DIGITAL DEEP-SUBMICRON CMOS FREQUENCY SYNTHESIS FOR RF WIRELESS APPLICATIONS

APPROVED BY SUPERVISORY COMMITTEE:

Poras T. Balsara, Chair

Dinesh Bhatia

William Frensley

William Krenik

Mehrdad Nourani

To my parents: Kazimierz and Irena

and my wife Sunisa

DIGITAL DEEP-SUBMICRON CMOS FREQUENCY SYNTHESIS FOR RF WIRELESS APPLICATIONS

by

Robert Bogdan Staszewski, B.S.E.E., M.S.E.E.

DISSERTATION

Presented to the Faculty of

The University of Texas at Dallas

in Partial Fulfillment

of the Requirements

for the Degree of

DOCTOR OF PHILOSOPHY IN ELECTRICAL ENGINEERING

The University of Texas at Dallas

August, 2002

ACKNOWLEDGEMENTS

My sincere thanks and regards to my advisor Prof. Poras T. Balsara, whose constant support and patient guidance provided a clear path for my research. His original suggestion to work on an all-digital phase-locked loop architecture was, in retrospect, extremely valuable in both my academic and professional careers.

My thanks to Prof. Dinesh Bhatia, Prof. William Frensley and Prof. Mehrdad Nourani for serving on my committee, reading my dissertation and making useful comments. My thanks to Prof. Murat Torlak for originally serving on my committee.

Special thanks to Dr. Bill Krenik for serving on my committee, sponsoring the project at Texas Instruments, and providing very useful feedback throughout from the very days the idea was conceived.

I would like to acknowledge and thank Dr. Dirk Leipold, a physicist and my colleague at Texas Instruments, for many long and fruitful discussions leading to refinements of the digital RF architecture. His orthogonal thinking and expertise in process technology are unparalleled.

I would also like to acknowledge and thank Dr. Chih-Ming Hung for his detailed circuit design and layout supervision of the digitally-controlled oscillator and the RF power amplifier.

Many thanks to Ken Maggio, manager of the RF-CMOS group in Texas Instruments for his feedback and day-to-day activities to make this testchip a success.

I wish to acknowledge valuable technical discussions with Stanley Goldman, a recognized PLL expert at Texas Instruments.

I prepared this dissertation using the typesetting program LaTeX.

I wish to give special thanks to my wife, Sunisa, for her love, support and patience without which I could not have endured.

Finally, I would like to thank Prof. Dale Byrne for encouragement and guidance in my last undergraduate year at UT Dallas when he was my senior project advisor.

DIGITAL DEEP-SUBMICRON CMOS FREQUENCY SYNTHESIS FOR RF WIRELESS APPLICATIONS

Publication No. _____

Robert Bogdan Staszewski, Ph.D.
The University of Texas at Dallas, 2002

Supervising Professor: Poras T. Balsara

Traditional designs of commercial frequency synthesizers for multi-GHz mobile RF wireless applications have almost exclusively employed the use of a charge-pump *phase-locked loop* (PLL), which acts as a *local oscillator* (LO) for both transmitter and receiver. Unfortunately, the circuits and techniques required are extremely analog intensive and utilize a process technology which is incompatible with a digital baseband.

The author's research related to low-power and low-cost radio solutions has led to a novel *all-digital* synthesizer architecture that exploits strong advantages of a deep-submicron digital CMOS process technology as well as advances in digital *very large scale of integration* (VLSI) field. Its underlying theme is to maximize digitally-intensive implementation by

operating in a synchronous phase domain. Chief benefit obtained with this architecture is to allow to integrate the RF front-end with the digital back-end onto a single silicon die.

The presented frequency synthesizer naturally combines the transmitter modulation capability implemented in an all-digital manner. The pulse-shaping transmit filter and a class-E power amplifier are included to demonstrate the use of the proposed synthesizer in a targeted RF application.

The ideas developed in this research project have been implemented in a Texas Instruments' deep-submicron CMOS process and demonstrated in a working silicon of BLUETOOTH transmitter for short-range communications.

TABLE OF CONTENTS

LIST OF FIGURES

LIST OF TABLES

CHAPTER 1

INTRODUCTION

1.1 Motivation

With the explosive growth of the wireless communication industry, research related to communication circuits and architectures has received great attention. The major issues being addressed are low-cost, low-voltage and low-power designs, which achieve necessary performance while being able to be economically manufactured in high volumes. Recently, there has been an additional emphasis on *integration* of heterogeneous parts that constitute a communication device.

1.1.1 CMOS vs. Traditional RF Process Technologies

The *radio frequency* (RF) front-end circuits have been traditionally implemented by Gallium Arsenide (GaAs) MESFET, Si-bipolar junction transistors (BJTs), III-V heterojunction bipolar junction transistors (HBTs) and Silicon-Germanium (SiGe) HBTs, while the baseband *digital signal processing* (DSP) and analog circuits are being implemented exclusively using CMOS technologies. As the emphasis of wireless applications moves towards the *personal communication systems* (PCS) and *wireless local area networks* (WLAN) as well as the wireless entertainment electronics, light-weight, small-dimension, low-cost, low-power and a higher level of integration are becoming ever critical. These have spurred

the interests in low cost CMOS technologies. RF wireless systems using CMOS technologies are being intensively investigated mainly due to their low cost, high yield and higher level of integration including baseband circuits.

On the device physics side, there is a fundamental limitation of the bipolar technology that prevents from using it in low-voltage applications in favor of CMOS. Operation at 1 V power supply is difficult to achieve since the *base-emitter voltage* V_{BE} of bipolar transistors lies around 0.7 V and the base-to-collector junction must be reverse biased. By contrast, the MOS transistor can admit a *drain-source voltage* V_{DS} lower than the *gate-source voltage* V_{GS}, which in turn can be reduced by lowering threshold voltage of the MOS device. Therefore, if the saturation voltage is kept at a few hundred mV, there is enough room for some output dynamic range.

1.1.2 Deep-Submicron CMOS

As the feature size of the digital logic CMOS technology becomes smaller, the frequency at which CMOS transistors can operate while delivering acceptable performance becomes higher [1] [2] [3] [4].

In terms of CMOS transistor application for RF circuits, there are several *figure-of-merit* (FOM) parameters: cutoff frequency f_T, maximum oscillation frequency f_{max}, minimum noise figure F_{min}, flicker noise ($1/f$), power-added efficiency PAE and power gain G_A. A conventional deep-submicron CMOS technology designed for logic applications has shown to exhibit useful RF device characteristics [3].

Scaling of CMOS devices increases the $1/f$ flicker noise, which is caused by carrier

trapping near the thin oxide-silicon interface. On the other hand, there is no such junction in bipolar transistors. The flicker noise could be converted into close-in phase noise of *voltage-controlled oscillators* (VCO) and appear at the output of down-conversion mixers. Fortunately, as shown in [1], this noise frequency translation mechanism can be avoided through half-circuit topological symmetry.

Both f_T and f_{min} peak values already exceed 100 GHz for sub-0.1 μm deep-submicron CMOS logic devices and are predicted to double every three years [1]. Despite the continual RF performance lag behind the latest SiGe bipolar processes, at this point, their performance is adequate for wireless communication bands for up to 5 GHz.

This has opened the possibility for CMOS RF circuits that meet the stringent requirements of communication systems. Efforts have been made to show the feasibility of CMOS front-end circuits [5] [6] [7], and the performance has become comparable to those of BJT circuits. The ultimate goal is to take advantage of the CMOS process to implement a *single chip* radio by integrating the RF front-end, *intermediate frequency* (IF) modulation/demodulation circuits and analog baseband signal processing circuits with the digital baseband section.

In this research work, a fundamental property of the digital deep-submicron process is utilized as a *new paradigm*:

> In a deep-submicron CMOS process, time-domain resolution of a digital signal edge transition is superior to voltage resolution of an analog signal.

This is in a clear contrast with the older process technologies, which rely on a large supply voltage (originally 15 V, then 5 V, and finally 3.3 V and 2.5 V) and a stand-alone configura-

tion with few extraneous noise sources in order to achieve a good signal-to-noise ratio and resolution in the voltage domain, often at a cost of long settling time. In a deep-submicron process with its low supply voltage (at and below 1.5 V), relatively high threshold voltage (0.6 V and often higher due to the body effect) the available voltage headroom is quite small. Moreover, massive switching noise of a large digital circuitry around makes it harder to resolve signals in the voltage domain. At the same time, the switching characteristic of a MOS transistor are excellent with rise and fall times on the order of tens of picoseconds.

1.1.3 Digitally-Intensive Approach

Table 1.1. C035 process technology parameters

interconnection material	copper
minimum metal pitch	0.35 μm
transistor nominal voltage	1.5 V
L drawn	0.11 μm
L effective	0.08 μm
gate oxide	29 A
substrate resistivity	$\leq 50\ \Omega \cdot$ cm

A great reduction of the transistor feature size in recently developed deep-submicron CMOS processes shifts the design paradigm towards more digitally-intensive techniques. In a monolithic implementation, the manufacturing cost of a design is measured not in terms of a number of devices used but rather in terms of the occupied silicon area, and is little dependent on the actual circuit complexity. The testing part of the overall cost does indeed depend on the circuit complexity, but a large number of digital gates typically have a higher test coverage and lower testing cost than even a small analog circuit.

Each new digital CMOS process advance occurs roughly 18 months while increasing

the digital gate density by a factor of two (known as the Moore's Law). A typical digital cellular phone on the market today contains over a million transistors. Analog and RF circuits, on the other hand, do not scale down very well. For example, the latest Texas Instruments' CMOS process (C035 [8] summarized in Table 1.1) with 0.08 μm L-effective feature size achieves an astonishing digital gate density of 150K equivalent (2-input NAND) gates per mm^2, which is an order of magnitude greater than with more traditional RF BiCMOS process technologies. An average-size inductor for an integrated LC oscillator occupies about 0.5 mm^2 of silicon area! A low-noise charge pump, or a low-distortion image-reject mixer, both good examples of classical RF transceiver components, occupy roughly about the same area, which could be traded for tens of thousands of digital gates, which is a lot of DSP power! Consequently, there are numerous incentives to look for digital solutions.

Migrating to the digitally-intensive RF front-end architecture could bring forth the following well-known advantages of a conventional digital design flow:

- Fast design turn-around cycle using automated CAD tools (VHDL or Verilog hardware-level description language, synthesis, auto-place and auto-route with timing-driven algorithms, parasitic backannotation and postlayout optimization).
- Much lower parameter variability than with analog circuits.
- Ease of testability.
- Lower silicon area and dissipated power that gets better with each CMOS technology advancement (also called a "process node").
- Excellent chances of first-time silicon success. Commercial analog circuits usually

require several design, layout and fabrication iterations to meet marketing requirements.

1.1.4 System Integration

There is a wide array of opportunities that integration presents. The most straightforward way would be to merge various digital sections into a single silicon die, such as DRAM or flash memory embedded into DSP or controller. More difficult would be integrating the analog baseband with the digital baseband. Care must be taken here to avoid coupling of digital noise into the high-precision analog section, usually through substrate or power/ground supply lines. In addition, the low amount of voltage headroom challenges one to find new circuit and architecture solutions. Integrating the analog baseband into RF transceiver section presents a different set of challenges: The conventional Bi-CMOS RF process is tuned for high-speed operation with a number of available passive components and does not fundamentally stress high precision.

Sensible integration of diverse sections results in a number of advantages:

- Lower total silicon area. In a deep-submicron CMOS design, the silicon area is often bond-pad limited. Consequently, it is beneficial to merge various functions onto a single silicon die to maximize the core to bond-pad ratio.
- Lower component count and thus lower packaging cost.
- Power reduction. There is no need to drive large external inter-chip connections.
- Lower printed-circuit board (PCB) area, thus saving the precious ”real estate.”

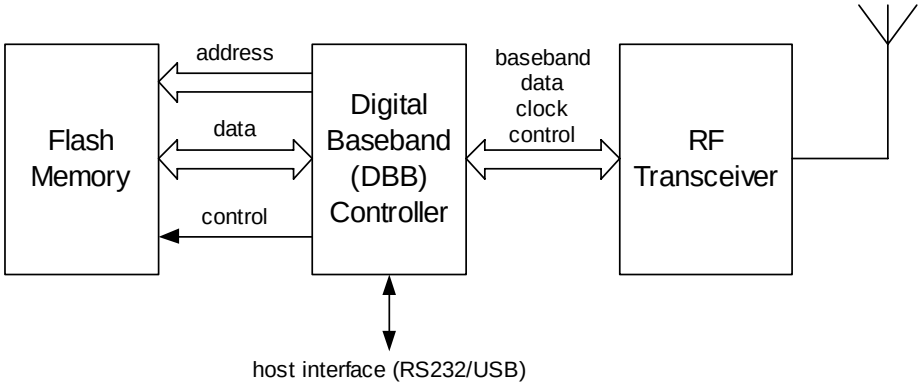

Figure 1.1. Example in ultimate mobile wireless integration: single-chip BLUETOOTH radio

The ultimate goal in mobile wireless integration is a single-chip digital radio as shown in Fig. 1.1. The *digital baseband controller* (DBB) is usually based on a *digital signal processor* (DSP) or the ARM7 microprocessor and is responsible for taking the digital data stream from the RF transceiver and performing any necessary digital signal processing on it to convert the digital data stream into a stream of user data. Examples of the processing performed by the DBB controller may include digital filtering, data encoding and decoding, error detection and correction. It also implements the GSM[1] cellular or BLUETOOTH [9] protocol layer stack which is controlled by a software program stored in a non-volatile flash memory. The RF transceiver module implements the physical layer by converting the information bits to/from the RF waveform. The advanced deep-submicron CMOS process total integration leads to an extremely compact and economic implementation of this sophisticated and highly functional communication system.

[1]GSM originally stood for *Groupe spe'cial mobile* but later renamed to *Global System for Mobile communications* for marketing reasons

1.1.5 System Integration Challenges for Deep-Submicron CMOS

Deep-submicron CMOS processes present new integration opportunities on one hand, but make it extremely difficult to implement traditional analog circuits, on the other. For example, frequency tuning of a low-voltage deep-submicron CMOS oscillator is an extremely challenging task due to its highly nonlinear frequency vs. voltage characteristics and low voltage headroom making it susceptible to the power/ground supply and substrate noise. In such low supply voltage case, not only the dynamic range of the signal suffers but also the noise floor rises, thus causing even more severe degradation of the signal-to-noise ratio. At times, it is possible to find a specific solution, such as utilizing a voltage doubler [10]. Unfortunately, with each CMOS feature size reduction, the supply voltage needs also to be scaled down, which is inevitable in order to avoid breakdown and reliability issues [11].

Moreover, the high degree of integration leads to generation of substantial *digital switching noise* that is coupled through power supply network and substrate into noise sensitive analog circuits [12]. Furthermore, the advanced CMOS processes typically use low resistance P-substrate which is an effective means in combating latchup problems, but exacerbates substrate noise coupling into the analog circuits. This problem only gets worse with scaling down of the supply voltage. Fortunately, there is a serious effort today among major IC fabrication houses to develop CMOS processes with higher resistivity silicon substrates.

Circuits designed to ensure the proper operation of RF amplifiers, filters, mixers, and oscillators depend on circuit techniques that operate best with long-channel, thick-oxide devices with supply voltage of 2.5 V or higher. The process utilized for this work is optimized for short-channel, thin-oxide devices operating as digital switches at only 1.5 V.

In order to address the various deep-submicron RF integration issues, some new and radical system and architectural changes have to be discovered. In this research, alternative approach and architectures for RF front-end are explored. This will allow easy integration of RF section into digital baseband.

1.2 Frequency Synthesizer as an Integral Part of an RF Transceiver

RF synthesizers, specifically, remain one of the most challenging tasks in mobile RF systems because they must meet very stringent requirements of a low-cost, low-power and low-voltage monolithic implementation while meeting the phase noise and switching transient specifications. They are being selected and ranked according to the following set of criteria:

- Phase noise performance – as any analog circuits, oscillators are susceptible to noise, which causes adverse affects in the system performance during receive and transmit.
- Discrete spurious noise performance – unwanted frequency components to appear in the oscillator output spectrum.
- Switching speed – very important in modern communications systems which utilize channel and frequency hopping (see Fig. 1.2) in order to combat various wireless channel impairments (fading, interference, etc.). Since the system switches carrier frequency often (as fast as once every 1.6 ms in BLUETOOTH), a fast switching and stable frequency synthesizer is essential for proper operation. Switching speed is also important in a fixed-channel *time-division multiple access* (TDMA) systems for quick handoff.

- Frequency and tuning bandwidth – the frequency range has to cover the operational band and have enough margin for process-voltage-temperature variations.
- Power consumption – important for battery operated mobile communication units.
- Size – important for mass production deployment.
- Integrateability – utilizing the deep-submicron CMOS process technology in order to integrate with digital baseband.
- Cost – no extra cost added to the process. Requires minimal amount of external components (so called “bill of materials”).
- Portability – ability to transfer the design from one application to another and from one process technology node to the next. An important issue in digital VLSI and for *intellectual property* (IP)-based applications. Designs described in a hardware description language (HDL) are very portable.

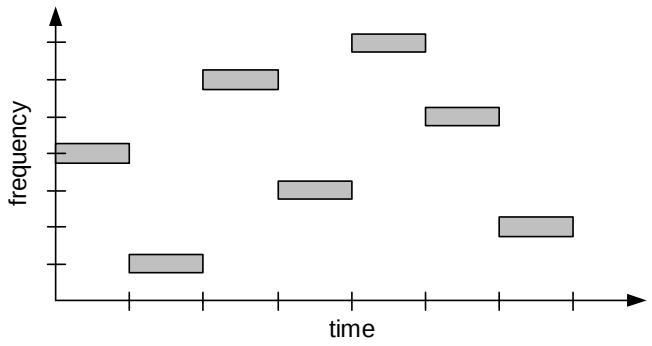

Figure 1.2. Channel hopping of a transmitted signal

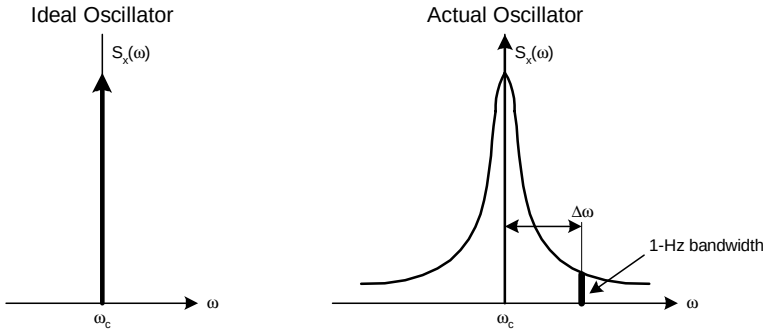

Figure 1.3. Output spectrum of ideal and actual oscillators

1.2.1 Noise in Oscillators

Phase noise is normally characterized in the frequency domain [13] [14]. For an ideal oscillator operating at frequency ω_c, the voltage output can be expressed as $x(t) = A \cdot \cos(\omega_c t + \phi)$, where A is an amplitude and ϕ is an arbitrary, but fixed, phase reference. The power is concentrated at a single frequency ω_c. Consequently, its spectrum $S_x(\omega) = \frac{A^2}{2}\delta(\omega - \omega_c)$ is the shape of a Dirac impulse as depicted in Fig. 1.3. In a practical oscillator, however, both the amplitude and the phase are time-varying fluctuations and the spectrum will exhibit "skirts" around the carrier frequency and spread into nearby frequencies. In most cases, the disturbance in the amplitude is negligible or unimportant, since it could easily be removed by a limiter circuit, and, therefore, only a random deviation of the phase is considered:

$$x(t) = A\cos(\omega_c t + \phi(t)) \tag{1.1}$$

where $\phi(t)$ is a small random excess phase representing variations in the period and is commonly called "phase noise." For a small value of the phase noise fluctuation $|\phi(t)| \ll 1$ rad, Eq. 1.1 could be simplified to

$$x(t) \approx A\cos\omega_c t - A\phi(t)\sin\omega_c t \tag{1.2}$$

which means that the spectrum of $\phi(t)$ is frequency-translated to $\pm\omega_c$.

To quantify this phase noise, we consider a 1-Hz unit bandwidth at an offset of $\Delta\omega$ from the carrier, calculate the noise power in the band, and divide this result by the carrier power [14]. This is the single-sided spectral noise density in units of deciBell carrier per Hertz [dBc/Hz]. By convention, "c" in "dBc" means with "respect to carrier."

$$\mathcal{L}\{\Delta\omega\} = 10 \cdot log\left(\frac{\text{noise power in a 1-Hz bandwidth at frequency } \omega_c + \Delta\omega}{\text{carrier power}}\right) \quad (1.3)$$

The single-sided phase noise of Eq. 1.3 is simply one-half of the phase noise spectrum, which contains both positive and negative frequency components:

$$\mathcal{L}\{\Delta\omega\} = \frac{S_\phi(\Delta\omega)}{2} \quad (1.4)$$

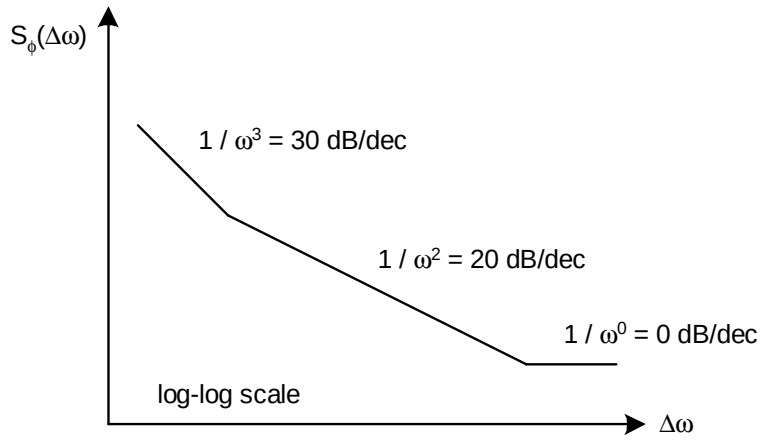

Figure 1.4. Phase noise spectrum of an actual oscillator

Fig. 1.4 shows a typical oscillator phase noise spectrum. In this log-log plot, phase noise normalized to dBc/Hz is plotted against the offset frequency $\Delta\omega$ from the carrier ω_c. The phase noise profile follows the curve shown, where it traverses through $1/\omega^3$, $1/\omega^2$ and

$1/\omega^0$ slope regions. The region $1/\omega^2$ is generally referred to as the thermal noise region, since it is caused by white or uncorrelated timing fluctuations in the period of oscillation. The $1/f$ flicker noise of electronic devices is also substantial for lower offset frequencies. It gets upconverted to the $1/\omega^3$ region. Finally, the $1/\omega^0$ region is the thermal electronic noise added to the clock outside of the oscillator, such as in an output buffer, and which does not affect the oscillator time base.

Let us consider the effect of a single sinusoidal tone in the phase, $\phi(t) = \phi_p \cdot \sin(\omega_m t)$. Eq. 1.1 now becomes

$$x(t) \approx A\cos\omega_c t + A\frac{\phi_p}{2}\left[\cos(\omega_c + \omega_m)t - \cos(\omega_c - \omega_m)t\right] \tag{1.5}$$

Therefore the oscillator output voltage *power spectral density* (PSD) is directly related to the phase noise PSD. Using single-sided spectral densities, we have

$$S_\phi(\omega) = \frac{\phi_p^2}{2} \cdot \delta(\omega - \omega_m) \tag{1.6}$$

$$S_x(\omega) = \frac{A^2}{2}\left[\delta(\omega - \omega_c) + \frac{1}{2}S_\phi(\omega - \omega_c) + \frac{1}{2}S_\phi(\omega_c - \omega)\right] \tag{1.7}$$

Eq. 1.7 shows that the phase noise is directly shifted in frequency towards the carrier and put on the left and right side of the synthesized frequency. It is graphically depicted in Fig. 1.5.

Fig. 1.5 also demonstrates an *undesired* systematic fluctuation in the oscillator phase noise giving rise to a spurious tone. Spurious tones are normally caused by the phase/frequency

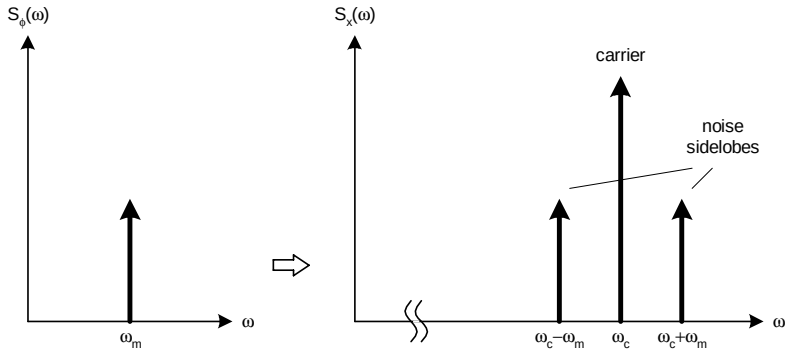

Figure 1.5. Equivalence between PSD and single-sided phase noise

detector and divider circuits in the classical PLL-based synthesizers. In the time domain, the presence of systematic timing fluctuations in an oscillator waveform represents a periodic timing error. In the frequency domain, it manifests as undesired tones in the frequency spectrum. Ideally, an oscillator output spectrum centers at a single frequency with no spurious tones. In reality, the presence of spurious tones causes other frequency components to appear in the oscillator output spectrum. Spurious tones are measured in dBc (or dB referenced to the carrier) at a specific frequency location in the spectrum. It is simply the power difference between the carrier and the spurious tone signals in dB.

At times, it is necessary to relate the PSD of the instantaneous frequency deviation $\Delta\omega(t)$ to the phase noise PSD $S_\phi(\omega)$ and to the single-sided phase noise $\mathcal{L}\{\omega\}$. Since frequency is the derivative of phase, we get

$$S_{\Delta\omega}(\omega) = \omega^2 S_\phi(\omega) = 2\omega^2 \mathcal{L}\{\omega_m\} \tag{1.8}$$

The oscillator perturbations seen in the frequency domain have the underlying cause in the time domain, where the exact time of one period of oscillation will differ from the other. The period has an average value T_0 and a timing error ΔT. The timing error variance

$\sigma_{\Delta T} = \sqrt{\overline{\Delta T^2}}$ is called jitter. A first-order formula is given in [15] that relates jitter to phase noise

$$\mathcal{L}\{\Delta\omega\} = \frac{2\pi \cdot \omega_c}{\Delta\omega^2} \cdot \left(\frac{\sigma_{\Delta T}}{T_0}\right)^2 \tag{1.9}$$

The region modeled by Eq. 1.9 is the $1/\omega^2$ upconverted thermal noise, which is the dominant noise mechanism in oscillators.

1.2.2 Transmitter

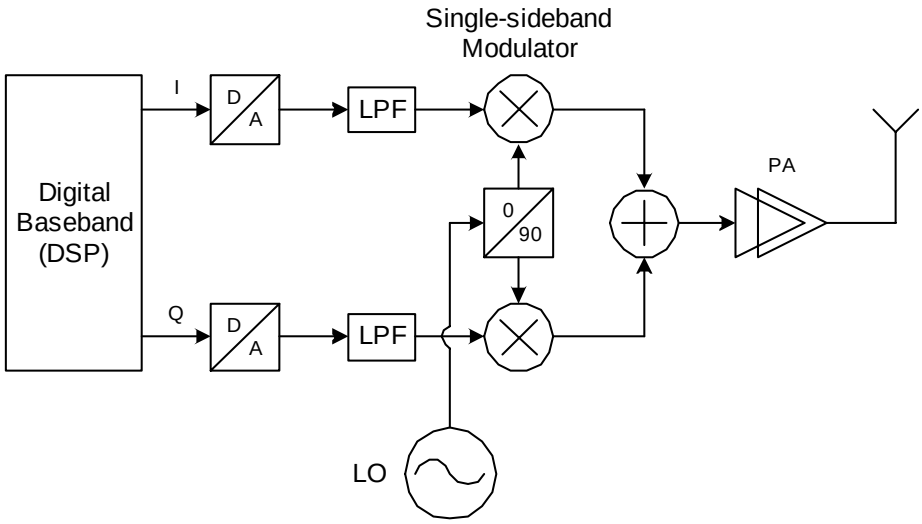

Figure 1.6. Conventional direct up-conversion transmitter

The RF frequency synthesizer is used as a *local oscillator* (LO) in the transmitter to perform frequency translation. Fig. 1.6 shows a conventional direct up-conversion transmitter. The *in-phase* (I) and *quadrature* (Q) pulse-shaped digital baseband signals are converted into analog domain with *digital-to-analog* (D/A) converters. Due to their digital nature, the D/A outputs exhibit strong sampling-time harmonics and switching noise which have to be conditioned with *low-pass filters* (LPF) before being up-converted to the RF carrier by a modulator, which is a critical RF/analog block. The *power amplifier* (PA) is the last stage of

the transmitter path. It performs antenna impedance matching and brings the emitted signal to the required power level. A major weakness of this mixer-based transmitter architecture is that even a small mismatch in phase shift or amplitude gain between the I and Q paths can significantly impair the system performance. Furthermore, because of a certain amount of inherent frequency shift between the modulator input and output (it performs frequency translation by design), the strong power amplifier signal can cause frequency pulling of the oscillator through injection locking. This mechanism finds parasitic paths, such as substrate, power and ground lines as well as electromagnetic radiation to feed strong PA signal into most sensitive parts of the oscillator.

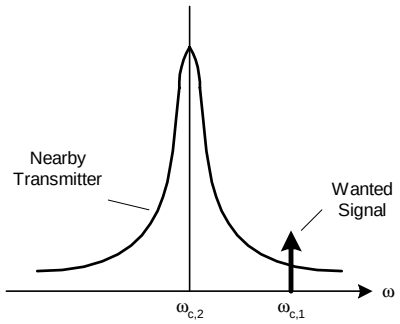

Figure 1.7. The effect of LO phase noise in a transmitter

The phase noise generated by the LO in the transmitter must be minimized due to the following reason. Fig. 1.7 indicates a problem when a noiseless receiver must detect a weak desired signal at frequency $\omega_{c,1}$, while a powerful, nearby transmitter generates a signal at frequency $\omega_{c,2}$ with substantial phase noise. The wanted signal will be corrupted by the phase noise tail of the transmitter. This leads to a challenging noise requirement for the noise skirt of the RF LO. The tough standards for modern wireless communication systems result in very tight specifications on the close-in and far-out phase noise of synthesizers [16].

1.2.3 Receiver

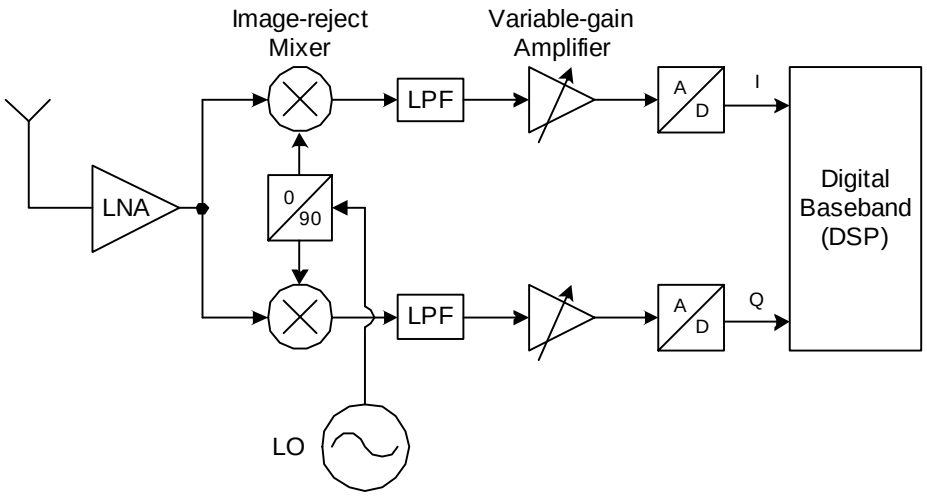

Figure 1.8. Zero-IF receiver topology

The RF frequency synthesizer is used as a *local oscillator* (LO) in the receiver to perform frequency translation and channel selection. Fig. 1.8 shows a modern zero-IF receiver architecture in which the LO is used to down-convert the RF signal directly into baseband without any *intermediate frequency* (IF) stages. Alternatively, due to dc-offset issues plaguing the true zero-IF architecture [17], the conversion could preferably be near-zero-IF. In this case, the IF signal does not get demodulated exactly to dc but a fraction of the signal bandwidth away.

The received RF signal at the antenna output port is immediately strengthen by a *low-noise amplifier* (LNA). It is then down-converted by an image-reject mixer into baseband. The *low-pass filters* (LPF) are used to keep the unwanted frequency components from being fed further in the receive chain. Variable-gain amplifiers bring the signal to the required level before it gets converted into digital form by the *analog-to-digital* (A/D) converter. The digital baseband processes both the *in-phase* (I) and *quadrature* (Q) parts and performs

detection and other DSP processing.

This direct down-conversion architecture is considered to be the most suited for on-chip integration since it does not rely on external high-Q tuning circuits for IF filtering. Unfortunately, it suffers from DC-offset problems due to LO leakage as well as self-mixing and various DSP-based cancellation methods must be used, which is an area of active research.

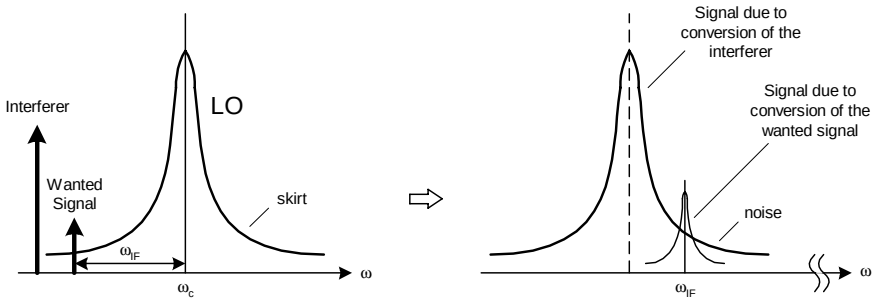

Figure 1.9. The effect of LO phase noise in a receiver

The requirement of the BLUETOOTH receiver is that a -70 dBm signal power in the 1 MHz bandwidth must be detected with a *bit error rate* (BER) of less than 10^{-3}. The expected *signal-to-noise ratio* (SNR) at the non-coherent detector should be no less than 17 dB. The system must also be able to reject -10 dBm out-of-band single-tone blocking signals with a signal level of -67 dBm. The system must also pass an intermodulation test with a -39 dBm single-tone signal close to a -39 dBm BLUETOOTH-modulated interferer when the signal level is at -64 dBm. These impose stringent requirements on the RF LO. Figure 1.9 illustrates the effect of LO purity in a receiver. Since the wanted signal power is small and the interferer is large, the noise skirt of the LO must be low so that the wanted signal would not be corrupted after downconversion.

1.3 Towards the Direct Transmit Modulation

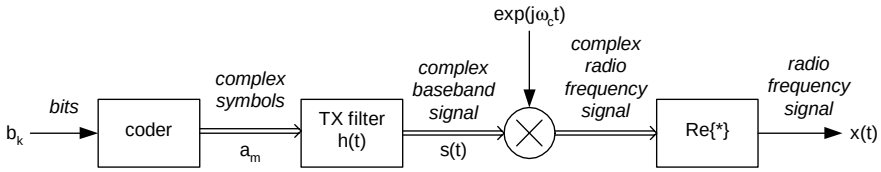

Figure 1.10. PAM modulation with complex signals

Fig. 1.10 illustrates a general block diagram of a transmit *pulse amplitude modulation* (PAM) using complex signals. It mathematically describes an arbitrary modulation process. The incoming bit stream b_k is fed to a coder, which converts the ”0” or ”1” digital bits into a stream of symbols a_m. A symbol assumes values from an alphabet. Since the coder may map multiple bits into a single data symbol, a distinction must be made between the symbol rate and the bit rate. In BLUETOOTH and GSM there is a one-to-one correspondence between the bits and symbols: $\{0, 1\} \rightarrow \{-1, +1\}$. More advanced encoding schemes, such as QPSK, for example, pack two bits into a symbol.

Symbols are applied to a transmit filter, which normally produces a continuous-time signal for transmission over the continuous-time channel. The main purpose of employing the baseband transmit filter is to properly and efficiently constrain the bandwidth occupied by the modulated RF spectrum. When rectangular pulses are passed through a bandlimited channel, the pulses will spread in time, and the pulse for each symbol will smear into the time intervals of succeeding symbols [18]. This causes *intersymbol interference* (ISI) and leads to increased probability of the receiver making an error in detecting a symbol. Out-of-band radiation in the adjacent channel in a mobile system should generally be 40 dB to 80 dB below that in the desired passband. Since it is difficult to directly manipulate the

transmitter spectrum at RF frequencies, spectral shaping is done through baseband.

The impulse response $h(t)$ of the transmit filter is called the pulse shape and could be raised-cosine or Gaussian. The raised-cosine rolloff filter belongs to the class of filters which satisfy the Nyquist criterion of no ISI at sampling instances. Gaussian filters, on the other hand, have a smooth transfer function but do not satisfy the Nyquist criterion and allow for a certain amount of ISI at zero-crossings. However, they can employ power-efficient non-linear amplifiers and are commonly used with frequency modulated signals.

In modern implementations, the pulse shape is oversampled by a "chip" clock, which here is an integer multiple of the symbol clock, and represented digitally throughout the pulse filtering process, even though the filter output $s(t)$ is usually, in the end, brought back to the continuous-time domain by performing a digital-to-analog conversion and subsequent low-pass filtering.

The digital baseband data bits b_k are synchronous to the baseband clock, whereas the digital filter output samples are synchronous to the "chip" clock, which is conventionally a multiple of the data rate.

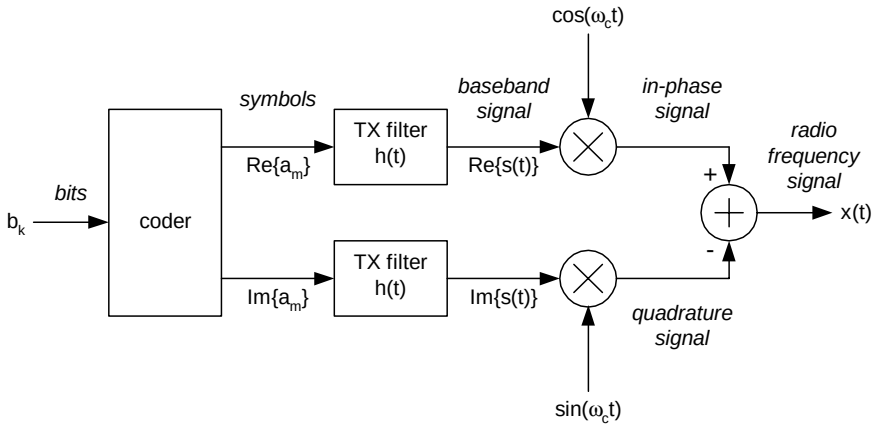

Figure 1.11. PAM modulation with I and Q baseband signals

Complex signal representation requires two physical wires that carry both real-valued parts of a complex number. Fig. 1.11 shows a block diagram of a PAM transmit modulation using *in-phase* (I) and *quadrature* (Q) signals that represents a natural progression towards a more physically-realizable representation. This realization is the basis for the conventional transmit modulator introduced in Sec. 1.2 and can handle a wide range of modulation schemes. However, its I/Q imbalance and carrier feedthrough usually leads to poor sideband suppression.

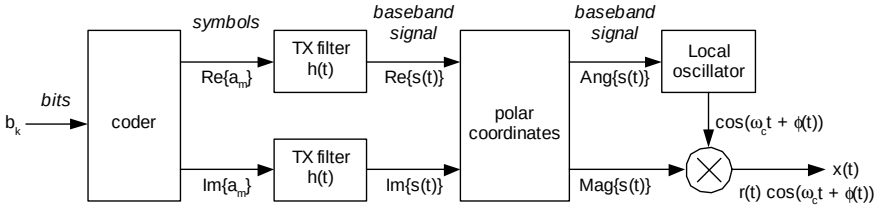

Figure 1.12. PAM modulation with a direct phase and amplitude modulation

Fig. 1.12 shows a block diagram of a PAM transmit modulation using a polar alternative in a form of direct amplitude and phase modulation. The direct phase modulation is conventionally performed by modulating the oscillator tuning input in a feed-forward manner with a possible PLL loop compensation method as shown in [19] and [20]. The direct amplitude modulation might be performed by a conventional method of regulating the supply voltage to a saturation-mode power amplifier. The PAM polar method is clearly the best choice for digital integration of mobile RF transceivers because it does not use the traditional RF/analog-intensive up-conversion mixer of Sec. 1.2.

Constant-envelope transmit modulation schemes such as *gaussian frequency shift keying* (GFSK), which is used in BLUETOOTH and GSM (acronym for Global System for

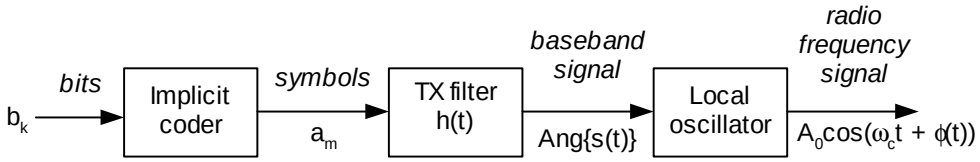

Figure 1.13. GFSK modulation as a PAM modulation with constant envelope

Mobile Communications)[2] standards, permit the use of nonlinear power amplifiers that have a better power efficiency than their linear counterparts. This is important in keeping the total system power consumption at a minimum since the RF power amplifier constitutes a significant portion of the overall power budget. With constant-envelope modulation, the dynamic amplitude control is not required and only a slow-varying output power regulation might need to be implemented. Fig. 1.13 shows the GFSK modulation as being a special case of Fig.1.12 with a constraint of constant envelope. This architecture is selected as the preferred embodiment for the implemented BLUETOOTH transmitter.

The RF output of an angle (i.e., phase or frequency) modulated system can be expressed as

$$s(t) = A_c \cos(\omega_c t + \phi(t)) \tag{1.10}$$

where A_c is a carrier amplitude, ω_c is an angular carrier frequency [rad/s] and $\phi(t)$ is a modulating phase [rad]. In GFSK, the information bearing quantity is frequency, i.e., the instantaneous frequency $f(t)$ of the carrier signal is proportional to the modulating signal $y(k)$

$$f(t) = k_f \cdot y(t) \tag{1.11}$$

where k_f is the *frequency modulation* (FM) proportionality constant. Since $\phi(t)$ phase is

[2]GSM actually uses *gaussian minimum shift keying* (GMSK) which is a special case of GFSK

an integral of f frequency

$$\phi(t) = 2\pi \int_{-\infty}^{t} f(\tau) d\tau \tag{1.12}$$

Eq. 1.10 could be re-written as

$$s(t) = A_c \cos\left(\omega_c t + 2\pi \int_{-\infty}^{t} f(\tau) d\tau\right) \tag{1.13}$$

$$= A_c \cos\left(\omega_c t + 2\pi \int_{-\infty}^{t} k_f \cdot y(\tau) d\tau\right) \tag{1.14}$$

More complex modulation schemes, such as *quadrature phase shift keying* QPSK and *8 phase shift keying* (8PSK), require full symbol-rate RF amplitude control. The polar PAM method still is likely to be preferred in this case over the I/Q scheme. Here, a stripped-down version of *Envelope Elimination and Restoration* method (without the envelope detection and amplitude limitation parts) could be used, and is based on power-efficient non-linear saturation-mode *power amplifiers* (PA). In this method, the supply voltage of a non-linear PA is adjusted according to the desired amplitude of the output, while the input signal has a constant envelope with a duty cycle not far from 50%. Another saturation-mode PA method could be used: *Linear Amplification with Nonlinear Components* (LINC) adds two constant-envelope PA outputs of the properly phase shifted signals. However, it is more area-intensive and power-inefficient technique than the previous one and is currently used mainly in base stations. A method proposed in this work uses a novel digital pulse sliming circuit, instead.

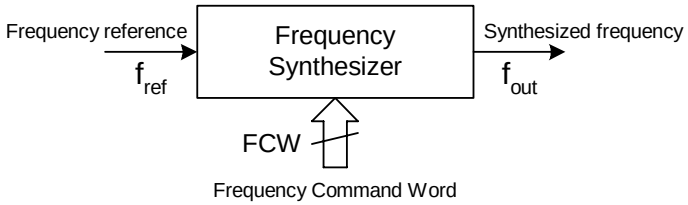

Figure 1.14. Frequency synthesis

1.4 Frequency Synthesis

1.4.1 Definition

The term *frequency synthesizer* generally refers to an active electronic device (Fig. 1.14) that accepts some *frequency reference* (FREF) input signal of a very stable frequency f_{ref} and then generates frequency output as commanded by the *frequency command word* (FCW), whereby the stability, accuracy, and spectral purity of the output correlate with the performance of the input reference. The desired value of the output frequency is an FCW multiple (generally, a real number) of the reference frequency according to the following equation:

$$f_{out} = FCW \cdot f_{ref} \tag{1.15}$$

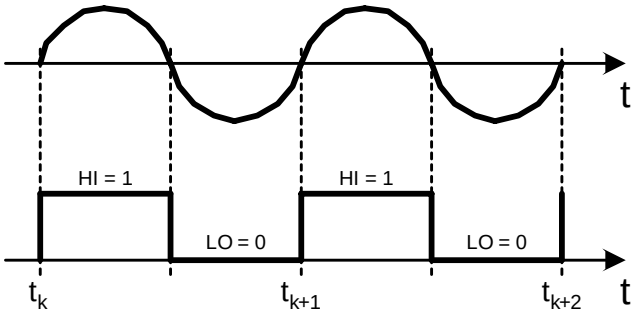

Figure 1.15. Possible digital waveforms of a synthesizer

Interestingly, the above definition does not specify shape of the synthesized output. It

could be a sinusoid or a rectangular signal (Fig. 1.15). The frequency and phase information is preserved in either the continuous-time waveform fit to the ideal sinusoid or in edge transition times, respectively. A clear advantage of the rectangular digital signal is taken here as more useful for digital CMOS process technology.

1.4.2 Frequency Synthesis Techniques

There are three major, conventional frequency synthesis techniques:

- Direct-analog mix/filter/divide,
- Direct-digital,
- Indirect or *phase-locked loop* (PLL), and
- Hybrids – any combination of the above three methods.

Each of these methods has advantages and disadvantages, hence each application requires selection based upon the most acceptable combination of compromises.

Direct-Analog Synthesis

Direct-analog synthesis, also called mix/filter/divide, uses echelons of multiplication, division and other mathematical manipulations to produce the desired new frequency [21]. The process is called ”direct” because the error correction process is avoided, hence the quality of the output correlates directly with the quality of the input. Phase noise is typically excellent because of the direct process and switching speed can be very fast. Unfortunately, the broadband mix/filter/divide synthesizer requires many references, which makes

it extremely expensive. Because of its high cost and high power disadvantages, the direct-analog synthesis method is mainly used in instrumentation and is not practical for mobile communications.

Direct-Digital Synthesis

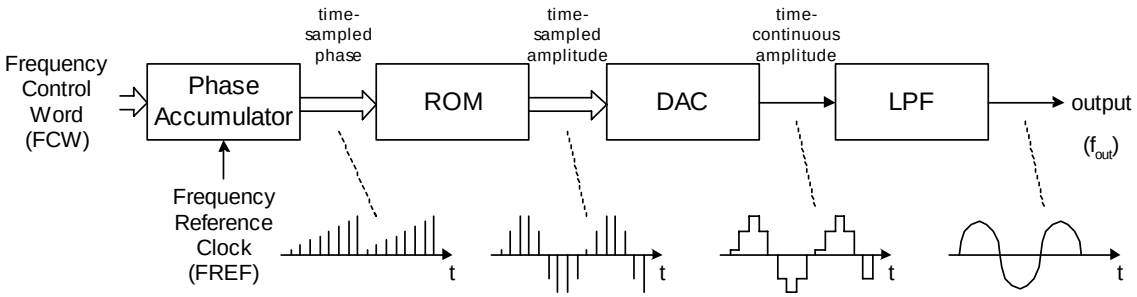

Figure 1.16. Direct digital frequency synthesis (DDFS)

Direct-digital frequency synthesis (DDFS) is the most recently developed frequency synthesis technique dating back from the early 1970s [22]. A DDFS uses logic and memory to digitally construct the desired output signal, and a data conversion device (DAC) to convert it from the digital to the analog domain, as shown in Fig. 1.16. Therefore, the DDFS method of constructing a signal is almost entirely digital, and the precise amplitude, frequency, and phase are known and controlled at all times. For these reasons, the switching speed is considered extremely high, but the power consumption could be excessive at high clock frequencies. The DDFS method is not entirely digital in the true sense of the word since it requires a digital-to-analog converter (DAC) and a low-pass filter (LPF) to attenuate spurious frequencies due to the digital-switching. In addition, a very stable frequency reference clock of at least three times the output frequency is required. Considering this and the fact that the DAC and LPF might be difficult to build and would consume ex-

cessive amount of power at GHz range, the DDFS solution is not acceptable for mobile communications.

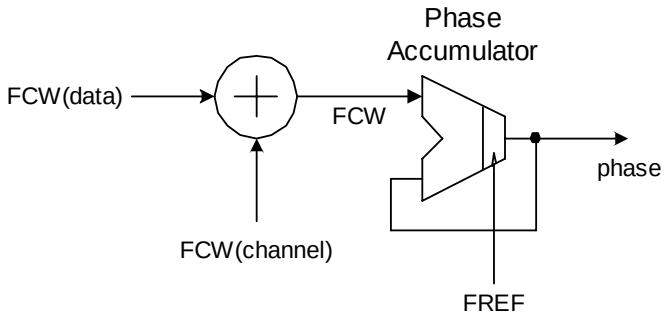

Figure 1.17. Phase accumulator front-end of the DDFS with frequency modulation capability

Due to its digital waveform reconstruction nature, the DDFS technique is best suited for implementing wideband transmit modulation as well as fast channel-hopping schemes [23]. As an example, Fig. 1.17 shows the phase accumulator front-end of the DDFS with an arithmetic adder that combines the FCW components of the selected channel and the frequency-modulating data.

Let the wordlength of the accumulator be W. For a given frequency command word FCW and clocking frequency f_{ref}, the output frequency f_{out} of the synthesizer is given by

$$f_{out} = \frac{f_{ref} \cdot FCW}{2^W} \tag{1.16}$$

and the frequency resolution is

$$\Delta f = \frac{f_{ref}}{2^W} \tag{1.17}$$

Because it is very costly to implement a DDFS at frequencies of interest for wireless communications (multi-GHz range), this technique is directly used so far only in military applications. From yet another perspective, the DDFS method is fundamentally not the

best choice of generating RF signals. As explained above in Sec. 1.4.1, the digital clock of Fig. 1.15 is preferred in a deep-submicron process technology over the sinusoidal signal, of which the DDFS technique concerns itself. The digital *reconstruction* of the entire waveform is simply too wasteful if only the zero-crossings are what is ultimately needed.

