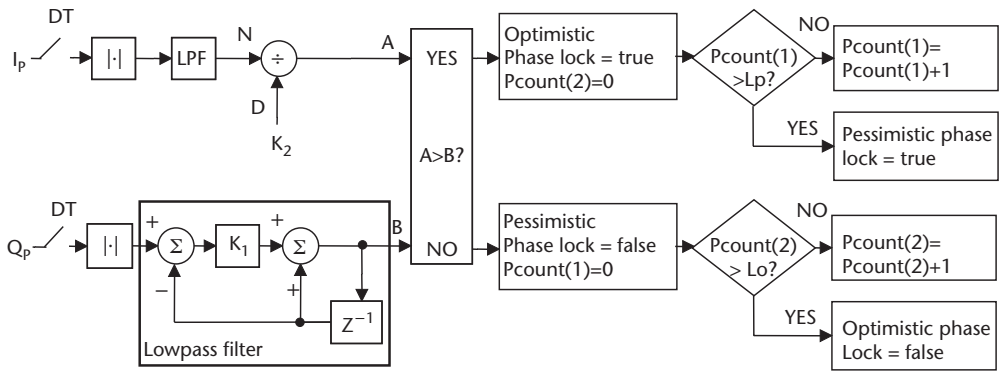


Figure 5.43 Basic phase lock detector with optimistic and pessimistic indicators.

phase lock indicator is not reset until several loss of phase lock comparisons are made in a row. Typical values for the design parameters are: $DT = 20$ ms, $K_1 = 0.0247$, $K_2 = 1.5$, $L_P = 50$, and $L_0 = 240$.

5.11.3 False Frequency Lock and False Phase Lock Detector

False frequency lock can occur in FLL. This can be detected when the DLL velocity state does not match the FLL velocity state. Since both exist, only a comparison check is necessary in FLL to correct the FLL velocity state.

False phase lock can occur in PLL operation when the phase lock indicator declares phase lock but the PLL replica frequency state is incorrect. The incorrect frequency is typically some multiple of 25 Hz for the C/A and P(Y) code signals. The FLL-assisted PLL loop design ordinarily prevents false phase lock, but it is prudent to implement a false phase lock indicator to detect this possible false carrier loop condition. The false phase lock indicator is used only when the phase lock indicator declares that a phase lock condition exists.

Figure 5.44 is a design example of a false phase lock indicator. It performs a frequency discriminator function on a pair of prompt in-phase and quadraphase samples, I_{Pi-1}, Q_{Pi-1}, I_{Pi} and Q_{Pi} formed into the cross product, C, and dot product, D, functions shown as inputs into the detector. Typically, the in-phase and quadraphase samples are collected every 10 ms and the C and D products are formed every $DT = 20$ ms, then applied to the input. These are integrated and dumped for N samples (typically $N = 50$). Typically, at $N \times DT = 1$ second intervals, the four-quadrant arctangent is computed with output E. The absolute value of E represents the change in phase in 1 second, F, which is in units of hertz; this is compared to a threshold, G (for this example, this is 15.5 Hz). If D exceeds G, then the pessimistic phase lock indicator is set to "false" and a 25-Hz correction is applied to the carrier accumulator based on the sign of E.

5.12 Use of Digital Processing

The use of digital processing was as important to the feasibility of the GPS navigation concept as the advancement of reliable space qualified atomic standards. The computational burdens in the GPS space segment, control segment, and user segment are such that digital signal processing is an indispensable asset. Even the earli-

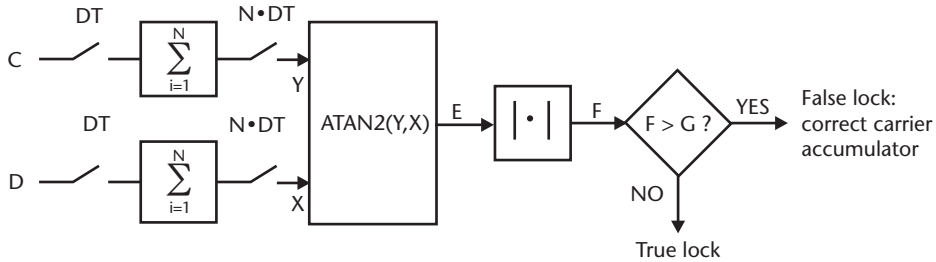


Figure 5.44 False phase lock detector.