Indirect Synthesis

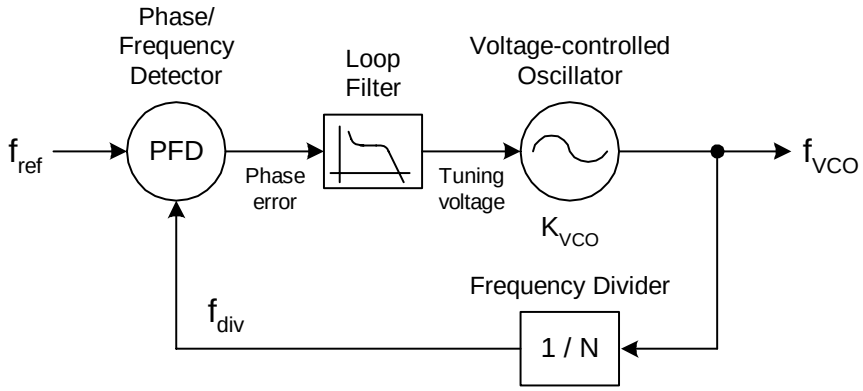

Figure 1.18. Phase-locked loop (PLL)

Indirect synthesis, also called *phase-locked loop* (PLL), compares the output phase of a *voltage-controlled oscillator* (VCO) with a phase of a reference signal f_{ref} as shown in Fig. 1.18. As the output drifts, detected errors produce correction commands to the VCO, which responds in a negative feedback manner. Phase and frequency deviation monitoring occurs in the *phase/frequency detector* (PFD), which adds phase noise close to the carrier, though a PLL can outperform direct synthesis at larger offsets. Fine frequency steps degrade phase noise, and fast switching is difficult to achieve with a PLL design even with the use of aggressive VCO pretuning techniques.

In general, the indirect synthesizer uses a PLL loop and a programmable fractional-N divider which multiplies the stable reference frequency f_{ref}. In the loop, a *loop filter* (LF)

is present so as to suppress spurs produced in the phase detector so that they do not cause unacceptable frequency modulation in the VCO. However, the filter causes the degradation in transients, which limits the switching time. Therefore, the requirements for both the frequency switching time and the suppression of spurs are in conflict.

The classical PLL-based frequency synthesizers are only suitable for narrowband frequency modulation schemes, in which the modulating data rate is well within the PLL loop bandwidth.

Synthesis Based on Hybrid Structure

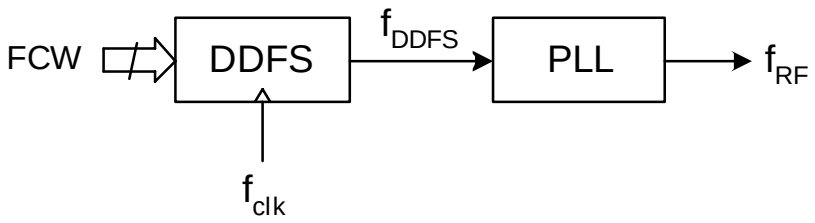

Figure 1.19. Hybrid of DDFS and PLL

In certain applications it is necessary to combine the two (rarely three) major synthesis techniques such that the best features from each basic method are emphasized. Most likely it is the hybrid of DDFS and PLL structures that is found in certain wireless devices. As shown in Fig. 1.19, the wideband modulation and fast channel-hopping capability of the DDFS method, that now operates at lower frequency, could be combined with a frequency multiplication property of a PLL loop that up-converts it to the RF band.

As another example of a hybrid approach, A. Hafez [24] describes a 900 MHz band hybrid synthesizer structure that uses a 1.10–1.85 MHz low-frequency DDFS to generate a stable frequency reference to the main PLL loop. The PLL loop, instead of conventional

digital frequency divider or prescaler, uses subsampler mixing to translate the RF frequency down to the f_{ref}. The frequency resolution of the synthesizer is established by the DDFS and the PLL loop is mainly used as a frequency integer multiplier. Since the DDFS operates at low frequency, its major limitation of high power is not a concern. Unfortunately, the subsampling process introduces an excessive noise.

1.4.3 Frequency Synthesizers for Mobile Communications

A great majority of RF wireless synthesizers for mobile applications are based on a charge-pump PLL structure [25]. Under locked condition, the average output frequency of a PLL bears an exact relationship with the reference input frequency, so the frequency accuracy is extremely fine. Unfortunately, its acquisition time is rather limited since the phase/frequency detector evaluates the frequency difference between the reference and generated clocks by means of the phase difference. In modern wireless applications, the fast acquisition characteristic of the frequency synthesizer is crucial (e.g., in channel hopping). The acquisition time is directly proportional to the initial frequency difference Δf_0 and inversely proportional to the loop bandwidth f_{BW} [26]. Consequently, in order to reduce the acquisition time, small initial frequency difference and wide loop bandwidth are desirable. However, it is not always possible to achieve small Δf_0 at the designing stage due to the process, voltage and temperature (PVT) variations, therefore, a self-calibrating mechanism would be preferred [27]. In addition, an attempt to enhance the acquisition time through wide loop bandwidth increases the contribution of the reference phase noise.

The frequency difference could be measured directly, as in [28], in order to significantly speed up the acquisition to just a few clock cycles independent of the initial frequency

difference. Our proposed architecture is capable of performing a simultaneous phase and frequency estimation and instantaneous correction, therefore, the acquisition performance should be even better. This feature, however, is not emphasized for the targeted wireless application, because its performance is already an order of magnitude better than required.

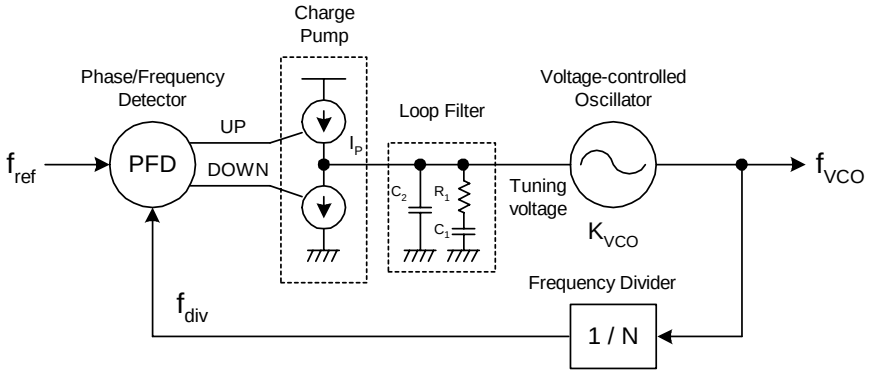

Figure 1.20. Typical charge-pump-based PLL for RF wireless applications

As shown in Fig. 1.20, the PFD estimates the phase difference between the frequency reference f_{ref} input and the divided-by-N VCO clock f_{div} by measuring time between their respective closest edges and generates either UP or DOWN pulse depending on the measured time difference. This signal, in turn, produces a current pulse I_P with the proportional duty cycle in a charge pump block. At the loop filter, this current is converted into a VCO tuning voltage. C_2 is an integrating capacitor and introduces a pole at dc, thus giving rise to the type-II loop. It main task is to suppress the glitch generated by the charge pump on every phase comparison instant. The glitch arises from mismatches between the width of UP and DOWN pulses produces by the PFD as well as charge injection and clock feedthrough mismatches between PMOS and NMOS devices in the charge pump. This periodic glitch will modulate the VCO output frequency, thus giving rise to frequency spurs. The primary sidebands will be located on both sides of the carrier at the compare frequency

offset (equal to the frequency reference).

The double integrator configuration is only marginally stable and it is necessary to stabilize the loop with a zero introduced by R_1 and C_1.

Integer-N Architecture

Conventional PLL has an integer divider ratio of N such that $f_{vco} = N \cdot f_{ref}$. The letter N is used historically in the PLL field to designate its frequency multiplication property and is equivalent to FCW as defined by Eq. 1.15 (page 24). Resolution is equal to the reference frequency which is usually selected to be the same as the channel spacing. Narrow loop bandwidths are undesirable because of long switching times, inadequate suppression of the VCO phase noise, and susceptibility to the supply and substrate noise.

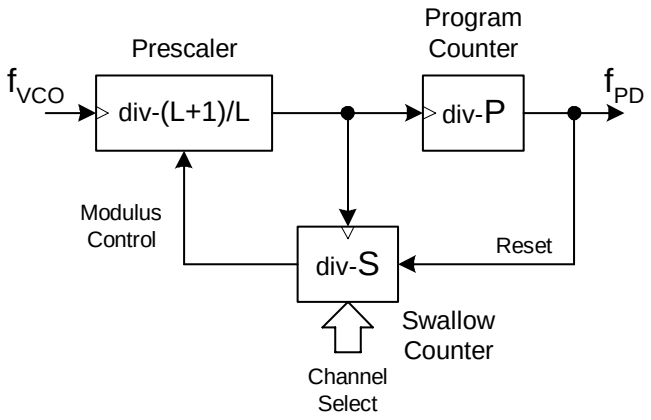

Figure 1.21. Pulse swallow frequency divider

The PLL building blocks for the RF applications require extreme care due to the high frequency of operation, matching and linearity. For example, it is impractical to build a programmable frequency divider of Fig. 1.18 (page 28) directly at the RF frequency. A pulse swallower structure shown in Fig. 1.21 is commonly used instead. It consists of a

high-frequency “prescaler” that either divides by L or $L + 1$, where L is usually a low power-of-two number (8, 16 or 32 are used in practice), and so-called a “program counter” and a “swallow counter.” The program counter always divides the prescaler output by P, whereas the swallow counters divides the prescaler output by a selectable value of S that determines the desired channel. Both counters operate at lower frequency f_{vco}/L. It could be easily shown that if $S < P$, the resulting division ratio is $N = PL + S$. The S counter value controls the channel selection by integer multiple of f_{ref}.

Unfortunately, the charge-pump PLL structure does not easily lend itself to silicon integration. Because of the spur reduction requirements, the loop filter, usually realized as a charge pump as in Fig. 1.20, has large resistors and capacitors, most likely external to the IC chip, in order to achieve a low PLL bandwidth of several kHz. Realizing a monolithic capacitance on the order of a few hundred pF would require a prohibitively large area if implemented as a high-quality *metal-insulator-metal* (MIM) capacitor. Implementing it as a MOS capacitor would take less area, but it would likely be unacceptable because of its high leakage current and nonlinearity.

Another major disadvantage of this analog-intensive synthesizer is lack of portability from one process technology to another.

Fractional-N Architecture

In fractional-N synthesizers, the output frequency can increment by fractions of the reference frequency, allowing the latter to be much greater than the required channel spacing. This allows wide loop filter design at the expense of fractional spurs, resulting with

improved loop dynamics and attenuation of the oscillator-induced noise [13]. The PLL bandwidth is usually set at roughly 10% of the reference frequency to avoid any significant feedthrough of the reference tone and may now span several channels. In response to a change in a frequency control word, the PLL output frequency settles to the programmed value with a time-constant inversely related to the loop bandwidth.

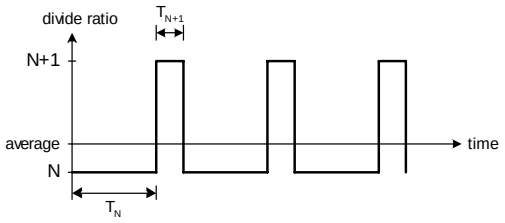

Figure 1.22. Alternating divide ratio of fractional-N PLL

Fractional-N PLL can achieve arbitrarily fine time-averaged frequency division ratio of $(N.f)$ by modulation of the instantaneous integer division ratio of N and $N+1$ (In practice, multi-bit modulus could be used as demonstrated in [29]). Fig. 1.22 reveals the principle in which the integer division is periodically altered from N to $N+1$. The resulting average divide ratio will be increased from N by the duty cycle of the $N+1$ division, as shown by Eq. 1.18.

$$N_{avg} = \frac{N \cdot T_N + (N+1) \cdot T_{N+1}}{T_N + T_{N+1}} = N + \frac{T_{N+1}}{T_N + T_{N+1}} = N + (.f) = (N.f) \qquad (1.18)$$

Phase detector will operate at a frequency of $f_{ref} + (.f/N)f_{ref}$, the phase error of the phase detector causes VCO fractional spurs at multiple of the offset frequency $(.f)f_{ref}$. There are several methods to suppress the fractional spurs. A more conventional method is the analog fractional-N compensation scheme that uses an accumulator and a DAC and is based on the observation that the phase error perturbation is periodic and deterministic

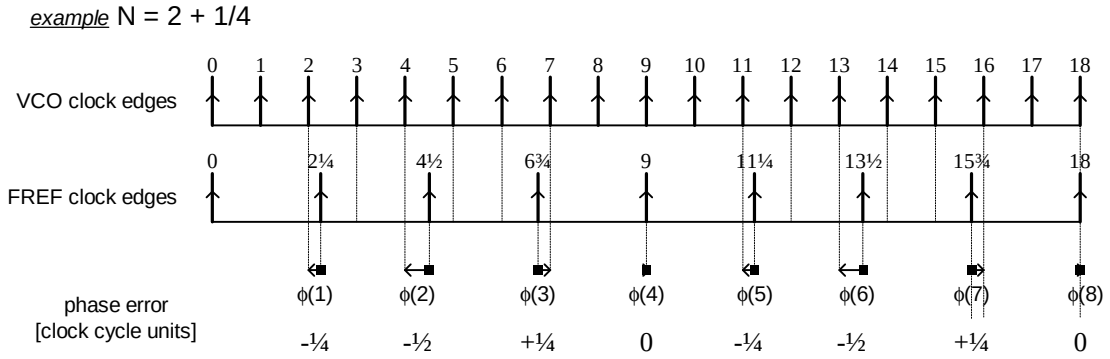

Figure 1.23. Periodic and deterministic phase error in a fractional-N PLL

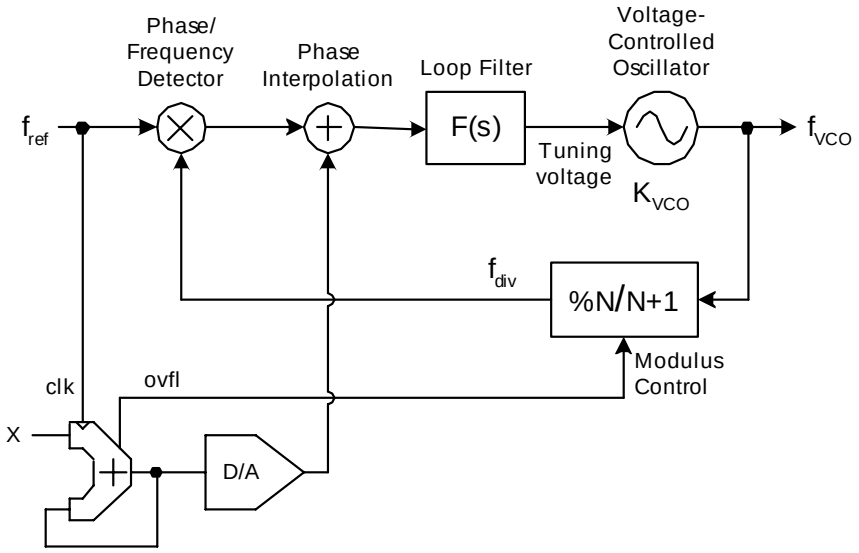

Figure 1.24. Fractional-N PLL incorporating phase interpolation

(Fig. 1.23) and could be canceled out by a tracking circuit. This form of correction is called phase interpolation and is presented in Fig. 1.24. The fractional spurs are reduced to the extent that the phase interpolation signal exactly matches the phase error. In practice, it is difficult to achieve fractional spurs lower than -70 dBc. This requires a precision DAC and carefully designed phase detector and sampler circuitry. Due to its analog complexity, the interpolation scheme is not suitable in most applications.

The second method uses a $\Sigma\Delta$ modulated clock divider as described by B. Miller et al.

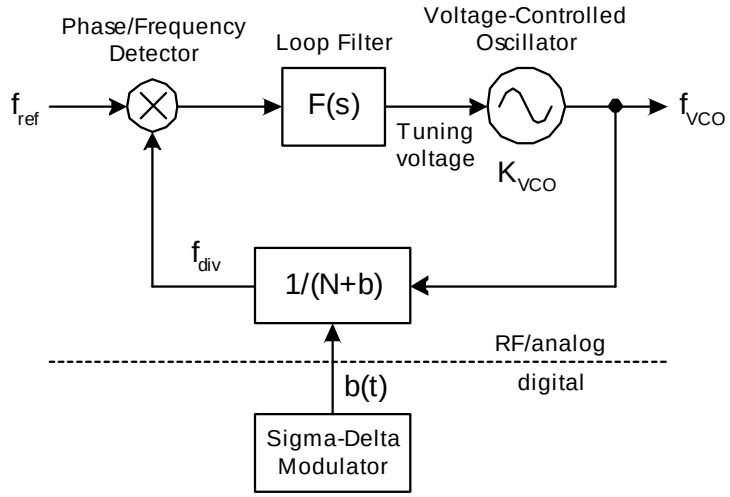

Figure 1.25. Fractional-N synthesizer using a ΣΔ modulated clock divider

[30] and T. Riley et al. [31] and is shown in Fig. 1.25. This solution is more digital in nature since it does not rely on precise analog component matching of the previous technique. It trades the reduction in fractional spurs for the increase in the noise floor.

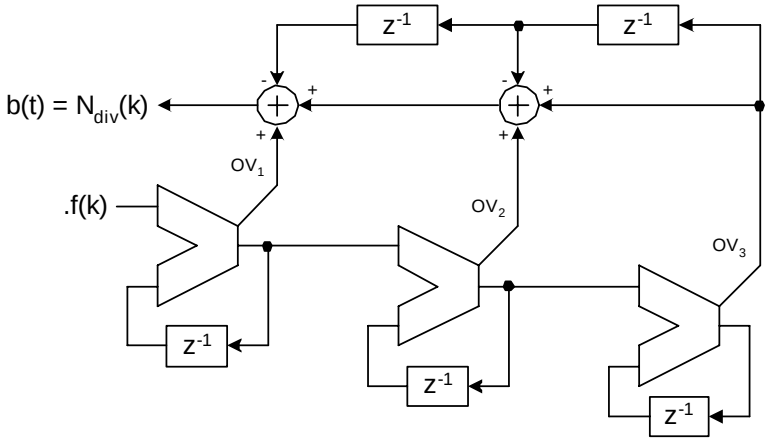

Figure 1.26. MESH-3 ΣΔ digital modulator divider

Fig. 1.26 shows a third order ΣΔ digital modulator divider disclosed in [30]. It uses three accumulator stages in which the storage is performed in the accumulator feedback path. The modulator input is a fractional fixed-point number and its output is a small

integer stream. It is shown in the reference that its transfer function is

$$N_{div}(z) = .f(z) + (1 - z^{-1})^3 E_{q3}(z) \tag{1.19}$$

where E_{q3} is the quantization noise of the third stage, and it equals the output of the third stage accumulator. The first term is the desired fractional frequency, and the second term represents noise due to fractional division.

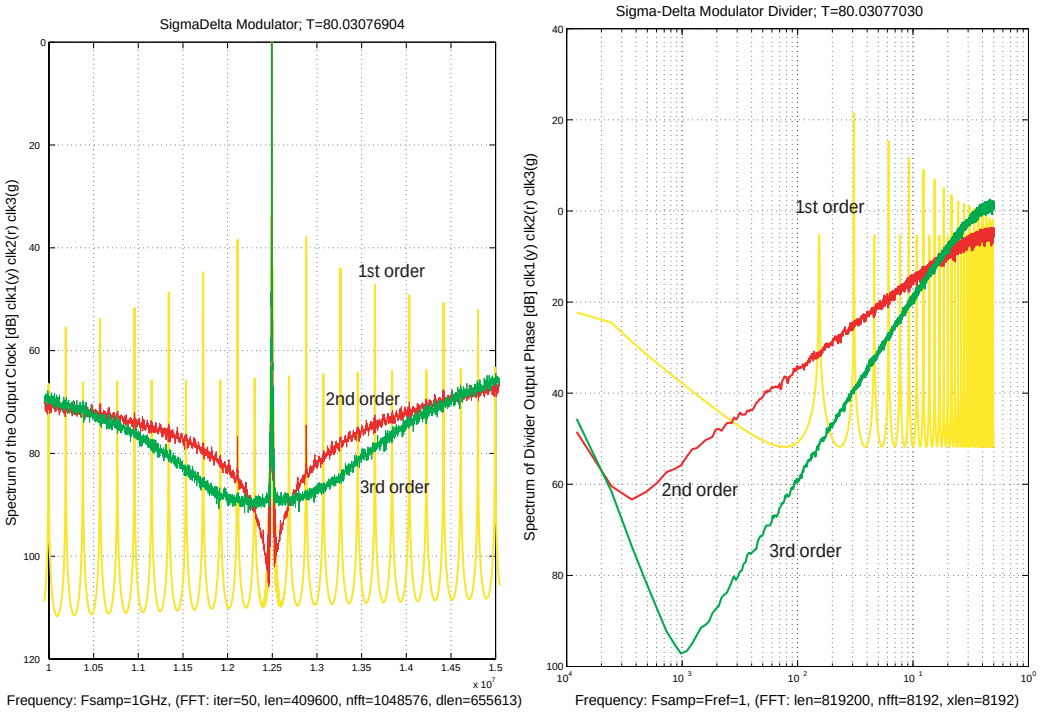

Figure 1.27. $\Sigma\Delta$ divided clock: clock output spectrum (left), phase spectrum (right)

Fig. 1.27 shows quantization noise shaping properties of the first, second and third-order $\Sigma\Delta$ modulation of the clock division ratio. The first order of $\Sigma\Delta$ operation turns out to be equivalent to the conventional, but uncompensated, fractional-N architecture and exhibits systematic division ratio patterns that produce unacceptably large frequency tones. In this case, the quantization noise is concentrated at discrete frequencies rather than spread continuously and merged into the noise floor as in the second and higher $\Sigma\Delta$ orders. The

left plot shows the power spectral density (S_x as defined in Sec. 1.2.1) of the divided clock and is centered around f_{div}. As shown, the third order of $\Sigma\Delta$ dithering introduces enough randomness to completely eliminate any frequency spurs that are clearly shown with the second and first order $\Sigma\Delta$ dithering. The right plot shows the PSD of the divided clock phase (S_ϕ as defined in Sec. 1.2.1) and it demonstrates that the second order $\Sigma\Delta$ dithering performs the high-frequency shaping of the division ratio quantization noise of 20 dB per decade, whereas the third order produces 40 dB per decade. The noise at higher frequencies then gets filtered out in the loop filter. While the phase spectrum S_x is important to determine the noise floor, it is not so reliable in dealing with frequency spurs.

The $\Sigma\Delta$ modulators are usually built as a multistage structure of single-bit modulators [32]. As derived in [30], the quantization noise shaping is related to the number of modulator stages m by the following formula and is given in units of rad^2/Hz.

$$\mathcal{L}\{f\} = \frac{(2\pi)^2}{12 f_{ref}} \left[\frac{f}{f_{ref}/2\pi}\right]^{2(m-1)} \tag{1.20}$$

The fractional-N frequency synthesizer architecture lends itself well to an indirect narrowband frequency modulation which could be implemented entirely in a digital manner. As long as the modulation data rate is lower than the PLL loop bandwidth, the average division ratio N digital command word, which corresponds to a desired channel, could be augmented by the instantaneous value of the modulation frequency deviation. There has been some research to increase the data rate by compensating for the PLL loop high frequency attenuation by boosting the high frequency components of the modulation signal as shown in [33]. After the equalized modulation signal passes through the PLL, the modulation spectrum could be restored to its original form. The digital equalizer could be

embedded in the GFSK filter with little extra overhead. However, the precise loop compensation requirement makes this architecture not very practical for manufacturing.

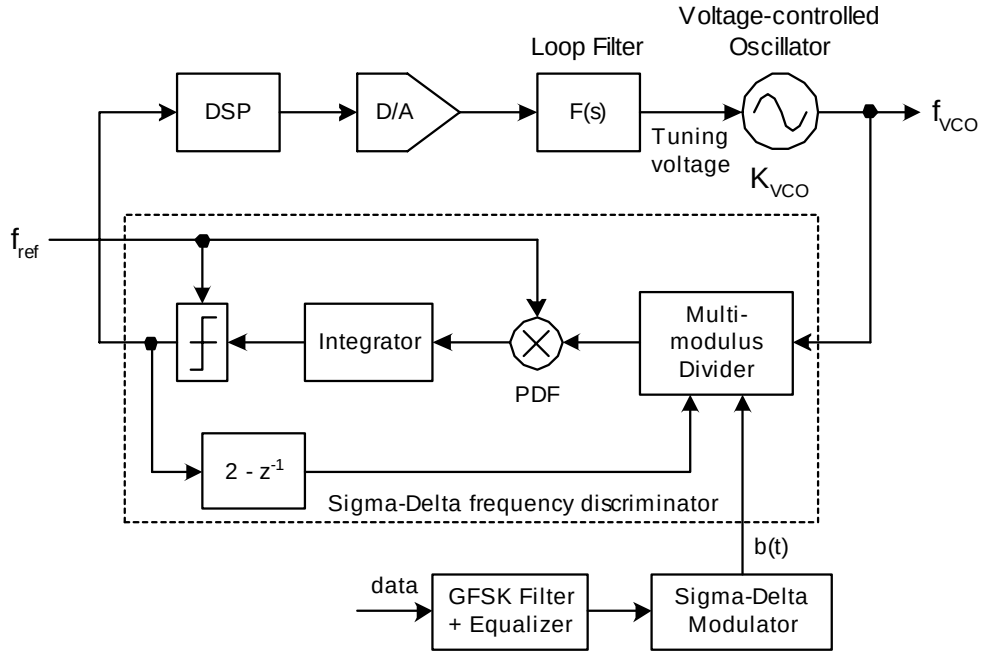

Figure 1.28. Modulating wideband fractional-N synthesizer [34]

The above problem of mismatch between the digital compensation filter and the analog PLL got addressed by W. Bax et al. [34]. The fractional-N PLL is re-architectured there to place a $\Sigma\Delta$ frequency discriminator, whose transfer function is set digitally and well controlled, in the feedback path (Fig. 1.28). As a result, the only analog component left that requires a substantial amount of matching is the VCO.

1.4.4 All-Digital PLL Approach

At this time of writing, there is no reported research on successful low-cost and low-power synthesizer architectures for mass-market RF mobile communications that would employ an *all-digital* approach. There are, however, several reports on all-digital PLL (ADPLL)

synthesizers [35] [36] [37] [38] [39] used for clock recovery applications in wireline communication circuits (Ethernet, asynchronous transfer mode, fiber optics, etc.), as well as clock generation in microprocessors and DSP's. In most ADPLL's, the oscillator is controlled by digital commands, as opposed to the analog tuning voltage. The rest of the controller parts can be designed at the HDL level. However, intensive SPICE simulation still has to be carried out to ensure that target frequency band remains achievable under PVT variations. Specific transistor sizing and layout design of the oscillator have to be "handcrafted" as the design specification or process changes [36].

The above applications have much more relaxed requirements on the phase noise and spurious content as compared with RF wireless applications. In fact, they are *all* based on a ring oscillator structure which inherently features relatively poor phase noise characteristics. Another limitation is their integer-only multiplication ratio N.

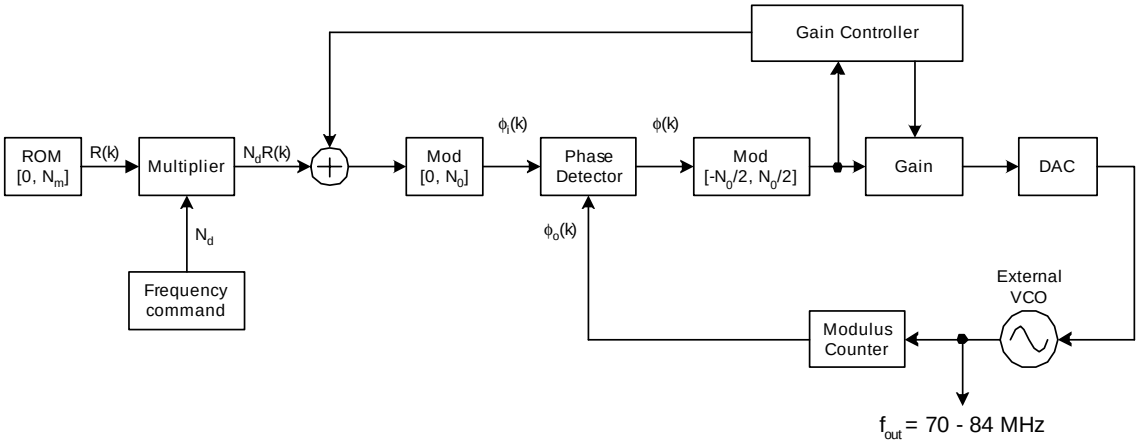

Figure 1.29. Phase-domain PLL based on Kajiwara et al. [40]

A 1992 paper by A. Kajiwara et al. [40] is worth mentioning here. The authors describe a digitally-intensive architecture (shown in Fig. 1.29) that is suitable only for lower-frequency synthesizers. It addresses the chief limitation of the traditional PLL-based fre-

quency synthesizers, namely the slow frequency switching times, which make them less desirable for advanced portable wireless applications that use spread spectrum and frequency hopping techniques. On the other hand, the direct digital synthesizers, whose switching time is extremely fast, cannot be used in a straightforward manner at wireless frequencies. The paper proposes a *digital signal processor* (DSP)-based phase-domain PLL structure that features fast switching times. This architecture, however, has a major limitation of handling only integer-N division ratios.

The authors make a very important observation that since the reference phase and oscillator phase are in a linear form, their difference produced by the phase detector is also linear with no spurs and the loop filter is not needed. This is in contrast with conventional PLL's with correlation detector, and especially based on a charge pump, which generate significant amount of spurs, and which require a filter that degrades the transients and limits the switching time (Fig. 1.20 on page 31). The settling time is only several reference clock cycles for the larger loop bandwidth setting. The acquisition time of this synthesizer is barely influenced by the step frequency, due to the lack of a loop filter which accumulates the phase error.

The metastability problem that arises due to asynchronous timing between the two digital phase detector inputs is handled by converting the terminating path into analog domain through the DAC, which does not require any resampling. For this reason, the architecture is not entirely digital.

1.5 Main Contributions of this Work

This research work deals with a deep-submicron CMOS implementation of a high-speed frequency synthesizer as part of a modern wireless *transceiver* (transmitter/receiver) front-end of a mobile communications channel. It can be deployed as a local oscillator (LO) in both a transmitter path (such as in Fig. 1.6 on page 15) and a receiver path (such as in Fig. 1.8 on page 17).

The synthesizer shall further have a natural capability to perform a frequency/phase modulation that would greatly simplify the transmitter structure by eliminating the analog-intensive I/Q mixing modulator with D/A converters and anti-aliasing filters as presented previously in Fig. 1.6 (page 15). In order to demonstrate salient advantages of the proposed synthesizer architecture, a few blocks will be added that will turn it into a full transmitter. Of course, the LO capability for the receiver is not compromised.

The ideas developed in this research project are commercially implemented in a Texas Instruments' 0.08 μm L-effective CMOS IC chip (codenamed X1743) for the short-range wireless BLUETOOTH connectivity and networking standard. The resulting synthesizer acquisition and phase noise performance appears to be promising enough to be even considered for the widely-popular, but extremely demanding, GSM standard used in cellular phones.

Fig. 1.30 shows the scope of this research within the framework of a BLUETOOTH transmitter. It covers the entire RF front-end that starts with the physical-layer 1 Mbps data bit stream and ends with the RF signal feeding the antenna. This digitally-intensive transmitter was integrated into the same silicon die with the Texas Instruments' C54X

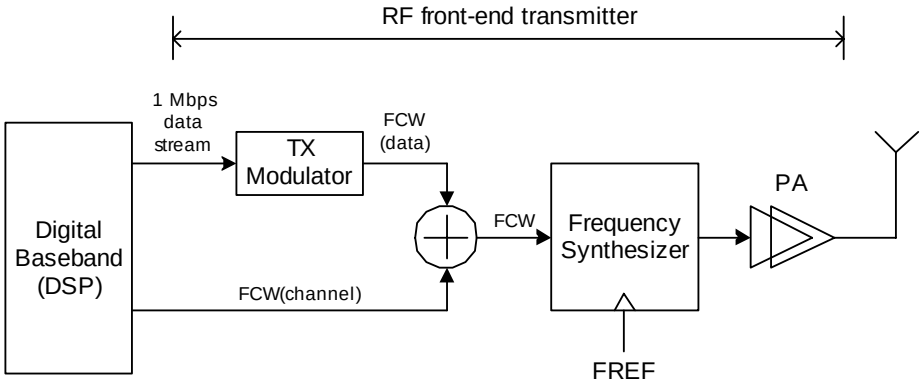

Figure 1.30. Research scope: RF front-end of the synthesizer-based transmitter

DSP processor (used in cellular phones) in order to demonstrate feasibility of the single-chip radio as illustrated in Fig. 1.1 (page 7). It should be noted that the same transmitter architecture could be used for GSM cellular phones with the differences of a slower data rate (about 280 kbps), more stringent RF performance specifications and a required external power amplifier, since at 1.5 V it is not possible to drive signal power on the order of a watt.

Since this is a novel work in the field of the digital RF frequency synthesizers, I am taking a more comprehensive view that would describe the frequency generation process as a digital-to-frequency conversion operation. Various compensation techniques currently used in the *analog-to-digital converter* (ADC) research field, such as the $\Sigma\Delta$ randomization and *dynamic element matching* (DEM), are utilized.

The frequency targeted is the 2.4 GHz *industrial scientific medical* (ISM) band allocated by Federal Communications Commission (FCC) for short-range communications, such as BLUETOOTH [9] and HOMERF [46], and *wireless local area networks* WLAN, such as IEEE 802.11b [47] standard. At such a high frequency, there are extreme challenges for implementing an integrated RF front-end even using conventional BJT technologies.

What is specifically novel about this research, is that the modulating frequency synthesizer is implemented in a digitally-intensive manner in order to take full advantage of the incredible digital gate density (150K equivalent gates per mm^2) and still-untapped capabilities of the latest deep-submicron CMOS processes. However, the 2.4 GHz digital portion, including the *digitally-controlled oscillator* (DCO) and the class-E *power amplifier* (PA), is implemented with some assistance of more traditional analog and RF design techniques. The scope of this research covers the architecture but not the detailed circuit component design techniques of these RF blocks. As a side note, each of these individual RF blocks, or even their portions, has been subject of entire doctoral dissertations.

In order to further improve design productivity and time-to-market, an ASIC design methodology flow was chosen over the full custom approach. A recent paper by D. Chinnery [42] compared a "best practice" ASIC-based Texas Instruments 550 MHz disk drive read channel (in which the author was involved [43] [44] [45]) with best full custom designs and found that an automated ASIC approach can indeed close the gap in speed and area with proper design methodology orchestration and attention to key design factors.

In summary, this research investigation has culminated in a silicon demonstration of a novel RF frequency synthesizer optimized for short-range wireless communication transceivers with the following characteristics and constraints:

1. 2.4 GHz ISM band – FCC unlicensed band providing great potential for consumer application of short-range wireless devices, such as BLUETOOTH, HOMERF or 802.11b WLAN.

2. Low area – mass production deployment in short range wireless-enabled devices and

cellular phones, whose battery size continuously decreases.

3. Low power – battery operated mobile communication units.

4. Fully monolithic implementation with minimal count of external components (internal VCO, no charge pump capacitors, no analog filters).

5. Integrateable – utilizing the deep-submicron CMOS process technology in order to integrate with digital baseband and take advantage of high digital gate density.

6. Fully digital – to take advantage of the incredible digital gate density of the advanced CMOS processes, fast design turnaround time and portability. Minimize analog and RF circuit content.

7. Naturally capable of performing direct *gaussian frequency shift keying* (GFSK) modulation by the transmit data without I/Q image reject mixers, which are analog in nature.

8. Fit well with the receiver and transmitter architecture – acting as an optimized *local oscillator* (LO) capable of performing channel hopping, with fast switching times and low phase noise.

9. Top-down modeling and simulation methodology by designing the entire system in the VHDL high-level hardware description language, as described in [41].

1.6 Organization of the Dissertation

A bottom-up, instead of top-down, approach to present the development of ideas and guide the reader through the design steps was chosen. The organization of the thesis is as follows.

In Chapter 2, the frequency synthesizer design starts from a raw *digitally-controlled oscillator* (DCO), which is a foundation of a novel digital architecture. A time-domain model, which will be extensively used for analysis and VHDL simulation, is introduced.

In Chapter 3, a hierarchical layer of arithmetic abstraction is added over the DCO which makes it easier to operate algorithmically. The main task of this overlay block is to perform DCO calibration and normalization such that the *normalized DCO* (nDCO) transfer function is largely independent from the process and environmental factors. Other improvements, such as increasing the frequency resolution through $\Sigma\Delta$ dithering and dynamic element matching, are also introduced.

In Chapter 4, a phase correction mechanism is built around the normalized DCO oscillator such that the frequency drift or wander performance of the system is as good as the stable external frequency reference.

In Chapter 5, a frequency modulation capability is added to the synthesizer core to enable it to efficiently perform a transmit data modulation. A full transmitter is completed with an addition of two blocks. The first of these blocks is the transmit pulse filter operating in baseband. The second block is a class-E power amplifier that is capable of emitting an RF signal with the power level of several mW. The full transmitter core implementation is then described.

In Chapter 7, the behavioral modeling and simulation methodology used in the design are described. Simulation plots are presented.

In Chapter 8, the IC chip micrograph and experimental results are presented.

CHAPTER 2

DIGITALLY-CONTROLLED OSCILLATOR (DCO)

2.1 Discrete-Time Oscillator

As discussed in Sec. 1.4.1, the phase/frequency information of a *discrete-time* oscillator is contained not in the local waveform fit to the ideal sinusoid but in the significant (rising or falling) edge transition times. If the transition time-stamp represents a positive zero-crossing of an arbitrary-amplitude sinusoid, then this should provide sufficient amount of phase information. However, it is necessary for practical reasons for the digital signal to have the same frequency as the desired output signal, therefore, falling edge transitions are naturally generated half-way between the positive edge transitions.

From the information theory standpoint, this is a very efficient mechanism to represent a signal containing the phase and frequency information. It is quite in alignment with the fundamental strength of the digital deep-submicron CMOS processes stated in Sec. 1.1.2: time-domain resolution is superior to the voltage-domain resolution.

The presented oscillator is a cell with only digital *inputs/outputs* (I/O's) operating in the discrete-time domain, even though the underlying functionality is mainly continuous-time and continuous-amplitude in nature. This is a very important consideration since it stops the analog nature from propagating up in the hierarchy, right at the interface level. The analog design, modeling and simulation constraints of the system are thus vastly simplified. An

analogy should be drawn here to a flip-flop and its fundamental role in the sequential digital circuits, even though its underlying nature and internals are analog.

A *digitally-controlled oscillator* (DCO) is used in this work as a foundation to perform the *digital-to-frequency conversion* (DFC). Its output is a periodic waveform whose frequency f is a certain function of the input *oscillator tuning word* (OTW).

$$f = \mathrm{f}(OTW) \tag{2.1}$$

In general, the $\mathrm{f}(OTW)$ mapping of the digital input to the frequency of oscillation is a nonlinear function. The frequency setting function is not known precisely and varies with the process spread and environmental factors (voltage and temperature). The instantaneous value of the frequency also depends on power/ground and substrate noise, as well as truly random phenomena, such as thermal and flicker noise. The DCO building block accomplishes only the most bare minimum of functionality and it has to be conditioned by normalization circuits, later described in Chapter 3. The DCO also provides necessary means for higher-level blocks to perform self-calibration.

2.2 Digital Control of Oscillating Frequency

Frequency tuning of a low-voltage deep-submicron CMOS oscillator is quite a challenging task due to its highly nonlinear frequency-vs.-voltage characteristics and low voltage headroom. Fig. 2.1 shows normalized representative curves of a MOS varactor capacitance vs. control voltage (C-V curve) for both a traditional CMOS process and a deep-submicron process. Previously, a large linear range of the C-V curve could be exploited for a precise and wide operational control of frequency. With a deep-submicron process, the linear range

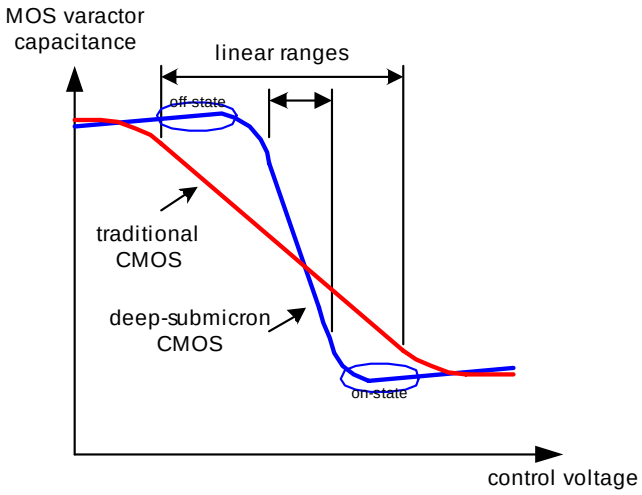

Figure 2.1. Idealized capacitance vs. voltage curves of a MOS varactor

now is very compressed and has undesirable high gain ($K_{VCO} = \Delta f/\Delta V$) which makes the oscillator extremely susceptible to noise and operating point shifts.

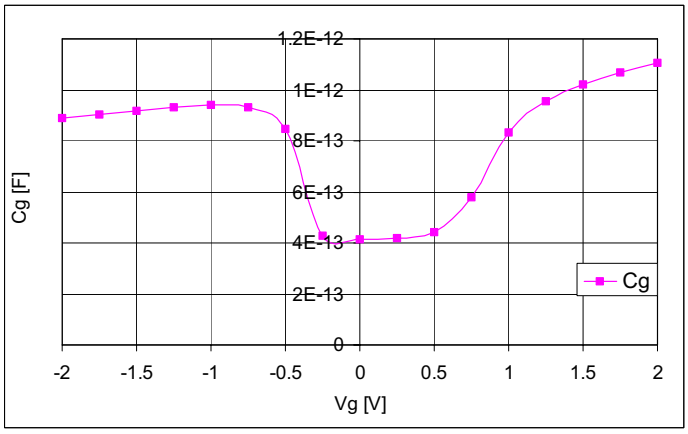

Figure 2.2. Gate capacitance vs. gate voltage of a measured PMOS varactor: C035.a process, PPOLY/NWELL, inversion type, single-contacted gate, L=0.5 μm, W=0.6 μm, N=8 fingers x 12 x 2, freq=2.4 GHz

An example C-V curve of an actual PMOS varactor used in the proposed design for the acquisition mode is shown in Fig. 2.2. The data was measured and de-embedded from a test structure by the TI internal Silicon Modeling Lab (SML). Because of the well isolation properties in this N-well process, the PMOS device (Fig. 2.3) is a better candidate for a var-

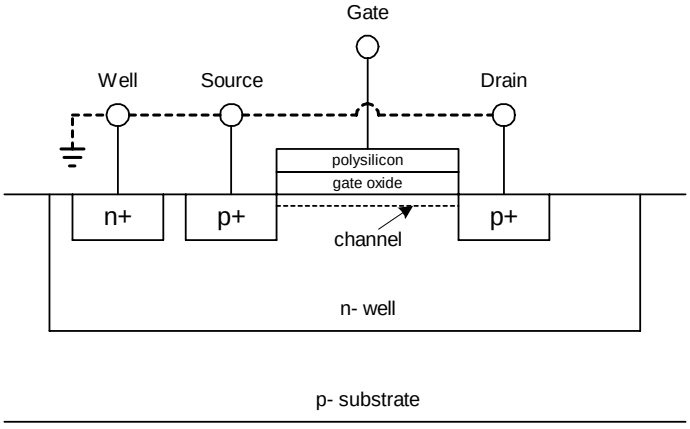

Figure 2.3. Physical structure of a PMOS transistor used as a varactor when the source, drain and well tie-offs are tied to ground

actor. It was experimentally found that in this process the NPOLY/NWELL inversion-type varactor features more distinctly defined operational regions than does the accumulation-type varactor. The device has the following channel length and width dimensions and finger multiplicity: L=0.5 μm, W=0.6 μm, N=8 fingers x 12 x 2. The measurements were performed at the intended frequency of operation of 2.4 GHz. In this configuration, the source, the drain and the well are all tied to ground.

Still referring to Fig. 2.2 and Fig. 2.3, let the gate potential V_g start at +2 V, at the right end of the C-V x-axis. The positively charged gate attracts a large number of electrons, which are the majority carriers of the N-type well. The varactor capacitance is relatively high because this structure behaves like a parallel-plate capacitor with only the silicon oxide dielectric in between. The gate conductor forms one plate of the capacitor and the high concentration of electrons in the N-well forms the second plate. This region of operations is termed the *accumulation* mode. As V_g is lowered, less and less electrons are attracted to the region below the gate and its concentration drops. This causes the effective "bottom" plate to be further separated, thus lowering the gate-to-well capacitance. As soon as the

gate potential is close to zero and enters negative values, the electrons start being repelled causing a depletion region under the gate. Now the structure is in the *depletion* mode. The capacitance gets lower and lower while the depletion region increases. Lowering V_g further below the (negative) threshold level V_t results in holes being attracted to the region under the gate. This gives rise to a conductive layer of holes and this region of operation is called the *inversion* mode. Because the "bottom" plate of the capacitor is just below the gate oxide, the gate capacitance is high again. A strong inversion layer exists at V_G = -2 V.

The slight drop of capacitance in the "flat" strong inversion region in Fig. 2.2 had not been of any practical significance until the advent of deep-submicron CMOS processes. It is due to the depletion layer being created in the gate polysilicon [48] which is less doped and much thinner than in the past.

In this varactor structure, the source, drain and backgate are tied to the same zero potential. This is very similar to the classical MOS capacitor structure, except that the latter does not have the source and drain. The inversion region in the MOS capacitor relies on a process of thermal regeneration of electron and hole pairs, which takes an extremely long amount of time (on the order of μs) to create a channel. Consequently, the channel never manages to get created and destroyed for the RF range of frequencies. In the MOS varactor, on the other hand, the source and drain regions serve as vast and ready reservoirs of electrons, so this problem does not exist.

The proposed solution to control the oscillating frequency could generally be summarized as follows: A method of weighted binary switchable capacitance devices, such as varactors, is proposed. An array of varactors could be switched into a high-capacitance mode or a low-capacitance mode individually by a two-level digital control voltage bus,

thus giving a very coarse step control for the more-significant bits, and less coarse step control for the less-significant bits. In order to achieve a very fine frequency resolution, the LSB bit could possibly be operated in an analog fashion. (A similar idea is used in [27] which employs a hybrid of digital oscillator control for PVT and *analog* control for acquisition and tracking.) However, this requires a DAC and does not fundamentally solve the problem of the nonlinear VCO gain (K_{VCO}) characteristics. A better solution is to dither the LSB digital control bit (or multiple bits), thus controlling its time-averaged value with a finer resolution. Consequently, each varactor could be allowed to stay in only one of the two regions where the capacitance sensitivity is the lowest and the capacitance difference between them is the highest. These two operating regions are shown by the ovals in Fig. 2.1.

It should be noted here that, at the time of writing, there have not been any reports in the literature on the *fully* digital control of oscillators for RF applications. Lack of the fully digital control is a severe impediment for the total integration in a deep-submicron CMOS process for the reasons expressed in Chapter 1. Due to the fact that there are several disclosures on the ring-oscillator-based DCO's for clock recovery and clock generation applications [35] [36] [37] [38] [39], where the frequency resolution and spurious tone level are quite relaxed, it seems that the latter two concerns have been an effective deterrent against digital RF synthesizers for wireless communications. The proposed combination of various circuit and architectural techniques has brought to fruition a fully digital solution that has an extremely fine frequency resolution with low spurious content and low phase noise.

The proposed idea of high-rate dithering of LSB capacitors is illustrated in Fig. 2.4.

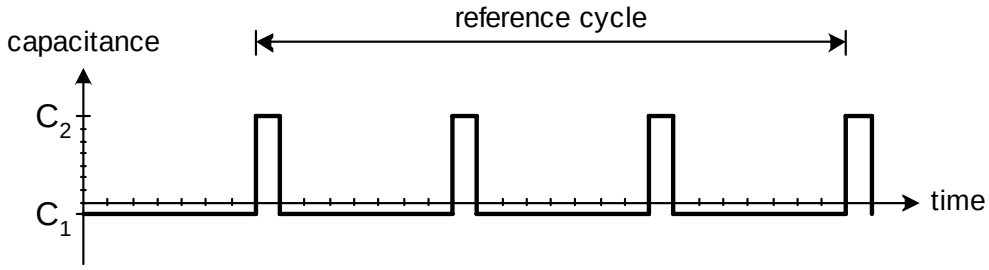

Figure 2.4. DCO dithering by changing the discrete capacitance at high rate

It is similar in principle to the fractional-N division ratio dithering presented earlier in conjunction with Fig. 1.22 (page 34). Instead of applying a constant input that would select capacitance C_1 or C_2, where $C_2 = C_1 + \Delta C$ with ΔC being an LSB capacitor, during the entire reference cycle, the selection alternates between C_1 and C_2 several times during the cycle. In the example, C_2 is chosen one-eighth of the time and C_1 is chosen the remaining seven-eighths. The average capacitance value, therefore, will be one-eighth of the $C_2 - C_1$ distance over C_1. If the dithering speed is performed at a fast enough rate, the resulting spurious tone at the oscillator output could be made vanishingly small (see Appendix A). It should also be noted that the resolution of the time-averaged value relies on the dithering speed. Without any feedback that would result in a supercycle, the dithering rate has to be higher than the reference cycle rate times the integer value of the resolution inverse (eight in this case). Therefore, there is a proportional relationship between the frequency resolution improvement and the dithering rate.

The dithering pattern shown in Fig. 2.4 is not random at all and is likely to create spurious tones. It is equivalent to the first order $\Sigma\Delta$ modulation [30]. In Chapter 3, a second and third order of $\Sigma\Delta$ randomization is proposed in order to effectively eliminate the spurious content.

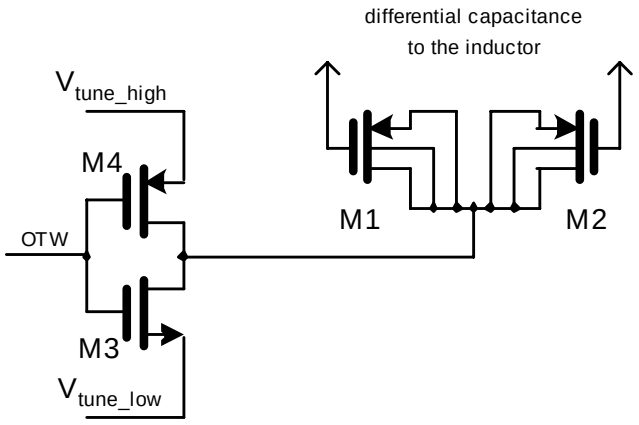

Figure 2.5. Differential varactor and an inverting driver

Fig. 2.5 shows an implementation of the differential varactor and the preceding driver stage. The V_{tune_high} and V_{tune_low} rail supply levels of the inverter are set to correspond with the two stable operating points, off-state and on-state, respectively, as shown in Fig. 2.1. The varactor used in this work is a differential configuration built upon the basic structure described in conjunction with Fig. 2.2 and Fig. 2.3. The balanced capacitance is between the gates of both PMOS transistors M1 and M2, whose source, drain and backgate connections are shorted together and tied to the M3/M4 inverter output. Since the voltage control is now applied to the backgate and source/drain, the negative and decreasing values of V_g in Fig. 2.2 covering the inversion mode are of interest. Because of the differential configuration, only one-half of the single PMOS capacitance is achieved.

The circuit of Fig. 2.5 also reveals a phase noise contribution mechanism from the static tuning input. When either of the driving transistors (M3 or M4) is turned on, its channel resistance generates a thermal noise

$$\overline{e_n^2} = 4kTR\Delta f \tag{2.2}$$

where $\overline{e_n}$ is the rms open-circuit noise voltage generated by the driving resistance R over

the bandwidth Δf at a given temperature T; k is a Boltzmann's constant. As an example, a 50 Ω resistance generates about 0.9 nV of rms noise over a bandwidth of 1 Hz. This noise is added to the stable control voltage which then perturbs the varactor capacitance. This, in turn, perturbs the oscillating frequency and gives rise to the phase noise. These observations favor selection of large W/L ratios of the driver stage transistors in order to reduce the driving resistance and hence thermal voltage noise, and a careful selection of the operational states on the C-V curve (Fig. 2.2) that would result in the smallest possible capacitance sensitivity to the voltage noise.

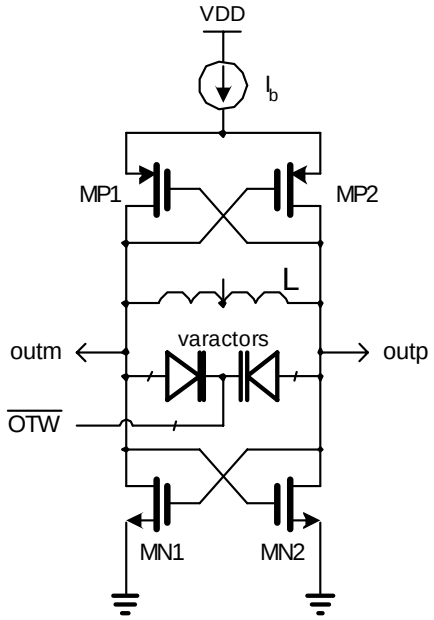

Figure 2.6. Ideal schematic of the DCO oscillator

Fig. 2.6 shows an ideal schematic of the DCO oscillator. The inductor is connected in parallel with an array of the differential varactors. NMOS transistors MN1 and MN2 comprise the first cross-coupled pair that provide a negative resistance to the LC tank. PMOS transistors MP1 and MP2 provide second such pair. The current source I_b limits the amount of current the oscillator is allowed to draw. The oscillator output is differential

with “outp” and “outm” pins being fed to the differential-to-complementary circuit whose purpose is to square the near-sinusoidal outputs and make them insensitive to common mode level. This structure of forming the negative resistance by double cross-connection of transistor pairs is found in literature [14]. It was chosen for its inherent low power since the current used for amplification is utilized twice. What is novel in this work is the replacement of “analog” varactors with digitally-controlled varactor array. This research work only deals with system-level design of the DCO and does not address the detailed component- and circuit-level design issues. The latter part, by itself, could well be another topic for a complete doctoral dissertation.

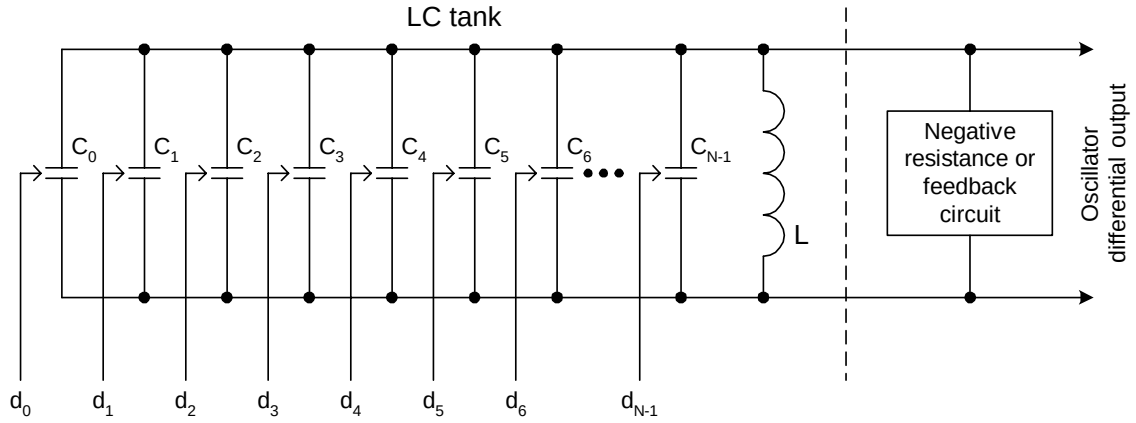

Figure 2.7. LC-tank-based oscillator with switchable capacitors

The idea of the digitally-controlled LC tank oscillator is shown from a higher system level in Fig. 2.7. The resonating frequency of the parallel LC tank is established by the following formula:

$$f = \frac{1}{2\pi\sqrt{L \cdot C}} \tag{2.3}$$

The oscillation is perpetuated by a negative resistance device, which is normally built as a positive feedback active amplifier network.

The frequency f could be controlled by either changing the inductance L or the capacitance C. However, in a monolithic implementation it is more practical to keep the inductor fixed while changing capacitance of a voltage-controlled device, such as a varactor.

Since the digital control of the capacitance C is required, the total capacitance is quantized into an N number of smaller digitally-controlled varactors, which do not necessarily follow the binary-weighted pattern of their capacitance values. Eq. 2.3 now becomes

$$f = \frac{1}{2\pi\sqrt{L \cdot \sum_{k=0}^{N-1} C_k}} \tag{2.4}$$

The digital control signifies that each of the individual capacitors (of index k) could be placed in either a high capacitative state $C_{1,k}$, or a low capacitative state $C_{0,k}$ (see Fig. 2.1). The capacitance difference between the high and low capacitative states of a single bit k is $\Delta C_k = C_{1,k} - C_{0,k}$ and is considered the effective switchable capacitance. Since the frequency of oscillation grows with lowering the capacitance, increasing the digital control value must result in the increased frequency of oscillation. Therefore, the digital control state is opposite to the capacitative state, so the digital bits need to be *inverted* such that the kth capacitor could be expressed as

$$C_k = C_{0,k} + \overline{d_k} \cdot \Delta C_k.$$

The bit inversion turns out to be quite convenient from the implementational point of view. Fig. 2.5 reveals that it is necessary to provide a buffering scheme that would (1) isolate the "raw" varactor input from the noisy digital circuits, (2) have sufficiently low driving resistance to minimize the thermal and flicker noise, and (3) establish two stable low and high voltage levels for the best varactor operation.

Eq. 2.4 could be re-written to include the digital control details.

$$f = \frac{1}{2\pi\sqrt{L \cdot \sum_{k=0}^{N-1}(C_{0,k} + \overline{d_k} \cdot \Delta C_k)}} \tag{2.5}$$

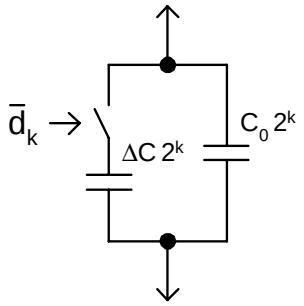

Figure 2.8. Binary-weighted switchable capacitor of index k

Fig. 2.8 shows a model of a single-cell binary-weighted switchable capacitor of index k, that is equivalent to the weight of 2^k. The basic unit cell is created for the weight of 2^0. The next varactor of weight 2^1 is created not as a single device of double the unit area but it is built of two unit cells. This is done for matching purposes. It mainly ensures that the parasitic capacitance due to fringing electric fields, which is quite significant for a deep-submicron CMOS process and is extremely difficult to control and model, is well ratioed and matched. Each next cell consists of double the number of the unit cells. Even though the total occupied silicon area of the device multiplicity method is somewhat larger than the straightforward method of progressively larger uniform devices, it allows to easily achieve the economical component matching resolution of eight bits.

When the d_k digital control bit is 1, the only capacitance seen by the oscillating circuit is C_0 times the weight. This capacitance is always present signifying that the varactor could never be truly turned off. For this reason it could be considered a “parasitic” shunt capacitance. The total sum of these contributions C_0 sets the upper limit of the oscillating

frequency for a given inductance L. When the digital control bit is 0, the ΔC capacitance times the weight is added. The index k of the binary-weighted capacitance can thus be described as

$$C_k = C_{0,k} \cdot 2^k + \overline{d_k} \cdot \Delta C_k \cdot 2^k \tag{2.6}$$

making the total binary-weighted capacitance of size N:

$$\begin{aligned} C &= \sum_{k=0}^{N-1} C_k = \sum_{k=0}^{N-1} (C_{0,k} \cdot 2^k + \overline{d_k} \cdot \Delta C_k \cdot 2^k) && (2.7) \\ &= \sum_{k=0}^{N-1} C_{0,k} \cdot 2^k + \sum_{k=0}^{N-1} \overline{d_k} \cdot \Delta C_k \cdot 2^k && (2.8) \\ &= C_0 + \sum_{k=0}^{N-1} \overline{d_k} \cdot \Delta C_k \cdot 2^k && (2.9) \end{aligned}$$

Contributions from all the static shunt capacitances are lumped into C_0, so the only adjustable components are the effective capacitances in the second term of Eq. 2.9.

2.3 Open-Loop Narrowband Digital-to-Frequency Conversion

From the functional perspective, the above operation can be thought of as a *digital-to-frequency conversion* (DFC) with the digital word comprising d_k bits, where $k = 0, 1, \ldots, N-1$, directly controlling the output frequency f. In order to illustrate that a straightforward DFC conversion to the RF range is not likely to work, let's consider the following example. For the BLUETOOTH application with the oscillating frequency in the RF band of 2.4 GHz and a frequency resolution of 1 kHz, at least 22 bits of DFC resolution is required. It is clearly utmost difficult to achieve this kind of precision even with the most advanced component matching techniques. The best one could hope to economically achieve is 8 to 9 bits of capacitor matching precision [49], without resorting to elaborate matching schemes

that often require numerous and time consuming design, layout and fabrication cycles. In fact, better than 10-bit resolution would normally require some digital error correction techniques [50].

There is one aspect of digital-to-frequency conversion for wireless communications that significantly differs from the general digital-to-analog conversion and that is taken advantage of here. Namely, it is the narrow-band nature of the wireless communication transmission. Consequently, even though the frequency command steps must be very fine, the overall dynamic range at a given time instant is quite small. For example, the nominal frequency deviation of the BLUETOOTH GFSK data modulation scheme is 320 kHz. For a 1 kHz frequency resolution, only 9 bits could be made required ($320kHz/1kHz = 320 < 2^9$). If not handled carefully, a much higher dynamic range is usually necessary to cover frequency channels of the RF band. For the BLUETOOTH band of 80 MHz, 17 bits of full 1 kHz resolution are thus required. Many more extra bits would be necessary to account for process and environmental (voltage and temperature) changes which could reach over +/-20% of the operational RF frequency.

The proposed solution to the above dynamic range problem is to proportionately lower the frequency resolution whenever a higher dynamic range is expected. This is accomplished by traversing through the three major operational modes with progressively lower frequency range and higher resolution such that the intrinsically economical component matching precision of 8 bits is maintained (Fig. 2.9). In the first step, the large oscillating frequency uncertainty due to the *process-voltage-temperature* (PVT) variations is calibrated. After the PVT calibration, the nominal center frequency of the oscillator will be close to the center of the BLUETOOTH band. Since this uncertainty could easily be in the

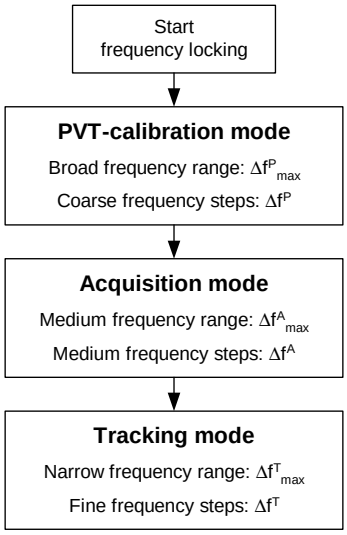

Figure 2.9. Progression flowchart of the DCO operational modes

hundreds of megahertz range, a one or two MHz increments are satisfactory. In this case, an 8-bit resolution is sufficient. The second step is to acquire the requested operational channel within the available band. For an 8-bit resolution, half-MHz steps would span over 100 MHz which is enough for the 80 MHz BLUETOOTH band.

The third step is the finest, but with the most narrow-band range, and serves to track the frequency reference and to perform data modulation within the channel. In the proposed solution, the 1 MHz channel spacing resolution of the BLUETOOTH band already starts at the first step (PVT) but because of the very coarse frequency selection grid possibly covering multiple channels, the best that could be achieved is to get near the neighborhood of the desired channel. It is in the second step (the acquisition mode) that the channel is approximately acquired. However, the fine selection of the requested channel could only be accomplished in the third step (the tracking mode), which is most refined of them all. Therefore, the tracking mode dynamic range has to additionally cover the resolution grid of the preceding acquisition mode. For the BLUETOOTH example, if frequency in the

acquisition mode cannot be resolved to better than 500 kHz and the frequency modulation range is 320 kHz, then the dynamic range of the tracking mode should be better than 10 bits [$(500kHz + 160kHz)/1kHz = 660 < 2^{10}$].

From the operational perspective, the varactor array is divided into three major groups (varactor banks) that reflect three general operational modes: process-voltage-temperature (PVT), acquisition and tracking. The first and second groups approximately set the desired center frequency of oscillation initially, while the third group precisely controls the oscillating frequency during the actual operation. During PVT and acquisition, the frequency range is quite high but the required precision is relatively low, therefore the best capacitor array arrangement here is the binary-weighted structure with a total capacitance of (based on Eq. 2.9)

$$C^P = C_0^P + \sum_{k=0}^{N^P-1} \overline{d}_k^P \cdot (\Delta C^P \cdot 2^k) \tag{2.10}$$

$$C^A = C_0^A + \sum_{k=0}^{N^A-1} \overline{d}_k^A \cdot (\Delta C^A \cdot 2^k) \tag{2.11}$$

where N^P is the number of PVT-mode varactors, N^A is the number of acquisition-mode varactors, ΔC^P and ΔC^A are the unit capacitance of the LSB varactors, $\overline{d}_k^P$ and $\overline{d}_k^A$ are the inverted PVT and acquisition bits, respectively, of the DCO tuning word that control capacitance of the varactor devices.

It is important to note that, at any given time, only varactors that belong to the same bank are allowed to switch. Consequently, only the varactors in each bank need to be matched. This is the key principle behind achieving an extremely fine digital frequency resolution with only 8-bit basic resolution of component matching.

The process/environmental subgroup corrects the center oscillating frequency of the

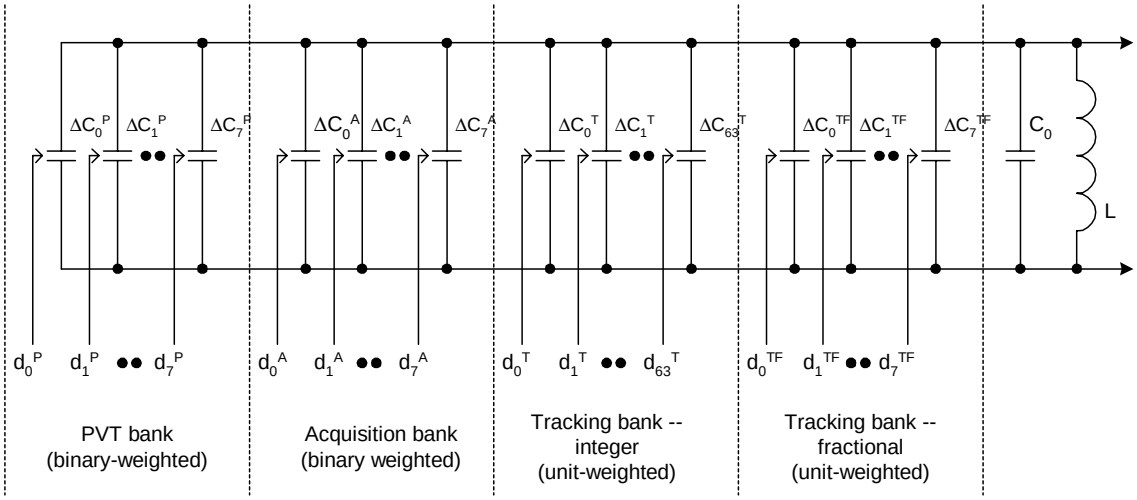

Figure 2.10. LC-tank with dedicated capacitor banks

operational band due to process-voltage-temperature variations and could be performed at manufacturing, on power-up or on "as needed" basis. The channel select varactor group controls the frequency acquisition process for the desired transmission channel. Both groups are best implemented using individual binary-weighted capacitance structures, but their ranges could be overlapping. There is no need to preserve the binary-weight continuity between the process/environmental and channel select structures due to the different origin of their respective control inputs. The PVT correction is infrequent and could be usually done through register interface (e.g., lookup table created during factory calibration), whereas the channel select DCO tuning is performed dynamically and is an integral part of the synthesizer PLL loop. Fig. 2.10 shows the dedicated capacitor banks, which are connected in parallel to create a larger quantized capacitance. Only the effective switchable capacitors are shown forming the banks. The individual shunt capacitances are indistinguishable from each other, therefore, they are lumped together as C_0. Also shown in the figure is the fractional-resolution tracking varactor bank for the high-speed dithering, which will be discussed later.

The tracking-mode operation presents, on the other hand, a different set of requirements. The frequency range is relatively low but the required resolution is quite high. The binary-weighted capacitance arrangement of the acquisition mode is a poor choice here due to the following reasons: binary switching noise (changing a value by 1 LSB might require many bits to toggle; for example, incrementing decimal 31 causes six bits to flip), poor device matching of different size devices (2X precision matched capacitor is rarely implemented as twice the area – usually two identical devices are in parallel next to each other), etc. A better structure would be an array of unit devices of fine but identical dimensions. The total capacitance is

$$C^T = C_0^T + \sum_{k=0}^{N^T-1} \overline{d}_k^T \cdot \Delta C^T \tag{2.12}$$

where N^T is the number of tracking-mode varactors, ΔC^T is the unit switchable capacitance of each varactor and $\overline{d}_k^T$ are the inverted tracking bits of the DCO tuning word.

Since the relative capacitance contribution of the tracking bank is quite small as compared to the acquisition bank, the frequency deviation due to the tracking capacitors could be linearized by the df/dC derivative of Eq. 2.4. Consequently, the frequency resolution or granularity of the LC tank oscillator is a function of the operating frequency f:

$$\Delta f^T(f) = f \cdot \frac{\Delta C^T}{2C} \tag{2.13}$$

where ΔC^T is the tracking-bank unit switchable capacitance and C is the total capacitance. The total tracking-bank frequency deviation is:

$$f^T(f) = \Delta f^T \cdot \sum_{k=0}^{N^T-1} d_k^T = f\frac{\Delta C^T}{2C} \cdot \sum_{k=0}^{N^T-1} d_k^T \tag{2.14}$$

The tracking-bank encoding is classified as a redundant arithmetic system since there are many ways to represent a number. The simplest encoding would be a thermometer

scheme with a predetermined bit order. A less restrictive numbering scheme was chosen in order to facilitate a dynamic element matching – a technique to linearize the frequency-vs.-code transfer function. This idea will be described in Chapter 3.

Further refinement of the frequency resolution is obtained by performing a high-speed dither of one or few of the tracking bits. It is implemented with a first, second or third order of a $\Sigma\Delta$ modulator of the fractional part and will be discussed in Chapter 3.

The DCO operational mode progression could be mathematically described in the following way. Upon power-up or reset, the DCO is set at a center or "natural" resonant frequency f_c by appropriately presetting the d_k inputs. This corresponds to a state in which half or approximately half of the varactors are turned on, in order to maximally extend the operational range in both directions. The total capacitance value of the LC-tank is C_c and the "natural" frequency is

$$f_c = \frac{1}{2\pi\sqrt{L \cdot C_c}} \tag{2.15}$$

During PVT mode, the DCO will approach the desired frequency f by appropriately setting the d^P control bits so that the new total capacitance is $C_{tot,P} = C_c + \Delta C^P$. The resulting final frequency of the PVT mode is

$$f_c^P = \frac{1}{2\pi\sqrt{L \cdot C_{tot,P}}} \tag{2.16}$$

The acquisition mode will start from a new center frequency of f_c^P. It will approach the desired frequency f by appropriately setting the d^A control bits so that the new total capacitance is $C_{tot,A} = C_c + \Delta C^P + \Delta C^A$. The resulting final frequency of the acquisition

mode is expressed as

$$f_c^A = \frac{1}{2\pi\sqrt{L \cdot C_{tot,A}}} \tag{2.17}$$

The following tracking mode will commence from a new center frequency of f_c^A. It will reach and maintain the desired frequency f by appropriately setting the d^T control bits so that the new total capacitance is $C_{tot,T} = C_0 + \Delta C^P + \Delta C^A + \Delta C^T$. The resulting frequency of the tracking mode is set by Eq. 2.3.

The above-described mode progression process of Fig. 2.9 contains two mode switching events during which the center frequency is "instantaneously" shifted closer and closer towards the desired frequency. At the end of the PVT and acquisition modes, the terminating-mode capacitor state is frozen and it now constitutes a new center frequency (f_c^P or f_c^A) from which the frequency offsets, during the following mode, are calculated.

2.4 Implementation

The frequency tuning of the DCO is accomplished by means of the quantized capacitance of the LC-based tank oscillator. There are three operational modes of the DCO with the following arithmetic word encoding, nominal frequency resolution and range as implemented in the X1743 test chip:

- *Process/voltage/temperature (PVT)-calibration mode*: Active during cold power-up and on as-needed basis. Places the nominal center frequency of the DCO in the middle of the BLUETOOTH band. It is also possible to use this mode on a regular basis as an ultra-fast acquisition before the regular acquisition mode. Uses an 8-bit binary-weighted encoding. For best matching, the binary weight is obtained by

means of finger multiplicity of a unit-size varactor. Frequency resolution $\Delta f^P = 2316$ kHz.

- *Acquisition mode*: Active during channel select. Uses an 8-bit binary-weighted encoding. For best matching, the binary weight is obtained by means of finger multiplicity of a unit-size varactor. Frequency resolution $\Delta f^A = 461$ kHz. Frequency range $\Delta f^A_{max} = 118$ MHz.

- *Tracking mode*: Active during the actual transmit and receive. 64-bit unit-weighted encoding and 8-bit unit-weighted encoding for the fractional resolution. Frequency resolution $\Delta f^T = 23$ kHz. Frequency range $\Delta f^T_{max} = 1.472$ MHz.

The frequency numbers were originally obtained through calculation using the capacitor and inductor models obtained from the SML modeling lab. They were confirmed through SPICE and SpectreRF simulations. They were finally verified through lab measurements of the working silicon chip. Unfortunately, due to uncertainty of early varactor models, the frequency of oscillation was lower by about 15%. A conscious decision was made to rather err on the lower frequency side than on the higher frequency side for the following reason: Since initial models of any emerging process technology are not fully debugged, an error was expected. If the actual capacitance is larger than estimated, then the proper center frequency could be achieved with only metal-layer mask change by disconnecting some varactors. It would be much more difficult had the actual capacitance been lower than predicted. This misalignment has been fixed in a subsequent testchip.

The inductor used was an integrated planar inductor [14] realized as a center-tap octagonal structure constructed of all five layers of metal. It is the biggest single component on the

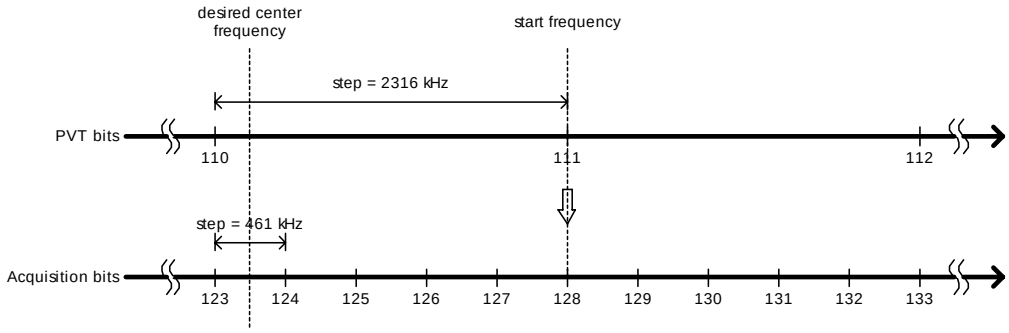

Figure 2.11. Frequency transversal example for the implemented DCO: PVT to acquisition

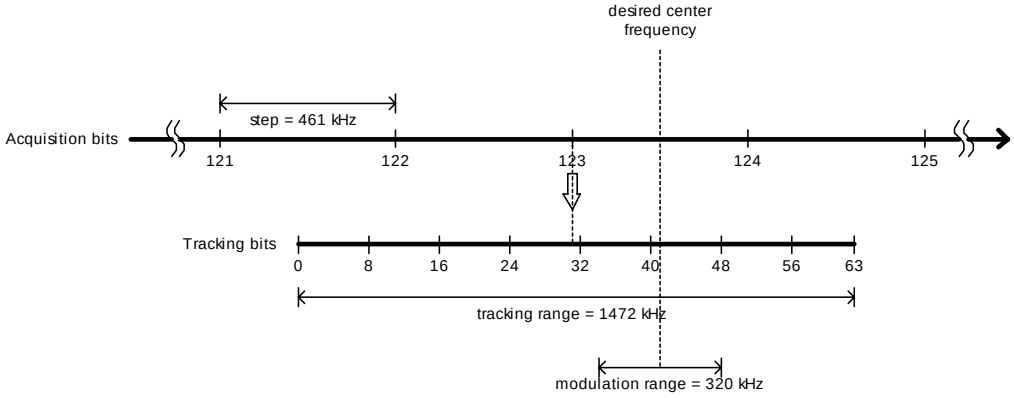

Figure 2.12. Frequency transversal example for the implemented DCO: acquisition to tracking

entire chip and is clearly discernable in lower left corner of the chip micrograph in Fig. 6.2 (page 213). This photo dramatically illustrates the high cost (in terms of digital gates) of conventional RF components in high-density modern CMOS processes. Consequently, the number of classical RF components shall be minimized with proper architectural and circuit design choices.

Fig. 2.11 and Fig. 2.12 demonstrate numerical example of the frequency transversal for the implemented DCO. Let's assume that the PVT mode has been calibrated to the middle of the BLUETOOTH band by fixing the selection to the code of 111. The acquisition mode, therefore, starts from the mid-point reset value of 128, where roughly half of the varactors

are turned on and the other half are turned off. Let's further assume that the desired center frequency lies two channels or 2 MHz lower from the center channel of the BLUETOOTH band. This translates to between four and five acquisition steps. As a result, the loop will first quickly move several steps lower from the starting code of 128. After reaching the point about 2 MHz away, it will dither between the codes of 123 and 124, not being able to resolve any finer. In this example, the transition to the tracking mode happens when the acquisition code is 123 . This code stays frozen for the duration of the packet. The tracking mode always starts from the mid-point value of 31, as shown in Fig. 2.12. It happens that the desired center frequency is located about 230 kHz higher from that point. This corresponds to 10 tracking steps. During transmission, another 160 kHz is allocated on both sides to the frequency modulation.

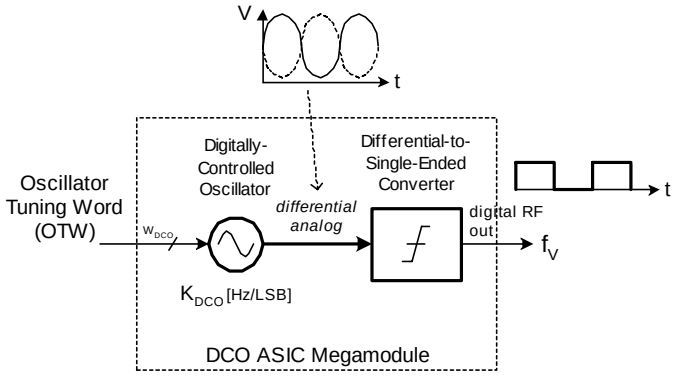

Figure 2.13. DCO as an ASIC cell with digital I/O's

In this implementation, the oscillator is built as an ASIC cell (Fig. 2.13) with truly digital I/O's, even at the RF frequency of 2.4 GHz. The RF signal digitizer is a differential-to-digital converter (with complementary outputs) that transforms the analog oscillator waveform into the zero-crossing digital waveform with a high degree of common-mode rejection.

2.5 Time-Domain Mathematical Model of the DCO

Due to the fact that the conventional RF synthesizers are based on the frequency-domain model, whereas the proposed architecture is rooted in time domain, this section introduces some basic time-domain equations and modeling concept that will be used throughout this work.

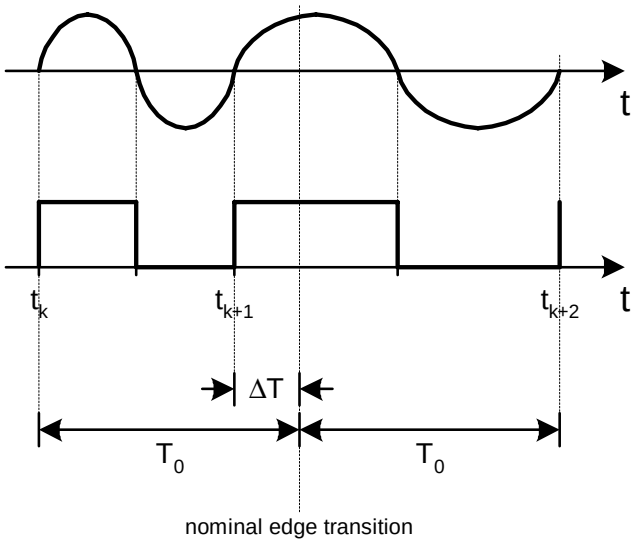

Figure 2.14. Frequency deviation as a clock period deviation

Let the nominal frequency of oscillation be f_0. It is related to the nominal clock period T_0 by its inverse $f_0 = \frac{1}{T_0}$. If the clock period is shortened by ΔT (see Fig. 2.14), the new clock period will be $T = T_0 - \Delta T$. This will result in a higher frequency of oscillation of $f = f_0 + \Delta f$. Let's determine the relationship between Δf and ΔT. Expanding $f = \frac{1}{T}$ results in

$$f_0 + \Delta f = \frac{1}{T_0 - \Delta T} = \frac{1/T_0}{1 - \Delta T/T_0} = \frac{f_0}{1 - \Delta T/T_0} \tag{2.18}$$

For $\frac{\Delta T}{T_0} \ll 1$, using the approximation formula $\frac{1}{1-\varepsilon} \approx 1 + \varepsilon$, Eq. 2.18 simplifies to

$$\Delta f \approx f_0 \frac{\Delta T}{T_0} = f_0^2 \Delta T = \frac{\Delta T}{T_0^2} \tag{2.19}$$

Therefore, for small frequency and clock period deviations, their relationship is linear by the following proportionality relation

$$\frac{\Delta T}{T_0} = \frac{\Delta f}{f_0} \tag{2.20}$$

Eq. 2.19 is used extensively in this work as a conversion formula for system analysis and simulation. As described later in Chapter 7, the simulation environment is VHDL which, being an event-driven digital simulator, is foreign to the concept of frequency and exclusively operates in the native time domain.

Table 2.1. Timing deviation vs. frequency deviation at different points in a BLUETOOTH band

Period deviation ΔT	Center frequency f_0	Frequency deviation Δf
1 fs	2400 MHz	5760 Hz
1 fs	2402 MHz (first channel)	5770 Hz
1 fs	2440 MHz (middle channel)	5953 Hz
1 fs	2480 MHz (last channel)	6159 Hz
1 fs	2500 MHz	6250 Hz

Table 2.1 relates (based on Eq. 2.19) the DCO frequency deviation due to 1 femptosecond of a period deviation for several frequencies around beginning and end of the BLUETOOTH band. It is obvious that a fine timing resolution is required at RF frequencies for time-domain simulation tools. In fact, it was necessary to resort to the finest timing resolution of 1 fs that the VHDL standard provides. From a physical viewpoint, a femptosecond time deviation is quite meaningless for a single observation[1], and only an averaged value could make sense.

[1]One femptosecond is such a small period of time that, during which, the visible light could only travel less than its wavelength

Table 2.2. Frequency resolution and corresponding DCO period deviation for the DCO modes at the middle of the BLUETOOTH band, f_0 = 2440 MHz

Mode	Frequency resolution	Equivalent time resolution
PVT	2326 kHz	390.7 fs
Acquisition	461 kHz	77.43 fs
Tracking	23 kHz	3.863 fs

Table 2.2 relates the DCO frequency resolution for various modes to the DCO period deviation at the middle of the BLUETOOTH band.

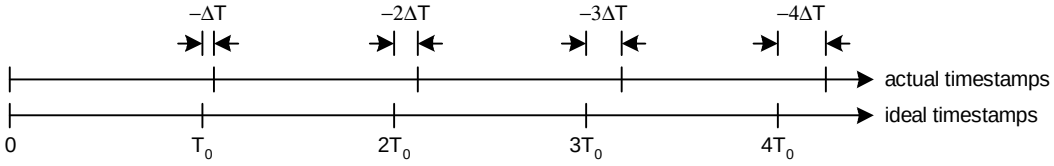

Figure 2.15. Development of an accumulative timing deviation (TDEV)

For a time-invariant oscillator, ΔT period deviation will result in ΔT deviation from ideal timing instances within one oscillator clock period, $2\Delta T$ within two clock periods, etc., as shown in Fig. 2.15. Within N oscillator clock cycles, the accumulated *timing deviation* (TDEV) will reach

$$TDEV(N \cdot T_0) = N \cdot \Delta T = N \cdot \frac{\Delta f}{f_0^2} \tag{2.21}$$

Eq. 2.21 states that the TDEV defined as the difference between actual and ideal timing instances is an integral of the oscillator frequency deviation. The ΔT direction is selected towards shortening the period such that ΔT and Δf signs agree.

The time-domain DCO model is presented in Fig. 2.16. The oscillator tuning word OTW [bits] at the DCO input (d_k bits in Fig. 2.10) will change its operating frequency by $\Delta f_V = OTW \cdot K_{DCO}$ [Hz]. On every rising DCO edge event, the DCO event output

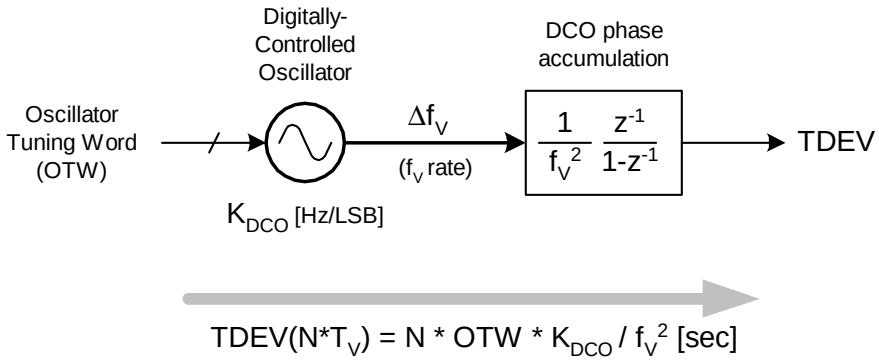

Figure 2.16. DCO time-domain model

Δf_V [Hz] multiplied by a "constant" $\frac{1}{f_V^2}$ [1 / Hz^2] will be accumulated. At the end of N cycles, it will accumulate the TDEV timing deviation [sec] according to Eq. 2.21. The z-domain representation of the accumulation, as opposed to the more commonly used s-domain, signifies that the accumulated timing deviation is only defined at the end of the DCO clock cycle with each rising clock edge. It should be noted that because the phase is fundamentally a time integral of frequency, the DCO phase accumulation is a pure time development process and is not tied to any hardware. The timing deviation is a measure of "badness" and signifies the departure from the desired timing instances that has to be corrected by a feedback loop mechanism which will be introduced in Chapter 4.

The frequency resolution Δf of each mode will be referred to in the following chapters as the DCO gain, K_{DCO}, expressed in Hz/LSB units. The oscillating frequency f will be referred to as the variable frequency f_V for reasons apparent in Chapter 4.

2.6 Summary

In this chapter, a first building block that is at the heart of the proposed RF frequency synthesizer is introduced. The digitally-controlled oscillator (DCO) starts a series of bottom-up progressive development levels which ultimately culminate in a full RF transmitter. The presented DCO features only a minimal but necessary set of functionalities such that it is able to be constructed as a digital I/O ASIC cell that generates a digital RF output clock in response to a digital input. The task of calibrating its transfer function is left to the higher-level blocks described in the following chapters. The oscillator must have a sufficiently low amount of phase noise and spurious tone content.

A novel idea is to use an array of digitally-controlled varactors in place of "analog" varactors used in conventional designs. An advantage is taken here of a narrowband nature of the wireless communication system and a progressive refinement of the effective frequency resolution is proposed. However, as shown, the resulting resolution would still be too coarse to be of any use for RF applications. In order to address this, an idea of dithering the LSB varactors is proposed.

Finally, a time-domain model of the oscillator is introduced. It will be refined in subsequent chapters while more functionality is added.

CHAPTER 3

NORMALIZED DCO

3.1 Overview

The *digitally-controlled oscillator* (DCO) of Chapter 2 provides only a raw and bare minimum of functionality. This chapter introduces a circuitry built around it for the purpose of adding a hierarchical layer of arithmetic abstraction that makes it easier to operate the DCO from the outside.

The oscillator frequency dependence on the process spread and environmental factors, such as voltage or temperature, is tracked by a normalization circuit that belongs to this layer. Consequently, the oscillator proposed in this chapter is referred to as the *normalized DCO* (nDCO). The DCO normalization block also includes circuitry to control the precise application of the tuning word in order to reduce the spurious noise level. This is one of advantages of operating the oscillator in the discrete-time domain that is not possible in conventional continuous-time designs.

As also mentioned in Chapter 2, the DCO is encapsulated at the top I/O level as a discrete-time system. Consequently, this view could be extended for the normalized DCO. This brings numerous benefits of being able to tap the rich body of knowledge from the *digital-signal processing* (DSP) field.

There have not been at the time of writing any reports on the fully-digital oscillators for RF applications. Since all the ideas presented in this chapter pertain to the digital-control and discrete-time nature of the oscillator, they are believed to be novel.

3.2 Oscillator Transfer Function and Gain

At the heart of the frequency synthesizer lies the digitally-controlled oscillator. It generates an output with a frequency of oscillation f_V being a physically-inherent function of the digital *oscillator tuning word* (OTW) input. The $f_V = \mathrm{f}(OTW)$ mapping was defined by Eq. 2.1 (page 48).

In general, $\mathrm{f}(OTW)$ is a nonlinear function of the input. However, within a limited range of operation it could be approximated by a linear transfer function. In this case, f(OTW) is a simple gain K_{DCO}. This allows us to rewrite Eq. 2.1 in a more linear form.

$$f_V = f_0 + \Delta f_V = f_0 + K_{DCO} \cdot OTW \tag{3.1}$$

where Δf_V is a deviation from a certain center frequency f_0. f_0 could be one of the mode-adjusted center frequencies as described in Sec. 2.3. Δf_V must be sufficiently small such that the linear approximation is satisfied.

K_{DCO} is specifically defined as a frequency deviation Δf_V (in Hz) from a certain oscillating frequency f_V in response to 1 LSB of the input change. Within a linear range of operation, the DCO gain can also be expressed as

$$K_{DCO}(f_V) = \frac{\Delta f_V}{\Delta(OTW)} \tag{3.2}$$

Within a limited range, K_{DCO} should be fairly linear with respect to the input, otherwise the DCO gain could be generalized as being also a function $K_{DCO}(OTW)$ of the specific input.

$$K_{DCO}(f_V, OTW) = \frac{\Delta f_V}{\Delta(OTW)} \tag{3.3}$$

3.3 DCO Gain Estimation

Due to its analog nature, the K_{DCO} gain is subject to the process and environmental factors which cannot be known precisely. It belongs to one of a few unknown system parameters whose estimate, $\widehat{K}_{DCO}$, must be determined. As described later, the $\widehat{K}_{DCO}$ estimate could be calculated entirely in the digital domain by observing the phase error responses to the past DCO phase error corrections. The actual DCO gain estimation involves arithmetic operations, such as multiplication or even division, and averaging, and could be performed by a dedicated hardware or a *digital signal processor* (DSP).

The frequency deviation Δf_V of Eq. 3.2 cannot be directly measured, except perhaps in a lab or a factory setting. Due to the digital nature of the synthesizer, Δf_V can be, however, indirectly measured on-the-fly by harnessing the power of the existing phase detection circuitry. Discussion of this method is deferred until Sec. 4.20.

3.4 DCO Gain Normalization

At the higher level of abstraction, the DCO oscillator, together with the DCO gain normalization $f_R/\widehat{K}_{DCO}$ multiplier, logically comprise the *normalized DCO* (nDCO), as illustrated in Fig. 3.1. The DCO gain normalization decouples the phase and frequency in-

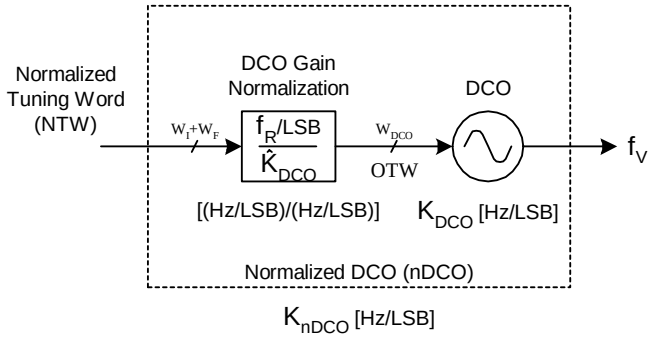

Figure 3.1. Normalized DCO

formation throughout the system from the process, voltage and temperature variations that normally affect the K_{DCO}. The phase information is normalized to the clock period of the oscillator T_V, whereas the frequency information is normalized to the value of the external *reference frequency* f_R. The digital input to the nDCO is a fixed-point *normalized tuning word* (NTW), whose integer part LSB bit corresponds to f_R. The reference frequency is chosen as the normalization factor because it is the master basis for the frequency synthesis as defined in Sec. 1.4.1. Another reason is that the clock rate and update operation of this discrete-time system is established by the frequency reference.

The quantity K_{DCO} should be contrasted with the process-temperature-voltage-independent oscillator gain K_{nDCO} which is defined as the frequency deviation (in Hz units) of the DCO in response to the 1 LSB change of the integer part of the NTW input. If the DCO gain estimate is exact, then $K_{nDCO} = f_R/LSB$, otherwise

$$K_{nDCO} = f_R/LSB \cdot \frac{K_{DCO}}{\widehat{K}_{DCO}} \tag{3.4}$$

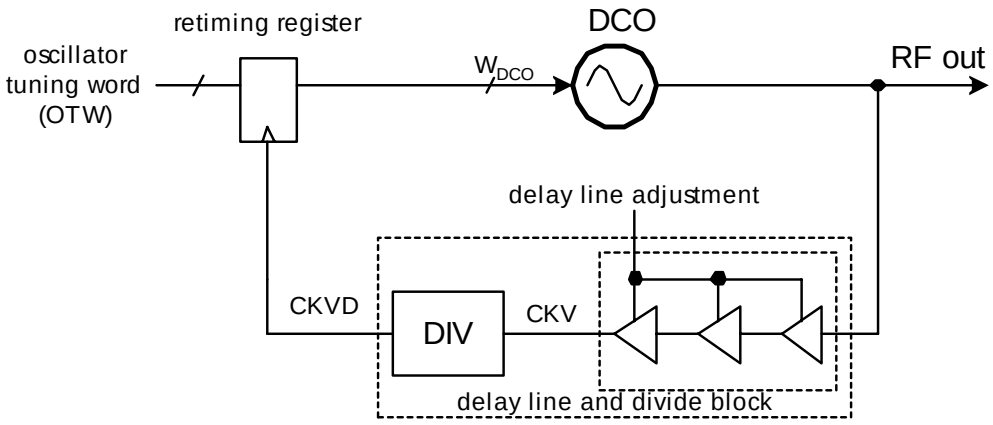

Figure 3.2. Synchronously-optimal sampling and timing adjustment of the DCO input

3.5 Principle of Synchronously-Optimal DCO Tuning Word Retiming

Fig. 3.2 proposes the principle of the synchronously-optimal DCO input tuning word retiming method. This idea is based on the observation that changing the tuning control input of an oscillator, in order to adjust its phase/frequency in a normal PLL operation, is quite a disturbing event that reveals itself as jitter or phase noise [51] [52] [53]. This is especially noticeable in case of a sample-mode oscillator, such as the DCO, where its oscillating frequency is commanded to change at discrete times. For example, if the oscillating frequency of an LC tank is controlled by a voltage-to-capacitance conversion device (e.g., varactor), the instances when the oscillating energy is fully stored in a capacitor are the worst moments to change the capacitance. Changing the capacitance at those moments causes the electrical potential to change ($\Delta V = Q/\Delta C$), since the total charge must be preserved, thus introducing the largest perturbation, as shown on the lower plot in Fig. 3.3. Changing the varactor capacitance at times when it is fully discharged will hardly affect its voltage and thus will not contribute to the oscillating jitter (upper plot in Fig. 3.3).

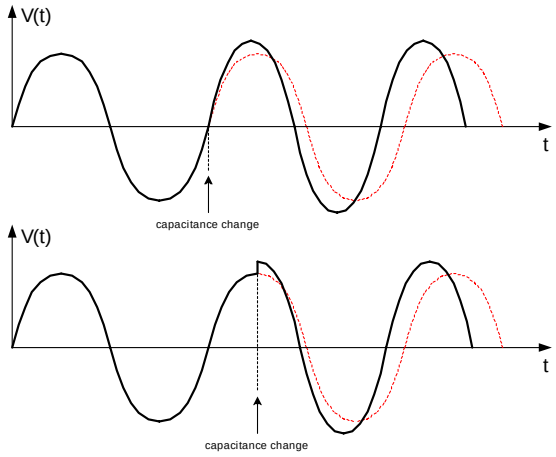

Figure 3.3. Waveforms for capacitance change of an LC-tank oscillator

The proposed solution is to control the timing moments when the varactor capacitance change is allowed to occur, thus minimizing jitter due to the tuning word update. This is implemented by feeding the delayed oscillator edge transitions back as the clock input to the *synchronous* register retiming stage, as shown in Fig. 3.2. The retiming stage ensures that the input control data, as seen by the oscillator, is allowed to change only at precise and *optimal* time after the oscillator zero-crossings. The feedback loop delay is set algorithmically to minimize the oscillator jitter. The actual delay could be accomplished by a voltage-controlled delay line. In the testchip, we chose a long string of inverters while externally controlling their V_{dd} supply voltage. The allowed range of the delay line V_{dd} voltage is quite limited (from 1 V to about 1.8 V) so the delay change contribution per inverter is not very significant. However, the delay multiplied by the total number of inverters can exceed the DCO clock cycle, thus guaranteeing the full 360 degree coverage.

3.6 Time Dithering of the DCO Tuning Input

The phase error is calculated and applied to correct the DCO frequency at regular intervals as normally determined by the frequency reference. These regularly spaced events are likely to introduce sharp spurs at the output. The nature of these spurs is similar as in the fractional-N clock division method described in Sec. 1.4.3, where Fig. 1.27 (page 37) depicts the deterministic and $\Sigma\Delta$ randomized spectrum. The digital control of the oscillator makes it possible to somewhat randomize these events in a manner described below. It should be noted that it is generally difficult to perform an accurate time dithering in the continuous-time domain.

The ideas revealed in this section have not been implemented in this testchip in the exact embodiment of Fig. 3.7, Fig. 3.8, Fig. 3.9. However, the time dithering concept is coupled to the frequency reference retiming by the DCO clock, which will be described in Sec. 4.10.

3.6.1 Oscillator Tune Time Dithering Principle

Fig. 3.4 shows the proposed principle behind the oscillator time dithering scheme. Instead of calculating and applying the tuning word input to the oscillator at evenly-spaced and deterministic time intervals, as conventionally defined by the frequency reference clock, “random” time-stamp deviations at each update are exercised. The statistical properties of these time-shift deviations will determine how much of the spectral spur energy gets spread into the background. The time dithering of the oscillator tuning input could be fundamentally implemented in one of two ways: time dithering of the OTW itself or time

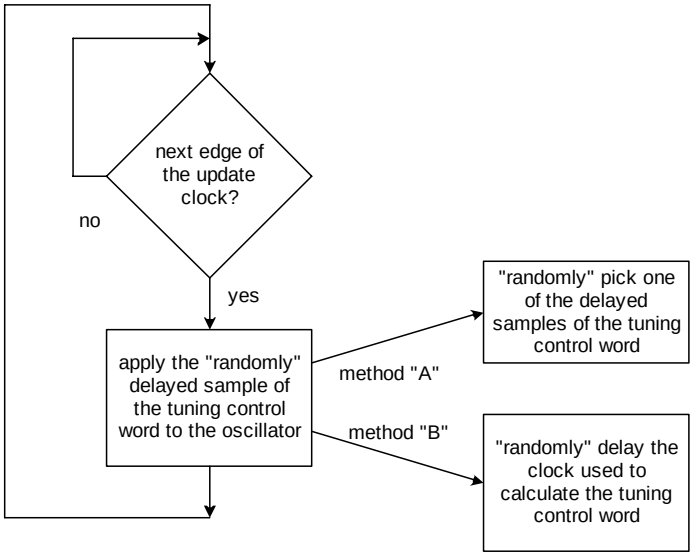

Figure 3.4. Flowchart of the oscillator tune time dithering

dithering of the actual time the OTW gets calculated and applied.

3.6.2 Direct Time Dithering of the Tuning Input

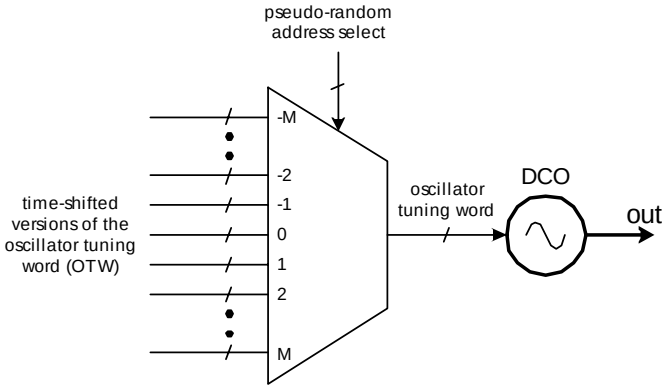

Figure 3.5. Basic idea of discrete time dithering of the DCO tuning input

Fig. 3.5 shows the main idea behind the tuning input time dithering scheme. The *oscillator tuning word* (OTW) is a digital signal and is synchronous to the compare events of the

phase detection operation[1]. The oscillator tuning word would normally be connected to the DCO input after a gain stage (with possible low-level signal conditioning) if a loop filter is not used. In this scheme, time-shifted replicas of the OTW signal are pseudo-randomly selected to randomize the exact time-stamps of regular DCO frequency updates.