est GPS receivers with high analog front end design content utilized digital processing in the receiver baseband processing, receiver control, navigation, and user interface areas. The custom component digital technology known as ASICs, FPGAs, DSPs, and general purpose microprocessors are advancing so rapidly that all GPS receiver manufacturers currently use digital processing at higher levels of the signal processing functions. Also, the processing speed along with built-in floating point processors enables the modern GPS receiver designer to use optimum algorithms rather than approximations. Fortunately, as the feature sizes of ASICs and FPGAs become smaller, their power consumption is reduced and their speed is increased. These advances in technology not only reduce the component count, which reduces cost and power and increases reliability, but also can greatly improve performance. For this reason, outdated analog GPS receiver processing techniques were not discussed.

The partitioning between the microprocessor or DSP and the custom digital components depends on the digital signal processing throughput capability (bit manipulation and computational speed) of the microprocessor or DSP. Eventually DSPs may take over the role of ASICs and FPGAs. It is important to keep in mind that every process performed in the microprocessor is performed in sequential steps, whereas the custom digital components (ASICs or FPGAs) typically perform their processing in parallel. However, the speed of ASICs and FPGAs has increased to the point where digital multiplexing (time sharing of the same function) is now used.

The digital data is sampled data, and there is real time between these samples in which the DSP or microprocessor can process the previous data. The processor is interrupted every time the sampled data is updated. This is called real-time processing, and the real-time processor must have completed all of the tasks within the interrupt time line and have some throughput resources left over for additional processes that are scheduled for completion on longer time lines. Fortunately, the nature of GPS digital signal processing is such that as the processing steps become more complex, there is also more real time allowed between the processes to complete the signal processing. For example, in a C/A code digital GPS receiver, the digital samples containing the spread spectrum signals at IF of all of the visible SVs must be processed at a rate of 2 to 6 MHz (and even faster for modern wideband narrow correlator designs). The replica C/A code generators (one per SV tracked) must operate at 1.023 MHz. At these processing speeds, it is unlikely that the speed-power product of general purpose microprocessors will make them the suitable choice to perform the carrier and code wipeoff functions in the very near future—perhaps never—but specialized DSPs might eventually make a showing. However, after the SV signals have been despread, the processing rate per SV seldom exceeds 1 kHz and typically is of the order of 50 to 200 Hz per SV, which is well within the real-time signal processing capability of a modern microprocessor. The navigation process in a GPS receiver seldom exceeds 1 Hz, even for a high dynamics application.

Since extensive use is made of non-real-time computer modeling and simulations for all aspects of a GPS receiver design, including external aiding, the ideal design environment is one that permits the stripping away of test features not required for real-time operation and the porting of the real-time programs into the actual GPS receiver hardware without modification of the source code [23].

5.13 Considerations for Indoor Applications

The generic receiver design described in the previous sections is a continuously tracking design for outdoor (direct LOS) applications where the C/N_0 may be modestly deteriorated by signal attenuation due to foliage, multipath, antenna gain roll-off, and so on, or by signal interference due to unwanted in-band RF signals. The principles described thus far assume that the carrier and code loops are closed after the search process, and, as long as the C/N_0 remains above the tracking threshold, the receiver will operate satisfactorily. The E-911 requirement for autonomous cellular phone location has motivated the development of highly specialized GPS receivers that can obtain range measurements indoors where the C/N_0 is 20 dB or more lower than the normal situation. These receivers, which are integrated with cellular phone handsets, maximize the use of aiding from communication networks to avoid the need to read the satellite navigation message data or any other tracking mode that requires continuous satellite signal tracking. The objective of these specialized designs is to obtain a reasonable estimate of the user position by extensive averaging of the code correlations with several weak GPS SV signals until there is enough signal processing gain to obtain a set of transmit time measurements from them. The indoor measurements do not have to be nearly as accurate as is expected for the continuously tracking GPS receiver designs. In indoor applications, it is often acceptable for the position determination to only be sufficiently accurate so as to identify the building location. As a result, indoor GPS applications are probably the only operational situation where multipath signals are considered to be friendly signals by the GPS receiver since there is little or no direct GPS signal energy inside most buildings.

There are numerous variations of network-aided GPS techniques. This section outlines some basic principles to obtain the high sensitivities needed for indoor applications. Section 9.4 provides a detailed overview of network-assistance techniques, including industry standards.

One common practice for GPS receivers in cellular handsets is to never close any tracking loops, but rather to dwell on the GPS signals in a controlled (network-aided) search mode long enough to extract enough crude range measurements to provide a snapshot position. The rover receivers utilize communications aiding in order to dwell within the correlation range of the C/A code (about 300m per correlator) and within the carrier Doppler bin in a signal search mode. To accomplish this, they utilize a reference receiver with a clear view of the sky within the cellular network to provide the information needed to replicate their code phase and to synthesize their Doppler estimate during an extended search dwell. If the aiding is such that the SV signals remain within the Doppler bin and the correlation window during the extended search dwell, then sufficient processing gain is achieved to produce valid transmit time measurements. A number of factors make this practical. The user dynamics are small inside buildings, and their distance from the relay (GPS reference) station is not large; therefore, the difference between the user LOS range and Doppler to the SVs is not much different from that of the reference station. Extending the correlator range and the Doppler tolerance is part of the innovative designs of these specialized receivers. Also, the latency of aiding is tolerable to the user receiver since the aiding is applied open loop to the carrier wipeoff process (i.e.,

the total Doppler error from all sources must not exceed the Doppler bin width, and this width is determined by the dwell time for each coherent set of code correlations). Unless data wipeoff information is provided (and this is seldom used), this limits the coherent dwell times to 20 ms or less depending on the sophistication of the aiding. The coherent results are summed together noncoherently for as long as possible—typically less than 600 ms. The greatest limitation to the amount of noncoherent integration that can take place in the rover receiver is the stability of the reference oscillator, which is usually provided by the cellular phone. These reference oscillators are not normally specified with the stringent stability requirements of GPS receivers, but there is more tolerance in this case because there is no attempt to achieve phase lock with the satellite signals.

The methods for obtaining pseudorange measurements in indoor applications are similar to the techniques for acquisition described in Section 5.8. A major difference is that, as described earlier, the communications aiding typically results in a very narrow window in the two-dimensional codephase/Doppler parameter space. This restricted space allows much longer dwells than would be practical for an unaided receiver. As an example of the sensitivities that can be achieved by various levels of coherent and further noncoherent integration, Figure 5.45 presents the probabilities of detecting a GPS signal as a function of $(C/N_0)_{dB}$ for various values of the coherent integration period, T, and the number, K, of successive I and Q outputs