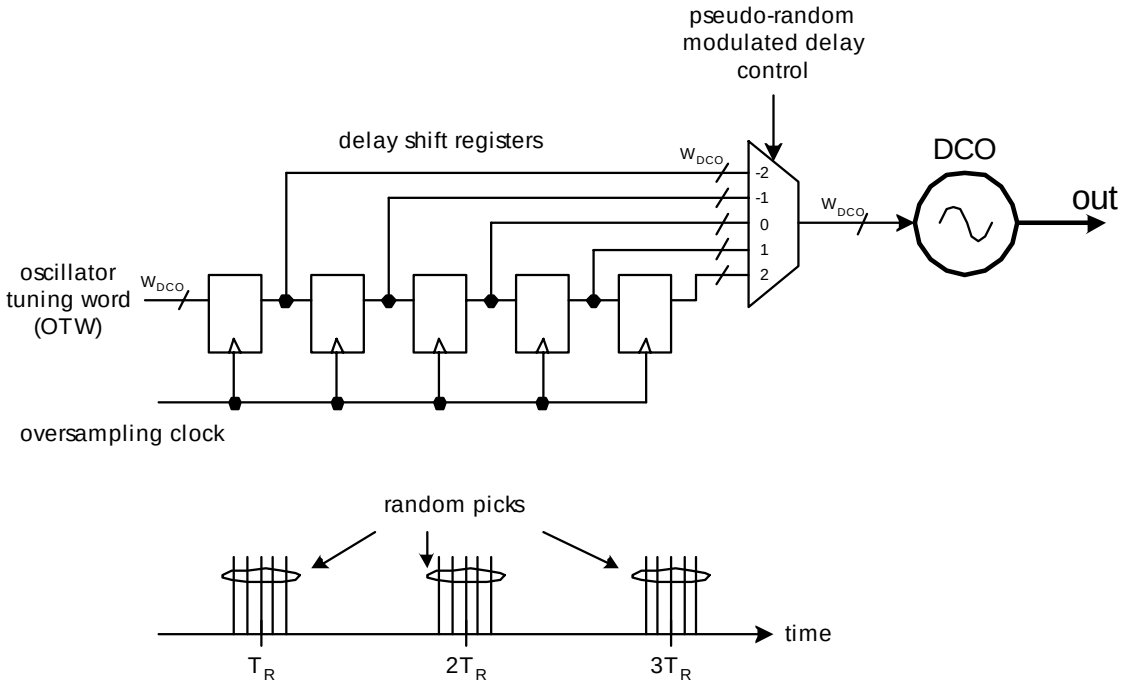

Figure 3.6. Time dithering method of the DCO tuning input through a multiplexer

Fig. 3.6 reveals a simple time-causal method to obtain the time-shifted replicas of the OTW with delay stages. An accurate discrete-time dithering of the OTW signal is obtained by reclocking it by the high-frequency oversampling clock and passing it through a delay shift register. A multibit input multiplexer synchronously selects the appropriate output of the delay register chain. This method provides a means of dynamically offsetting the actual DCO update timing, which is done at the frequency reference rate, discretely by the oversampling clock. The discrete delay control stream is further constrained to promote

[1]The phase detection operation will be described in Chapter 4

favorable properties of the resulting OTW signal. Consequently, it is advantageous to use $\Sigma\Delta$ modulated time-shift control to move the time-dither quantization energy into higher frequencies where it is easy to filter it out by the Q of the oscillator, power amplifier and the antenna band filter. Properties of the modulator should be selected based on the desired quantization noise characteristics.

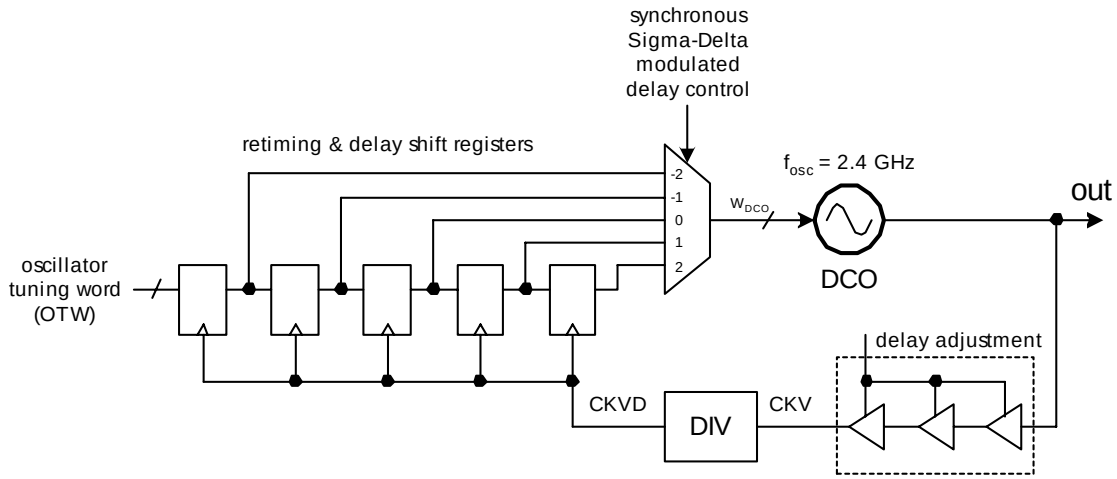

Figure 3.7. Time dithering with the DCO synchronous tuning input retiming

The digitized RF output of the synthesizer would be used as the high-frequency over-sampling clock directly or after an appropriate frequency division by the edge-divider[2] block DIV as shown in Fig. 3.7. In this embodiment, the oscillator is implemented as an ASIC cell with truly digital I/O's, even at the RF frequency (shown in Fig. 2.13 on page 69). Consequently, this circuit and even the entire normalized DCO are implemented in a digital fashion. This figure also reveals the synchronously-optimal DCO input tuning word retiming method (described in Sec 3.5) which could be optionally used in conjunction with the discrete-time dithering method.

[2]"Frequency divider", "edge divider" and "clock divider" terms are interchangeable in the frequency synthesizer design field

The order of fine time delay (covering at least one RF clock period) and edge division could be reversed, but usually it consumes less power to delay the lower-repetition-rate clock edges.

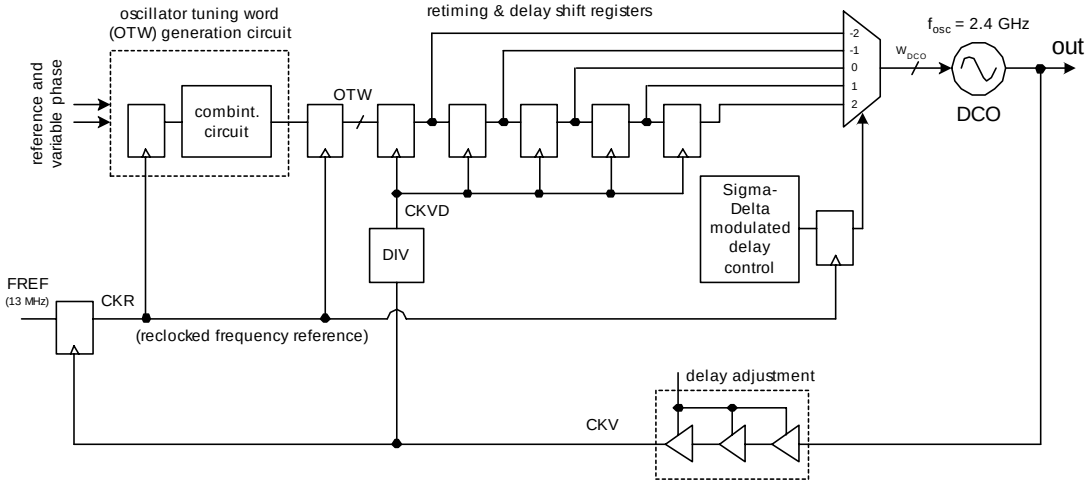

Figure 3.8. Time dithering with an additional frequency reference retiming

Fig. 3.8 reveals further details within the ADPLL architecture settings[3]. The operation of generating the oscillator tuning word (that also includes calculating the phase error between the reference and oscillator phase information) is run synchronously to the reclocked frequency reference (CKR) and is synchronously reclocked by the oversampling clock CKVD. The CKR clock has the same long term period as the frequency reference FREF and its edges are forced to be synchronous to the oscillator clock CKV (and its derivatives, such as CKVD) by the retiming stage. The oversampling clock CKVD is obtained, as in Fig. 3.7, by edge dividing of the oscillator clock CKV. This configuration preserves the mutually-synchronous clock planes of CKR and CKVD greatly simplifying the tuning word interface between the clock domains.

[3]The ADPLL loop will be described in Chapter 4

The Δ or $\Sigma\Delta$ modulator randomizes the small discrete timing deviations to the actual repetitive update of the DCO oscillator such that the compare-frequency spurs are sufficiently blurred into the background noise.

Not all the ideas presented above have been implemented in the actual silicon. They are presented here to illustrate the logical development steps in the research process and help to understand the final architecture of the proposed synthesizer.

3.6.3 Update Clock Dithering Scheme

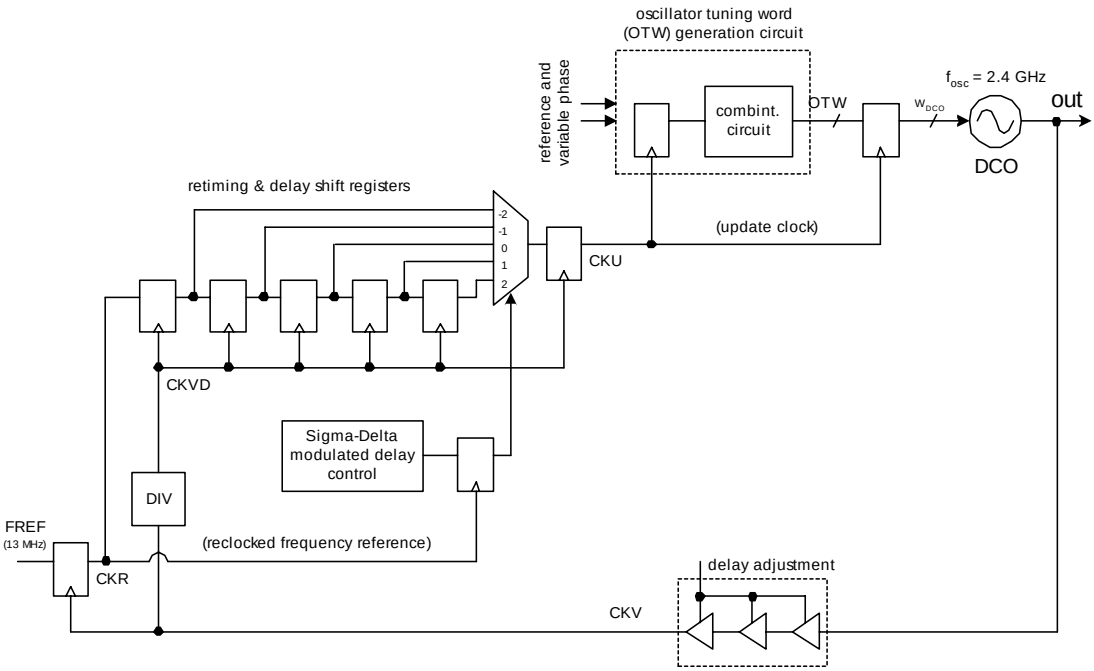

Figure 3.9. Time dithering implementation of the update clock

Fig. 3.9 discloses an improved time dithering method (“method B” in Fig. 3.4) which randomizes sampling edges of the *update clock* (CKU) instead of the oscillator tuning word input, as in the previous method. The update clock is then used to trigger the generation and sampling of the DCO tuning input. The operational order of calculating the tuning word and

time dithering is thus reversed. This leads to substantial hardware savings since delaying the clock, which takes a single bit, is preferred to delaying a multibit tuning word. Another clear benefit in a digitally-intensive system is that the complex OTW calculation operation is more randomly spread in time and exhibits less temporal correlation. Consequently, this further leads to reduction of frequency spurs.

If the silicon chip die also contains a microprocessor and a DSP on the same substrate, which is often the case with modern RF transceivers, it is advantageous to clock these processors *synchronously* to the time-dithered update clock CKU. Two significant benefits could thus be obtained: First, randomly modulating the clock period prevents substrate noise with strong periodical correlation to couple from the digital baseband to the RF section. Second, if the processor clock exhibits enough delay from the synthesizer update clock, the phase detection and tuning word adjustment operations occur during the "quiet" periods of the DSP.

3.7 Implementation of PVT and Acquisition DCO Bits

Fig. 3.10 shows an implementational block diagram of the three separate DCO loop filter gain paths for the three modes of operation: PVT, acquisition and tracking, as originally defined in Fig. 2.9 (page 61). The blocks on the left side of the diagram (phase detector and the loop gain section) will be formally introduced in Chapter 4. The tracking path additionally splits into integer and fractional parts, mainly due to their significantly different clock rates. Each of the switched capacitor array banks (first introduced with Fig. 2.10 on page 63) is individually controlled by a respective oscillator interface circuit.

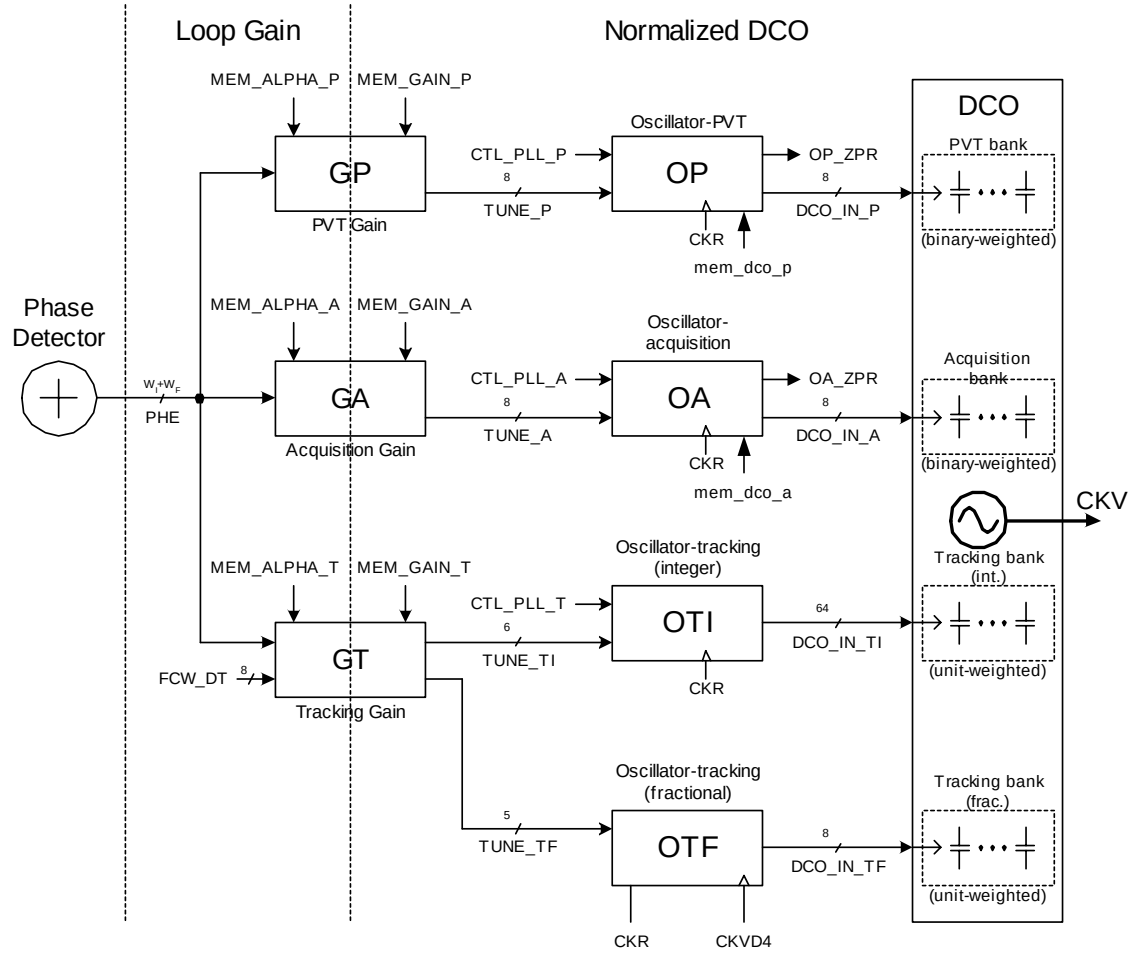

Figure 3.10. DCO gain paths – implementational block diagram

Also shown in Fig. 3.10 is the phase detector output signal PHE being fed into the three gain circuits (GP, GA, GT for the PVT, acquisition and tracking modes, respectively). Due to their vastly different gain ranges, each gain circuit could use a different subset of the full range of the phase error. The gain circuits multiply the phase error by the associated factors, which are split into two parts: the loop normalizing gain (MEM_ALPHA set to α [4]) and the DCO normalization gain (MEM_GAIN set to $f_R/\widehat{K}_{DCO}$). Only the second normalizing multipliers formally belong to the nDCO layer, but are physically combined

[4]Loop normalizing gain α will be defined in Chapter 4

with the loop gain multipliers for implementational reasons. The outputs of the gain circuits are the oscillator tuning words (OTW) that control the oscillator control circuits OP, OA, OTI and OTF.

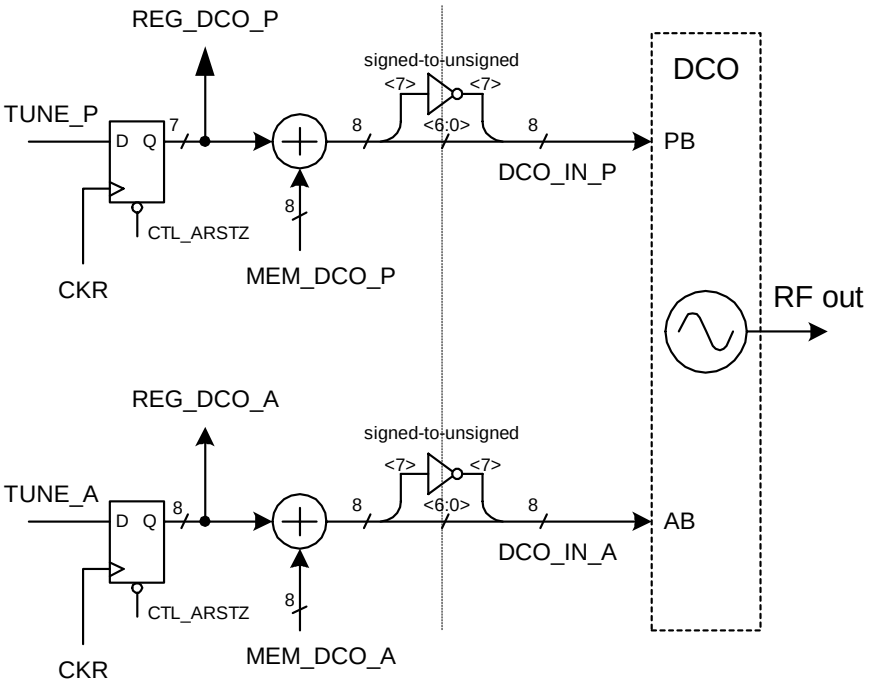

Figure 3.11. Oscillator interface with the PVT and acquisition bits

The PVT and acquisition oscillator interface is shown in Fig. 3.11. Both capacitor banks are built as 8-bit binary-weighted arrangements. Their direct digital control is arithmetically expressed as *unsigned* number arguments into the total contributive capacitances C^P and C^A of Eq. 2.10 (page 62) and Eq. 2.11 (page 62), respectively. The digital PLL loop control, however, natively operates at an offset frequency with respect to a certain center or "natural" frequency. Consequently, the control logic inherently uses an arithmetic encoding representation that is in a *signed 2's complement* notation. A conversion mechanism, by simply inverting the MSB bit, is thus only required at the interface point. In this scheme, $-2^7 \cdots 0 \cdots (2^7 - 1)$ maps to $0 \cdots 2^7 \cdots (2^8 - 1)$, so the "MSB bit inversion" could be

thought of as an addition of $+2^7$ to the 8-bit 2's complement signed number with the carry outs disregarded. It should be noted that the intrinsic offset of 2^7 on the DCO side is part of establishing the "natural" oscillating frequency.

The above observation brings a very important point. The center or natural frequency of a digitally-controlled oscillator (DCO) must be handled differently than that of a voltage-controlled oscillator (VCO). In a VCO, the natural frequency is defined at the zero or ground level (or halfway between the positive and negative supply rails) of the tuning voltage input. This results in a maximum tuning range in both frequency directions. With a DCO, however, the situation is not so straightforward, especially now with the multiple tuning words. The issue we are facing could be considered a generalization of the oscillator tuning natural frequency which so far had not seen practical consequences with VCO-based designs due to difficulty of freezing analog tuning voltages.

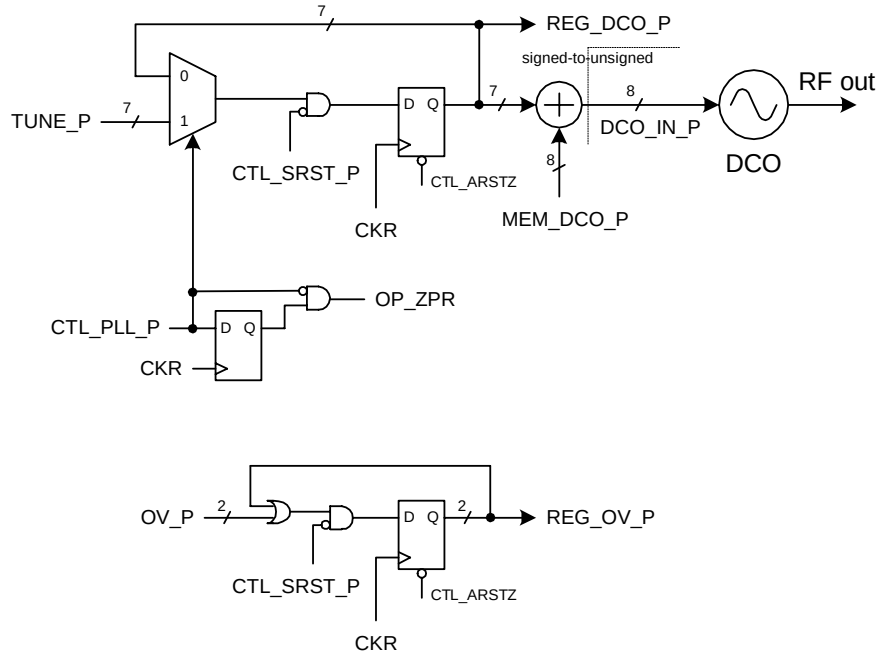

Figure 3.12. Control circuit of the oscillator PVT bits (OP)

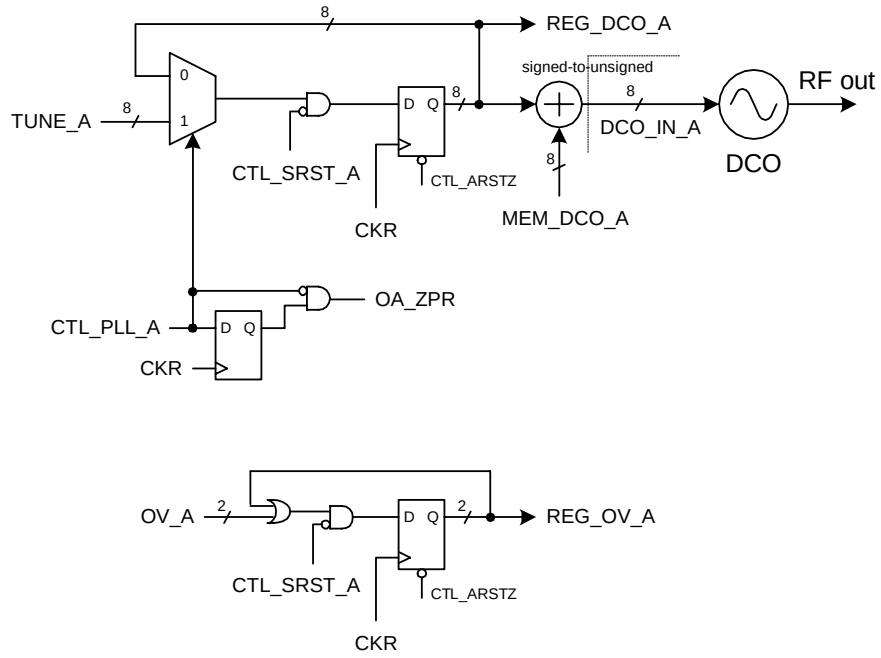

Figure 3.13. Control circuit of the oscillator acquisition bits (OA)

Fig. 3.12 and Fig. 3.13 illustrate almost identical interface control structures (OP and OA blocks of Fig. 3.10) for both PVT and acquisition bits, respectively, while Fig. 3.11 previously showed only the very interface details. There are sets of two register memory interface words. MEM_DCO_P and MEM_DCO_A could be the last frequency estimate from the controller's lookup table in order to speed up the loop operation. REG_DCO_P and REG_DCO_A are the frequency offset status words reported back to the controller.

At reset, the DCO is placed at the center of the operational frequency range (possibly redefined by MEM_DCO_P and MEM_DCO_A) through asynchronous clear CTL_ARSTZ of the driving registers. This is an important mechanism that prevents the oscillator from failing to oscillate if the random power-up values of tuning word registers set it above the oscillating range, which might happen at the slow process corner.

During the active mode of operation, the new tuning word is latched by the register with

every clock cycle. Upon the DCO operational mode change-over, the last stored value of the tuning word is maintained by the register. Consequently, during the regular operation, only one path of Fig. 3.10 can be active at a given time, whereas the previously executed modes maintain their final DCO control states. The *zero phase restart* (ZPR) is used to zero out the phase detector output to avoid any discontinuities in the oscillator tuning word during the mode switchover. A short explanation of the ZPR principle is as follows: At the mode switchover, the tuning word of the last mode corresponds to a certain value of the phase error. This tuning word is now frozen, so the phase error value that maintains it is not longer needed. However, the new mode is always referenced to the new center frequency established by the last mode. Consequently, it operates on the *excess* phase error rather than absolute. Therefore, the old value of the phase error that corresponds to the frozen tuning word of the last mode would have to be constantly subtracted from the new phase error. A better solution is to use the proposed method of zero phase restarting. In this way, a hitless progression through the three DCO operational modes is accomplished. This method is described in more detail in Chapter 4.

3.8 Implementation of Tracking DCO Bits

The tracking bits of the DCO oscillator need a much greater care and attention to detail than the PVT and acquisition bits. The main reason is that these very bits are used during the normal operation. The PVT and acquisition bits, on the other hand, are used in the preparatory steps to quickly establish center of the operating frequency and are inactive during the normal operation when the synthesized frequency is used. Consequently, any phase noise or spurious tone contribution of the tracking bits will degrade the synthesizer

performance, so they need to be treated with care.

3.8.1 High-speed Dithering of the Fractional Varactors

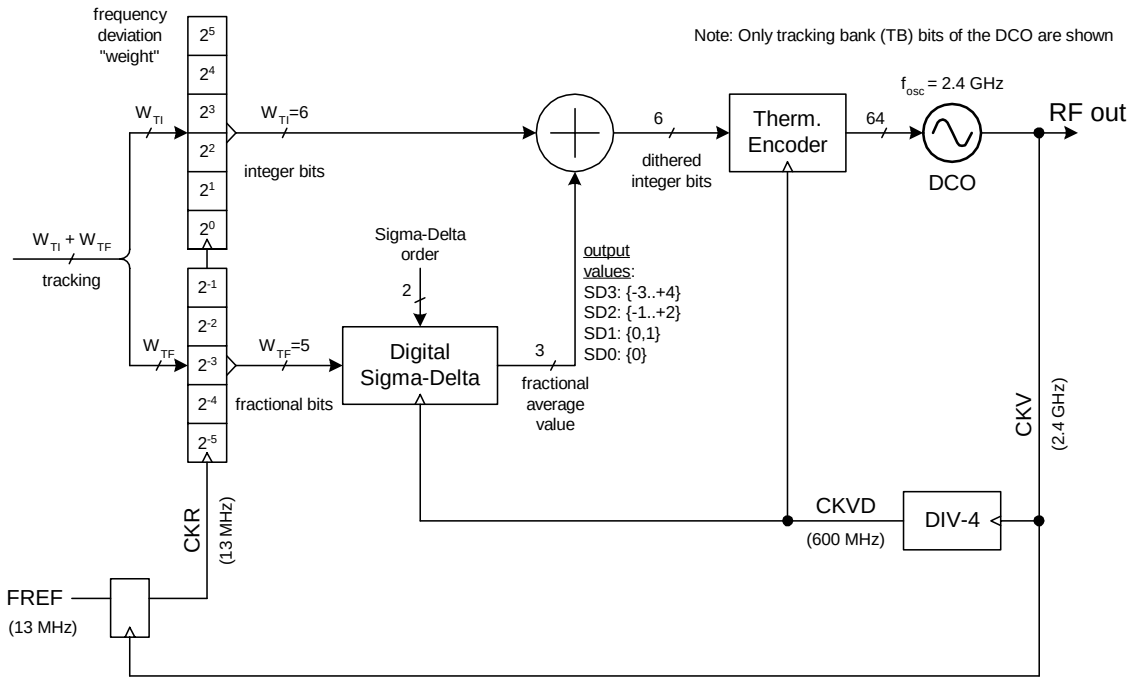

Figure 3.14. Improving frequency resolution with $\Sigma\Delta$ dither of DCO tracking bits

Fig. 3.14 shows the proposed idea to increase frequency resolution of the DCO. Tracking part of the *oscillator tuning word* (OTW) is split into two components: $W_{TI} = 6$ integer bits and $W_{TF} = 5$ fractional bits. The LSB of the integer part corresponds to the basic frequency resolution of the DCO oscillator. The integer part could be thermometer encoded to control the same-size DCO varactors of the LC-based tank oscillator. In this scheme, all the varactors are unit weighted but their switching order is predetermined. This guarantees monotonicity and helps to achieve an excellent linearity, especially if their switching order agrees with the physical layout. The transients are minimized since the number of switching varactors is no greater than the code change. This compares very favorably with

the binary-weighted control, where a single LSB code change can cause all the varactors to toggle. In addition, due to equal load throughout for all bits, the switching time is equalized in response to code changes.

In this implementation, a slightly more general unit-weighted capacitance control is used to add some extra coding redundancy which lends itself to various algorithmic improvements of the system operation, as described below.

The fractional part, on the other hand, employs a time-averaged dithering mechanism to further increase the frequency resolution. The dithering is performed by a digital $\Sigma\Delta$ modulator that produces a high-rate integer stream whose average value equals the lower-rate fractional input. Appendix A proves that the DCO spurs introduced by the switching of a varactor can be made vanishingly small if performed at a fast enough rate.

$\Sigma\Delta$ techniques have been used successfully for over two decades in the field of analog data converters. This has developed a rich body of knowledge for other applications to draw upon [54].

The integer part of the tuning word is then added to the integer-valued high-rate-dithered fractional part. The resulting binary signal is thermometer encoded to drive the sixty-four tracking bank varactors. In this simplest embodiment, the high-rate fractional part is arithmetically added to the low-rate integer part thus making its output, as well as the entire signal path terminating at the varactors inside the DCO, high rate. A preferred solution to implement this approach is presented below.

The $\Sigma\Delta$ modulator is built as a third-order MESH-type structure [32] that could be efficiently scaled down to a lower order. It is clocked by the 600 MHz divided-by-4 oscillator

clock, CKVD, which can be gated off if not needed.

As explained earlier in Sec. 2.2, the dithering method trades the sampling rate for the granularity. As an example of the implemented design, if the frequency resolution of the 2.4 GHz DCO is $\Delta f^T = 23$ kHz with a 13 MHz update rate, then the effective time-averaged frequency resolution, within one reference cycle, after the 600 MHz $\Sigma\Delta$ dither with five sub-LSB bits would be $\Delta f^{T-\Sigma\Delta}$ = 23 kHz / 2^5 = 718 Hz. The frequency resolution improvement achieved here is $2^5 = 32$. This roughly corresponds to the sampling rate speedup of 600 MHz / 13 MHz = 26.

The above point requires further elaboration. As mentioned in Sec. 2.2, the finest resolution improvement per one reference cycle is upper bounded by the sampling rate speedup. In the example, this condition is not quite satisfied. In fact, the *operational* frequency resolution achieved is much finer. It is determined by the fractional wordlength $W_F = 15$ of the reference phase accumulator and equals 13 MHz / 2^{15} = 396.7 Hz. The difference between the two frequency resolution numbers is that the former assumes a single reference cycle during which the loop does not make any corrections. The latter, on the other hand, involves multiple FREF cycles and harnesses the PLL loop averaging power over the longer observation period. The closed loop operation is deferred until Chapter 4.

The structure of the digital $\Sigma\Delta$ modulator is depicted in Fig. 3.15. It is implemented as a 3-rd order MESH-type architecture. Its topology is based on [30] and was shown previously in Fig. 1.26 (page 36). The original structure is not the best choice for high-speed designs because the critical path spans through all the three accumulator stages and the carry sum adders. A critical path retiming transformation needed to be performed in order to shorten the longest timing path to only one accumulator so that the 600 MHz clock

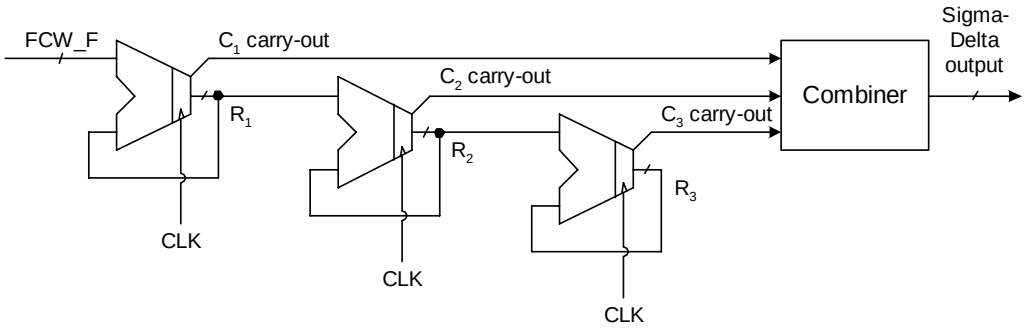

Figure 3.15. MESH-3 $\Sigma\Delta$ digital modulator structure

operation could be reached.

Since the structure is highly modular, the lower-order modulation characteristics (plotted in Fig. 1.27 on page 37) could be set by disabling the tail accumulators through gating off the clock, which is a preferred method from power saving standpoint.

The combiner circuit (originally shown in Fig. 1.26 on page 36) merges the three single-bit carry-out streams such that the resulting multi-bit output satisfies the 3-rd order $\Sigma\Delta$ spectral property. The $\Sigma\Delta$ stream equation below is a result of register retiming of the architecture originally described in [30].

$$out_{\Sigma\Delta} = C_1 \cdot D^3 + C_2 \cdot (D^2 - D^3) + C_3 \cdot (D - 2D^2 + D^3) \tag{3.5}$$

where $D \equiv z^{-1}$ is the delay element operation. This equation is easily scaled down to the second or first order $\Sigma\Delta$ by disregarding the third or third and second terms, respectively.

Fig. 3.16 reveals the preferred method of implementing the integer and fractional oscillator tracking control (OTI and OTF of Fig. 3.10) from a lower power standpoint. The fractional path of the DCO tracking bits, which undergoes high-rate dithering, is entirely separated from the lower-rate integer part. It even has a dedicated DCO input just to avoid "contaminating" the rest of the tracking bits with frequent transitions. The switch matrix,

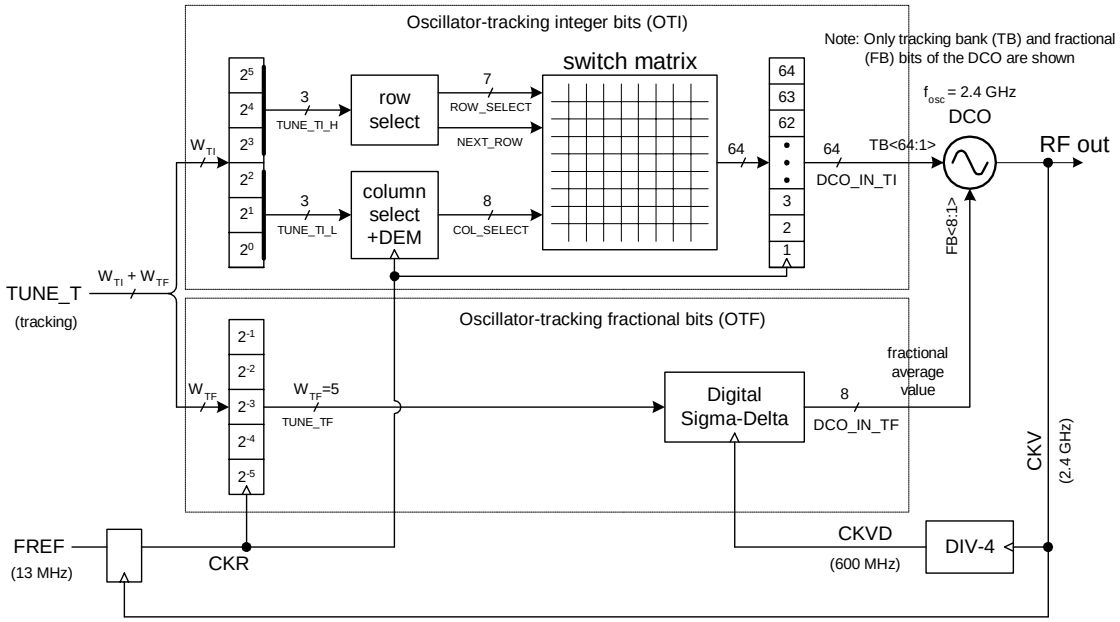

Figure 3.16. Implementation block diagram of the DCO tracking bits (OTI + OTF) with DEM of the integer part and $\Sigma\Delta$ dithering of the fractional part

together with the row and column select logic, operates as a binary-to-unit-weight encoder in response to the integer part of the tracking tuning word. The $\Sigma\Delta$ modulator is responsive to only the fractional part of the tracking tuning word. The actual merging of both parts is performed inside the oscillator through time-averaged capacitance summation at the LC tank.

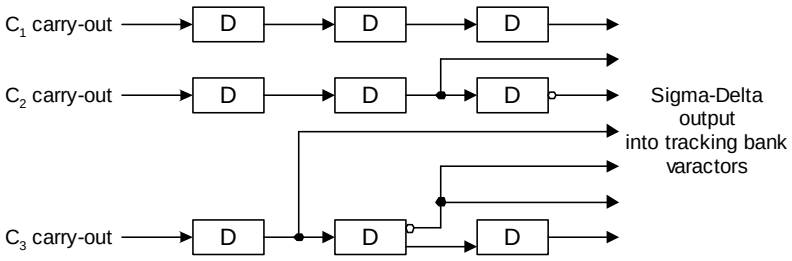

Figure 3.17. $\Sigma\Delta$ modulator carry-out combiner structure

Another important benefit of the chosen approach is that the high-speed arithmetic operation of the Eq. 3.5 combiner is now trivial. Fig. 3.17 shows the proposed implementation.

All that is required are flip-flop registers (for the delay operation) with complementary outputs (for the negation). The arithmetic addition is performed inside the oscillator through capacitance summation.

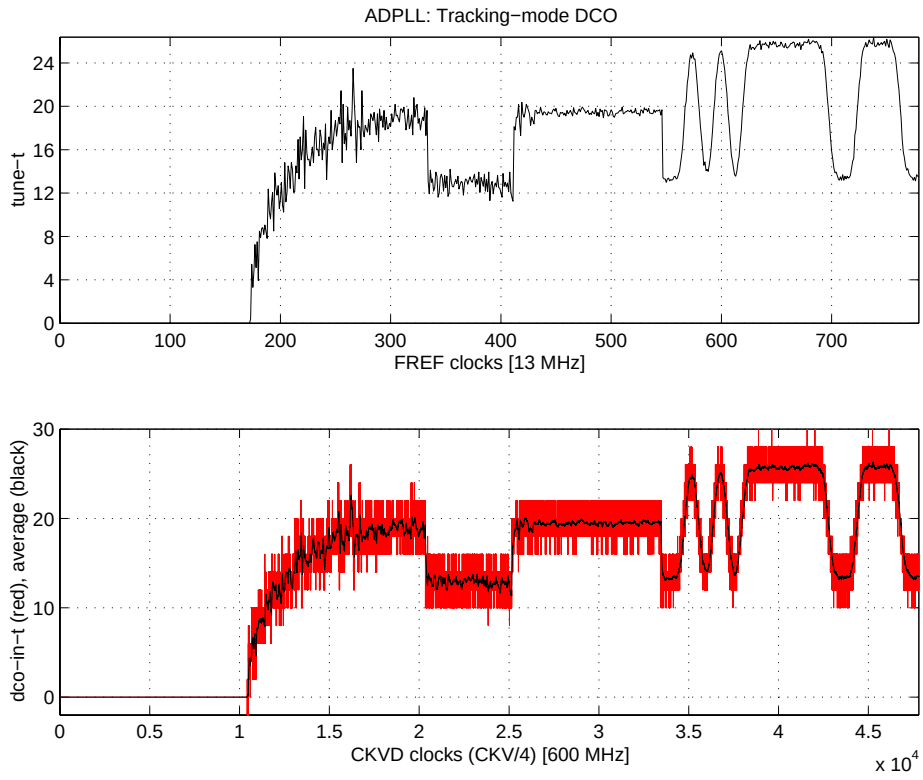

Figure 3.18. Simulation plot using the $\Sigma\Delta$ modulation of the fractional part of the tracking tuning word; (top: fixed-point tuning word; bottom: decoded merged DCO integer input)

Fig. 3.18 illustrates the second-order MESH-type $\Sigma\Delta$ modulation of the fixed-point tracking DCO tuning control word with five fractional bits. The fixed-point tuning word (TUNE_T, upper plot) consists of six integer bits and five fractional bits and is clocked at the 13 MHz reference frequency. The $\Sigma\Delta$ modulates the five-bit fractional part at 600 MHz clock rate and outputs the integer stream that controls the DCO frequency. The lower plot shows the $\Sigma\Delta$ output stream (DCO_IN_TF) "merged" with the 6-bit integer part stream (DCO_IN_TI). For the purposes of visualization only, the DCO_IN_TI stream is mathe-

matically decoded into an unsigned number representation and added to the mathematically decoded DCO_IN_TF signed stream. This mathematical operation is performed by a MATLAB software package based on the data files generated by a VHDL MODELSIM simulator[5]. The solid black curve on the lower plot is the running average and it faithfully reproduces the fixed-point tuning control input.

3.8.2 Dynamic Element Matching of the Varactors

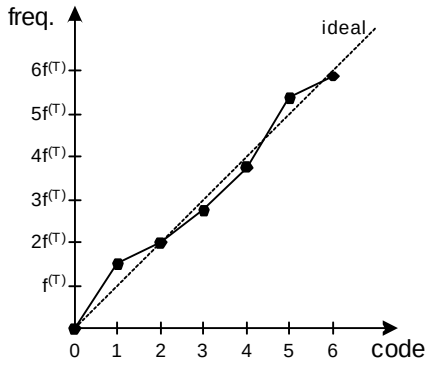

Figure 3.19. Cumulative nonlinearity of the DCO tracking bits

Ideally, each of the unit-weighted capacitors of the tracking bank has the exact same capacitative value. Using real-world fabrication process, however, the capacitative value of each capacitor will vary slightly from the ideal. As capacitors are turned on and off by the integer tracking oscillator controller OTI of Fig. 3.10, non-linearities will be evident in the output due to variations in capacitative values, as shown in Fig. 3.19.

A method to improve the digital-to-frequency conversion linearity is also revealed in Fig. 3.16. It cyclically shifts the unit-weighted varactors using the *dynamic element match-*

[5]Simulation methodology will be described in Chapter 7

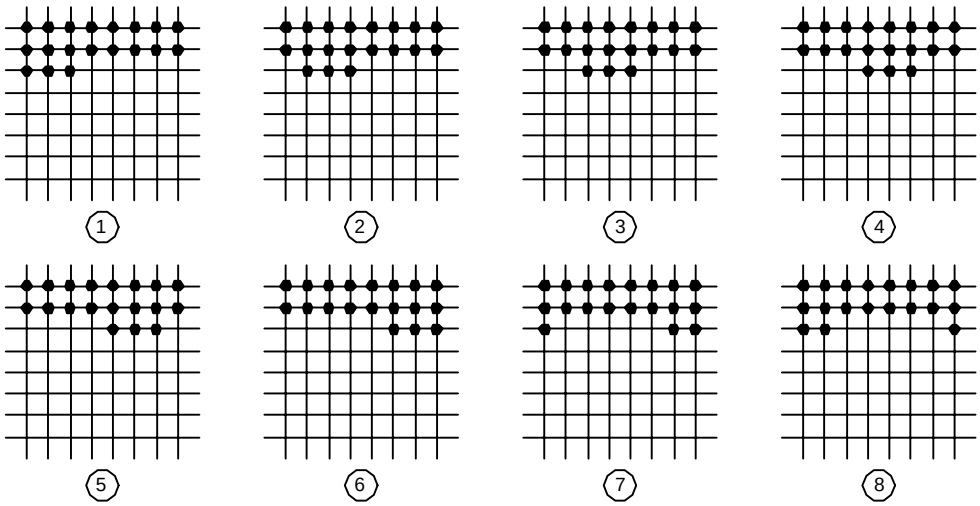

Figure 3.20. Dynamic element matching through cyclic shift within a matrix row

ing (DEM) method recently being employed in digital-to-analog converters [55]. The integer part of the tuning word is split into upper and lower bits. The upper bits are encoded and control the row selection of the switch matrix. The lower bits are also encoded and select the next column of the switch matrix. The cyclic shift of unit-weight varactors is performed within the row (see Fig. 3.20) but could also extend to other rows. However, the number of active switches does not change for the same control input.

On Fig. 3.20, the capacitors associated with an unfilled row (“next row” signal of Fig. 3.16) of the switch matrix are rotated on each CKR clock cycle. Initially, the first three columns of row three are enabled. On the next clock cycle, columns two through four, rather than columns one through three are enabled. On the next clock cycle, columns three through five are enabled, etc. Accordingly, on each clock cycle, the set of capacitors used in the 64-element array changes slightly. Over time, the non-linearities shown in Fig. 3.19 average out, thereby producing a much more accurate output.

With this DEM scheme, the enabled switches for a single row are rotated. This is

accomplished by modulo incrementing the starting column of the enabled switches on each clock cycle. This method could be varied slightly by including two (or more) rows in the rotation. As a result, a larger frequency range would be subject to the beneficial time averaging, by including a greater number of capacitors in the rotation, but at a cost of a longer repetition cycle. An alternative method of increasing the DEM frequency span would be to lengthen the number of columns per row, thus creating a non-square matrix.

The output bits of the switch matrix are individually coupled to the bank of sixty-four resampling drivers, which are implemented as flip-flop registers. Each driver controls a single unit-weighted varactor of the LC-tank. Using resampling by the CKR clock eliminates delay mismatches due to path differences, such that the timing points of varactor transitions coincide. This helps with the spurious noise control. It should be noted that while the switch matrix is shown (from an algorithmic standpoint) in a row/column configuration, the actual implementation is not a precise grid. In fact, a group of rows could be physically combined into a single line.

The principal difference in DFC vs. DAC specification requirements is that the full dynamic range is not required for the available number of controlled units. In the DFC application, the frequency headroom is required because it is not expected that the oscillator operates at the precisely specified frequency before entering the tracking mode.

3.8.3 DCO Varactor Rearrangement

As illustrated in Fig. 3.21, the sixty-four integer tracking-bit varactors of the LC tank have a physical layout of two long columns and the fractional tracking-bit varactors are arranged

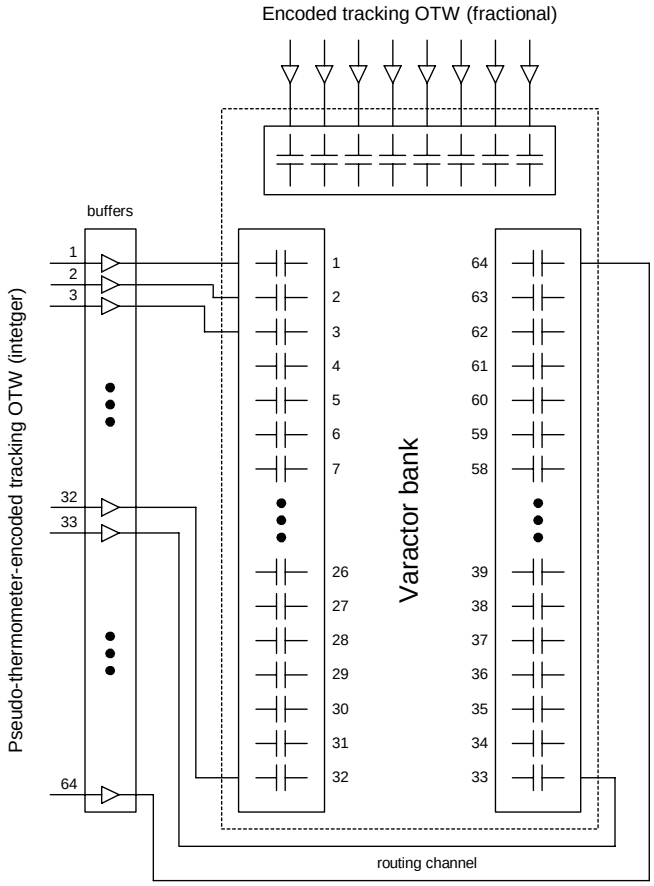

Figure 3.21. Layout diagram of the tracking capacitors

separately. However, the controlling circuitry is located only on one side. This creates an unbalanced structure in which routing to one varactor column is significantly shorter with easier access than to the other and, therefore, their transient response is different. Moreover, the spatially separated devices are likely to be more mismatched due to the process gradient. If, during the course of operation, the varactor selection transitions through the column boundary, it is likely to create larger switching perturbations. It is proposed that before entering the finest tracking mode, a rearrangement of DCO varactors to be performed such that “lower quality” capacitors be filled in order to maximize the frequency dynamic range of the most preferred capacitor section. In this architecture, this could be done upon

switchover from fast tracking to tracking.

It should be noted that, while in other designs, the tracking capacitors could be arranged differently depending upon various layout issues, a certain set of capacitors will always be favored based on the proximity metric to the control logic.

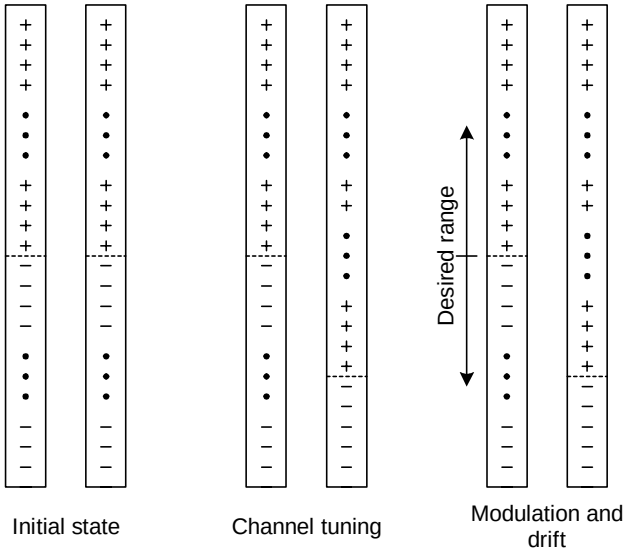

Figure 3.22. Tracking capacitors rearrangement

Fig. 3.22 illustrates a method of improving the quality of the DFC conversion. In the initial state, half of the capacitors in each column are turned on (as designated by a “+”) and half are turned off (as designated by a ”-”). During fast tracking, the capacitors of the less desirable right-hand column are enabled or disabled in order to fine tune the oscillator to the selected channel, to the extent possible. If additional capacitors need to be enabled or disabled, the capacitors from the left-hand column may be used, preferably those capacitors at the edges of the column. After channel tuning, the capacitors in the left-hand column are used for modulation and drift control. In this way, the most desirable capacitors are used for maintaining lock and for generating the signal once data is being transmitted.

It should be noted that while the preferred center point was shown as the middle of the left-hand column, the preferred center point could be set at any location that is far from layout and routing discontinuities.

3.9 Time-Domain Model

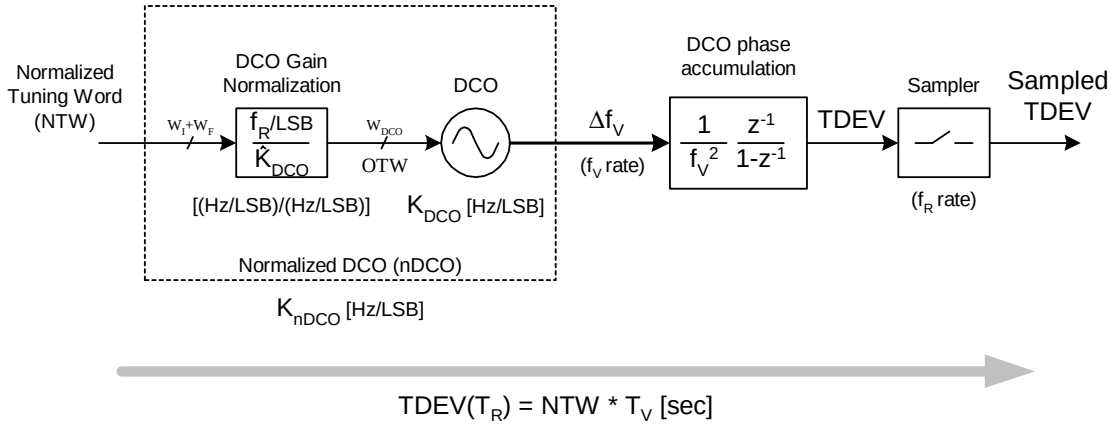

Figure 3.23. nDCO time-domain model

The time-domain nDCO model is presented in Fig. 3.23. It is build upon the DCO time-domain model presented in Fig. 2.16 (page 73). Provided the DCO gain is estimated correctly, the normalized tuning word NTW [bits] at the nDCO input will change its operating frequency by $\Delta f_V = NTW \cdot f_R$ [Hz]. On every rising edge event, the DCO event output Δf_V [Hz] multiplied by a "constant" $\frac{1}{f_V^2}$ [1 / Hz^2] will be accumulated. The accumulation interval is established by the inverse of the reference frequency $T_R = \frac{1}{f_R}$. It is related to the nominal oscillation frequency f_V by $T_R = N \cdot T_V$. At the end of N cycles, the accumulated timing deviation TDEV [sec] will be sampled with value

$$TDEV(T_R) = N \cdot \Delta T_V = N \cdot \frac{\Delta f_V}{f_V^2} = NTW \cdot \frac{N f_R}{f_V^2} = NTW \cdot T_V \qquad (3.6)$$

As stated previously, the DCO phase accumulation is not tied to any hardware. It simply reflects the workings of a progression of time. However, the sampling mechanism requires an explicit hardware that would make periodic snapshots of the evolving TDEV.

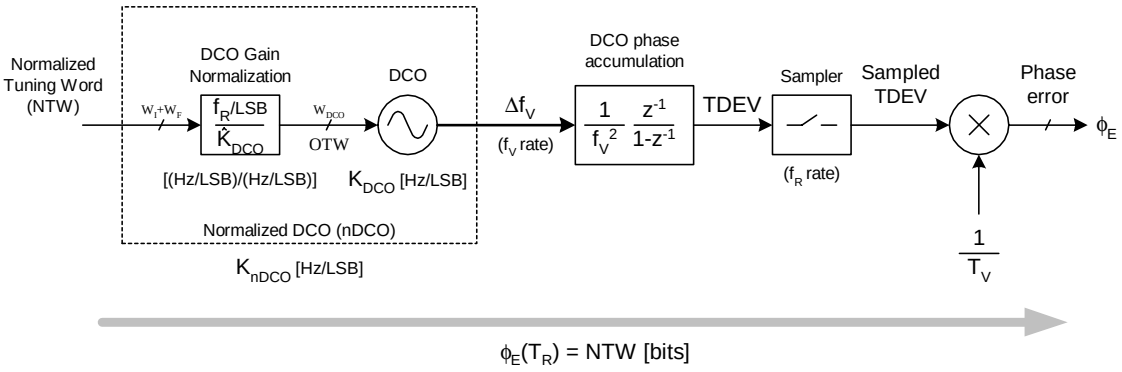

Figure 3.24. nDCO time-domain model with phase detection

A glimpse into the next chapter is given in Fig. 3.24. A generic hardware circuitry is added to the nDCO that would detect TDEV and convert it to the digital bit format. At the same time, it deals with the troublesome unit of time by performing normalization to the clock period of the DCO oscillation, defined as *unit interval* (UI). The diagram does not suggest any particular mechanism. It merely indicates and mathematically describes a timing deviation detector that would determine any frequency and phase deviations of the oscillator which would then be fed back as loop corrections. The transfer function from the normalized tuning word input to the detector outputs is 1 [bits/bits] within one FREF clock cycle.

3.10 Summary

This chapter presented the first hierarchical layer of arithmetic abstraction over the raw digitally-controlled oscillator (DCO) in order to make it easier to operate it algorithmi-

cally. The main task of this overlay block is to perform DCO calibration and normalization such that the *normalized DCO* (nDCO) transfer function is largely independent from the process and environmental factors. Other improvements, such as increasing the frequency resolution through $\Sigma\Delta$ dithering and dynamic element matching, are also introduced. This layer serves the purpose of concealing the implementational details to the layers above in order to make their algorithms and implementation simpler.

Block diagram of the datapath portion of the normalized DCO layer is presented on the right hand-side of Fig. 3.10 (page 88). It consists of the oscillator interface (OP, OA, OTI and OTF) and normalizing gain blocks (second part of GP, GA and GT) for each of the operational modes. The non-datapath operation of the K_{DCO} estimation is performed in software. It utilizes full synthesizer features and will be described in later chapters.

CHAPTER 4

ALL-DIGITAL FREQUENCY SYNTHESIZER

4.1 Overview

The *digital-to-frequency conversion* (DFC) of the *normalized digitally-controlled oscillator* (nDCO) described in the previous chapter operates in an open-loop manner. Consequently, its stability is quite poor due to drift or wander of the self-generated phase and frequency. This chapter proposes a phase correction mechanism by which the output phase, and hence frequency, is periodically corrected by comparison with a stable reference phase as established by the FREF frequency reference input of Fig. 1.14 (page 24). This way, the long-term frequency stability of the synthesizer matches that of the reference.

The phase correction mechanism is performed entirely in the digital domain by phase locking the generated DCO clock to the reference input. Design and building of its constituent blocks also followed the digital methodology. Due to these reasons, the frequency synthesizer described in this chapter is also referred to as *all-digital phase-locked loop* (ADPLL).

This chapter also introduces a gear-shift and zero-phase restart techniques that, while collaborating with the synthesizer loop operation, switch the progressive refinement of resolution as the operational frequency approaches the desired frequency. These techniques implement the algorithm described in Fig. 2.9 (page 61).

4.2 Phase Domain Operation

Let's define the actual clock period of the variable (VCO, DCO or a generally-controllable oscillator) output CKV as T_V and the clock period of the frequency reference FREF as T_R.

Let's assume that the oscillator runs appreciably faster than the available reference clock, $T_V \ll T_R$, which is in alignment with a majority of frequency synthesis applications and certainly with this RF synthesizer, where generated RF carrier is of orders of magnitude higher frequency than the crystal reference. Let's further assume, in order to simplify the initial analysis, that the actual clock periods are constant or time-invariant,

The CKV and FREF clock transition timestamps t_V and t_R, respectively, are governed by the following equations:

$$t_V = i \cdot T_V \tag{4.1}$$

$$t_R = k \cdot T_R + t_0 \tag{4.2}$$

where $i = 1, 2, \ldots$ and $k = 1, 2, \ldots$ are the CKV and FREF clock transition index numbers, respectively, and t_0 is some initial time offset between the two clocks, which is absorbed into the FREF clock.

It is convenient in practice to normalize the transition timestamps in terms of actual T_V units (referred to as *unit intervals*, UI) since it is easy to observe and operate on the actual CKV clock events. Let's define dimensionless variable and reference "phase".

$$\theta_V \equiv \frac{t_V}{T_V} \tag{4.3}$$

$$\theta_R \equiv \frac{t_R}{T_V} \tag{4.4}$$

The term θ_V is only defined at CKV transitions and indexed by i. Similarly, θ_R is only defined at FREF transitions and indexed by k. This results in

$$\theta_V(i) = i \tag{4.5}$$

$$\theta_R(k) = k \cdot \frac{T_R}{T_V} + \frac{t_0}{T_V} = k \cdot N + \theta_0 \tag{4.6}$$

The normalized transition timestamps $\theta_V(i)$ of the variable clock, CKV could be estimated by accumulating the number of significant (rising or falling) edge clock transitions.

$$R_V(i \cdot T_V) = \sum_{l=0}^{i} 1 \tag{4.7}$$

Without the frequency reference retiming (described later in Sec. 4.3), the normalized transition timestamps $\theta_R(k)$ of the frequency reference clock, FREF, could be obtained by accumulating the *frequency command word* (FCW) on every significant (rising or falling) edge of the frequency reference clock.

$$R_R(k \cdot T_R) = \sum_{l=0}^{k} FCW \tag{4.8}$$

FCW is formally defined as the frequency division ratio of the *expected* variable frequency to the reference frequency.

$$FCW \equiv \frac{\mathscr{E}(f_V)}{f_R} \tag{4.9}$$

The reference frequency is usually of excellent long term accuracy, at least as compared to the variable oscillator. For this reason, we do not use the expectation operator on f_R.

Alternatively, FCW could be defined in terms of the division of the two clock periods in the mean sense.

$$FCW \equiv \frac{T_R}{\mathscr{E}(T_V)} \tag{4.10}$$

where $\mathscr{E}(T_V) \equiv \overline{T_V}$ is the average clock period of the oscillator. Eq. 4.10 gives another interpretation of the phase domain operation. The FCW value establishes how many high-frequency CKV clocks are to be contained within one lower-frequency FREF clock. It suggests counting the number of CKV clocks and dividing it by the number of FREF cycles in order to get the estimate. It should also be noted here that the instantaneous clock period ratio might be slightly off due to the phase noise effects of the DCO oscillator. However, the long-term value should be very precise and approach FCW in the limit.

FCW control is generally expressed as being comprised of an integer (N_i) and fractional (N_f) parts.

$$FCW = N = N_i + N_f \tag{4.11}$$

The PLL operation achieves, in a steady-state condition, a zero averaged phase difference between the variable $\theta_V(i)$ and the reference $\theta_R(k)$ phases. Attempt to formulate the phase error as this unitless phase difference $\phi_E = \theta_R - \theta_V$ would be unsuccessful due to the nonalignment of the time samples.

Additional benefit of operating the PLL loop with phase domain signals is to alleviate the need for the frequency detection function within the phase detector. This allows us to operate the PLL loop as type-I (only one integrating pole due to the DCO frequency-to-phase conversion), where it is possible to eliminate a low-pass filter between the phase detector and the oscillator input, resulting in a high bandwidth and fast response of the PLL loop. It should be noted that conventional phase-locked loops such as a charge-pump-based PLL (Fig. 1.20 on page 31) do not truly operate in the phase domain. There, the phase modeling is only a small-signal approximation under the locked condition. Their

reference and feedback signals are edge based and their closest distance is measured as a proxy for the phase error. F. Gardner describes this as “converting the timed logic levels into analog quantities” [25]. Deficiencies, such as false frequency locking, are direct results of not truly operating in the phase-domain.

A more rigorous treatment of the phase domain operation appears in Appendix B.

4.3 Reference Phase Retiming

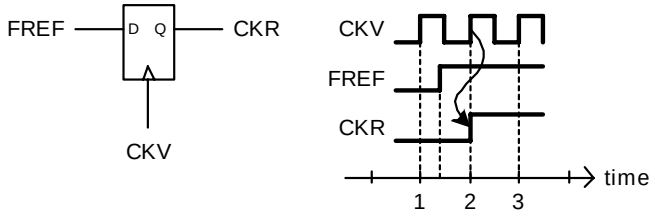

Figure 4.1. Concept of synchronizing the clock domains by retiming the frequency reference (FREF)

It must be recognized that the two clock domains as described in Sec. 4.2 are entirely asynchronous, and it is difficult to physically compare the two digital phase values at different time instances t_V and t_R without having to face metastability problems. (Mathematically, $\theta_V(i)$ and $\theta_R(k)$ are discrete-time signals with incompatible sampling times and cannot be directly compared without some sort of interpolation.) Therefore, it is imperative that the digital-word phase comparison be performed in the same clock domain. This is achieved by over-sampling the FREF clock by the high-rate DCO clock, CKV, (see Fig. 4.1) and using the resulting CKR clock to accumulate the reference phase $\theta_R(k)$ as well as to synchronously sample the high-rate DCO phase $\theta_V(k)$, mainly to contain the high-rate transitions. Since the phase comparison is now performed synchronously at the

rising edge of CKR, the equations Eq. 4.5 and Eq. 4.6 ought to be re-written as follows.

$$\theta_V(k) = k \tag{4.12}$$

$$\theta_R(k) = k \cdot N + \theta_0 + \varepsilon(k) \tag{4.13}$$

The set of phase estimate equations (Eq. 4.7 and Eq. 4.8) should be augmented by the sampled variable phase.

$$R_V(k \cdot T_R) = \sum_{l=0}^{i} 1 \Big|_{iT_V = kT_R} \tag{4.14}$$

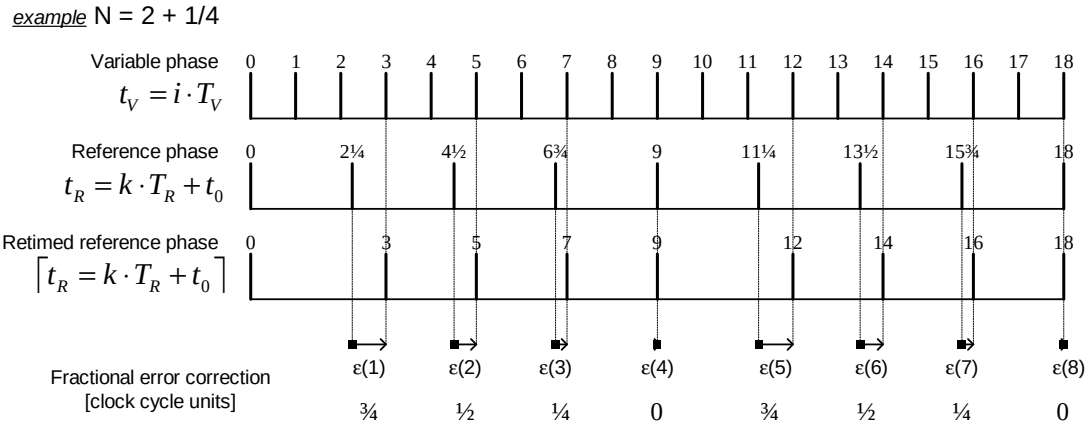

Figure 4.2. Fractional-N division ratio timing example

The index k is the now kth transition of the retimed reference clock CKR, not the kth transition of the reference clock FREF. By constraint, it contains an integer number of CKV clock transitions. $\varepsilon(k)$ is the CKV clock edge *quantization error*, in the range of $\varepsilon \in (0, 1)$, that could be further estimated and corrected by other means, such as the fractional error correction circuit. This operation is graphically illustrated in Fig. 4.2 as an example of integer-domain quantization error for a simplified case of the frequency division ratio of $N = 2\frac{1}{4}$. Unlike $\varepsilon(k)$, which represents rounding to the next DCO edge, conventional definition of the phase error represents rounding to the closest DCO edge and is shown as

$\phi(k)$ in Fig. 1.23 (page 35). An exact definition of the correction signal is not extremely important, as long as it is consistent and properly provides a negative feedback.

The reference retiming operation (shown in Fig. 4.1) can be recognized as a quantization in the DCO clock transitions integer domain, where each CKV clock rising edge is the next integer and each rising edge of FREF is a real-valued number. Since the system must be time-causal, quantization to the next DCO transition (next integer), rather than the closest transition (rounding-off to the closest integer), could only be realistically performed.

Table 4.1. ADPLL clock names

notation	name	frequency
CKV	variable (DCO) clock	f_V
FREF	frequency reference	f_R
CKR	retimed frequency reference clock	f_R

Table 4.1 summarizes major clocks used in the described architecture. Because of the clock edge displacement as a result of the retiming, the CKR clock is likely to have an instantaneous frequency different from its *average* frequency.

4.4 Phase Detection

Fig. 4.2 suggests that a new definition of the phase error is more appropriate for this architecture. Conventionally, it is defined as the difference between the reference and variable phases. Here, a third term will be added to augment the timing difference between the reference and variable phases by the ε correction.

$$\phi_E(k) = \theta_R(k) - \theta_V(k) + \varepsilon(k) \tag{4.15}$$

Additionally, dealing with the units of radian is not useful here because the system operates on the whole and fractional parts of the variable clock cycle and true unitless variables are more appropriate.

The initial temporary assumption made in Sec. 4.2 about the actual clock periods to be constant or time-invariant could now be relaxed at this point. Instead of producing a constant ramp of the detected phase error ϕ_E, the phase detector will now produce an output according to the real-time clock timestamps.

The phase error can be estimated in hardware by the phase detector operation defined by

$$\widehat{\phi}_E(k) = R_R(k) - R_V(k) + \varepsilon(k) \tag{4.16}$$

It is possible to rewrite Eq. 4.16 in terms of independent integer and fractional parts such that the integer part of the reference phase $R_{R,i}$ is added to the integer-only R_V, and the fractional part of the reference phase $R_{R,f}$ is added to the fractional-only ε.

$$\widehat{\phi}_E(k) = [R_{R,i}(k) - R_V(k)] + [R_{R,f}(k) + \varepsilon(k)] \tag{4.17}$$

In light of the above equation, the fractional error correction ε is to track the fractional part of the reference phase $R_{R,f}$, which is similar in operation to the variable phase R_V tracking the integer part of the reference phase $R_{R,i}$. Therefore, the three-term phase detection mechanism performs dual phase error tracking, with separate paths for the integer and fractional parts. The fractional-term tracking should be contrasted with the integer-term tracking due to the apparently different arithmetic operations. The former is complement-to-1 tracking (both fractional terms should ideally add to one), whereas the latter is 2's

complement tracking (both terms should ideally subtract to zero). The not-so-usual application of the unsigned 2's complement operation (complement-to-1) is a result of the ε definition and has no implications on circuit complexity. Even the resulting bias of one is easily absorbed by the variable phase accumulator.

Table 4.2. Phase detection signal names cross-reference

math notation	implem. notation	name	bus width
N	FCW	frequency command word	$W_I + W_F$
$\theta_R(k)$	-	reference phase	$-$
$R_R(k)$	PHR	reference phase (estimated)	$W_I + W_F$
$\theta_V(i)$	-	variable phase	$-$
$R_V(i)$	PHV	variable phase (estimated)	W_I
$\theta_V(k)$	-	sampled variable phase	$-$
$R_V(k)$	PHV_SMP	sampled variable phase (estimated)	W_I
$\varepsilon(k)$	PHF_F	fractional error correction	W_F
$-$	PHF_I	TDC edge skip	1
$\phi_E(k)$	PHE	phase error	$W_I + W_F$

Table 4.2 summarizes the major phase-domain signals and cross-references them with the implementational notation used later. For the implemented architecture, W_I = 8 and W_F = 15.

Fig. 4.3 illustrates a general block diagram of the phase detection mechanism of Eq. 4.16. It consists of the phase detector itself, which operates on the three phase sources: reference phase $R_R(k)$, variable phase $R_V(k)$ and the fractional error correction $\varepsilon(k)$. The actual variable phase $R_V(i)$ is clocked by the CKV clock of index i and it must be resampled by the CKR clock of index k. After the PHV resampling, all the three phase sources are synchronous to the CKR clock which guarantees the resulting phase error $\phi_E(k)$ to be also synchronous. The blocks are summarized in Table 4.3. An extra output bit from the

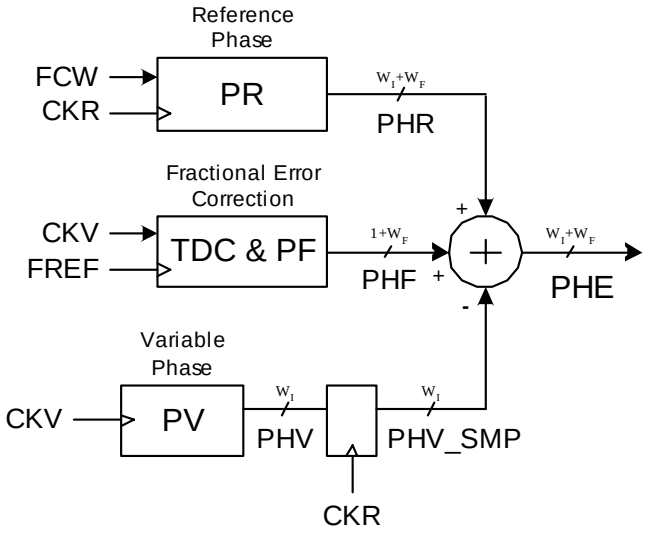

Figure 4.3. General block diagram of the phase detection

fractional error correction PF is due to metastability avoidance and will be explained in Sec. 4.8.

Table 4.3. Phase detection block names

PV	variable phase accumulator (incrementer)
PR	phase reference accumulator
PF	fractional error correction
TDC	time-to-digital converter (part of PF)
PD	phase detector

Fig. 4.4 reveals the internal structure of the phase detector circuit. All inputs are synchronous. The integer and fractional parts of the fixed-point phase signals are split and processed independently with proper bit alignment. The integer portion uses modulo arithmetic in which W_I-width rollovers are expected as a normal occurrence.

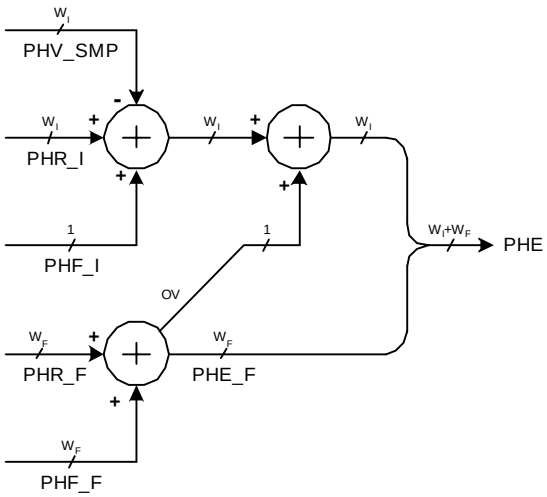

Figure 4.4. Phase detector structure

4.5 Integer-Domain Operation

If $\varepsilon(k)$ could not be estimated, this operation might be compensated in the phase domain by the ceiling operation of the reference phase.

$$\tilde{R}_R(k) = \lceil R_R(k) \rceil \tag{4.18}$$

The reference phase $R_R(k)$ is generally a fixed-point arithmetic signal with a sufficiently large fractional part to achieve the required frequency resolution as governed by Eq. 4.11. The ceiling operation of Eq. 4.18 could be easily implemented in hardware by discarding the fractional bits and incrementing the integer bits. This method improperly handles the case when the fractional part is zero but this has no practical consequences. In fact, the above statement agrees quite well with Eq. 4.17, where ε tracks the fractional part of the reference phase by perfectly complementing-to-1 it such that $R_{R,f}(k) + \varepsilon(k) = 1$.

$$\tilde{\phi}_E(k) = R_{R,i}(k) - R_V(k) + 1 \tag{4.19}$$

It should be emphasized that even though the integer-domain quantization error $\varepsilon(k)$ due to reference phase retiming in Eq. 4.13 is compensated by next-integer rounding operation (ceiling) of the reference phase in Eq. 4.18, the phase resolution still cannot be better than $\pm\frac{1}{2}$ of the DCO clock cycle. It means that the CKV clock drift within its full clock cycle cannot be detected, and consequently, compensated by the loop. In other words, the transfer function of the variable phase is quantized and slight phase error variations of less than CKV clock cycle are uncorrected. This could also be illustrated by the timing diagram of Fig. 4.1. If the FREF edge wanders between the first and second CKV edges, then this movement cannot be detected and the CKR always transitions on the second CKV edge. Consequence of operating in the integer domain without the ε correction is a larger phase noise at lower frequencies, which for many non-wireless applications might be quite adequate.

The integer phase detector in the synchronous digital phase environment can now be realized as a simple arithmetic subtractor of the DCO phase from the reference phase performed on every rising edge of the FREF clock.

$$\tilde{\phi}_E(k) = \tilde{R}_R(k) - R_V(k) = \lceil R_R(k) \rceil - R_V(k) = R_{R,i}(k) - R_V(k) + 1 \tag{4.20}$$

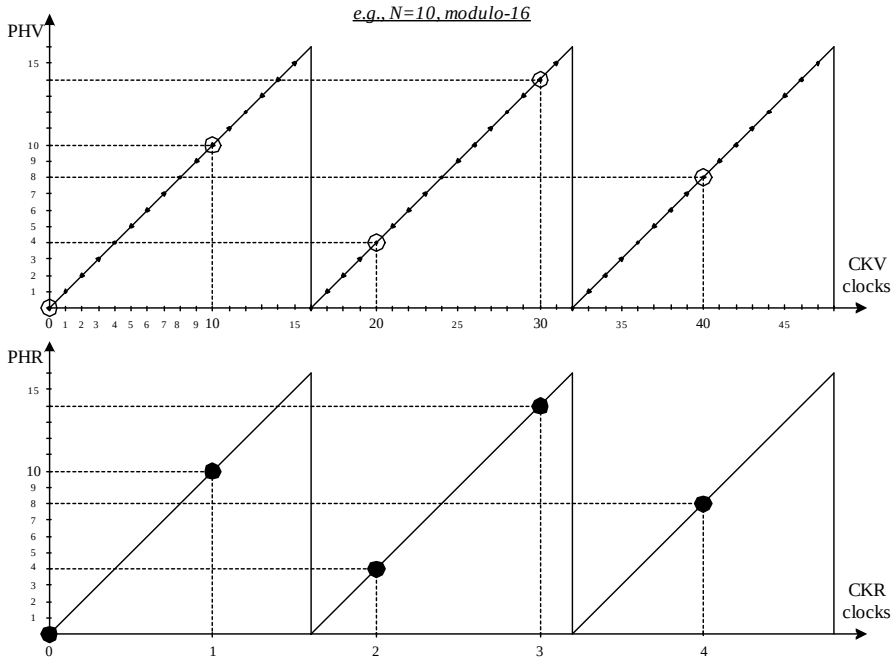

Figure 4.5. Modulo arithmetic of the reference and variable phase registers with zero phase alignment

4.6 Modulo Arithmetic of the Reference and Variable Phases

The variable and reference accumulators $R_V(i)$ and $R_R(k)$, respectively, are implemented in a modulo arithmetic in order to practically limit wordlength of the arithmetic components. Table 4.2 reveals that the integer part of the accumulators is W_I (=8 in this testchip) and the fractional part of the reference accumulator is W_F (=15 in this testchip). These accumulators represent the variable and reference phases, θ_R and θ_V, respectively, which are linear and grow without bound with the development of time. The registers, on the other hand, cannot hold unbounded numbers, so they are restricted to a small stretch of line from zero to infinity, which repeats itself indefinitely such that any such stretch is an alias of the fundamental stretch from 0 to 2^{W_I}.

A good example of such an approximation is a modulo arithmetic. Any accumulator

of length W_I, whose carry out bits are simply disregarded and which does not perform saturation, is a modulo-W_I accumulator. In fact, the modulo arithmetic is very natural in the digital logic. Both the variable and reference modulo-W_I accumulators are not absolutely linear in the strictest sense of the word. They are, however, linear in the local sense (for a constant frequency, of course). The phase equations Eq. 4.7 and Eq. 4.8 (page 109) are now rewritten to acknowledge the implicit modulo operation.

$$R_V(i+1) = mod\left(R_V(i) + 1, 2^{W_I}\right) \tag{4.21}$$

$$R_R(k+1) = mod\left(R_R(k) + FCW, 2^{W_I}\right) \tag{4.22}$$

Fig. 4.5 shows an example of modulo-16 operation with the frequency division ratio of N=10 and perfect alignment between the two phases. If the system is in a settled state, then both phases follow a sawtooth trajectory with the same speed. The variable phase PHV (top plot) traverses all integers at a very fast pace. The reference phase PHR (bottom plot), on the other hand, moves infrequently (10 times less often) but with large steps (10 times bigger). As a net effect, their traversal velocity is the same and their respective locations will correspond to the same register readout (0, 10, 4, 14, 8, ...), which happens at the same time on the CKR clock, thus producing zero phase error. Also shown every 10 CKV clock cycles is the sampling process of PHV during activity at PHR.

A situation similar to Fig. 4.5, but exhibiting a small phase offset of 3 is depicted in Fig. 4.6. For this case, the difference between R_R (3, 13, 7, 1, 11, ...) and R_V (0, 10, 4, 14, 8, ...) readout sequences should always be 3, and it is except for the forth comparison. Here, the error of -13 lies beyond the (-5, 5) linear range and performing modulo-10 arithmetic will get it to 3, which is in line with the rest.

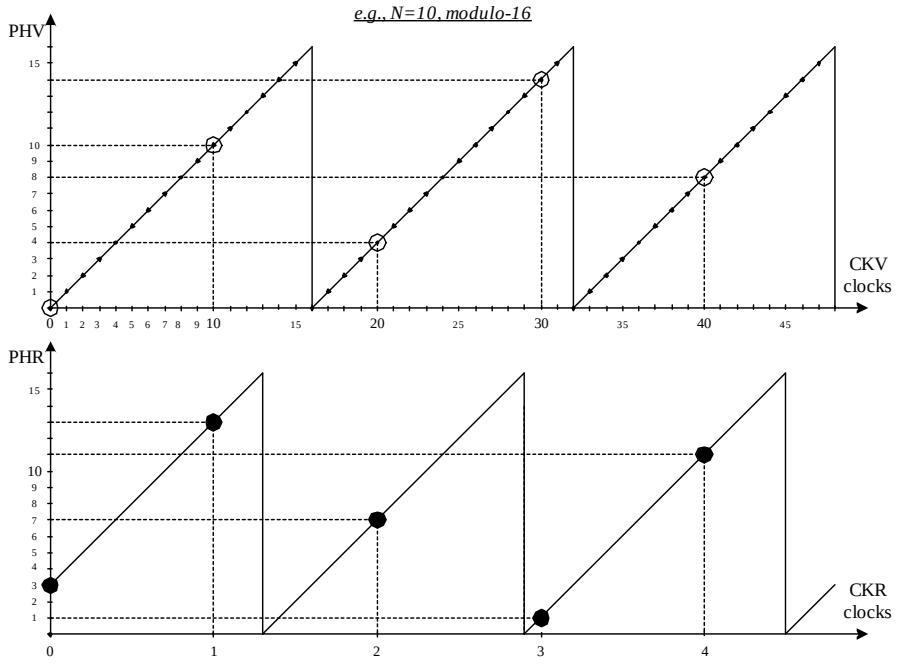

Figure 4.6. Modulo arithmetic for phase offset ϕ_E of 3

The modulo arithmetic on R_V and R_R could be visualized as two rotating vectors and the smaller angle between them constituting the phase error (Fig. 4.7). Both R_V and R_R are positive numbers and their maximum value possible without rollover depends on the counter width or integer part of the FCW, and equals 2^{W_I}. The phase error has the same range but is symmetric around zero, i.e., it is a 2's complement number. This figure also helps to understand that the phase detector is not only the arithmetic subtractor of two numbers, but also performs a cyclic adjustment so that, under no circumstance, the larger angle between the two vectors would be decided. This could happen if, for example, the larger vector appears just before the smaller vector which is already on the other side of the "zero" radius line. Because the PD output is a W_I-bit constrained signed number, the conversion is always implicitly made and the output lies within $(-2^{W_I-1}, 2^{W_I-1})$. Consequently, no extra hardware is required and none such is shown in Fig. 4.4.

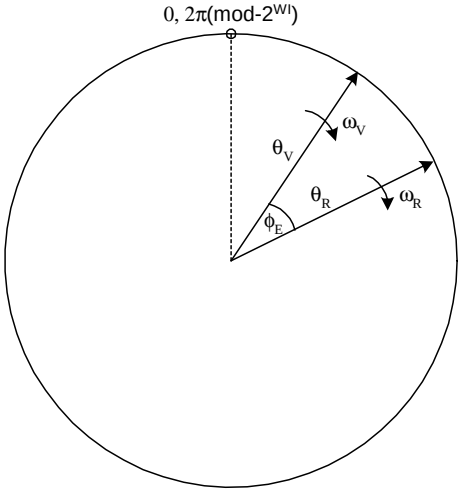

Figure 4.7. Rotating vector interpretation of the reference and variable phases

Due to the modulo arithmetic, there might be a possibility of aliasing for large values of FCW and large variable frequencies. For example, FCW = $2^{W_I} + x$ will alias to x. Similarly, $f_V = f_R \cdot (2^{W_I} + x)$ will alias to $f_V = f_R \cdot x$. The "Nyquist frequency" could be increased by making the integer width W_I sufficiently large. In our implementation, however, both variable and reference frequencies are tightly controlled and there is no possibility of getting into the foldover region.

4.7 Variable Phase Accumulator

The variable phase accumulator implements the DCO clock count incrementing as defined in Eq. 4.7 (page 109) with the rollover effect as described in Sec. 4.6.

The CMOS process is fast enough to perform an 8-bit binary incrementer at 2.4 GHz clock in one cycle using a simple carry-ripple structure. Critical timing of this operation would comprise a chain of seven half-adders and an inverter. However, for a commercial application it was necessary to add an extra timing margin in order to guarantee robust op-

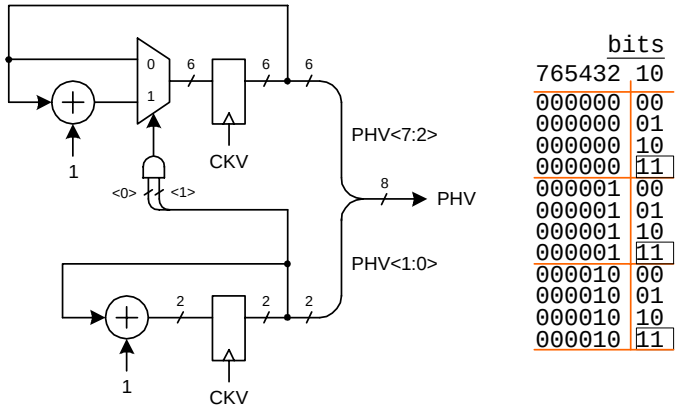

Figure 4.8. Variable phase incrementer with separate calculation between lower and higher order bits

eration with acceptable yield for all the process and environmental conditions, as well as anticipated clock distribution skew statistics. This extra margin was obtained by increasing the maximum operational speed through topological means. The carry-ripple binary incrementer was transformed into two separate smaller incrementers as shown in Fig. 4.8. The first incrementer operates on the two lower-order bits and triggers the higher order increment whenever its count reaches “11”. The second incrementer operates on the same CKV clock, but the 6-bit increment operation is allowed now to take four clock cycles. The long critical path of the 8-bit carry-ripple incrementer has thus been split into smaller parts allowing for the necessary timing margin.

It was discovered quite early in the design cycle that the critical path of the above partitioning was actually the increment trigger path from the lower-order registers, through the NAND gate and through the six single-bit multiplex select lines and terminating on the higher-order registers. The fanout of six was taking a disproportional toll on the delay, so a slight modification was done by retiming that control path. Final implementation version of the PV block is shown in Fig. 4.9. The triggering state is now one count earlier at “10”.

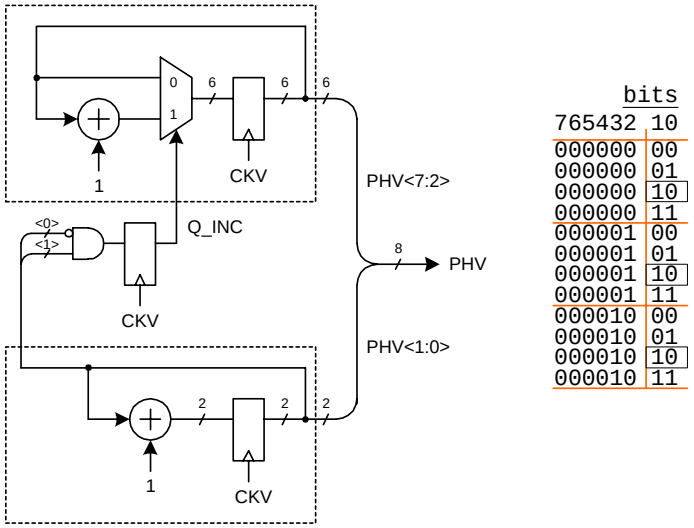

Figure 4.9. Implementation of the variable phase incrementer with higher order increment retiming

The chief improvement of this solution is that the register output Q_INC is now capable of driving six multiplex select lines.

4.8 Fractional Error Estimator – Time-to-Digital Converter

Due to the DCO edge counting nature of the PLL, the phase quantization resolution of the integer precision configuration, as described in Sec. 4.5, cannot be better than $\pm\frac{1}{2}$ of the DCO clock cycle. For wireless applications, a finer phase resolution might be required. This must be achieved *without* forsaking the digital signal processing capabilities. Fig. 4.3 shows the method by which the integer-domain quantization error $\varepsilon(k)$ gets corrected by means of a *fractional-period error estimator* (PF).

The fractional (sub-T_V) delay difference ε between the reference clock and the next significant edge of the DCO clock is measured using a *time-to-digital converter* (TDC) with a time quantization resolution of an inverter delay t_{inv} and the time difference is expressed

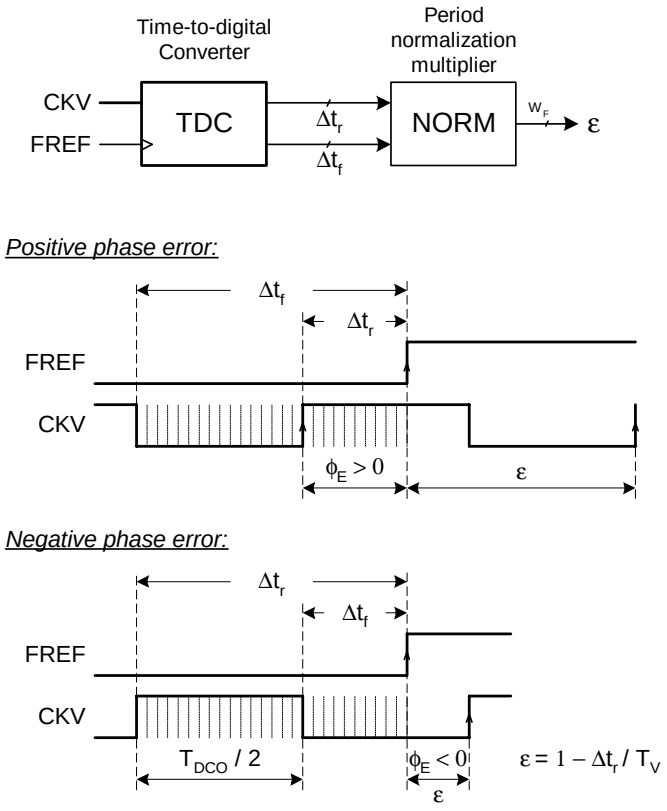

Figure 4.10. Fractional (sub-T_V) phase error estimation

as a fixed-point digital word. This operation is shown in Fig. 4.10.

The raw TDC output could not be used in the system in its integer form since the time resolution is a varying physical parameter, therefore, it has to be normalized by the oscillator clock period. Only the fractional error correction ε is used by the phase detector.

The maximum readily achievable timing resolution of the digital fractional phase detector is determined by a TDC inverter delay t_{inv} of the given CMOS process, which is about 30 ps for a typical L_{eff} = 0.08 μm process [8]. The string of inverters is the simplest possible implementation of time-to-digital conversion. In a digital deep-submicron CMOS process, the inverter could be considered a basic precision time-delay cell which has full digital-level regenerative properties.

It should be noted that it is possible to achieve a substantially better resolution than the inverter delay for the TDC function. An example is given in [56]. It utilizes a Vernier delay line with two non-identical strings of buffers. The slower string of buffers is stabilized by negative feedback through a delay line. The buffer time propagation difference establishes the resolution. The disadvantage of this approach is higher power consumption and extra analog circuitry.

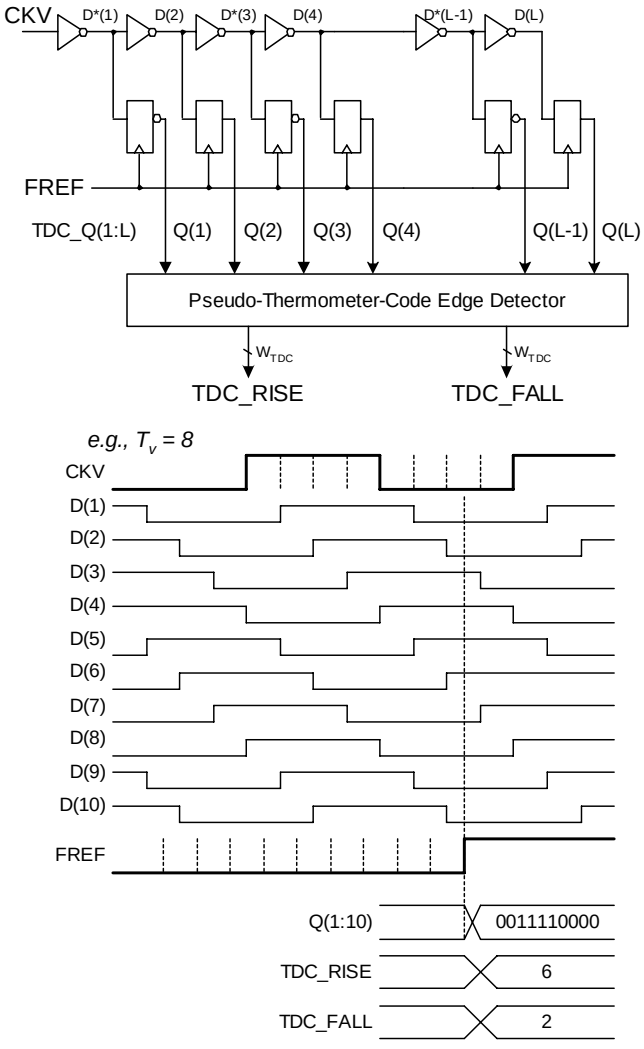

Figure 4.11. Time-to-digital Converter (TDC)

The digital fractional phase is determined by passing the DCO clock through a chain

of inverters (see Fig. 4.11), such that each inverter output would produce a clock slightly delayed than from the previous inverter. The staggered clock phases are then sampled by the same reference clock. This is accomplished by an array of registers, whose Q outputs form a pseudo-thermometer code. In this arrangement there will be a series of ones and zeros. In this example, the series of four ones (half period: T_V = 8 inverters) start at position 3 and extend to position 6. The series of four zeros follow starting at index 7. Position of the detected transition from 1 to 0 would indicate a quantized time delay Δt_r between the FREF sampling edge and the rising edge of the DCO clock, CKV, in t_{inv} multiplies. Similarly, position of the detected transition from 0 to 1 would indicate a quantized time delay Δt_f between the FREF sampling edge and the falling edge of the DCO clock, CKV. Because of the time-causal nature of this operation, both delay values must be interpreted as positive. This is fine if Δt_r is smaller than Δt_f. This corresponds to the positive phase error of the classical PLL loop in which the reference edge is ahead of the DCO edge and, therefore, the phase sign has to be negated. However, it is not so straightforward if Δt_r is greater than Δt_f. This corresponds to the negative phase error of the classical PLL loop. The time lag between the reference edge and the *following* rising edge of CKV must be calculated based on the available information of the delay between the *preceding* rising edge of CKV and the reference edge and the clock half-period, which is the difference $T_V/2 = \Delta t_r - \Delta t_f$. In general,

$$T_V/2 = \begin{cases} \Delta t_r - \Delta t_f & \Delta t_r \geq \Delta t_f \\ \Delta t_f - \Delta t_r & \text{otherwise} \end{cases} \tag{4.23}$$

The number of taps L required for the TDC of Fig. 4.11 is determined by how many

inverters are needed to cover the full DCO period.

$$L \geq \frac{\max(T_V)}{\min(t_{inv})} \tag{4.24}$$

If too many inverters are used then the circuit is more complex and consumes more power than necessary. For example, inverters 9 and 10 are beyond the first full cycle of eight inverters and are not needed since the pseudo-thermometer decoder is based on priority detection scheme and earlier bits would always be considered first. It is a good engineering practice, however, to keep some margin in order to guarantee proper system operation at the fast process corner and the lowest DCO operational frequency, even if it is below the operational band.

In this implementation, the conventional phase ϕ_E is not needed. Instead, Δt_r is used for the $\varepsilon(k)$ correction of Eq. 4.13 that is positive and $\varepsilon \in (0, 1)$. It has to be normalized by dividing it by the clock period (unit interval, UI) and complementing-to-1, in order to properly combine it with the fractional part of the reference phase output $R_{R,i}$. The fractional correction $\varepsilon(k)$ is represented as a fixed-point digital word.

$$\varepsilon(k) = 1 - \frac{\Delta t_r(k)}{T_V} \tag{4.25}$$

In practice, it is preferred to obtain the clock period T_V through longer-term averaging in order to ease the calculation burden and linearize the transfer function of $1/T_V$. The averaging time constant could be as slow as the expected drift of the inverter delay, possibly due to temperature and supply voltage variations. The instantaneous value of the clock period is an integer but averaging it would add significant fractional bits with longer operations.

$$\overline{T}_V = \frac{1}{N_{avg}} \sum_{k=1}^{N_{avg}} T_V(k) \tag{4.26}$$

It was found that accumulating 128 clock cycles would produce accuracy within 1 ps of the inverter delay. The length of the operation is chosen to be a power of 2 since the division by the number of samples N_{avg} could now be replaced with a simple right-shift.

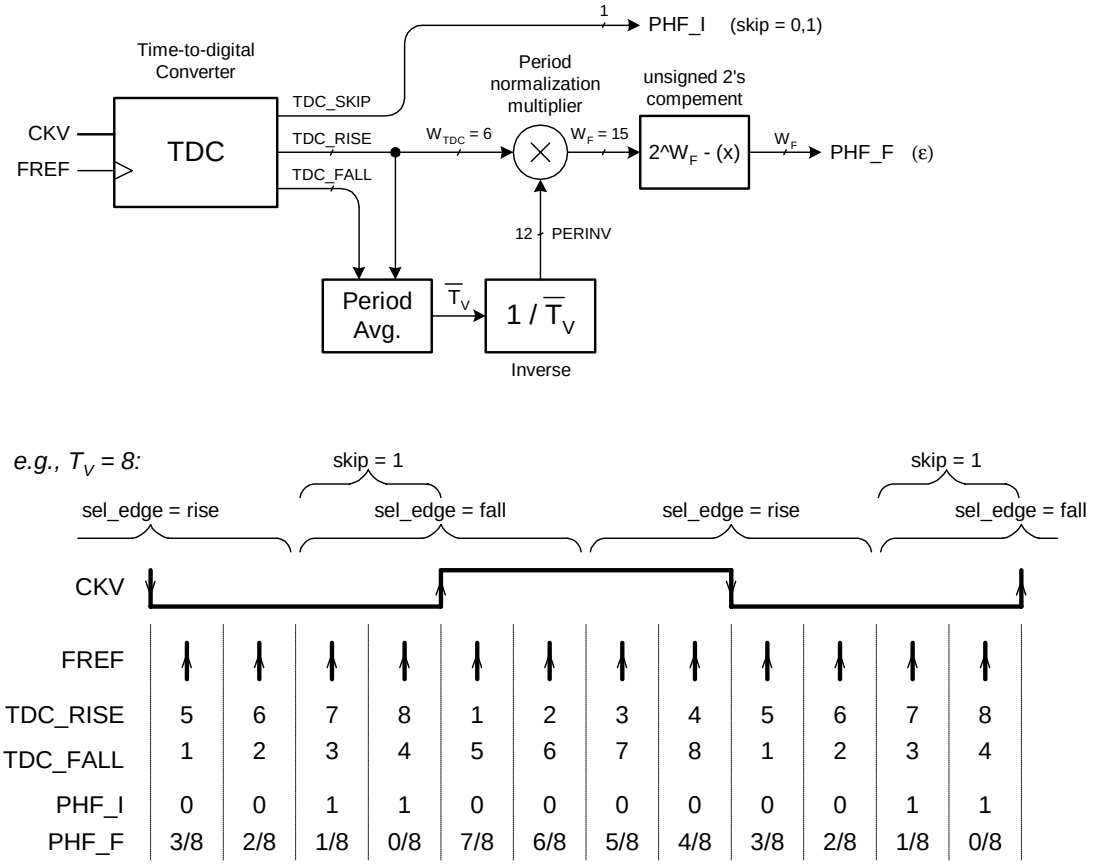

Figure 4.12. TDC normalization and edge-skipping operation

The actual fractional output of the ε error correction needs one extra bit due to the fact that the whole CKV cycle would have to be skipped if the rising edge of FREF transitions too close before the rising edge of CKV. As a safety precaution, the falling CKV edge would have to be used, and that is always resampled by the following rising edge of CKV. PHF_I is of the integer LSB weight. This scenario is illustrated in Fig. 4.12 in which there is a full-cycle skipping if FREF happens as close as two inverter delays before rising edge

of CKV. The clock skipping operation is further described in Sec. 4.10.

The proposed configuration of the ADPLL synthesizer was designed to work when CKV is much faster than FREF. This is typically the case in wireless communications, where FREF (created by an external crystal) is at most a few tens of MHz, and CKV (RF carrier) is in GHz range. In the implemented test chip, f_R = 13 MHz and f_V = 2.4—2.8 GHz, resulting in the division ratio N in the range of 180. The large value of N puts more emphasis on the CKV edge counting operation (Eq. 4.7), which is *exact*, and less emphasis on the ε determination (TDC operation), which is less precise due to the continuous-time nature of device delays. The proposed architecture would still work correctly if the N ratio is much smaller. The main requirement is that the resolution of the fractional error correction be at least an order of magnitude better than the CKV period.