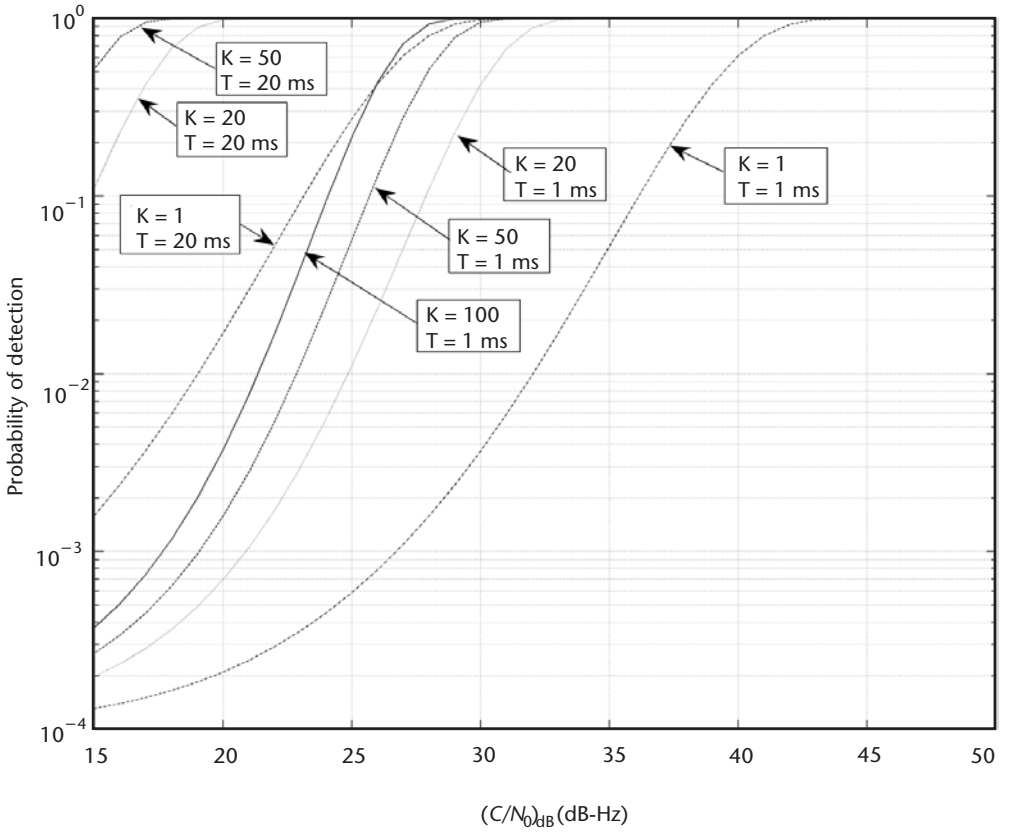


Figure 5.45 Probability of detection as a function of $(C/N_0)_{dB}$ using K noncoherent integrations at T ms dwells for 10^{-4} probability of false alarm.

that are squared and summed. As noted in Section 5.8, over one coherent integration period the envelope $\sqrt{I^2 + Q^2}$ has a Ricean pdf when the signal is present and a Rayleigh distribution when the signal is not present. Further noncoherent integration is typically achieved by summing $I^2 + Q^2$ from each of K successive coherent integration periods. The total resultant dwell time is KT. The sum of $I^2 + Q^2$ over the K periods has a central chi-square pdf with $2K$ degrees of freedom when the signal is not present and a noncentral chi-square pdf with $2K$ degrees of freedom when the signal is present (e.g., [24]). Figure 5.45 assumes that the thresholds for each pair of (T, K) values is set to achieve a false alarm probability of 10^{-4}.

The curves in Figure 5.45 ignore some practical issues such as cross-correlation among C/A code signals, which generally require the thresholds to be inflated somewhat to avoid excessive false alarms. As noted in Section 4.3.4, the cross-correlation levels of the C/A code signals can be as high as –21 dB. This can cause false acquisitions under certain Doppler differences and antenna gain conditions. For example, a user in a building might receive one C/A code signal through a window with very little attenuation and a second C/A signal through several floors of a building with very high attenutation. Discrimination among signals received with great strength differences can be very difficult for a C/A code receiver, resulting in an occasional false acquisition. Fortunately, the unwanted SV signal cannot usually be tracked for long because both the correlation properties and the Doppler change rapidly, resulting in loss of lock and a reacquisition process for the GPS receiver. It is important that the GPS receiver design implement sophisticated C/A code search procedures that avoid cross-correlation and autocorrelation sidelobe acquisitions. If the receiver successfully hands over from C/A code to P(Y) code, then this success provides a built-in assurance of the integrity of the C/A code measurements. However, if the receiver remains in C/A code, then there is always concern that false tracking has occurred under such marginal signal conditions.

It is also important to note that significant implementation losses can result from Doppler frequency errors in the carrier NCO and timing errors in the replica PRN code, particularly for large values of T. Thus, the curves in Figure 5.45 should really be viewed as upper bounds on achievable detection probabilities.