Table 4.4. TDC signal names cross-reference

math notation	implem. notation	name	bus width
–	TDC_Q	sampled timing state vector	48
Δt_r	TDC_RISE	CKV rising edge to FREF	$W_{TDC} = 6$
Δt_f	TDC_FALL	CKV falling edge to FREF	$W_{TDC} = 6$
–	TDC_SKIP	TDC edge skip	1

Table 4.4 summarizes the major TDC signals and cross-references them with the implementational notation. TDC_Q is the pseudo-thermometer encoded timing state vector in Fig. 4.11 feeding the priority decoder. TDC_RISE and TDC_FALL are small integer quantization of the Δt_r and Δt_f time delays. They are outputs of the priority decoder. The TDC_Q bus width was conservatively chosen to be 48, so 6 bits are required to represent the decoded data.

In this implementation of TDC, we have chosen a symmetric sense-amplifier-based flip-flop (adapted from [57]) with differential inputs to guarantee substantially identical delays for rising and falling input data.

4.9 Fractional Division Ratio Compensation

The $\varepsilon(k)$ samples are roughly constant if the DCO clock period T_V is an integer division of the frequency reference clock period T_R. For a more general case where this ratio is fractional, the $\varepsilon(k)$ samples increase linearly within the modulo $(0, 1)$ range. (Fig. 4.2 shows an example.) This sample pattern could be easily predicted in digital form that closely corresponds mathematically to the well-known analog fractional phase compensation scheme of fractional-N PLL frequency synthesizers. However, the proposed architecture tracks this periodic pattern naturally in a complement-to-1 manner, so no extra processing is needed. In fact, unlike the conventional PLL-based frequency synthesizers, the proposed architecture was designed from ground up to handle a real-valued general frequency multiplication, limited only by wordlength of the reference phase accumulator.

The complement-to-1 fractional phase tracking works as follows. The fractional rotational speed is determined by the fractional part of the fixed-point FCW control word. Let's assume FCW_F = 1/4, as per example in Fig. 4.2. Accumulating it on every FREF cycle would circularly ramp up the fractional part of the fixed-point reference phase register according to sequence: 0, 1/4, 2/4, 3/4, 0, 1/4, etc. The fractional error correction ε is always referenced to the next CKV edge, not from the *previous* CKV edge as before, so it follows the opposite pattern by *ramping down* according to sequence: 4/4, 3/4, 2/4, 1/4,

4/4, 3/4, etc. Hence, ε tracks the fractional rotation in a complement-to-1 manner. This is also described mathematically by Eq. 4.15.

4.10 Frequency Reference (FREF) Retiming by the DCO Clock

The frequency reference clock retiming was conceptually presented in Fig. 4.1 (page 111). Mathematically, the flip-flop register performs the ceiling operation of the real-valued FREF timestamps through retiming it by the integer-valued CKV timestamps. The resulting CKR clock timestamps are integer-valued and, of course, synchronous to the DCO clock. Unfortunately, this simple and elegant mathematical model stumbles in face of the real-world limitation of metastability.

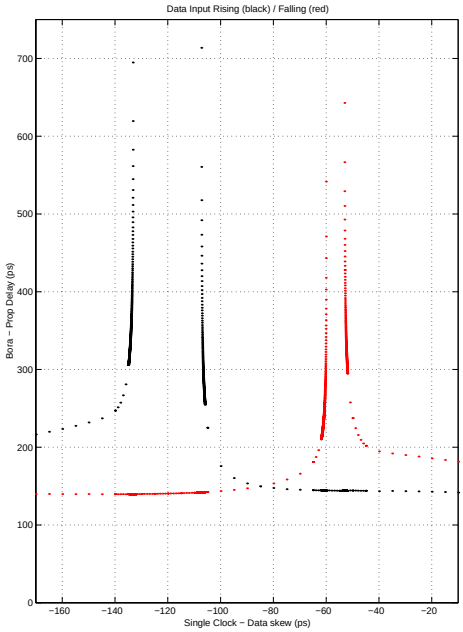

Figure 4.13. CLK-to-Q delay as a function of data-clock timing skew relationship – high performance standard-cell flip-flop (DTN20P)

Metastability is a physical phenomenon that limits performance of comparators and digital sampling elements, such as latches and flip-flops. It recognizes that it takes a nonzero

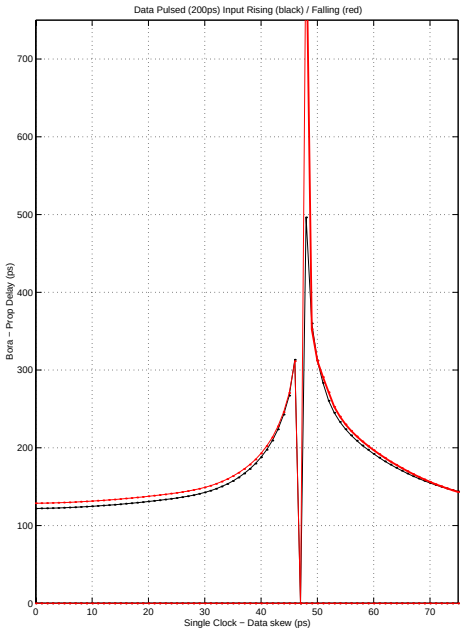

Figure 4.14. CLK-to-Q delay as a function of data-clock timing skew relationship – tactical flip-flop (“Bora”)

amount of time from starting of the sampling event in order to determine the input level or state [58] [59]. This resolution time gets exponentially larger if the input state change gets close to the sampling event. In the limit, if the input changes at exactly the same time as the sampling event, then it might theoretically take an infinite amount of time to resolve. During this time, the output can dwell in an illegal digital state, somewhere between 0 and 1. Fig. 4.13 shows a clock-to-output (CLK-to-Q) delay versus input-to-clock (D-to-CLK) skew of a high-performance flip-flop from the GS40 [8] digital standard-cell library distributed for the utilized C035 process. It reveals that the fall and rise transitions are not symmetric and the uncertainty zone differs in location by as much as 65 ps between the edges. This is clearly not acceptable if the required resolution would be below 65 ps. Fig. 4.14 shows the same plot for a tactical flip-flop (called “Bora”) developed specifically for 2.4 GHz digital operations. It enjoys an extremely small metastability window and

symmetric response for rise and fall transitions. It is further described in Sec. 4.10.1.

The metastable condition of the retimed reference clock CKR is not acceptable for two main reasons. First reason is general in nature: the metastability of any clock could introduce glitches and double clocking in the driven digital logic circuitry. Second reason is more specific: a non-bounded relationship between CKV and CKR violates the very principle of the common synchronous plane between them. It is quite likely that, within a certain metastability window between FREF and CKV, the CLK-to-Q delay of the flip-flop would make CKR to potentially span multiple of the DCO clock periods. This amount of uncertainty is not acceptable for the proper system operation.

The presented method stochastically solves the above metastability problem of the frequency reference retiming of Fig. 4.1.

4.10.1 Sense-amplifier-based Flip-flop

Fig. 4.15 shows a schematic of the tactical sense-amplifier-based “Bora” flip-flop ([57]). This topology was compared against other flip-flop architectures for the similar current consumption and was found to be the fastest. It was used by the author on the previous CMOS technology nodes for an ultra-high-speed read-channel as a sequential element of a *finite-impulse response* (FIR) filter [43] [44] and a companion *least-mean square* (LMS) adaptation algorithm circuit [45].

The tactical flip-flop consists of two blocks: a pulse generator *sense amplifier* (SA) (top of the figure) and a symmetric slave *set-reset* (SR) latch (bottom of the figure). It is different from the conventional master-slave latch combination in that the pulse generator

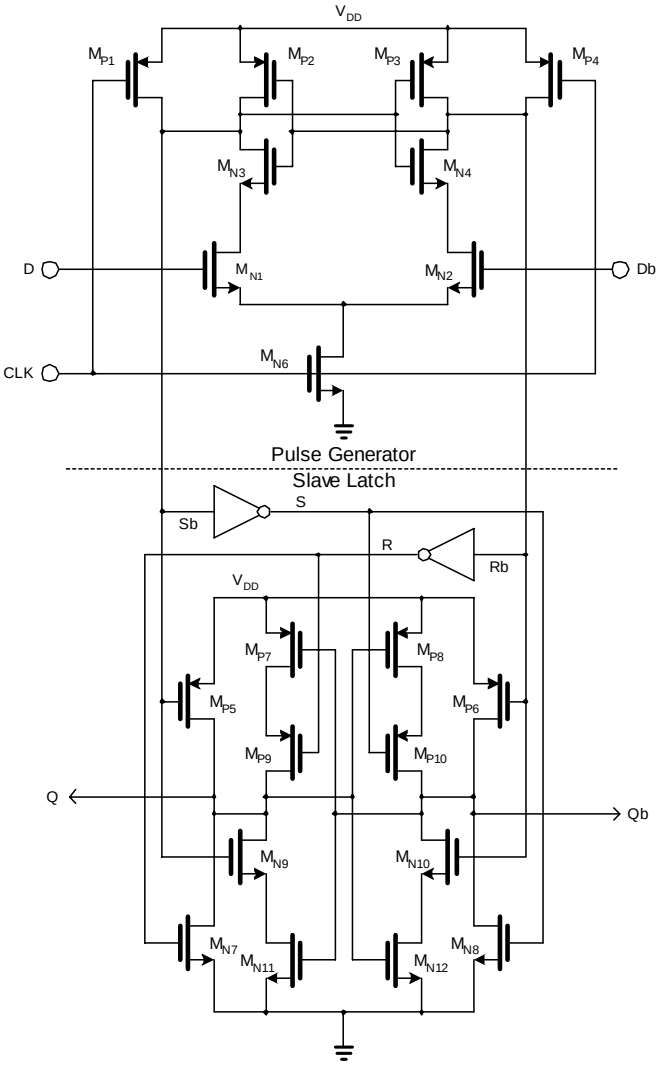

Figure 4.15. Schematic of the tactical “Bora” flip-flop

is not level sensitive but generates a pulse of sufficient duration on either Sb or Rb ouptuts as a result of changes in clock and data values. The SR latch captures the transition and holds the state until the next rising edge of the clock. After the clock returns to its inactive state, both Sb and Rb outputs of the SA stage assume logic high value.

4.10.2 General Idea of Clock Retiming

The proposed method of the frequency reference retiming by the oscillator clock is the key idea of the all-digital synthesizer architecture that allows it to work in a clock-synchronous manner.

This method is a general solution to the problem of retiming a lower frequency timing signal (or clock) by a higher frequency clock when both signals are asynchronous to each other. Normally, this problem is solved by passing the lower frequency signal through a series of flip-flops (registers) that are clocked by the higher frequency clock. There is a certain probability of a metastability condition per register stage and the overall probability of metastability at the system output decreases exponentially with each register stage. The number of register stages is established such that the *mean time between failures* (MTBF) rate is acceptably large. Unfortunately, the metastability condition brings the timing uncertainty of one high speed clock cycle (or possibly higher) since, during metastability, the output could be resolved at a given clock cycle or at the next. Even though the output levels are defined, this timing error is unacceptable in some applications.

In this architecture, the timing signal is the FREF frequency reference, the oversampling clock is the digitally-controlled oscillator output. Obviously, both of these signals are completely asynchronous to each other. In the ADPLL, we need to represent the reference phase, variable (DCO) phase, phase error and all other phase signals as fixed-point digital word signals that are synchronous to each other and which cannot be corrupted by noise. If this is accomplished, then the phase error could be simply an output of a synchronous arithmetic subtractor used as a phase detector. Thus retimed reference is used as a synchronous

system clock in the whole design.

The solution could be summarized as follows: Use the previously described *time-to-digital converter* (TDC) circuit to determine which edge of the higher frequency clock (oversampling clock) is farther away from the edge of the lower frequency timing signal. At the same time, the oversampling clock performs sampling of the timing signal by two registers: one on the rising edge and the other on the falling edge. Then, the register of ”better quality” retiming, as determined by the fractional phase detector decision, is selected to provide the retimed output.

The asynchronous retiming mechanism can be summarized as follows:

1. Sample the frequency reference clock (FREF) using both edges of an oversampling clock (CKV) that is derived from the controllable oscillator, such as DCO. The sampling is performed by a pair of clocked memory elements, such as flip-flops or registers, one operating on the positive or rising transition of the CKV clock, and the other operating on the negative or falling transition of the CKV clock. The effect of sampling the FREF clock by the CKV clock is to retime the FREF to either rising or falling edge of the CKV clock.

2. Delay both retimed clocks by a controllable amount.

   (a) The delay operation could be accomplished by inserting shift register stages clocked by CKV or its derived clock.

3. Retime the falling retimed clock with the rising edge of the CKV clock. If this path is chosen, the CKR clock will always be synchronous to the rising edge of the CKV clock and the ε definition will be consistent.

4. Feed the outputs of both rising and falling retimed paths into a selection element, such as a multiplexer.

    (a) Output of the multiplexer could be further resampled by the CKV clock or a clock derived from it.

5. Use the mid-edge detector to select one of the two retimed clock paths that would be sufficiently far away from metastability condition.

    (a) The mid-edge detector could be a time-to-digital converter (TDC) clocked by FREF that would determine which one of the two CKV edges lies father away (or just far enough – e.g., a few inverter delays) from the FREF edge. Use the selection choice to amend the TDC output if used as part of the phase-domain *all-digital PLL* (ADPLL) loop.

    (b) The mid-edge detector could be a register sampling the delayed CKV clock by the FREF clock such that the selected retimed clock would be sufficiently away from the metastability condition.

4.10.3 Implementation

Fig. 4.16 shows an implementation of the key idea that allows for the ADPLL to synchronize reference and variable clock planes. Its sole purpose is to resample the reference clock with a variable DCO clock as required in Fig. 4.1 in a manner that would stochastically avoid metastability. It does so by resampling the FREF by both edges of CKV and choosing the one that has a larger (or sufficiently large) clock separation. The select decision is based on the existing TDC output (a TDC portion of Fig. 4.11 is replicated here for clarity),

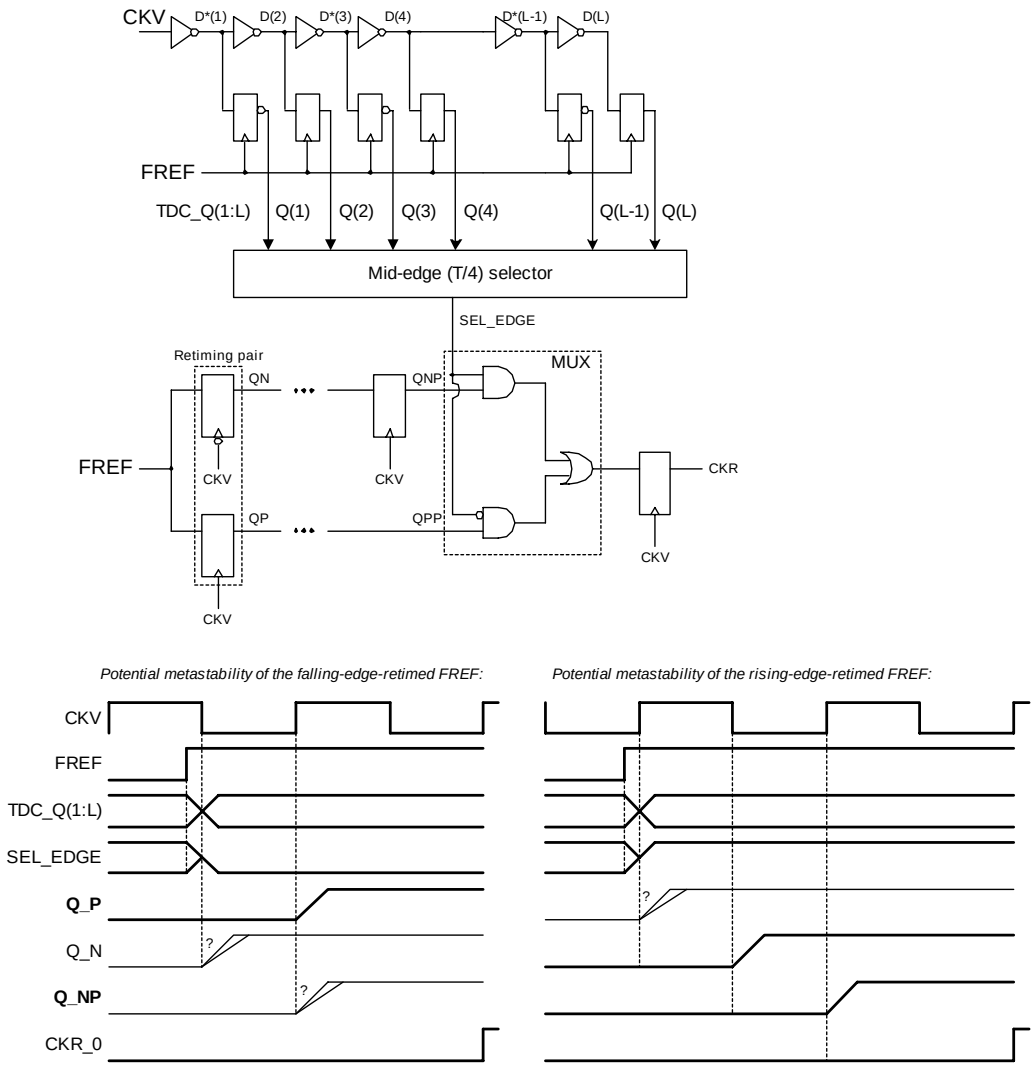

Figure 4.16. FREF retiming by the DCO clock

thus little additional cost is incurred. In case of FREF resampling with the falling edge of CKV, an additional rising-edge CKV retiming is required to satisfy the consistent definition of ε and in order for the CKR to have a fixed timing relationship with the rising edge of CKV. This figure also shows two examples of a potential metastability and how the right path selection is made in each case.

The following description of Fig. 4.16 is based on the asynchronous retiming mechanism steps of Sec. 4.10.2. On the rising edge of the FREF clock, the CKV delay state

is being sampled (Step 5a). A fraction of the DCO cycle afterwards, the FREF clock is sampled by both falling and rising edges of the CKV clock using a pair of flip-flops (Step 1) producing QN and QP outputs, one of which might have a potential for metastability condition. The QN and QP signals are delayed by several reclocking stages (Step 2) and the end of the QN path is further resampled by the rising edge (Step 3). Output of the mid-edge selector, SEL_EDGE, selects one of the two paths which is determined to be far enough from the metastability condition (Step 4). The selected multiplexer output is further resampled (Step 4a).

In this implementation, five additional pairs of rising/falling reclocking stages are inserted to allow extra time (in multiple of CKV clock cycles) for the SEL_EDGE selection signal to resolve its metastability. This number of reclocking stages was calculated based on the criteria of metastability such that the selection signal must be valid and free of metastability by the time the two reclocked signals arrive at the multiplexer input. It should be emphasized that not resolving the selection signals on time does not mean the final CKR clock will be metastable. Since the two reclocked signals pass through multiple FF stages, they will be essentially free of metastability at the MUX inputs (MTBF increases exponentially with the number of stages). If the selection signal has a valid digital level but does not come on time, then there is a 50% probability that the wrong decision might be made and a one-time ± 1.0 perturbation to ε injected into the PLL loop, from which the system can recover. Satisfying the above condition of stable FF output is quite easy with a sufficiently high gain on the selection signal path. This ensures that the output does not chatter but stays at a legal level until the internal bistable state gets resolved, even though the latter may dwell in an illegal zone until the final moment.

The real metastability can be injected into the CKR output if the reclocked signals arrive at the MUX input at the same time as the selection control. Probability of this happening is much smaller than in the above case, but the effects can be devastating if the metastable clock finds its path into the digital baseband controller and causes false clocking. There, the system-wide reset might be necessary to recover. For this reason, the output CKR clock gets further retimed by a lower-frequency CKV-derived clock that makes the final MTBF vanishingly small. It should be noted that because the ADPLL state machines get synchronously reset every packet (up to 3 ms of duration) any effect due to glitches or illegal states there has a very limited lifespan.

Fig. 4.17 shows the "raw" TDC output (TDC_Q) as well as the *extended* normalized fractional error correction ε (PHF_I and PHF_F) for all possible discrete timing between CKV and FREF when $T_V = 16 \cdot t_{inv}$. The timing diagram at the bottom is similar to the timing diagram in Fig. 4.12. The number of distinct bins, into which the rising edge of FREF can fall relative to CKV, has been increased from 8 to 16 simply because now there are not 8 but 16 inverters per CKV clock cycle.

A high degree of redundancy in the pseudo-thermometer-encoded TDC output vector could be exploited to obtain an extra error correction and metastability resolution. A majority voting scheme of at least three neighboring outputs was considered. In the end, however, in this application a simple extraction of the selection control is chosen as the best solution. This timing diagram also shows how a certain single TDC register output (TDC_Q(5) in this example) could be used for the edge selection. Inspecting various TDC_Q outputs, we can see that the 5th bit is best centered around the rising edge of CKV: it has four zeros before and for zeros after. In this case, the value of zero on that bit must be assigned to

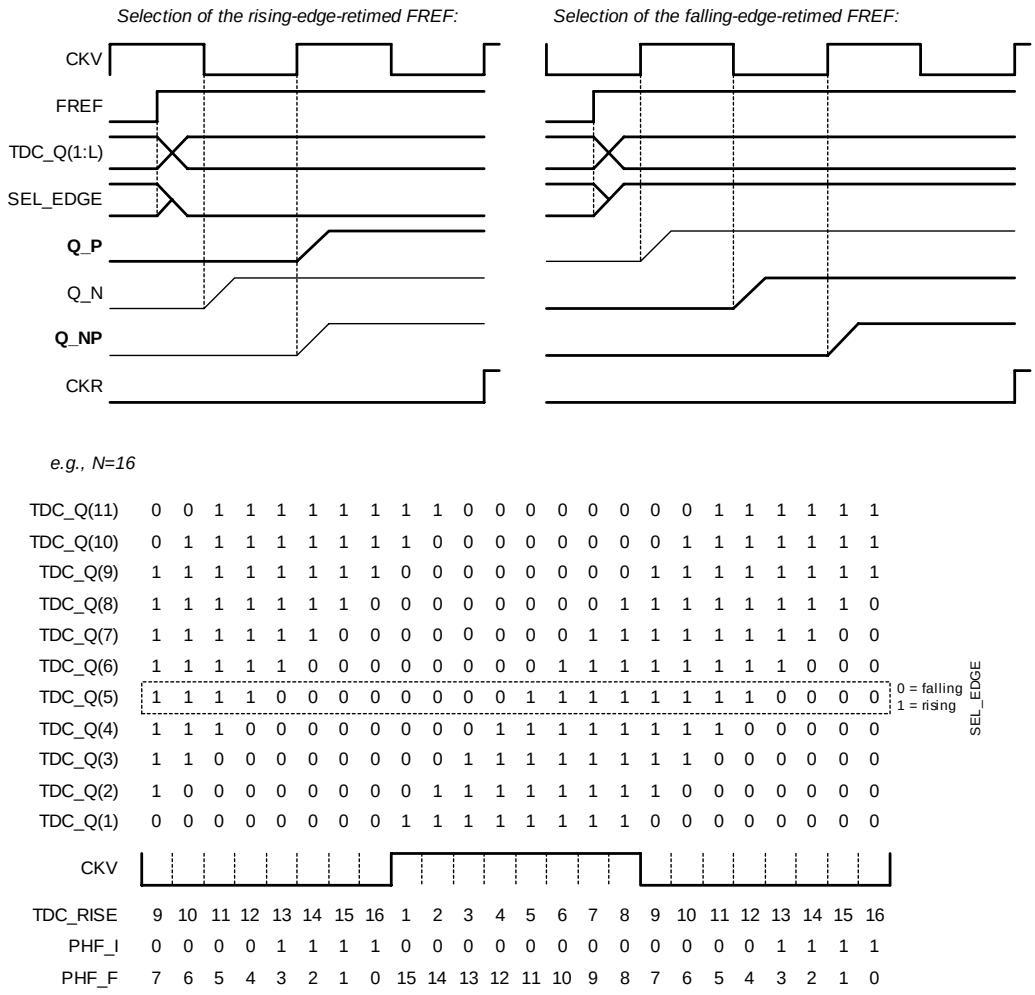

Figure 4.17. DCO clock retiming details

select the other edge, i.e. falling. Inspecting now the 5th bit around the falling CKV edge we can see that it is centered in a group of eight ones. In this case, the value of one on that bit must be assigned to select the other edge, i.e. rising. From the above analysis we can see that the selection of the proper CKV edge to sample the FREF signal *without* metastability could be simply accomplished by inspecting a single TDC_Q(5) bit of the sampling state vector. A question might arise here of what would happen if the TDC_Q(5) signal itself is metastable and cannot be resolved in a timely manner. Inspecting the region around $0 \rightarrow 1$ and $1 \rightarrow 0$ transitions on that bit, we can see that it is free from any activity on the CKV

clock line. The closest CKV transitions happen about a quarter of the CKV clock cycle, which is sufficiently long time to be considered “free of metastability”. In this region any of the two decisions could be made, so the SEL_EDGE selection value is immaterial.

It is also shown in Fig. 4.12, Fig. 4.16 and Fig. 4.17 that it is possible to skip the whole CKV clock cycle if the FREF rising edge is close to the CKV rising edge. If the FREF edge happens to be in any of the four bins before the rising CKV edge, TDC_Q(5) will select the falling edge sampling always followed by the rising edge resampling. Now the rising FREF edge is over full clock cycle away from the *final* rising CKV edge, implying that $\varepsilon > 1$. Consequently an extra bit of information is sent to the phase detector. This could be considered redefining the range of the fractional error correction to be $\varepsilon \in (0, 2)$.

4.11 TDC Resolution Effect on the Estimated Frequency Resolution

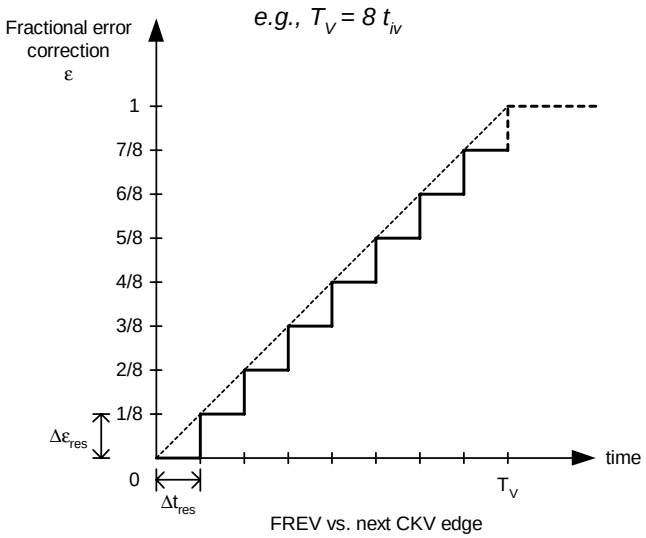

Figure 4.18. TDC quantized transfer function

In a conventional PLL loop, the phase detector is, at least theoretically, a linear device

whose output is proportional to the timing difference between the reference and the feedback oscillator clocks. In the proposed all-digital implementation, the ε fractional phase error correction is also linear but is quantized in Δt_{res} time units, where $\Delta t_{res} \approx t_{inv}$. Fig. 4.18 shows the quantization effects of the ε transfer function of Eq. 4.25 (page 128). The TDC quantum step Δt_{res} determines the quantum step of the normalized fractional error correction which is expressed as $\Delta\varepsilon_{res} = \Delta t_{res}/T_V$ in normalized units. The transfer function has a negative bias of $\Delta t_{res}/2$ but it is inconsequential since the loop will compensate for it automatically.

As noted previously in Sec. 3.9 while referring to Fig. 3.24 (page 105), the purpose of phase detection mechanism in the proposed architecture is to convert the accumulated timing deviation TDEV, which is a pure time-domain quantity, into a digital bit format. At the same time, as the TDC transfer function in Fig. 4.18 confirms, the phase detector is to perform the output normalization such that TDEV = T_V corresponds to unity.

Under these circumstances, the phase detector output ϕ_E could be interpreted as a frequency deviation estimator (from a center or "natural" frequency) of the output CKV clock and normalized to frequency reference f_R. Within one reference clock cycle, $T_R = 1/f_R$,

$$\widehat{\Delta f_V} = \phi_E \cdot f_R \tag{4.27}$$

The above estimate increases linearly with the number of reference cycles.

The resolution of the phase detector is directly determined by the TDC resolution, $\Delta\phi_{E,res} = \Delta\varepsilon_{res}$. Adopting the frequency estimation view of the phase detector, the quantum step in the f_V frequency domain, per reference cycle, would be

$$\Delta f_{PD,res} = \Delta\varepsilon_{res} \cdot f_R = \left(\frac{\Delta t_{res}}{T_V}\right) \cdot f_R = \left(\frac{\Delta t_{res}}{T_V}\right) \cdot \frac{1}{T_R} \tag{4.28}$$

For example, assuming Δt_{res} = 30 ps, f_V = 2.4 GHz and f_R = 13 MHz, the resulting frequency estimate quantization level of a single FREF cycle is $f_{V,res}$ = 935 kHz. Obviously, this is unacceptably high for our application, so an advantage is taken here of the fact that the frequency is a phase derivative of time and the frequency resolution could be enhanced with a longer observation period, i.e., over multiple FREF cycles. In this case, Eq. 4.28 could be modified by multiplying T_R by the number of FREF cycles.

4.12 Loop Gain Factor

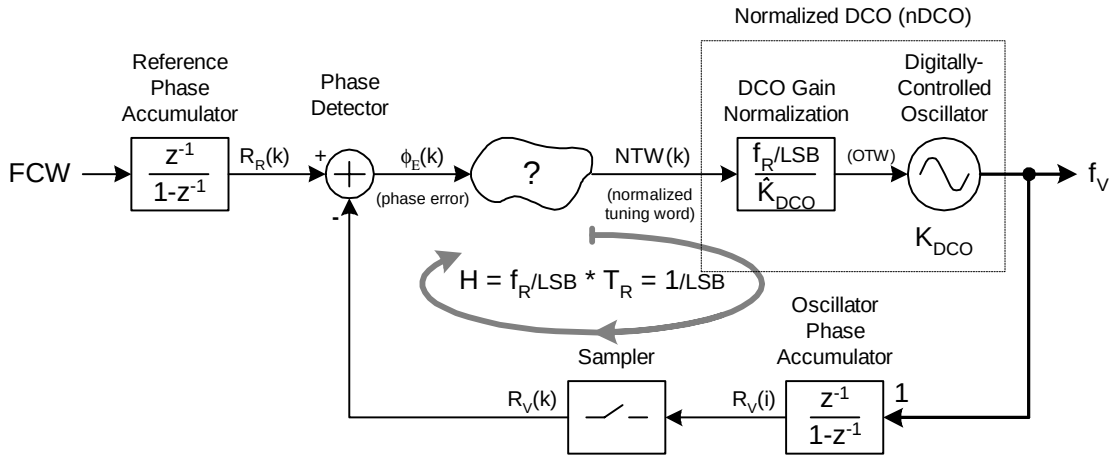

Figure 4.19. Calculating closed-loop gain of the ADPLL (assuming $K_{DCO} = \widehat{K}_{DCO}$)

Fig. 4.19 assembles various phase-domain blocks introduced thus far. The normalized DCO circuit was covered in Chapter 3. The oscillator phase accumulator R_V counts the number of rising edges of the DCO clock. Together with the following sampler, they directly implement Eq. 4.14 (page 112). The reference phase accumulator R_R addresses Eq. 4.8 (page 109) by accumulating FCW on each rising edge of the retimed frequency reference. The phase detector directly implements Eq. 4.16 (page 114). What is still missing at this point is closing of the loop such that the phase error ϕ_E would be used to correct for

the frequency and phase drift of the oscillator.

As stated before, the phase error ϕ_E is expressed in units of the reference frequency f_R. Similarly, the normalized DCO (nDCO) control is also normalized to f_R. As also revealed in Fig. 4.19, the frequency transfer function from the nDCO input to the phase detector output is unity. It means that a frequency perturbation Δf_V will be correctly estimated, within the quantization resolution $\Delta f_{V,res}$, in one FREF cycle T_R. Unfortunately, as shown in the example of Sec. 4.11, the quantization of the frequency estimator is excessive. For stability reasons, the output of the phase detector cannot be directly connected to the nDCO input. It must be attenuated.

Let's introduce a scaling factor α that controls of how much attenuation the phase error must undergo before affecting the nDCO frequency. In frequency domain, it controls fraction of the frequency detected in response to the frequency changed at the nDCO input. In time domain, it controls how much timing attenuation within a reference clock cycle one should see at the nDCO input in response to a certain change in the nDCO input in the previous clock cycle. For an overdamped system, $\alpha < 1$, for a critically damped system, $\alpha = 1$, and for an underdamped system $\alpha > 1$. This also establishes the loop stability.

$$\Delta f_{V,res} = \alpha \cdot \Delta\varepsilon_{res} \cdot f_R = \alpha \cdot \left(\frac{\Delta t_{res}}{T_V}\right) \cdot f_R = \alpha \cdot \left(\frac{\Delta t_{res}}{T_V}\right) \cdot \frac{1}{T_R} \tag{4.29}$$

Table 4.5 relates (based on Eq. 4.29) the DCO frequency quantization to the TDC resolution. One can see a tradeoff between the quantized loop frequency resolution and the loop dynamics. The frequency quantization of the DCO could be made finer at the cost of making the loop slower or of lower bandwidth.

Table 4.5. DCO frequency quantization

TDC resolution Δt_{res}	Center frequency f_0	loop gain α	frequency quantization Δf_V
30 ps	2450 MHz	1	955.9 kHz
30 ps	2450 MHz	1/8	119.5 kHz
30 ps	2450 MHz	1/32	29.87 kHz
30 ps	2450 MHz	1/256	3.733 kHz
30 ps	2450 MHz	1/1024	0.933 kHz

4.13 Phase Error Dynamic Range

A steady-state phase error signal in this architecture also indicates the frequency offset from the center DCO frequency. In order to demonstrate it, one should note that the tuning word directly sets the DCO operating frequency and there exists a proportionality factor α between the normalized tuning word and the phase error, as shown in Fig. 4.19. Consequently, the steady-state frequency offset could be expressed as

$$\Delta f_V = \phi_E \cdot \alpha \cdot f_R \tag{4.30}$$

Eq. 4.30 should be contrasted with Eq. 4.27, which is only a single reference cycle estimate that is a part of the detection process. Eq. 4.30 could also be explained from another perspective. If there happens a sudden frequency deviation Δf_V at the output, then in one FREF cycle the phase detector will estimate it as $\widehat{\Delta f_V}$ per Eq. 4.27. This will correct the DCO frequency by $\Delta f_V \cdot \alpha$. In the second reference cycle, the detected frequency at the phase detector will be $\Delta f_V(2-\alpha)$ leading to the DCO correction of $\Delta f_V(2-\alpha)\cdot\alpha$. This process of geometric sequence will continue until the DCO frequency gets fully corrected, and the phase detector develops the $\Delta f_V/\alpha$ offset.

The loop gain factor α also controls the dynamic range of the phase error. It was men-

tioned in Sec. 4.6 that both the variable phase R_V and reference phase R_R are wordlength limited to W_I bits in the integer part. This limitation carries over to the phase error ϕ_E. Consequently, the phase range of the ϕ_E signal is $(-2^{W_I-1}, 2^{W_I-1})$ times 2π rad of the CKV clock cycle and limits the dynamic range of the DCO to

$$\Delta f_{V,range} = 2^{W_I} \cdot \alpha \cdot f_R \tag{4.31}$$

4.14 Phase-Domain ADPLL Architecture

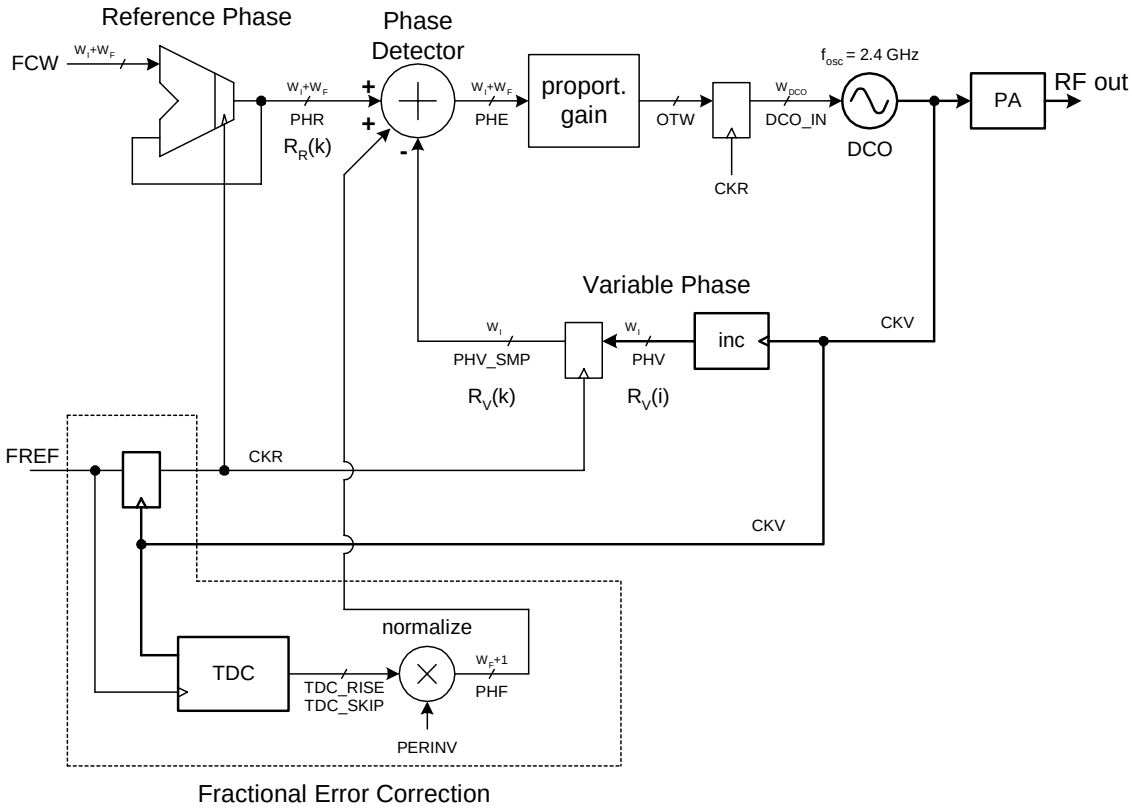

Figure 4.20. Phase-domain all-digital synchronous PLL synthesizer

At this point, all the major pieces comprising the phase-domain *all-digital PLL* (ADPLL) frequency synthesizer have been introduced. The block diagram is presented in

Fig. 4.20. The PLL loop is a fixed-point phase-domain architecture whose purpose is to generate an RF frequency in the 2.4 GHz unlicensed band for the BLUETOOTH standard.

The underlying frequency stability of the system is derived from a frequency reference FREF crystal oscillator, such as a 13 MHz *temperature-compensated crystal oscillator* (TCXO) for the GSM system (GSM is an acronym for Global System for Mobile Communications).

The chief advantage of keeping the phase information in fixed-point digital numbers is that, after the conversion, it cannot be further corrupted by noise. Consequently, the phase detector could be simply realized as an arithmetic subtractor that performs an exact digital operation. Therefore, the number of conversion places is kept at minimum: a single point where the continuously-valued clock edge delay is compared in a *time-to-digital converter* (TDC).

It should be emphasized here that it is very advantageous to operate in the phase domain for several reasons. First, the phase detector used is not a conventional correlative multiplier (such as shown in Fig. 1.20 on page 31) generating reference spurs. Here, an arithmetic subtractor is used and it does not introduce any spurs into the loop. Second, the phase domain operation is amenable to digital implementations, which is quite opposite to the conventional approach. Third, the dynamic range of the phase error could be made arbitrarily large simply by increasing wordlength of the phase accumulators. This compares favorably with the conventional implementations, which typically are limited only to $\pm 2\pi$ of the compare frequency with a three-state phase/frequency detector [26]. Fourth, the phase domain allows algorithmically higher precision than operating in the frequency domain, since the frequency is a time derivative of phase, and a certain amount of phase

quantization (such as in TDC) decreases its frequency error with the lapse of time.

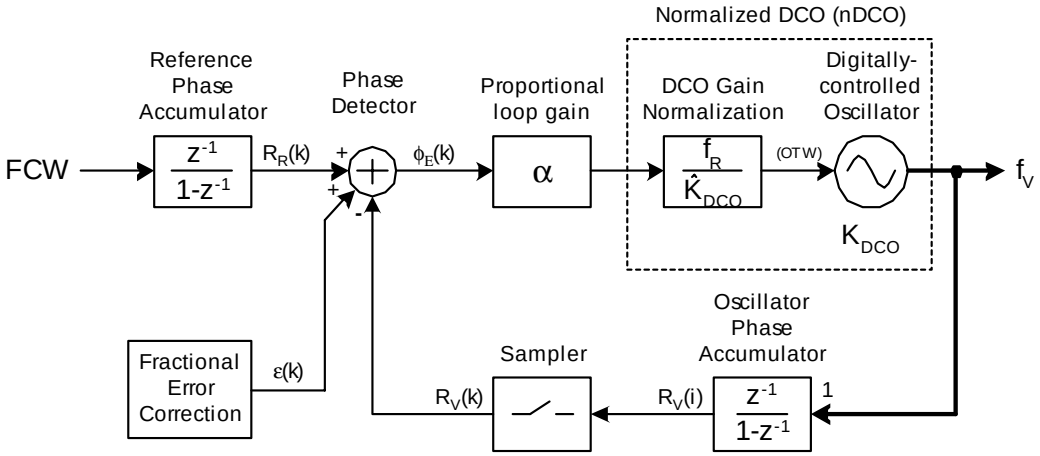

Figure 4.21. Phase-domain all-digital synchronous PLL synthesizer (a different perspective)

Fig. 4.21 shows the phase-domain ADPLL architecture from a different perspective. The central element is the 2.4 GHz *digitally-controlled oscillator* (DCO) and the PLL loop build around it is fully digital and of type-I (i.e., only one integrating pole due to the DCO frequency-to-phase conversion). Type-I loops generally feature faster dynamics and are used where fast frequency/phase acquisition is required or direct transmit modulation is used. The loop dynamics are further improved through avoiding using a loop filter. The issue of the reference feedthrough that affects classical charge-pump PLL loops and shows itself as spurious tones at the RF output is irrelevant here because, as discussed before, a linear, and not a correlation phase detector is used.

In addition, unlike in type-II PLL loops, where the steady-state phase error goes to zero in face of a constant frequency offset (i.e., frequency deviation between the actual and center DCO frequencies), the phase error in type-I PLL loop is proportional to the frequency offset. However, due to the digital nature of the implementation, this does not limit

the dynamic range of the phase detector or the maximum range of the DCO operational frequency.

Feeding to the nDCO is the normalized loop gain α multiplier. The normalized proportional loop gain constant α is a programmable PLL loop parameter that controls the loop bandwidth. It is defined as how much phase attenuation is expected to be observed at the phase detector output in response to a certain change in the phase detector output at the previous reference clock cycle.

The PLL loop is a synchronous all-digital phase-domain architecture that arithmetically compares the accumulated *frequency command word* (FCW) in the reference phase accumulator $R_R(k)$ with the DCO clock edge count in the variable phase accumulator $R_V(k)$ (the letter "v" is used here conventionally) in order to arrive at the phase error correction. The FCW input to the reference accumulator is used to establish the operating frequency of the desired channel and it is expressed in a fixed-point format such that 1 LSB of its integer part corresponds to the f_R reference frequency. FCW could also be viewed as a desired frequency division ratio $\frac{f_V}{f_R}$. Alternatively, FCW indicates the real-valued count of the DCO clock cycle periods T_V per one cycle of the reference clock, T_R.

Also shown in Fig. 4.21 is the coarse integer phase error compensation process by the finer fractional error correction in order to increase the phase resolution of the system to below the basic 2π rad of the oscillator phase.

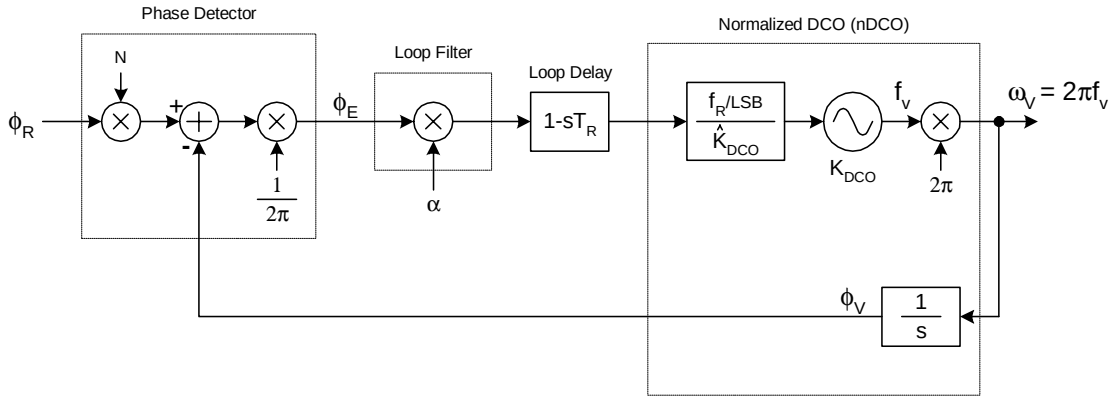

Figure 4.22. Linearized equivalent s-domain model of the ADPLL

4.15 PLL Frequency Response

Fig. 4.22 shows a general s-domain model of the ADPLL frequency synthesizer. It is a continuous-time approximation of a discrete-time z-domain model [60] and is valid as long as the frequencies of interest are much smaller than the sampling rate, which in this case equals f_R. (It is commonly accepted for this linear approximation to hold as long as the PLL loop bandwidth f_{BW} is at least ten times smaller than the sampling rate [25].) The loop filter is realized here as a normalized gain α stage thus giving rise to a type-I first-order PLL loop, which is defined as having only one integration pole due to the DCO frequency-to-phase conversion. For the purposes of this section only, quantities ϕ_R and ϕ_V are conventionally-defined reference and variable phases, respectively, in radian units. ω_V is the angular frequency in radians per second and is equal to $2\pi f_V$. Introduction of the angular frequency makes it necessary to modify the DCO transfer function by including the multiplication by 2π. Strictly speaking, this is viewed as modifying the normalizing gain component f_R [Hz] by 2π multiplication which results in ω_R [rad/s]. Quantity N is the frequency division ratio between the DCO clock and the FREF, and is equivalent

to FCW. Formally, K_{DCO} shall be expressed in units of radians/sec per LSB, but it is always divided by its estimate, so the radian units cancel out leaving a dimensionless unit of $K_{DCO}/\widehat{K}_{DCO}$. The $(1 - sT_R)$ operator is the control loop delay which degrades the phase margin by $2\pi(cycleDelay \cdot f_{BW}/f_R)$. In our case, there is only one cycle control loop delay and $f_{BW} \ll f_R$, therefore, the phase margin degradation is not an issue and this operator could be disregarded.

The open-loop transfer function $H_{ol}(s)$ is

$$H_{ol}(s) = \frac{1}{2\pi} \cdot \alpha \cdot \frac{2\pi f_R}{\widehat{K}_{DCO}} \cdot \frac{K_{DCO}}{s} = \alpha \cdot \frac{f_R}{s} \cdot \frac{K_{DCO}}{\widehat{K}_{DCO}} \tag{4.32}$$

Let's assume that the DCO gain is estimated correctly. $H_{ol}(s)$ then reduces to

$$H_{ol}(s) = \alpha \cdot \frac{f_R}{s} \tag{4.33}$$

There is one pole at dc, hence type-I classification of this ADPLL structure.

The closed loop transfer function can be expressed as

$$H_{cl}(s) = \frac{N \cdot \alpha \cdot \frac{f_R}{s}}{1 + H_{ol}} = \frac{N \cdot \alpha \cdot \frac{f_R}{s}}{1 + \alpha \cdot \frac{f_R}{s}} \tag{4.34}$$

which can be rearranged as

$$H_{cl}(s) = \frac{N}{1 + \frac{s}{\alpha \cdot f_R}} \tag{4.35}$$

Its magnitude response is shown in Fig. 4.23. Substituting $s = j\omega = j2\pi f$,

$$H_{cl}(f) = \frac{N}{1 + j\frac{2\pi f}{\alpha \cdot f_R}} \tag{4.36}$$

From this we obtain the bandwidth or 3-dB cut-off frequency of the low-pass closed-loop PLL (provided $f_{BW} \ll f_R$ in order for the s-domain approximation to hold) to be

$$f_{BW} = \frac{\alpha}{2\pi} \cdot f_R \tag{4.37}$$

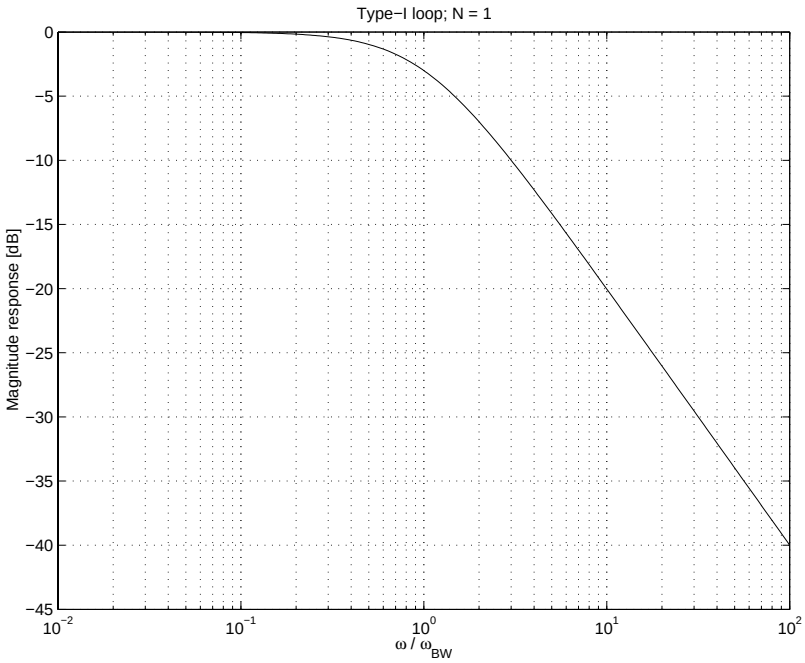

Figure 4.23. Magnitude response of the type-I PLL loop for normalized frequencies

It is interesting to note that the ratio divider N is not part of the open-loop transfer function and, hence, does not affect the loop bandwidth. This is in contrast to conventional PLL-based synthesizers in which the phase detector uses the divided-by-N oscillator clock (see Fig. 1.18), which is equal to the update rate under the locked condition. In that case, since the compared phase is expressed in units of radian (normalized to the update period and multiplied by 2π), the same amount of timing excursion (in units of second) translates into a smaller phase by a factor of N. The phase detection mechanism of the proposed architecture measures the oscillator timing excursion normalized to the DCO clock cycle, hence no phase division of ϕ_V.

The frequency reference input phase ϕ_R, on the other hand, needs to be multiplied by N since it is measured by the same phase detection mechanism normalized to the DCO clock cycle. The same amount of timing excursion on the FREF input translates into a larger

phase by a factor of N when viewed by the phase detector.

There is a multiplication factor of $\frac{1}{2\pi}$ at the output of the phase detector. It is simply due to the fact that the digital phase error is expressed not in units of radian of the DCO clock, but in the number of its cycles.

4.15.1 Conversion between s-domain and z-domain

The z-operator is defined as $z = e^{j\theta}$, where $\theta = \omega t_0$. $\omega = 2\pi f$ is the angular frequency and t_0 is the sampling period. In our case, $t_0 = \frac{1}{f_R}$ which leads to $z = e^{j\omega/f_R}$. For small values of ω in comparison with the sampling rate, we can make the following approximation.

$$z = e^{j\theta} \approx 1 + j\theta = 1 + j\omega/f_R = 1 + s/f_R \tag{4.38}$$

which results in

$$s = f_R(z - 1) \tag{4.39}$$

Transforming Eq. 4.33 and Eq. 4.35 gives

$$H_{ol}(z) = \frac{\alpha}{z - 1} \tag{4.40}$$

$$H_{cl}(z) = \frac{N}{1 + \frac{z-1}{\alpha}} \tag{4.41}$$

4.16 Noise and Error Sources

The ADPLL linear model including the phase noise sources is shown in Fig. 4.24. $\phi_{n,R}$ is the phase noise of the reference input that is external to the ADPLL. Its transfer function is expressed by Eq. 4.35. Internally to the system, there are only two places that the noise

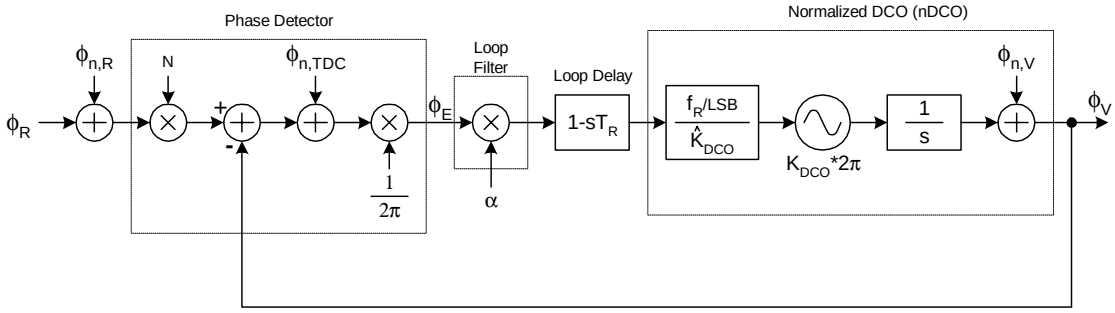

Figure 4.24. Linear s-domain model with noise sources added

could be injected. Due to its digital nature, the rest of the system is *completely* immune from any time-domain or amplitude-domain perturbations.

The first internal noise source $\phi_{n,V}$ is the oscillator itself. It undergoes high-pass filtering by the loop. A reference to the output frequency ω_V on Fig. 4.22 had to be removed because $\phi_{n,V}$ appears after the $1/s$ operation. Its closed-loop transfer function is

$$H_{cl,V}(s) = \frac{1}{1 + H_{ol}} = \frac{1}{1 + \frac{\alpha \cdot f_R}{s}} \tag{4.42}$$

which can be rewritten as

$$H_{cl,V}(f) = \frac{1}{1 - j\frac{\alpha \cdot f_R}{2\pi f}} \tag{4.43}$$

The above frequency transfer function indicates that the DCO noise has a high-pass characteristic with a bandwidth or 3-dB cut-off frequency of

$$f_{BW,V} = \frac{\alpha}{2\pi} \cdot f_R \tag{4.44}$$

The second internal noise source $\phi_{n,TDC}$ is the TDC operation of calculating ε. Even thought the TDC is a digital circuit, the FREF and CKV inputs are *continuous* in the time domain. The TDC error has several components: quantization, linearity and random due to thermal effects. It should be noted that the rest of the phase detection mechanism is digital

in nature and *does not* contribute any noise. The closed loop transfer function of the TDC noise can be expressed as

$$H_{cl,TDC}(s) = \frac{\alpha \cdot \frac{f_R}{s}}{1 + H_{ol}} = \frac{\alpha \cdot \frac{f_R}{s}}{1 + \alpha \cdot \frac{f_R}{s}} \tag{4.45}$$

which can be rearranged as

$$H_{cl,TDC}(s) = \frac{1}{1 + \frac{s}{\alpha \cdot f_R}} \tag{4.46}$$

or rewritten as below for easy inspection

$$H_{cl,TDC}(f) = \frac{1}{1 + j\frac{2\pi f}{\alpha \cdot f_R}} \tag{4.47}$$

The TDC-contributed noise has the same transfer function as the reference noise but without the gain of N.

4.17 Type-II ADPLL loop

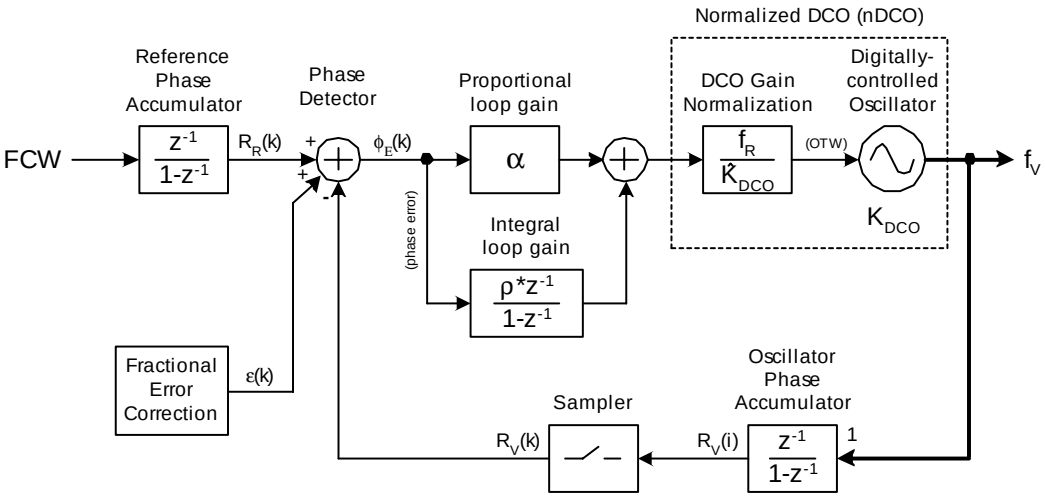

Figure 4.25. Type-II phase-domain all-digital synchronous PLL synthesizer

Fig. 4.21 is now modified to include a second pole at zero frequency thus giving rise to a type-II ADPLL loop. This is also known as a *proportional-integral* (PI) controller and is

universally used in all-digital PLL's for clock generation and recovery [35] [36] [37] [38] [39], and in hard-disk drive read channels [61]. A body of experience has been developed over the years on this type of structure.

The PI control is accomplished in digital domain by accumulating the phase error and scaling it by the integral loop gain ϱ and is shown in Fig. 4.25. Value of ϱ should normally be smaller than α. Contributions from both the proportional and the integral paths are added together.

A chief advantage of the type-II structure is its better noise filtering capabilities, leading to improvement in the phase noise performance. Type-I loop can provide only -20 dB/dec filtering of the reference phase noise, whereas type-II can provide up to -40 dB/dec under special conditions. On the DCO side, the type-I loop flattens the close-in phase noise region, whereas type-II has a capability to attenuate it by 20 dB/dec towards lower frequencies.

Another advantage of the type-II loop is that the phase offset between the reference and variable clocks approaches zero even in face of constant frequency offset. This is especially beneficial in clock recovery applications with integer-N frequency division ratios.

Due to the longer transients of the PI configuration, its use is beneficial only in the tracking mode. In order not to degrade the switching time, the integral part ϱ should be turned off during the acquisition. This way, a fast transient loop characteristic is used during acquisition and slower transient, but with a better filtering capability, is used during tracking.

4.17.1 PLL Frequency Response of Type-II Loop

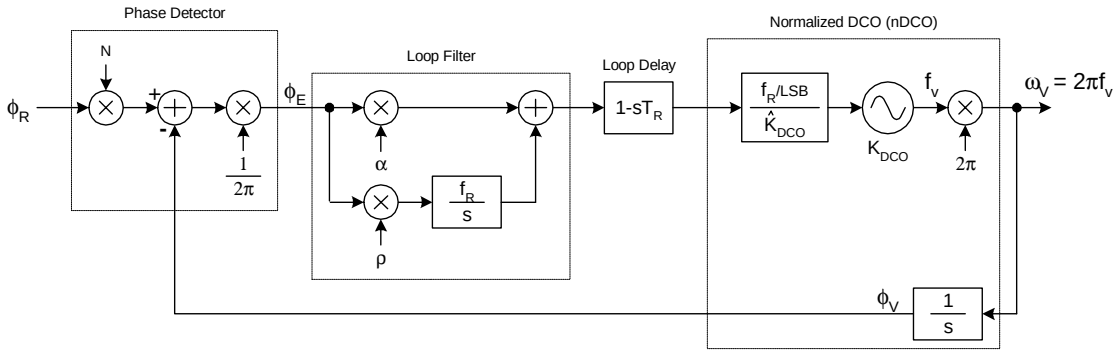

Figure 4.26. Linearized equivalent s-domain model of the type-II ADPLL

Let's start the analysis with finding the s-domain equivalent of the discrete-time accumulator $\frac{z^{-1}}{1-z^{-1}}$ in the integral path. We have $(z-1) = \frac{s}{f_R}$, and inverse of this is just $\frac{z^{-1}}{1-z^{-1}}$ which is shown in Fig. 4.26.

The open loop transfer function, assuming the K_{DCO} is estimated correctly, is

$$H_{ol}(s) = (\alpha + \frac{\varrho \cdot f_R}{s})\frac{f_R}{s} = \frac{\varrho \cdot f_R^2}{s} \cdot \frac{1 + \frac{s}{\varrho \cdot f_R/\alpha}}{s} \tag{4.48}$$

which shows two poles at origin and one complex zero at $\omega_z = j\frac{\varrho \cdot f_R}{\alpha}$

The closed-loop transfer function is

$$H_{cl}(s) = N\frac{(\alpha + \frac{\varrho f_R}{s}) \cdot \frac{f_R}{s}}{1 + (\alpha + \frac{\varrho f_R}{s}) \cdot \frac{f_R}{s}} = N\frac{\alpha f_R s + \varrho f_R^2}{s^2 + \alpha f_R s + \varrho f_R^2} \tag{4.49}$$

This can be compared to the classical two-pole system transfer function whose magnitude response is shown in Fig. 4.27.

$$H_{cl}(s) = N\frac{2\zeta\omega_n s + \omega_n^2}{s^2 + 2\zeta\omega_n s + \omega_n^2} \tag{4.50}$$

where ζ is the damping factor and ω_n is the natural frequency. Fitting the two equations yields

$$\omega_n = \sqrt{\varrho} f_R \tag{4.51}$$

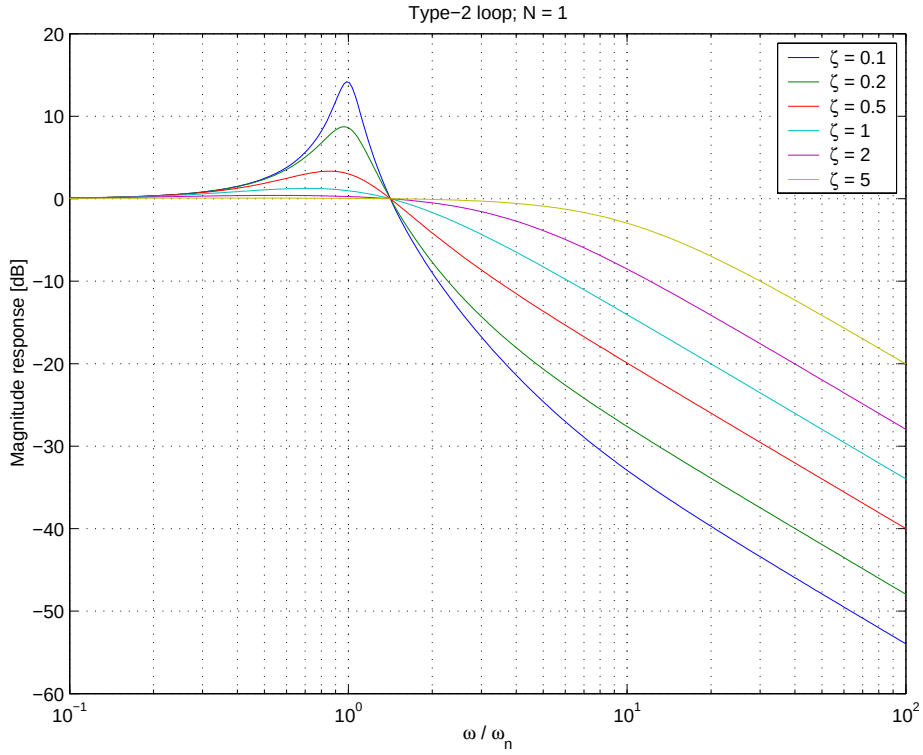

Figure 4.27. Magnitude response of the type-II PLL loop for normalized frequencies and for various values of ζ

$$\zeta = \frac{\alpha f_R}{2\omega_n} = \frac{1}{2} \cdot \frac{\alpha}{\sqrt{\varrho}} \tag{4.52}$$

Type-II loop gives one more tuning knob resulting in more flexibility to adjust the loop noise performance. The ϱ term contributes with a square-root effect on the natural frequency and damping factor, therefore it requires more bits to keep the same dynamic range as α.

Transforming Eq. 4.48 and Eq. 4.49 into z-domain gives

$$H_{ol}(z) = \frac{\alpha(z-1) + \varrho}{(z-1)^2} \tag{4.53}$$

$$H_{cl}(z) = N\frac{\alpha(z-1) + \varrho}{(z-1)^2 + \alpha(z-1) + \varrho} \tag{4.54}$$

4.18 Higher-order ADPLL Loop

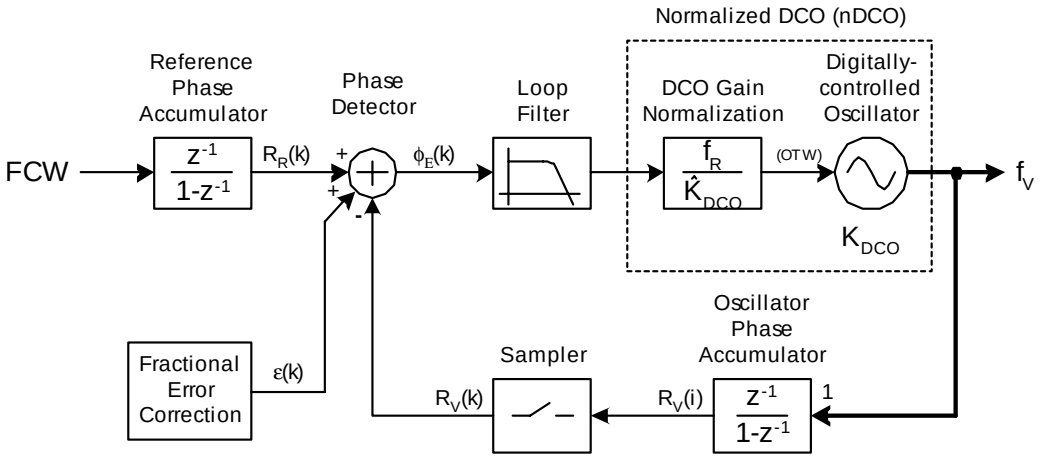

Figure 4.28. Higher-order phase-domain all-digital synchronous PLL synthesizer

Type-II PLL loop, as shown in Fig. 4.25, looses some of its earlier appeal in a fully-digital architecture with a digitally-controlled oscillator. Instead of a pole at dc, a low-pass filter with a transfer function of $L(z)$ is added as shown in Fig. 4.28.

Using analog techniques, the maximum practically achievable order of a PLL is third [37] [60] [62], mainly due to the stability issues, especially when process and temperature variation are taken into account. However, these restrictions do not exist with digital VLSI implementations and it is possible to create higher-order structures that would provide efficient noise reduction and accurate frequency response [63].

The all-digital loop renders a digital design of the loop filter possible, thus providing many benefits in terms of testability, flexibility and portability to various processes.

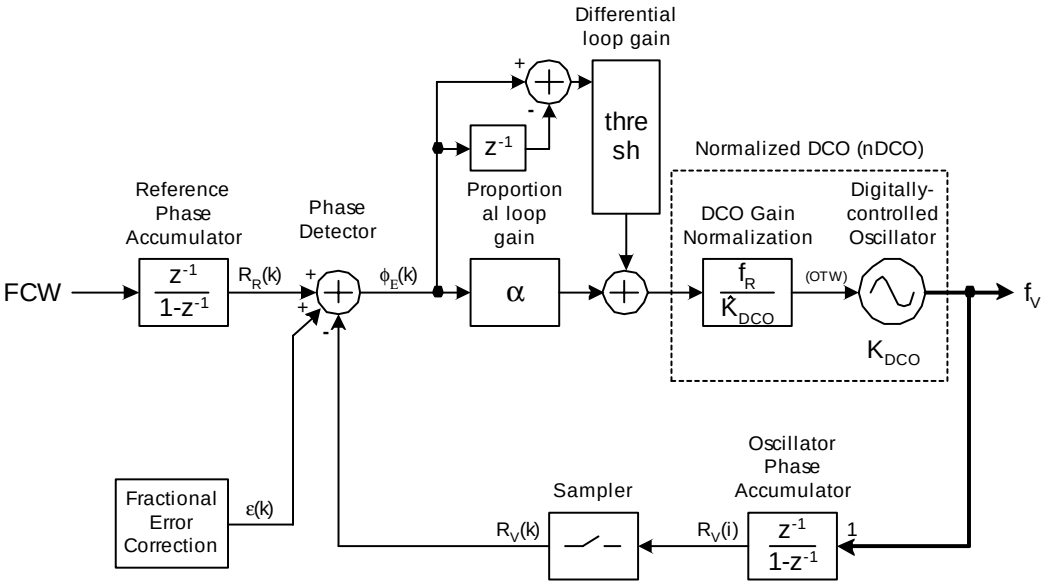

Figure 4.29. Phase-domain ADPLL synthesizer with non-linear differential term

4.19 Non-linear Differential Term of the ADPLL Loop

A differential term could be added to the ADPLL structure of either Fig. 4.21 or Fig. 4.25. Due to its noise-enhancement property, the differential term has to be filtered in a non-linear manner. This could be accomplished by a thresholder circuit that senses the absolute phase error difference between the current and previous samples and activates the DCO correction for large phase error steps as shown in Fig. 4.29. The differential term is very useful to handle situations in which an *occasional* rapid frequency perturbation occurs during the regular tracking operation when the PLL loop is settled and normally slower in response. The threshold should be set high enough as to avoid being triggered by the expected distribution of the thermal and flicker noise due to those loop components with continuous-domain characteristics.

These sudden changes in the oscillating frequency might be due to, for example, sudden

supply voltage drop when the integrated digital baseband starts a new activity. Relying on the proportional term to handle the sudden perturbation would normally require long time due to the narrow loop bandwidth. In order to filter out any transitory phase error perturbations, which might not necessarily indicate a consistent change in the oscillating frequency, it is necessary to qualify the new phase error for a number of clock cycles.

A gear-shifting procedure, as described below in Sec. 4.21, is normally required whenever the loop bandwidth α is modified.

4.20 DCO Gain Estimation by Using the PLL Loop

Incorrect estimate of the K_{DCO} gain affects the accuracy of the loop gain α, which in itself is no major concern. The best indication of this is that in the proposed implementation, the α is register programmable as a negative exponent of the radix 2, which is definitely very coarse. It was found through system analysis that a finer granularity of the loop gain control was not needed. Why then, the reader might ask, is this apparent stress on the DCO gain estimation? The main reason is the predictive loop operation for the transmit frequency modulation, which will be introduced in Chapter 5. This method does require a *precise* knowledge of the DCO gain.

As mentioned in Sec. 3.3, the DCO gain estimate $\widehat{K}_{DCO}$ can be computed by harnessing the power of the existing phase detection circuitry for the purpose of determining the oscillator frequency deviation Δf_V.

The DCO frequency deviation Δf_V can be calculated by observing the phase error difference $\Delta\phi_E$ (expressed as a fraction of the DCO clock period) in the observation interval

of the phase detector update, which is normally equal to the frequency reference clock period T_R. Eq. 4.27 (page 144) is rewritten here.

$$\Delta f_V = \frac{\Delta\phi_E}{T_R} = \Delta\phi_E \cdot f_R \tag{4.55}$$

Eq. 4.55 can be plugged into Eq. 3.3 (page 77) to provide an estimated DCO gain.

$$\widehat{K}_{DCO}(f_V, OTW) = \frac{\Delta\phi_E}{\Delta(OTW)} \cdot f_R \tag{4.56}$$

Eq. 4.56 *theoretically* allows us to calculate the local value, i.e., for a given DCO input OTW, of the oscillator gain K_{DCO} by observing the phase detector output $\Delta\phi_E$ being a response to the $\Delta(OTW)$ input perturbation at the previous reference clock cycle. Naturally, the reference frequency f_R is the system parameter which is, for all practical purposes, known exactly.

Unfortunately, as mentioned in Sec. 4.11, the above method of frequency estimation is a poor choice due to the excessive TDC quantization for realistic values of Δt_{res}. Instead, the difference between the steady-state phase error values is more appropriate. Eq. 4.30 (page 147) captures the relationship, which is also repeated here.

$$\Delta f_V = \phi_E \cdot \alpha \cdot f_R \tag{4.57}$$

Eq. 4.57 can be plugged into Eq. 3.3 (page 77) to provide an estimated DCO gain.

$$\widehat{K}_{DCO}(f_V, OTW) = \frac{\Delta\phi_E \cdot \alpha}{\Delta(OTW)} \cdot f_R \tag{4.58}$$

A significant advantage in operation can be obtained by noting that in a type-I PLL loop the phase error ϕ_E is proportional to the relative oscillating frequency. Consequently, not only the power of the phase detection circuitry could be harnessed but also the averaging

and adaptive capability of the PLL loop itself. Eq. 4.56 can be used now with the normal loop updates (unlike the general case) for an arbitrary number of FREF clock cycles. At the end of the measurement, the final $\Delta\phi_E$ and ΔOTW values are used. The loop itself provides the averaging and frequency quantization reduction.

In the above analysis, the OTW perturbation was the cause for the frequency deviation calculation by observing the resulting phase error change. This order could be beneficially reversed by setting the frequency deviation à priori and observing the resulting change in the tuning word.

4.21 Gear-shifting of the PLL Loop Gain

4.21.1 Proposed Idea

During normal course of a PLL operation, one can distinguish two unique intervals with likely conflicting requirements. In the first phase, the goal is to acquire the desired frequency as soon as practically possible. The loop bandwidth needs to be made wide even at the expense of enhanced phase noise and spurs, which are quite unimportant at that time. In the second phase, the goal is to maintain or track the desired frequency that got acquired during the first phase. The loop bandwidth there has to be kept lower in order to minimize the phase noise and spurs.

These conflicting requirements on the PLL loop characteristic necessitate a mechanism by which the loop bandwidth could be seamlessly shifted once the acquisition phase is completed. The proposed *gear-shifting* mechanism is the subject of this section. This idea has two embodiments: an autonomous gear-shifting that is fully contained in the oscilla-

tor gain block (GT) and the extended gear-shifting with zero-phase restart that involves multiple blocks.

The former embodiment is utilized in the testchip implementation for the gain switchover while using the tracking bank varactors. The extended gear-shift operation implements the progressive refinement of DCO resolution as the operational frequency approaches the desired frequency. That idea was first introduced in Fig. 2.9 (page 61).

The proposed mechanism deals with an introduction of a gear-shift or a switchover of the normalized PLL loop gain constant α. During frequency/phase acquisition, a larger loop gain constant, α_1, is used such that the resulting phase error is within limits. After the frequency/phase gets acquired, the developed phase error, which is a rough indication of the frequency offset, is in a steady-state. The following two operations are performed simultaneously (in a synchronous digital design it means within the same clock cycle) while making transition into tracking:

1. Add the DC offset to the phase error or the DCO tuning signal, and

2. Reduce the loop constant from α_1 to α_2.

Since (1) results in substantial lowering of the maximum phase error, α could now be safely reduced. It should be noted that in a type-I PLL loop the frequency deviation is directly proportional to the phase error and the gain constant, α.

The proposed idea makes an implementational sense mainly in digital systems. It is generally difficult to perform a PLL gear shifting in analog circuits because of the imperfect matching and voltage or charge losses during switching, thus resulting in phase *hits*

whenever a sudden perturbation (gear-shift) is introduced. The proposed solution is fully digital and provides an exact hitless operation, while requiring a low dynamic range of the phase detector.

While it is generally difficult to properly execute an instantaneous gear-shifting in analog domain, some attempts have been demonstrated. For example, [64] performs a time-continuous adaptive gear-shifting for clock recovery applications. The idea there is to gradually reduce the loop gain based on the filtered phase variations. As the loop settles, the phase detector output produces less and less variations at its output causing less charge to be stored on a capacitor. This is used to gradually reduce bias in the charge pump, thus reducing the overall loop gain. Unfortunately, since the charge pump current is dynamically controlled, this could likely become an additional source of phase noise at the VCO input. As another example, [65] switches the variable loop bandwidth by changing the charge pump current together with PLL loop filter parameters. However, it is believed that the speedup has not been as good as reported.

The proposed solution provides a hitless gear shifting mechanism of a PLL loop without requiring a large dynamic range of the phase error. It is a perfect match to the phase-domain all-digital PLL loop synthesizer architecture.

4.21.2 Autonomous Gear-shifting Mechanism

Fig. 4.30 shows the main idea behind the PLL gear-shifting scheme which could be fully contained in the DCO gain block. It specifically pertains to the type-I digital PLL loop in which the phase error $\phi_E(k)$ of the phase detector output is used to drive the tuning input

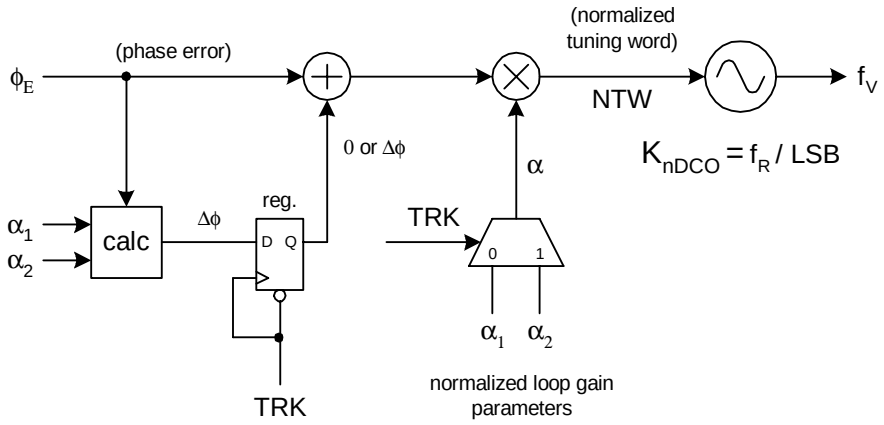

Figure 4.30. Gear-shifting mechanism of type-I PLL loop

of the oscillator without any additional filtering operation. This method is also applicable to higher-order PLL loops. In this case, the "phase error" input becomes the "filtered phase error" of the *loop filter* (LF) output as shown in Fig. 4.31. All signals in Fig. 4.30 and Fig. 4.31, with possible exception of the oscillator RF output in other embodiments, are digital. The asynchronous flip-flop controlled by the *tracking control* (TRK) sequencer signal is a short-hand notation for a latching mechanism of the $\Delta\phi$ phase error adjustment. In practice, it would be implemented as a state machine with a synchronous reset that stores a new $\Delta\phi$ value into a cleared register upon transition from acquisition to tracking mode.

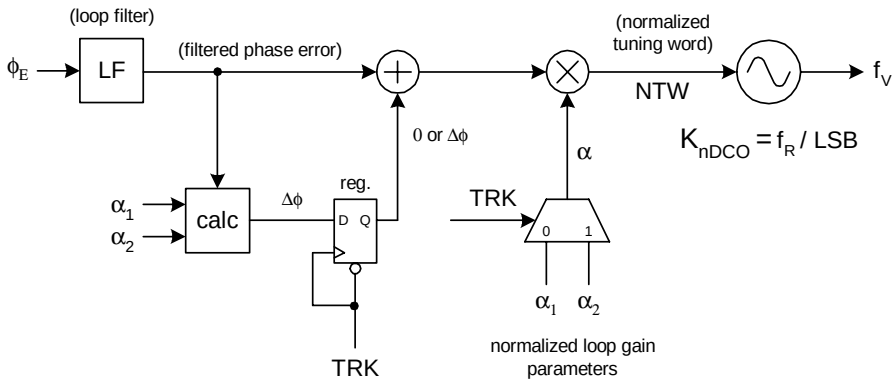

Figure 4.31. Gear-shifting mechanism of higher-order PLL loops

During the fast tracking mode, the PLL loop operates under the high loop bandwidth regime which is controlled by the normalized loop gain α_1. Just before the gear-shift switching instance, the phase error has a value of ϕ_1. At the gear-shift switching instance, the new phase error value $\phi_2 = \phi_1 + \Delta\phi$ should be adjusted for the new lower tracking-mode loop gain value α_2 such that there would be no frequency perturbation of the oscillator *before* and *after* the event:

$$\alpha_1 \cdot \phi_1 = \alpha_2 \cdot (\phi_1 + \Delta\phi) \tag{4.59}$$

The required phase error adjustment $\Delta\phi$ of Eq. 4.60 is derived from Eq. 4.59.

$$\Delta\phi = \frac{\alpha_1 - \alpha_2}{\alpha_2} \cdot \phi_1 = (\frac{\alpha_1}{\alpha_2} - 1) \cdot \phi_1 = \frac{\alpha_1}{\alpha_2} \cdot \phi_1 - \phi_1 \tag{4.60}$$

This $\Delta\phi$ value is then maintained as constant and added to the phase error samples throughout the subsequent tracking mode operation.

It is very advantageous to restrict the ratio of normalized loop gains $\frac{\alpha_1}{\alpha_2}$ to power-of-two values such that Eq. 4.60 simply reduces to the left bit-shift operation of the phase error just before the gear-shift instance ϕ_1 minus ϕ_1 itself.

It should be noted that the "effective" center frequency $f_{0,eff}$ in the tracking mode is now much closer to the desired frequency than that the "raw" oscillator center frequency f_0 at the beginning of the acquisition-mode operation.

$$f_{0,eff} = f_0 + \Delta\phi \cdot \alpha_2 \cdot K_{DCO} \tag{4.61}$$

This method of gear-shifting of the PLL loop gain could be naturally extended to the sequential reduction of three or more loop gain parameters α. With each reduction of α

as the loop gets closer to the desired frequency, the frequency dynamic range would be reduced, but the frequency resolution would become finer.

4.21.3 Extended Gear-shifting Scheme with Zero-Phase Restart

The proposed gear-shift scheme is self contained in the DCO gain block. An improvement could be further made to the DCO operational range, however, this requires some modification to either or both sources of the phase error signal: reference phase R_R and variable phase R_V. Since R_R operates at a much lower frequency than R_V it is much easier in practice to perform any phase adjustment to the reference phase accumulator.

Since the gear-switching operation is hitless, there is no perturbation to the unadjusted phase error $\phi(k)$ (from the phase detector). If the phase error value is at the maximum of the dynamic range at the end of the acquisition, it will remain there during tracking. The frequency range would be proportionately lower for the new α but the new "effective" center frequency is now closer to the desired frequency. The main idea behind the improved scheme is to make the effective center frequency after the gear-shifting switch to be exactly zero by an appropriate adjustment of the DC phase error correction $\Delta\phi$ and either the reference phase R_R or the DCO phase R_V.

The above could be simply implemented as performing the following steps during the gear-shifting operation:

1. Make R_R equal to R_V (or vice-versa, but this would not be as advantageous) in order to bring ϕ_2 to zero just after the gear-shift instance (still within the same clock cycle). This could be practically achieved by loading the variable accumulator value R_V into

the reference register R_R during the gear-shift clock cycle and performing the regular adjustment by the *frequency control word* (FCW). The phase error value ϕ_2 would not be zero, but be equal to FCW as normally expected follow-up to zero at the next clock cycle.

2. Modify $\Delta\phi$ of the original method to be now expressed as in Eq. 4.62.

$$\Delta\phi = \frac{\alpha_1}{\alpha_2} \cdot \phi_1 \tag{4.62}$$

It is very advantageous to restrict the acquisition-to-tracking normalized loop gain ratio $\frac{\alpha_1}{\alpha_2}$ to power-of-two values such that the Eq. 4.62 simply reduces to the left bit-shift operation of the phase error ϕ_1.

It should be noted that this improved method could be adapted to work for a higher-order PLL loop. In this case, the loop filter integrator would additionally have to be reset.

4.21.4 Zero-Phase Restart Mechanism

The improved gear shifting with ZPR is the basis of the DCO mode switchover operation of Fig. 2.9 (page 61)

Fig. 4.3 (page 116) illustrated a general block diagram of the phase detection mechanism. The phase detection is based on three signals: a reference phase PHR (also known as $R_R(k)$) calculated by the reference phase accumulator PR, a fractional error correction PHF (also known as $\varepsilon(k)$) calculated by a fractional error correction circuit (combination of TDC and PF), and a sampled variable phase correction PHV_SMP (also known as $R_V(k)$)

calculated by an integer-only variable phase accumulator PV. The phase error PHE is being arithmetically calculated (Eq. 4.16 on page 114 and repeated here for convenience) as

$$\phi_E(k) = R_R(k) - R_V(k) + \varepsilon(k) \tag{4.63}$$

with proper bit alignment to line up integer and fractional portions.

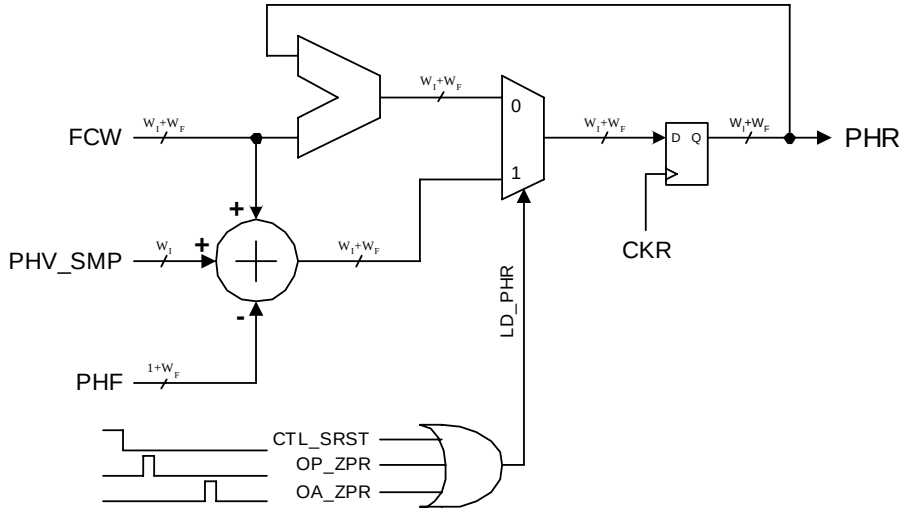

Figure 4.32. Reference phase accumulator (PR) with zero-phase restart

The modified reference phase accumulator circuit of Fig. 4.32 is used in conjunction with the PVT and acquisition bit controllers of Fig. 3.12 (page 90) and Fig. 3.13 (page 90), respectively, to restart the phase error PHE at the correct value during a mode change. The sequencer mode control signal (CTL_PLL_P or CTL_PLL_A) is constantly monitored for continuity by the oscillator PVT and acquisition control circuits in Fig. 3.12 and Fig. 3.13, respectively. During normal operation, the reference phase accumulator performs the following operation

$$R_R(k) = R_R(k-1) + N \tag{4.64}$$

where $N \equiv FCW$, $R_R(k-1)$ is the previous clock-cycle value and $R_R(k)$ is the current value. At the mode termination instance, the ZPR one-shot indication (OP_ZPR or

OA_ZPR) is asserted in those blocks. This causes a single cycle deviation from the normal accumulation operation of the PR circuit during which the register gets loaded with the following value:

$$R_R(k) = N + R_V(k-1) - \varepsilon(k-1) \tag{4.65}$$

Plugging Eq. 4.65 into Eq. 4.63 produces

$$\begin{aligned} \phi_E(k) &= [N + R_V(k-1) - \varepsilon(k-1)] - R_V(k) + \varepsilon(k) && (4.66) \\ &= N - ([R_V(k) - R_V(k-1)] - [\varepsilon(k) - \varepsilon(k-1)]) && (4.67) \end{aligned}$$

Applying expectation operator on both sides yields

$$\begin{aligned} \mathscr{E}\{\phi_E(k)\} &= N - \mathscr{E}\{[R_V(k) - R_V(k-1)] - [\varepsilon(k) - \varepsilon(k-1)]\} && (4.68) \\ &= 0 && (4.69) \end{aligned}$$

This is because the expected value of the variable phase change adjusted for the change in the fractional error correction is equal to the division ratio N.

As a result of the zero-phase restart, referring again to Fig. 4.3 or Fig. 4.20 (page 148) for a more comprehensive view, the phase error is forced to start again from a very small value close to zero. Any deviation from zero would be due to noise or non-linearity of the TDC.

The ZPR mechanism is additionally utilized as a substitute for a synchronous reset (active on CTL_SRST) of the variable phase accumulator PV. Referring back to Fig. 4.20 (page 148), the PV digital incrementer operates at the ultra high clock rate of 2.4 GHz and implementing a dedicated asynchronous or synchronous reset is expensive in terms of power dissipation and performance. Instead, an advantage is taken from the fact that, since

the reference and variable phases operate on a modulo arithmetic, their power-up absolute values do not matter – it is only their difference, phase error, that is propagated further. Consequently, performing the zero-phase restart at the power-up essentially accomplishes the task of synchronous reset.

4.21.5 Simulation Runs

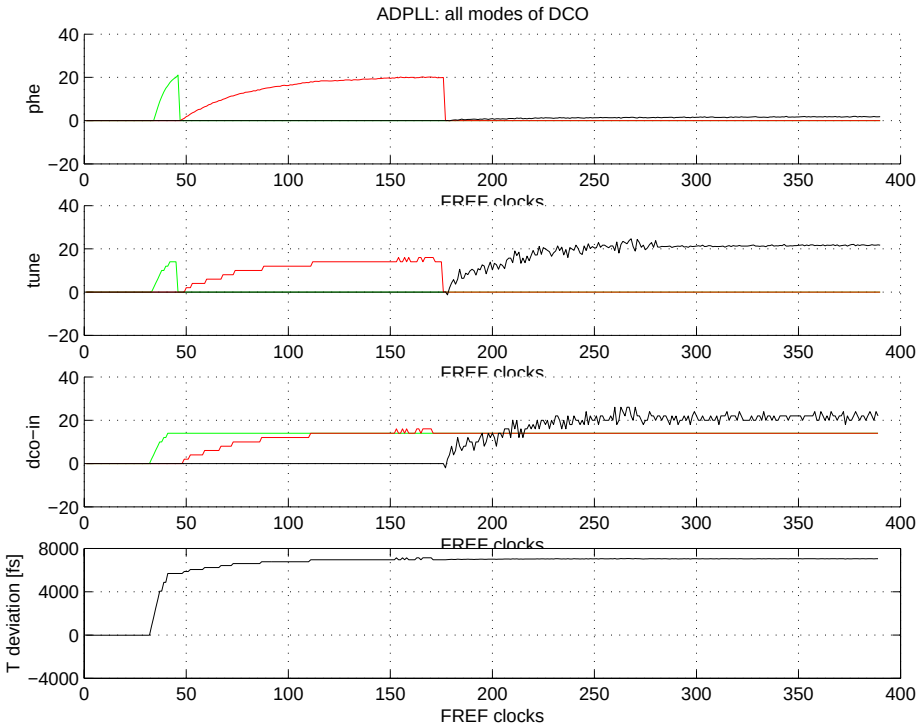

Figure 4.33. Simulation plots demonstrating the zero phase restart

The zero phase restart operation is demonstrated in Fig. 4.33. This plot consists of four subplots revealing the key ADPLL signals. The top three curves are further subdivided into three sequential operational modes: PVT (red), acquisition (green) and tracking (black). Time units on the x-axis are counts of FREF clock cycles.

1. “phe” – phase error ϕ_E in each of the operational modes.

2. “tune” – oscillator tuning words (OTW). In the PVT and acquisition modes it is an integer-valued signal. In the tracking mode it is a fixed-point signal with five fractional bits.

3. “dco-in” – DCO inputs. In the tracking mode, the displayed signal is an FREF-sampled version of the merged integer and fractional parts. Hence, it appears to have more noise than the above OTW signal. (The actual high-rate signal will be shown in a separate plot.)

4. “T deviation” – DCO period deviation in femptosecond units that is proportional to the DCO frequency deviation (1 fs = 5.77 kHz from Table 2.1 on page 71).

Initially, the loop operates at channel 0 (2402 MHz) of the BLUETOOTH band. Around time 40, a new value of the frequency command word (FCW) that corresponds to channel 40 (2442 MHz) is entered. The loop approaches the new frequency first in the PVT mode with the loop gain of $\alpha_P = 1/2^3$ and develops the tuning word of 14. Using entries from Table 2.2 (page 72), this translates to the frequency deviation of about 32.5 MHz. At around time 48, the loop enters the acquisition mode with $\alpha_A = 1/2^5$ by performing the zero phase restart. The developed phase error of 20 (confirming Eq. 4.30 on page 147: $20 \cdot 1/2^3 \cdot 13$ MHz = 32.5 MHz) drops instantly to zero while the PVT varactor bank gets frozen (“dco-in”) and the acquisition bank varactors start evolving in order to acquire the remaining distance of 40 MHz - 32.5 MHz = 7.5 MHz. The second ZPR happens at time around 170 and the developed phase error of 20 drop instantly to zero. At this time, the loop enters the fast tracking mode with $\alpha_{TA} = 1/2^5$.

At time around 280, after the steady-state is reached, the PLL loop transitions into the

normal tracking mode with a reduced loop gain of $\alpha_T = 1/2^8$. The reduced loop bandwidth exhibits itself as a smoother and more stable DCO frequency deviation curve. During the gear-shifting switch, the instantaneous values of the phase error, tuning word and the DCO period are maintained proving the correctness of implementing the hitless operation.

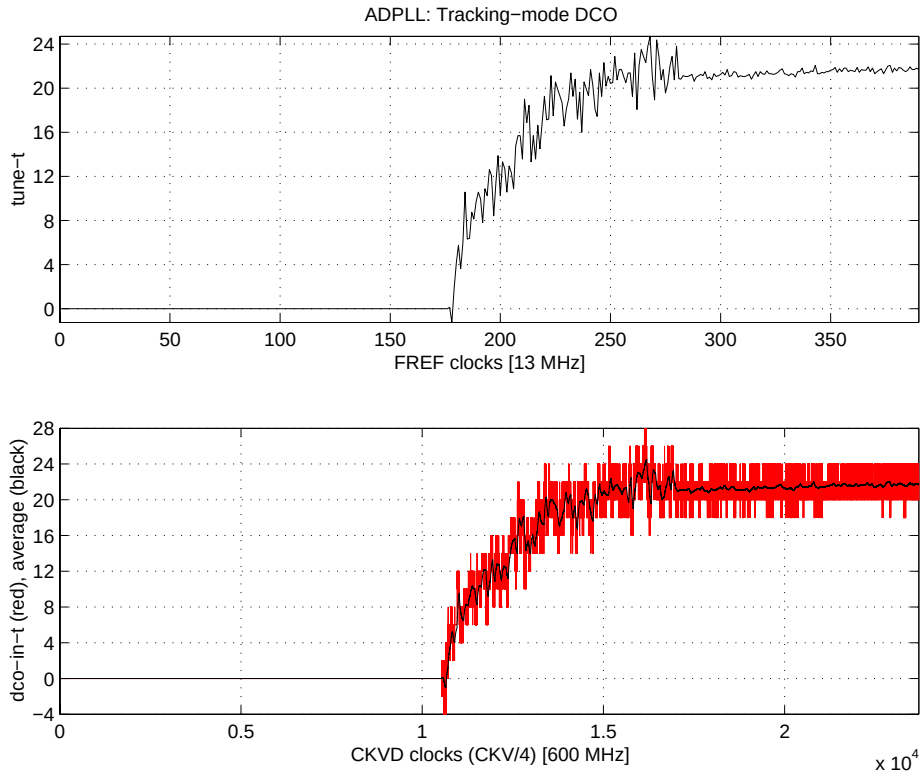

Figure 4.34. Simulation plots demonstrating correctness the gear-shifting

Fig. 4.34 further demonstrates the ADPLL gear-shifting operation and is similar in form to Fig. 3.18 (page 98) except for not performing the transmit modulation. The top plot shows the fixed-point tracking tuning word and the bottom plot shows the corresponding instantaneous merged DCO input. Worth noting is the hitless and continuous switchover of the loop bandwidth at time 280.

4.22 Summary

This chapter presented an all-digital phase-locked loop (ADPLL)-based frequency synthesizer that builds upon the normalized DCO (nDCO) described in Chapter 3. It revealed a novel phase correction mechanism which closes the loop around nDCO such that the oscillator's phase and frequency drift is corrected by means of the frequency reference. It also proposed the frequency reference (FREF) retiming performed in such a way as to stochastically avoid metastability. Block diagram of the proposed ADPLL was presented in Fig. 4.20 (page 148). It contains only two non-ideal parameters which are not known exactly and where distortion and noise could be introduced: the DCO gain (K_{DCO}) and the time-to-digital converter (TDC) resolution (Δt_{res}). The rest of the system is *exact* and is *completely* immune from any time-domain or amplitude-domain uncertainties and perturbations. Due to these reasons, the proposed architecture appears to be highly competitive or even exceed performance of conventional RF synthesizers, while consuming less power and area as well as enabling integration with the digital baseband.

CHAPTER 5

TRANSMITTER AS A SYNTHESIZER WITH FREQUENCY MODULATION

5.1 Overview

The synthesizer described in Chapter 4 has a convenient method of digitally controlling the oscillating frequency through the *frequency command word* (FCW) that feeds to the reference phase accumulator. The FCW fixed-point word could be easily augmented with a dynamically-changing modulating data in order to accomplish a frequency or phase (generally called "angle" in the communication theory) modulation of the synthesizer RF output. This chapter adds a necessary mechanism to accomplish this objective.

This chapter also introduces two other building blocks: pulse shaping filter (Sec. 5.3) and power amplifier (Sec. 5.5) with possible digital amplitude modulation (Sec. 5.6). These two blocks are not the main focus of this dissertation, although they contain novel ideas resulting from this research. Adding these blocks to the digital RF frequency synthesizer, which already has the direct frequency modulation capability, would now allow to complete the entire transmitter part of an RF transceiver for short-range digital communications. This would demonstrate the use of the proposed RF frequency synthesizer.

5.2 Oscillator Frequency Modulation

The oscillating frequency could be dynamically controlled by directly adding the appropriately scaled modulating data $y(k) = FCW_{data}(k)$ to the quasi-static frequency command word $FCW_{channel}$ at the reference accumulator input that is normally used for channel selection.

$$FCW(k) = FCW_{channel}(k) + FCW_{data}(k) \tag{5.1}$$

where k is a discrete-time index set by FREF.

This idea is depicted in Fig. 1.17 (page 27) that shows the front-end accumulator stage of a *direct digital frequency synthesizer* (DDFS) which is identical to the reference phase accumulator of the proposed architecture. Introducing the modulating data redefines the FCW, originally defined in Sec. 4.2, as the expected *instantaneous* frequency division ratio of the desired synthesizer output and the reference frequency.

$$FCW(k) = \frac{f_V(k)}{f_R} \tag{5.2}$$

Generally, the direct frequency or phase transmit modulation of a PLL loop of an RF frequency synthesizer is a challenging task. In order to attenuate the reference spurs and phase noise, the PLL bandwidth is usually kept low. This effectively prevents an application of closed loop modulation if the modulating data rate is not much smaller than the loop bandwidth [33] [34]. However, the direct closed-loop modulation of the DCO frequency still would be considered more cost-effective solution than the alternative of an image reject quadrature modulator.

5.2.1 Hybrid of Predictive/Closed PLL Loop Operation

The PLL loop operation could be dramatically enhanced by taking advantage of the predictive capabilities of the all-digital PLL loop. The idea is as follows. The DCO oscillator does not necessarily have to follow the modulating FCW command with the normal PLL loop response. In this fully digital implementation, where the DCO control and the resulting phase error measurement are in numerical format, it is easy to predict the current K_{DCO} gain of the oscillator by simply observing the past phase error responses to the DCO or nDCO corrections. With a good estimate of the K_{DCO} gain, the normal DCO control could be augmented with the "open loop" instantaneous frequency jump estimate of the new FCW command. The resulting phase error should be very small and subject to the normal closed PLL loop correction transients.

Since the time response of this type-I PLL loop is very fast (less than a few μs), the prediction feature is less important for channel hopping, where the allowed time is much greater. It is, however, essential to realize the direct frequency synthesizer modulation in the *Gaussian frequency shift keying* (GFSK) modulation scheme of BLUETOOTH or GSM[1] as well as the chip phase modulation of the 802.11b or Wideband-CDMA.

5.2.2 Prior Art

M. Bopp et al. report in [19] the idea of phase compensating the PLL loop by digitally integrating the transmit modulating data bits and using the integrator output to shift the phase of the reference clock signal, while the Gaussian filtered data directly modulates the

[1]GSM actually uses *gaussian minimum shift keying* (GMSK) which is a special case of GFSK

VCO frequency. However, this approach is rather analog in nature and requires a precise component matching, of not only the VCO but also the phase shifter. In contrast, the proposed solution is digital in nature and consumes little hardware overhead over the base PLL structure. Our solution requires only one component matching, of the DCO, which could be done adaptively with a very fine resolution.

Similar feed-forward compensation method, which also requires a precise knowledge of the VCO transfer function and other analog circuits is proposed in [20]. It uses DSP to calculate inverse of the VCO transfer function, which is obtained through lab measurements, followed by a high-precision DAC to pre-tune the VCO control voltage to the desired excursion. Even though the large frequency shifts could be quickly performed within the VCO accuracy, the problem of narrow PLL loop bandwidth still remains for residual frequency offsets.

5.2.3 Direct Oscillator Modulation with the PLL Loop Compensation

The proposed idea is to directly modulate the DCO frequency in a feed-forward manner such that it effectively removes the loop dynamics from the modulating transmit path. However, the rest of the loop, including all error sources, operates under the normal closed-loop regime. Fig. 5.1 shows the idea schematically.

The modulating data $y(k)$ at the upper feed directly affects the oscillating frequency with the open-loop transfer function:

$$h_{dir}^{f}(k) = \frac{1}{\alpha} \cdot \alpha \cdot \frac{f_R}{\widehat{K}_{DCO}} \cdot K_{DCO} = f_R \cdot \frac{K_{DCO}}{\widehat{K}_{DCO}} \tag{5.3}$$

expressed in [Hz/LSB] units.