The numerous design tradeoffs and solutions that have been adopted within industry for indoor applications are discussed further in Section 9.4.

5.14 Codeless and Semicodeless Processing

Since only the P(Y) code signal is normally broadcast on L2 by the GPS SVs up through the Block IIR-Ms, this denies direct two-frequency operation to SPS users when AS is activated by the control segment (see Section 4.3.1). Two-frequency carrier-phase measurements are highly desirable for surveying applications (see Section 8.4) and dual-frequency pseudoranges are needed to accurately compensate for ionospheric delays in navigation applications (see Section 7.2.4.1). This has motivated the development of techniques to obtain L2 Y code pseudorange and carrier-phase measurements without the cryptographic knowledge for full access to this signal. These techniques are referred to as either *codeless* or *semicodeless* processing. Codeless techniques only utilize the known 10.23-MHz chip rate of the Y

code signal, and the fact that [2] assures that the same Y code signal is broadcast on both L1 and L2, whereas the semicodeless techniques further exploit a deduced relationship between the Y code and P code.

Since they operate without full knowledge of the Y code signal, the codeless and semicodeless designs operate at significantly reduced SNRs, which requires the tracking loop bandwidths to be extremely narrow. This, in turn, reduces their ability to operate in a high dynamic environment without aiding. Fortunately, robust aiding is generally available from tracking loops operating upon the C/A code signal. Typical codeless and semicodeless receiver designs use L1 C/A carrier tracking loops to effectively remove the LOS dynamics from the L1 and L2 Y code signals, and then extract the L1-L2 differential measurements by some variation of a signal squaring technique that does not require full knowledge of the replica code. Codeless techniques effectively treat the Y code PRN as 10.23-Mbps data, which can be removed through squaring or by cross-correlation of the L1 and L2 signals. Semicodeless techniques exploit some known features of the Y code (e.g., [25]). In addition to the signal-to-noise disadvantage mentioned earlier, codeless receivers suffer from other robustness problems. Although the parallel C/A-code processing provides access to the GPS navigation message, codeless processing of L2 does not allow decoding of the navigation data for the purpose of verifying that the desired SV is being tracked. Also, two SVs with the same Doppler will interfere with each other in the codeless mode; therefore, the scheme fails for this temporary tracking condition. Modern semicodeless receivers, on the other hand, provide relatively robust tracking of the L2 Y code signal with assistance from the L1 C/A code. These concepts will become obsolete when the modernized GPS civil signals become available.

References

[1] Ward, P. W., "An Inside View of Pseudorange and Delta Pseudorange Measurements in a Digital NAVSTAR GPS Receiver," *Proc. of ITC/USA/'81 International Telemetering Conference, GPS-Military and Civil Applications*, San Diego, CA, October 1981.

[2] IS-GPS-200, *NAVSTAR GPS Space Segment/Navigation User Interfaces (Public Release Version)*, ARINC Research Corporation, El Segundo, CA, December 7, 2004.

[3] Ward, P. W., "An Advanced NAVSTAR GPS Multiplex Receiver," *Proc. of IEEE PLANS '80, Position Location and Navigation Symposium,* December 1980.

[4] Kohli, S., "Application of Massively Parallel Signal Processing Architectures to GPS/Inertial Systems," *IEEE PLANS '92 Position Location and Navigation Symposium*, April 1992.

[5] Townsend, B., et al., "Performance Evaluation of the Multipath Estimating Delay Lock Loop," *Proc. of ION National Technical Meeting,* January 1995, pp. 277–283.

[6] Przyjemski, J., E. Balboni, and J. Dowdle, "GPS Anti-Jam Enhancement Techniques," *Proc. of ION 49th Annual Meeting*, June 1993, pp. 41–50.

[7] Ward, P., "Performance Comparisons Between FLL, PLL and a Novel FLL-Assisted-PLL Carrier Tracking Loop Under RF Interference Conditions," *Proceedings of the 11th International Technical Meeting of The Satellite Division of The Institute of Navigation*, Nashville, TN, September 1998, pp. 783–795.

[8] Stephens, S. A., and J. C. Thomas, "Controlled-Root Formulation for Digital Phase-Locked Loops," *IEEE Trans. on Aerospace and Electronic Systems*, January 1995.

[9] Ward, P., "Using a GPS Receiver Monte Carlo Simulator to Predict RF Interference Performance," *Proc. of 10th International Technical Meeting of The Satellite Division of The Institute of Navigation*, Kansas City, MO, September 1997, pp. 1473–1482.

[10] Fuchser, T. D., "Oscillator Stability for Carrier Phase Lock," *Internal Memorandum G(S)-60233,* Texas Instruments Incorporated, February 6, 1976.

[11] Betz, J. W., and K. R. Kolodziejski, "Generalized Theory of GPS Code-Tracking Accuracy with an Early-Late Discriminator," 2000, unpublished.

[12] Betz, J. W., and K. R. Kolodziejski, "Extended Theory of Early-Late Code Tracking for a Bandlimited GPS Receiver," *NAVIGATION: Journal of The Institute of Navigation*, Vol. 47, No. 3, Fall 2000, pp. 211–226.

[13] Betz, J. W., "Design and Performance of Code Tracking for the GPS M Code Signal," *Proc. of 13th International Technical Meeting of The Satellite Division of The Institute of Navigation*, Salt Lake City, UT, September 2000, pp. 2140–2150.

[14] Ward, P., "The Natural Measurements of a GPS Receiver," *Proc. of The Institute of Navigation 51st Annual Meeting,* Colorado Springs, CO, June 5–7, 1995, pp. 67–85.

[15] Simon, M. K., et al., *Spread Spectrum Communications Handbook*, rev. ed., New York: McGraw-Hill, 1994, pp. 751–900.

[16] Scott, L., A. Jovancevic, and S. Ganguly, "Rapid Signal Acquisition Techniques for Civilian and Military User Equipments Using DSP Based FFT Processing," *Proc. of 14th International Technical Meeting of The Satellite Division of The Institute of Navigation*, Salt Lake City, UT, September 2001, pp. 2418–2427.

[17] Ward, P., "GPS Receiver Search Techniques," *Proc. of IEEE PLANS '96*, Atlanta, GA, April 1996.

[18] Polydoros, A., "On the Synchronization Aspects of Direct-Sequence Spread Spectrum Systems," Ph.D. dissertation, Department of Electrical Engineering, University of Southern California, August 1982.

[19] Tong, P. S., "A Suboptimum Synchronization Procedure for Pseudo Noise Communication Systems" *Proc. of National Telecommunications Conference*, 1973, pp. 26D-1–26D-5.

[20] Fishman, P., and J. W. Betz, "Predicting Performance of Direct Acquisition for the M Code Signal," *Proc. of ION 2000 National Technical Meeting*, Institute of Navigation, January 2000.

[21] Betz, J. W., J. D. Fite, and P. T. Capozza, "DirAc: An Integrated Circuit for Direct Acquisition of the M-Code Signal," *Proc. of The Institute of Navigation ION GNSS 2004*, Long Beach, CA, September 2004.

[22] Shnidman, D. A., "The Calculation of the Probability of Detection and the Generalized Marcum Q-Function," *IEEE Trans. on Information Theory*, Vol. 35, No. 2, March 1989.

[23] Jovancevic, A., et al., "Open Architecture GPS Receiver," *Proc. of ION 57th Annual Meeting,* Albuquerque, NM, June 2001.

[24] Van Dierendonck, A. J., "GPS Receivers," in *Global Positioning System: Theory and Applications, Vol. I*, B. Parkinson and J. J. Spilker, Jr., (eds.), Washington, D.C.: American Institute of Aeronautics and Astronautics, 1996.

[25] Woo, K. T., "Optimum Semicodeless Processing of GPS L2," *NAVIGATION: Journal of The Institute of Navigation*, Vol. 47, No. 2, Summer 2000, pp. 82–99.