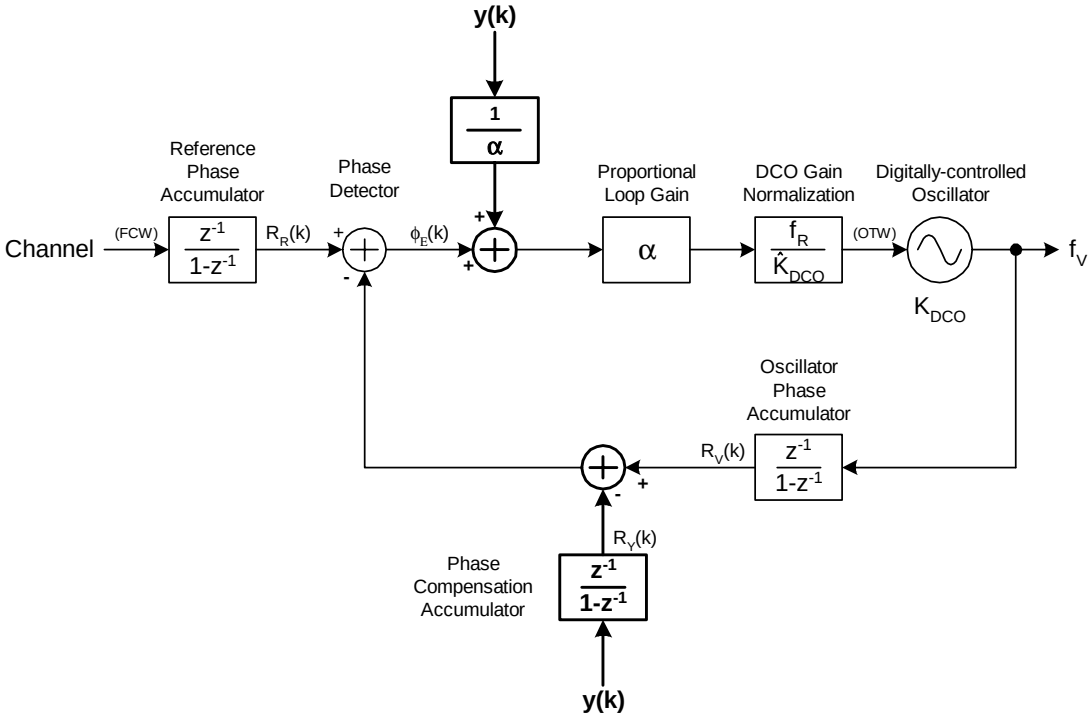

Figure 5.1. Direct oscillator modulation with a straightforward PLL loop compensation scheme

Unfortunately, the PLL loop will try to correct this perceived frequency perturbation integrated over the update period $1/f_R$. Its open-loop phase transfer function is:

$$h_{dir}(k) = f_R \cdot \frac{K_{DCO}}{\widehat{K}_{DCO}} \cdot \frac{1}{f_R} = \frac{K_{DCO}}{\widehat{K}_{DCO}} \tag{5.4}$$

expressed in [cycles/LSB] units. If the DCO gain estimate $\widehat{K}_{DCO}$ is accurate then $h_{dir} = 1$.

The transfer characteristic of the PLL closed loop with only the upper feed is high-pass. The low frequency components of the $y(k)$ data will be integrated in the variable accumulator, thus affecting the oscillator frequency base-line. It is necessary, therefore, to add a phase compensating circuit $R_Y(k)$, as shown at the lower $y(k)$ feed, that would completely subtract the phase contribution of the upper $y(k)$ direct modulation feed into the PLL loop if the DCO gain could be estimated correctly. The phase compensating open-loop

transfer function is expressed as:

$$h_{comp}(k) = 1 \tag{5.5}$$

expressed in [cycles/LSB] units. Consequently, if $\widehat{K}_{DCO} = K_{DCO}$ in Eq. 5.4, then the loop will be compensated perfectly [$h_{dir}(k) = h_{comp}(k)$] and the feedforward modulation will be exact. In case of non exact matching, the residual error $\frac{K_{DCO}}{\widehat{K}_{DCO}} - 1$ would have to undergo a normal high-pass loop response with the following s-domain transfer function.

$$H(s) = \frac{\frac{K_{DCO}}{\widehat{K}_{DCO}} - 1}{1 + \alpha \cdot \frac{f_R}{s}} \tag{5.6}$$

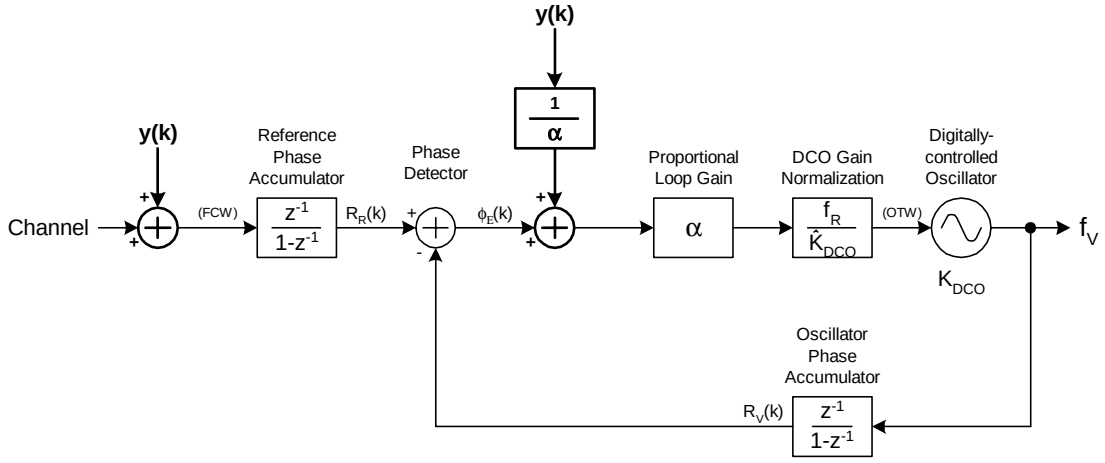

Figure 5.2. Direct oscillator modulation with an improved PLL loop compensation scheme

A more technically elegant variant is proposed in Fig. 5.2 that would merge the phase compensation accumulator $R_Y(k)$ with the reference phase accumulator $R_R(k)$. Now, the frequency command word is the sum of the channel and data signals, which is more intuitively appealing.

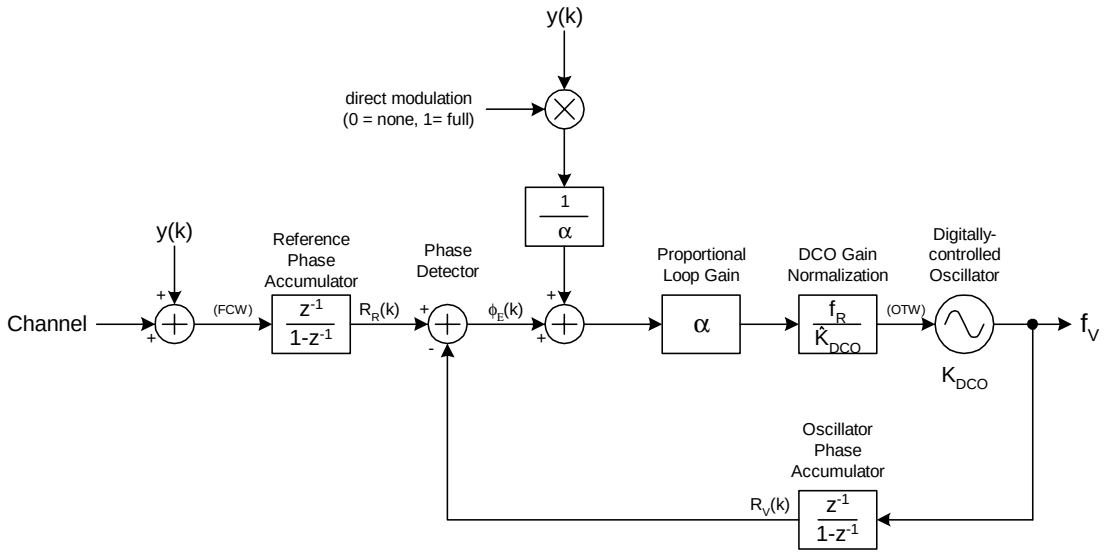

Figure 5.3. Partially-direct oscillator modulation with a PLL loop compensation scheme

5.2.4 Partially-Direct Oscillator Modulation

Fig. 5.3 shows the idea of the partially-direct frequency modulation. When the direct modulation "slider" gain is set to zero, the transmit data $y(k)$ undergoes the normal attenuation of the PLL loop low-pass filtering characteristics. However, when the direct modulation slider gain is set to one, the direct path, left $y(k)$, will fully undo the loop response due to the feedforward transmit data path, right $y(k)$. The slider gain value could also be set somewhere between the two extremes of 0 and 1 for partial direct modulation.

5.2.5 Improved Structure

If the loop parameter α is simply a power of two, which might seem quite adequate for most applications, then the architecture of Fig. 5.3 is a good choice since the α loop gain multiplier, here implemented as a right bit-shift operator, could be merged with the DCO gain normalization block, and the left bit-shift $1/\alpha$ operator is equally trivial.

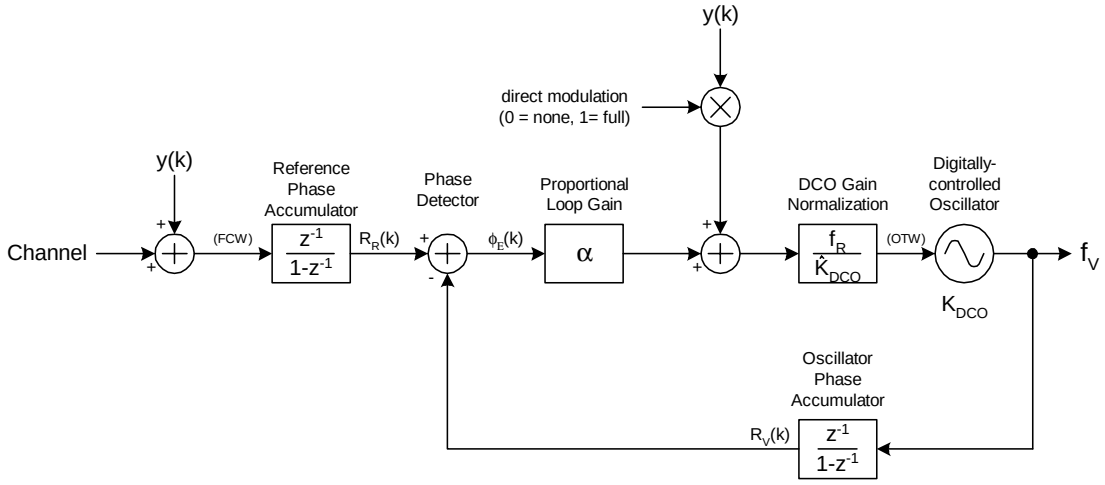

Figure 5.4. An improved structure without the $1/\alpha$ operation

However, if α is a fixed-point number or a combination of a few power-of-two numbers, i.e., low resolution mantissa, then an improved structure in Fig. 5.4 might be preferred. In this configuration, the $y(k)$ direct path feed is moved to the output of the α loop gain multiplier and, consequently, the inverse operation $1/\alpha$ is no longer needed. This particular topology is used in the implemented testchip.

5.2.6 Generalized Architecture

This direct oscillator modulation with the PLL compensating scheme works best in a digital implementation since almost a perfect compensation could be achieved. The presented idea would work equally well with a higher order PLL loop.

Fig. 5.5 shows how the direct oscillator modulation with the PLL loop compensation scheme of Fig. 5.3 or Fig. 5.4 could fit into a general digital PLL structure. A conventional digital PLL loop might consist of a VCO, frequency prescaler and divider, phase detector, loop filter and a digital-to-analog converter (ADC) that makes it possible to control the

oscillating frequency through a digital word. The modulating data $y(k)$ is dynamically added to the channel frequency information in order to affect frequency or phase of the oscillator output $f_{RF} = f_V$. This could be accomplished, for example, by controlling the frequency division ratio of the fractional-N PLL loop. The direct modulation structure is inserted somewhere between the loop filter and the oscillator. Gain of the direct modulating path from $y(k)$ to the oscillator input should be $\frac{f_R}{\hat{K}_{DCO}}$ if $y(k)$ is expressed as the unitless fractional division ratio.

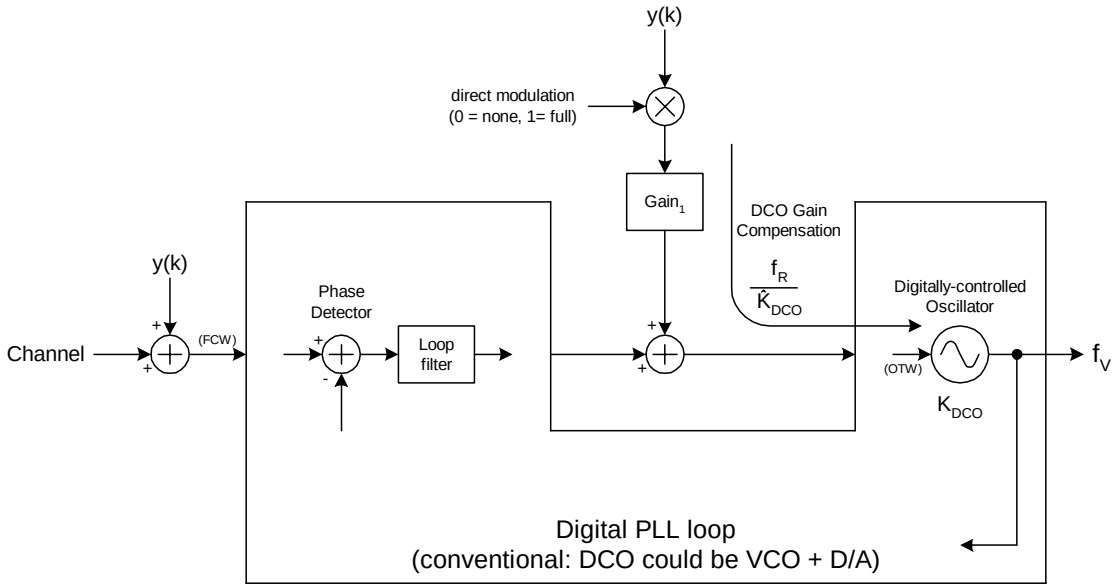

Figure 5.5. Oscillator modulation scheme with a PLL loop compensation within a general digital PLL architecture

5.3 GFSK Pulse Shaping of the Transmit Data

As mentioned in Sec. 1.3, it is required to perform pulse shaping operation on the transmitted symbols in order to limit the bandwidth occupied by the modulated RF spectrum before being emitted into the ether. Fig. 1.13 (page 22) revealed the chosen transmitter

architecture for the implemented GFSK-based BLUETOOTH transmitter. This section deals with the first two blocks of Fig. 1.13: bit coder and the transmit pulse shaping filter.

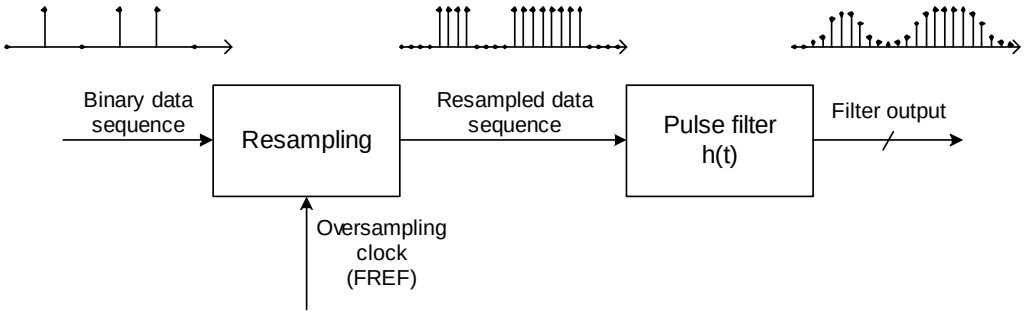

Figure 5.6. Operation principle of the digital transmit filter

Idea of the pulse shaping operation $h(t)$, performed entirely in digital domain, is illustrated in Fig. 5.6. It is mathematically described in Appendix C. The binary input data is being oversampled by a commonly-used 13 MHz frequency reference (FREF) clock generated by a crystal oscillator, which is an integer multiple of the 1 Mbps data (or symbol) rate. The oversampling clock is also called a baseband chip clock. In the GFSK system there is a one-to-one correspondence between data bits $\{0, 1\}$ and symbols $\{-1, +1\}$, with "0" $\rightarrow$ "-1" and "1" $\rightarrow$ "+1". For this reason, the coding conversion is being performed implicitly.

Fig. 5.7 plots an example output of the transmit filter with a symbol oversampling ratio of eight for the data sequence of "101110" with the initial state corresponding to bit "0". The "+1" and "-1" filter output levels correspond to the peak frequency deviation of 160 kHz at the transmitter RF output according to the formula

$$\Delta f_{pk} = \frac{m}{2} \cdot \frac{1}{T_s} \tag{5.7}$$

where m is the modulation index and T_s is the symbol period. For BLUETOOTH, m = 0.32

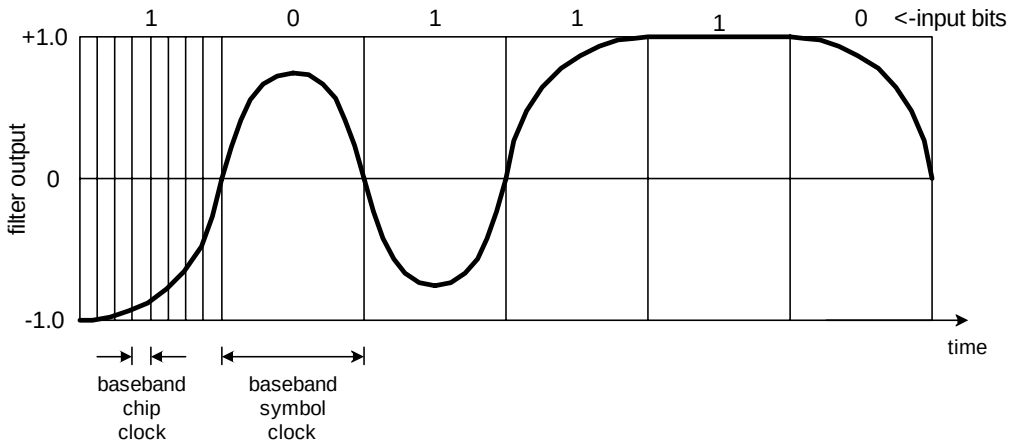

Figure 5.7. Output waveform of a transmit filter

(nominal) and T_s = 1 μs; for GSM, m = 0.5 and T_s = 3.6923 μs.

It is quite inefficient to perform the pulse shaping operation as literary shown in Fig. 5.6. It should be recognized that the FIR filtering of $h(t)$ has a number of properties that one could take advantage of in order to greatly reduce the implementational complexity. First, the input data is not a fixed-point number, it is a one-bit signal. Second, the input data is highly correlated since all the bits within symbol duration are the same due to oversampling. Third, the coefficients are fixed. The above observations allow us to operate the actual $h(t)$ filter as a 3-bit state look-up table that stores the *cumulative coefficients* $c(k)$, which are defined in Appendix C.

The state is based on the current and the previous two symbols (more symbols are required for GSM due to higher ISI) – it changes every symbol clock and has eight possible combinations. The data samples are precalculated per each state and are stored in a lookup table. They are read out on every FREF clock within the symbol duration. It will be shown below that there is a high amount of the data redundancy, such as symmetry, constant values for two states, etc., so the storage requirements could be quite relaxed. In fact, only one

state is stored in a ROM-like fashion and the other states are automatically calculated on power-up.

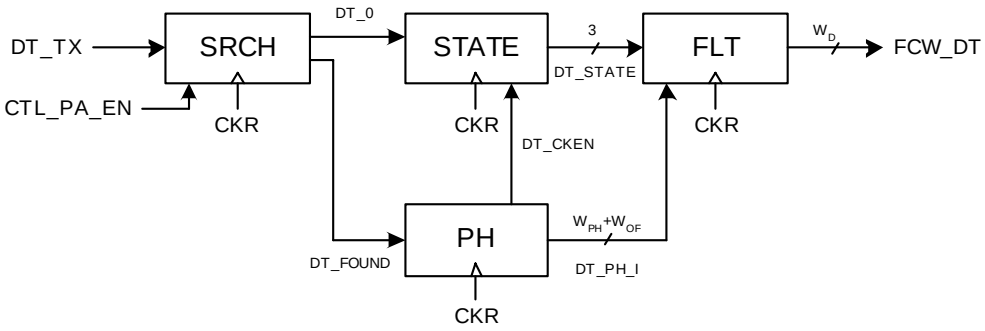

Figure 5.8. Top-level structure of the transmit filter

Fig. 5.8 shows an implementational block diagram of the proposed transmit filter. It consists of a search (SRCH), phase calculation (PH), state tracking (STATE) and the actual filter coefficient storage (FLT) subblocks. The search circuit starts with the power amplifier enable CTL_PA_EN control line asserted and the DT_TX data input from the digital baseband at the low level. It then waits for the starting bit sequence “010” on the DT_TX line that indicates beginning of a transmitted bit stream. When the first bit arrives, this circuit determines where the pulse lies with the *retimed frequency reference* (CKR) clock granularity. The initial mid-pulse location is used for sampling of all subsequent bits. The bit stream originates in the digital baseband which uses the same FREF clock, although with an arbitrary phase shift, as the transmitter. Therefore, once determined middle of the pulse holds for the entire packet duration.

Detection of the “010” starting bit sequence triggers the phase calculation circuit (with the data sync found DT_FOUND signal being asserted) that keeps track of the relative location of each CKR clock with respect to symbol boundaries. The state tracker (STATE)

is a simple shift register that keeps track of the previous two bits that, combined with the current bit, determine the state of the filter storage subblock. The filter coefficient storage subblock takes the state information (DT_STATE) and the phase (DT_PH_I) and produces an output (FCW_DT) by means of a lookup table. The output is conforms with the frequency command word (FCW) notation, where the LSB bit of the integer part corresponds to the reference frequency. The filtered data output FCW_DT is fed as the $y(k)$ input signal to the frequency modulator of Fig. 5.4.

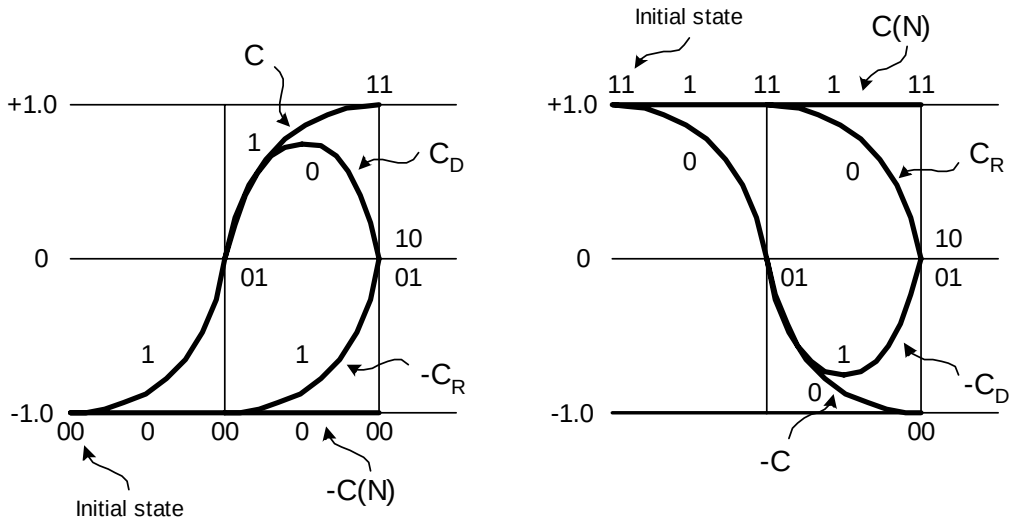

Figure 5.9. Curves for various state transitions based on previous symbol values

Fig. 5.9 illustrates curves for various state transitions based on the current symbol DT[0] and the previous two symbol values, DT[-1] and DT[-2]. It expands Fig. 5.7 into a detailed development. Each state selects one of two candidate curves: input bit zero and input bit one. On the left plot, the initial state is “00” (state format: “DT[-2] DT[-1]”). On the right plot, the initial state is “11”. In both cases the output curves are the same, except flipped over a horizontal axis. This reveals some redundancy which could be exploited to ease curve storage requirements.

Table 5.1. Transmit filter output curves

DT[-2]	DT[-1]	DT[0]	curve	coefficient set operation
0	0	0	$-C(N)$	negated max value
0	0	1	$-C_R(k)$	negated reversed order coefficients
0	1	0	$C_D(k)$	di-bit coefficients
0	1	1	$C(k)$	basic set of coefficients
1	0	0	$-C(k)$	negated basic set of coefficients
1	0	1	$-C_D(k)$	negated di-bit coefficients
1	1	0	$C_R(k)$	reversed order coefficients
1	1	1	$C(N)$	max value

The current output curve can be easily determined by the state field from the state circuit, as shown in Table 5.1. The fundamental state, from which curves of all other states are derived, is "011", where the right bit is the most recent. It is associated with the curve "C" which is defined as the cumulative coefficient $c(k)$ in Appendix C.

Curve $C(k)$ is a function of index k, where $0 \leq k \leq N-1$ and N is the oversampling ratio or integer number of CKR clock cycles per symbol. It is stored in a filter memory with eight bits per sample. From the $C(k)$ points, data points for all the other curves could be derived. The curve $C(N)$ is a shorthand notation for the constant maximum value of the accumulated coefficient that corresponds to maximum frequency deviation. Its value is roughly equal or just slightly higher than $C(N-1)$ since the slope and the incremental contributions are very small at this point.

The reversed-order coefficients $C_R(k)$ are computed as

$$C_R(k) = C(N-k) \tag{5.8}$$

where $C(N)$ is the maximum value of the accumulated coefficient as defined above. The

di-bit coefficients $C_D(k)$ feature the highest amount of ISI and are computed as

$$C_D(k) = C(k) + C(N-k) - C(N-1) \tag{5.9}$$

The remaining curves are the numeric negations of these curves. Only $C(k)$ coefficients are stored as "constants" – the rest are calculated on power-up through an arithmetic combinatorial logic.

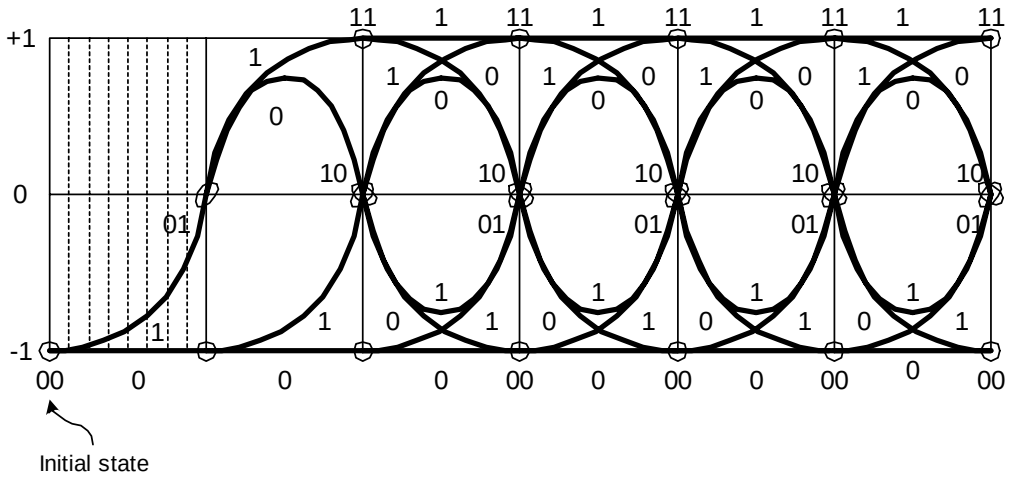

Figure 5.10. Trellis a GFSK transmit filter

Fig. 5.10 shows the evolution of all possible state trajectories starting from a state of "0". It is a waveform generalization of the plot shown earlier in Fig. 5.7. During each segment, a curve is selected based on the current symbol bit (0 or 1) and the 2-bit state, which depends on the previous two symbols. Since there are four states and symbol alphabet is of size two, then there are total of eight distinct curves. The waveform with the most amount of inter-symbol interference would be a "101010..." pattern, and the waveform with the output stuck at zero would be "000000...".

5.4 Just-in-time DCO gain calculation

The DCO gain estimation $\widehat{K}_{DCO}$, as defined in Chapter 3.2, could be conveniently and *just-in-time* calculated at the beginning of every packet. As alluded to in Sec. 4.20, the DCO gain could be alternatively estimated as the ratio of the *forced* oscillating frequency deviation Δf_V to the *observed* steady-state change in the oscillator tuning word $\Delta(OTW)$:

$$\widehat{K}_{DCO}(f_V) = \frac{\Delta f_V}{\Delta(OTW)} \tag{5.10}$$

Referring to Fig. 7.5 (page 230), at the end of the fast tracking and beginning of the regular tracking PLL operation, there is a sudden frequency jump marking the beginning of the proper transmit modulation mode. This Δf_{max} frequency jump corresponds to the maximum negative frequency deviation for data bit ”0” (that corresponds to the ”-1.0” symbol) and equals:

$$\Delta f_{max} = m/2 \cdot R \tag{5.11}$$

where m is the GFSK modulation index and R is the data rate. (For BLUETOOTH, m = 0.32 and R = 1 Mb/s resulting in Δf_{max} = 160 kHz; for GSM, m = 0.5 and R = 270.833 kb/s resulting in Δf_{max} = 67.708 kHz.) Since the frequency jump is precisely known as commanded by modulating data part of the frequency command word (FCW_DT), one needs to observe the tuning control word in the steady-state in order to determine the DCO gain.

$$\widehat{K}_{DCO}(f) = \frac{\Delta f_{max}}{\Delta(OTW)_{max}} \tag{5.12}$$

If the K_{DCO} gain is estimated correctly to start with, the precise frequency shift will be accomplished in one step, as shown in Fig. 5.11. However, if the K_{DCO} is not estimated

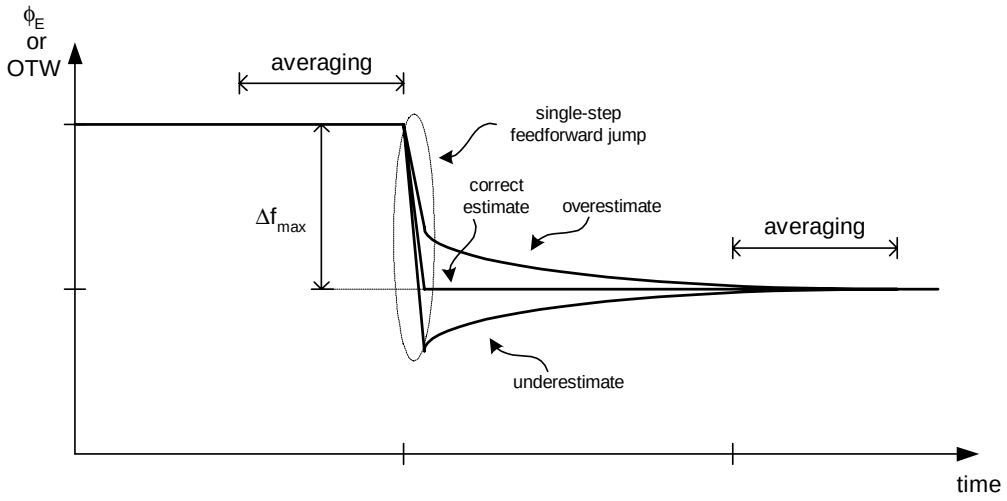

Figure 5.11. DCO gain estimate by measuring tuning word change in response to a fixed frequency jump

accurately, then the first frequency jump step will be off target by

$$\frac{K_{DCO}}{\widehat{K_{DCO}}} - 1$$

fraction and one would require a number of clock cycles to correct the estimation error through the normal PLL loop dynamics, preferably in the fast-acquisition mode, with the transfer function described by Eq. 5.6 (page 183). The K_{DCO} gain could then be simply calculated as the ratio of Δf_{max} to the *oscillator tuning word* difference. To lower the measurement variance, it might be required to average out the tuning inputs before and after the transition.

It should be noted that a frequency jump equal to the full modulation range is beneficial for two reasons: First, a little hardware overhead is required on top of the existing transmit modulator circuitry to execute the chosen frequency jump. Second, measuring the local gain value around the expected operational range is bound to provide the most accurate estimate.

In order to further improve the estimate, a larger frequency step of $2 \cdot \Delta f_{max}$ covering the whole data modulation range, could be performed.

5.5 Power Amplifier

This section deals with the last stage on the integrated transmitter path – power amplifier (PA). The purpose of a PA in a BLUETOOTH system is to deliver several mW of RF power to the antenna in an efficient manner.

Power amplifiers have been traditionally categorized under many classes: A, B, C, D, E and F [13]. Classes A, B and C are considered classical in the sense that both the input and output waveforms are sinusoidal. Voiding this assumption with class E and F operation leads to higher performance and efficiency.

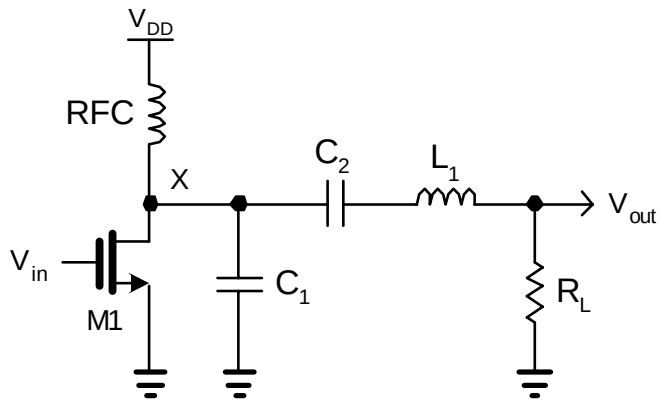

Figure 5.12. Class-E power amplifier

Class-E stage is a nonlinear amplifier that potentially achieves 100% efficiency while delivering full power. It has been shown to be best suited in a low-voltage environment [69]. An ideal schematic is shown in Fig. 5.12. Transistor M1 is used here as an on/off switch. RFC is a radio-frequency choke, a large external inductor (usually about 100 nH) that acts

as a bi-directional current source at RF frequencies, and which connects the switch to the supply voltage V_{DD}. C_1 is a capacitance connected in parallel to the switch and includes the parasitic capacitance of M1. The C_2-L_1 filter circuit is tuned to the first harmonic of the input frequency and only passes a sinusoidal current to the load R_L.

The values of C_1, C_2, L_1 and R_L are chosen such that V_X satisfies three conditions [13]:

1. As the switch turns off, V_X remains low long enough for the current to drop to zero.
2. V_X reaches zero just before the switch turns on.
3. dV_X/dt is near zero when the switch turns on

In this case, as almost universally in GHz-range applications, the load resistance R_L is 50 Ω. Inductor L_1 is realized as a bond-wire of a 3 nH value. C_2 is an external 1.5 pF capacitor. C_1 is an internal metal-to-metal 1.4 pF capacitor. The M1 transistor is a 32-finger NMOS of size W/L = 2.5/0.15.

During the time when the switch is closed, the voltage across it is zero. During the time when the switch is open, the current through it is zero. Since the voltage and current of the switch do not overlap, the power dissipation of the switch is ideally zero. When the switch turns off, the current through RFC splits between the two branches containing C_1 and R_L. The capacitance C_1 starts charging and slowly produces a voltage across the switch. Satisfying condition (1) is quite easy and it is guaranteed by C_1. Without C_1, V_X could rise as V_{in} is dropped introducing substantial power loss in M1. When the switch turns on, any charge stored on C_1 will be discharged to the ground resulting in a power loss. In order to avoid this, the circuit must be designed to satisfy conditions (2) and (3)

such that the voltage across M1 reaches zero at the turn-on time and stays there for some time.

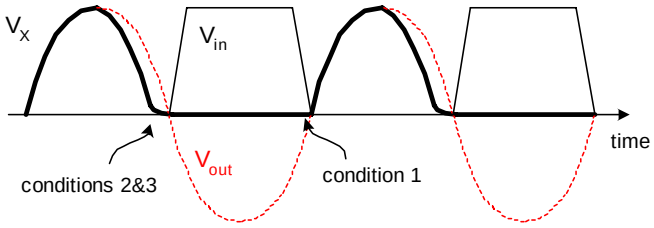

Figure 5.13. Waveforms of the class-E PA

After the switch turns off, the load network operates as a damped second-order system with overdamped, underdamped or critically damped response. If the quality factor Q of the network makes it critically damped, then the drain voltage of M1 will follow the V_X curve in Fig. 5.13. This will satisfy conditions (2) and (3). If the network response is underdamped, there would be a dying oscillatory response of V_X and condition (3) could not be met. If the network response is overdamped, V_X might not reach zero by the time M1 turns on. It should be noted that due to the inverting nature of the amplifier, the input and output waveforms are shifted by 180 degrees.

In the ideal situation mentioned above, the efficiency of a class-E amplifier is 100%. However, in practice, the switch has a finite on-resistance, and the transition times from the off-state to the on-state and vice-versa are not negligible. Both of these factors result in power dissipation in the switch and reduce efficiency [69].

The class-E power amplifier was chosen in the proposed architecture due to the following reasons:

- Low voltage operation – ideally suited for deep-submicron CMOS. The end stage

transistor operates as a switch. Unlike in class A, B and C stages where the transistor acts as a current source and the V_{ds} must be precisely controlled at all times to be high enough in order to avoid entering the triode region, the V_{ds} here can be arbitrarily low and no control is necessary. The only requirement is that the V_{gs} must be able to go higher than the threshold voltage for the transistor to turn on.

- Digital input – the transistor switch works best with digital input waveforms, preferably with sharp rise and fall times. Contrast it with the classical PA's which require sinusoidal inputs. This is where the deep-submicron strengths lie. The DCO output is already in a digital format. Duty cycle of the input waveform can conveniently control the output amplitude and power.

- Class-E stage is preferred over class-F, which is similar to class-E but has an additional filtering network to create a high impedance load at the transistor drain for the second or third harmonics, thus sharpening the edges. The filtering network requires an extra LC tank, which is quite area-expensive in a deep-submicron process. In addition, class-F amplifiers have consistently shown in practice worse performance than class-E amplifiers [69].

- High power efficiency: theoretically 100%, but in practice 80-90% have consistently been reported [69]. The efficiency does not degrade substantially with the output power.

Since the targeted output power for BLUETOOTH applications is only several mW, the efficiency is not as important as meeting the basic design specifications, which in itself is quite a challenge at 1.5 V supply. In this case, it is still advantageous to operate the power

amplifier with a digital switch, even though the class-E conditions might not be fully met.

5.6 Digital Amplitude Modulation

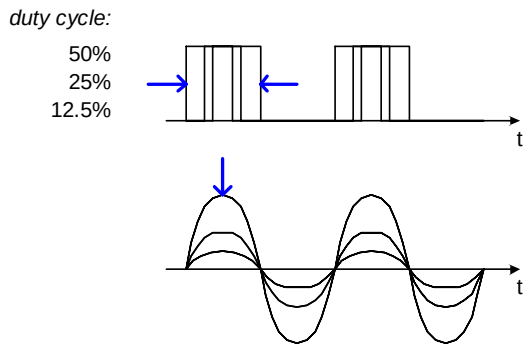

Figure 5.14. Output power control through duty cycle of the class-E PA input

As mentioned in Sec. 5.5, the output power of a class-E RF power amplifier could be controlled by changing the duty cycle or pulse width of its RF digital input. The pulse width controls how long the switch is turned on during the RF cycle and, consequently, how much energy gets transferred to the load R_L. This idea, shown in Fig. 5.14, is proposed to be used for the transmitted RF amplitude and power control. In the implemented BLUETOOTH testchip, only a static RF power control is required and this is here accomplished through the RF waveform amplitude control. The *dynamic* amplitude control presented here has not been fully implemented.

As of this writing, there have been no reports on using this kind of pulsewidth modulation in RF applications. Even a foremost authority on RF-CMOS, Thomas Lee, declared in his famous book [68] that this idea is "fairly useless at the gigahertz carrier frequencies of cellular telephones. (...) difficult to use pulsewidth modulation once carrier frequencies exceed roughly 10 MHz." This research successfully achieved the 2.4 GHz operation, mainly

due to the ultra-fast speed of operation of digital logic gates in this modern deep-submicron process.

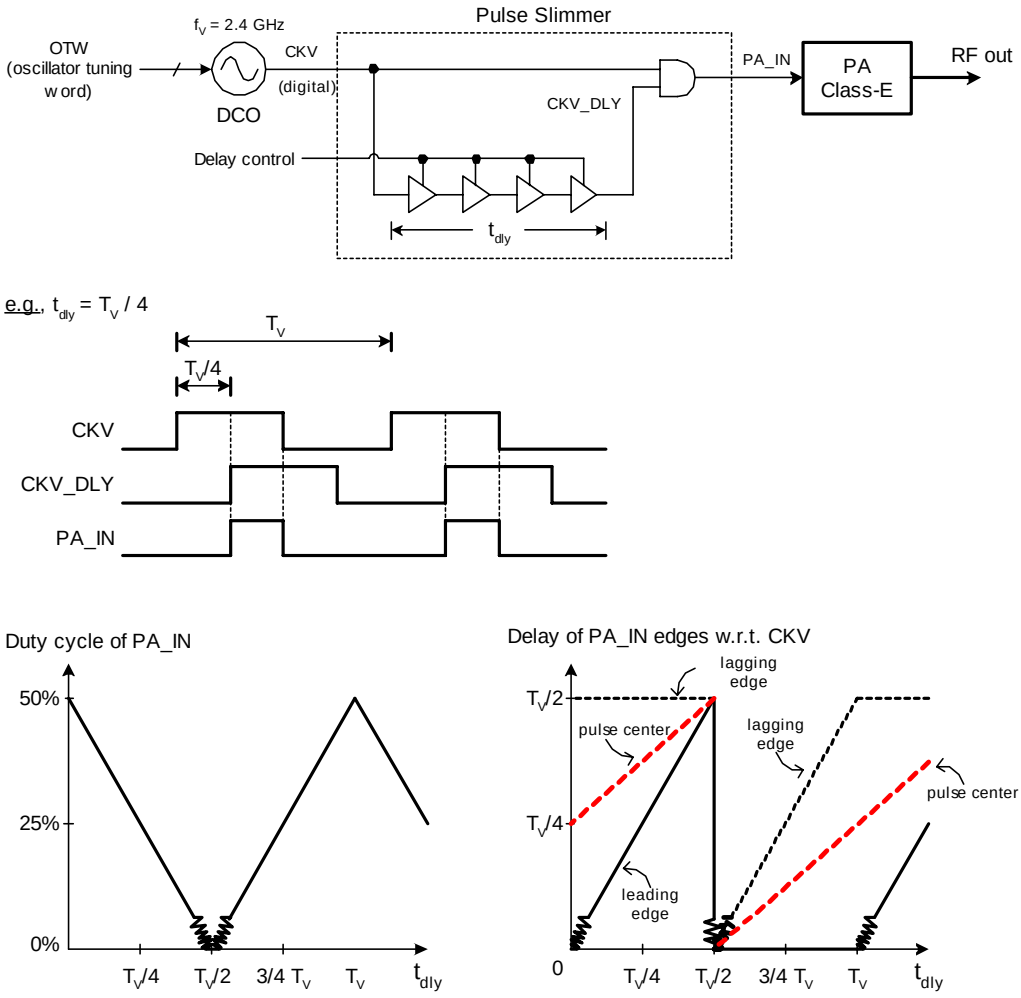

Figure 5.15. Digital amplitude control through pulse-width modulation

Fig. 5.15 shows an example implementation of an amplitude modulation using a digital pulse slimmer method. The digital oscillator output clock CKV is met at the digital AND gate with its delayed replica. The delay path could be constructed of a string of inverters or buffers, in which the delay could be controlled through a current-starving mechanism or variable capacitative load. In this implementation, delay control through a variable power

supply voltage was chosen. The AND gate output is connected to a class-E power amplifier input PA_IN. Depending on the relative time delay of the two paths, the timing and duty cycle of the AND gate output could be controlled. The duty cycle or pulse-width variation directly affects the turn-on time of the PA digital switch, thus establishing the RF output amplitude. The amplitude-vs.-pulsewidth relationship is quite linear, except for the very narrow input pulse which might not have enough energy to reliably turn on the switch. This non-linear region of operation could descriptively be called a "dead zone" – a reference to a commonly used term in conventional phase detectors. The dead zone could be entirely avoided at a system level by choosing modulation techniques that guarantee a certain minimum level of the signal envelope. For example, GFSK and GMSK are *constant*-envelope modulation schemes. Offset-8PSK is a modulation technique used in GSM-EDGE that purposefully rotates the I-Q constellation with every symbol so as to avoid the origin. These methods have been employed for a long time to improve efficiency of power amplifiers and to facilitate the use of a saturation mode of operation.

The timing diagram on the bottom right of Fig. 5.15 shows two regions of operation with different behavior of leading and lagging output edges with respect to the t_{dly} delay of the delay path. In the first region, the leading edge of the output traverses but the lagging edge does not. A reversed operation takes place in the second region (dotted line). Since the pulse position is determined of where its center lies, neither of the two provide orthogonality of the phase modulation in the oscillator and the amplitude modulation in the oscillator pulse slimmer circuit. Consequently, the phase adjustment is necessary with the amplitude change. This is not a difficult task since the phase control is in the digital domain through manipulation of the *oscillator tuning word* (OTW).

In order to save power and reduce jitter due to the long chain of buffers or inverters in the delay path, it might sometimes be beneficial to use an inverted CKV_DLY signal, which is equivalent to an extra half-cycle ($T_V/2$) periodic shift. This could be accomplished through either feeding the delay path from the inverted CKV clock output, or inverting the CKV_DLY signal itself. It is important to note that the maximum required amount of delay is never greater than half of the CKV clock cycle since the negated CKV (of the opposite phase) could always be used.

5.6.1 Discrete Pulse Sliming Control

The implementation of Fig. 5.15 is not completely digital since it requires a "delay control" signal, which is an analog voltage. It is proposed that the coarse delay control be performed by adding and subtracting a number of the inverters or buffers in the delay path. Finer delay control could be implemented by selecting taps of a delay line.

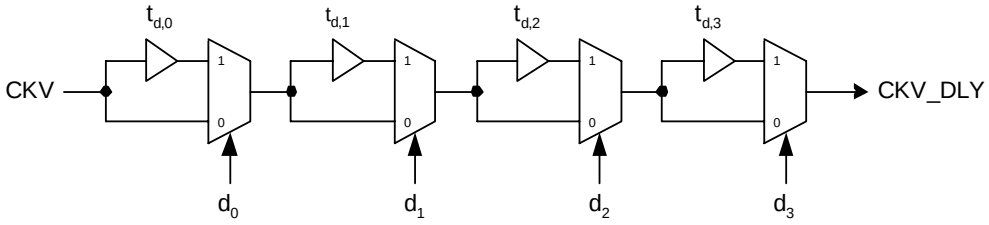

Figure 5.16. Discrete delay control of the delay path

Fig. 5.16 shows a delay path example of four buffer delay stages. The buffer delay could be the same in each stage. In this case, it would result in a total of five possible delay values, from zero to four. A better solution might be to have a binary-weighted arrangement of the buffer delays. In this case, it would result in a total of 16 possible delay values, from 0 to 15, not including the fixed multiplexer delays. The "effective" delay could be expressed by

the following equation:

$$t_{dly} = \sum_{j=0}^{N-1} d_j \cdot t_{d,0} \cdot 2^j = t_{d,0} \cdot \sum_{j=0}^{N-1} d_j \cdot 2^j \tag{5.13}$$

where N (=4 here) is the number of binary-weighted stages, d_j is the jth control word bit and $t_{d,0}$ is the basic element delay of weight 2^0. Each next stage contains twice the amount of the delay, which could be conveniently realized as doubling the number of inverters or buffers.

The delay control word must be synchronized to the CKV clock in order to avoid changing it while the signal is still propagating. A method similar to the DCO synchronous sampling and timing adjustment of Fig. 3.2 (page 79) could be used.

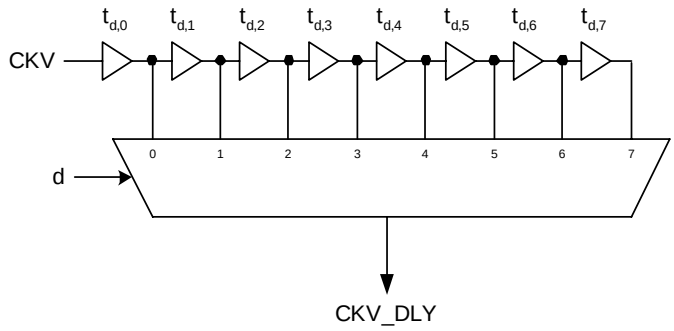

Figure 5.17. Discrete delay control of the delay path with a larger mux

The delay buffer arrangement of Fig. 5.16 is preferred over a transversal delay line configuration in which a large multiplexer selects various taps of a delay line comprised of a string of inverters or buffers, as shown in Fig. 5.17. This is mainly due to the difficulties of building a fast larger multiplexer with equalized delays for various inputs.

An alternative method to increase the effective delay resolution below that of a single inverter/buffer would be to change dynamically the number of inverters at a rate much higher than the symbol rate. The time-averaged delay value of the number of inverters could thus

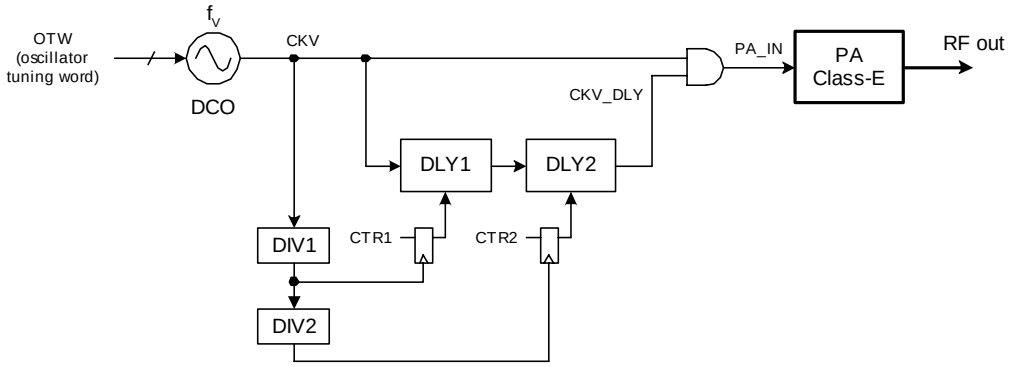

Figure 5.18. Discrete delay of PWM with additional high-speed dithering

be controlled with a fractional resolution, as previously illustrated in Fig. 1.22 (page 34) for a fractional frequency division ratio. Here, again, a $\Sigma\Delta$ digital dithering stream is a good choice due to its noise shaping properties. It should be noted here that a binary-weighted delay control would not work very well with the high-speed dithering. However, the delay path could be cascaded into a lower-rate binary-weighted structure and a higher-rate unit-weighted structure that would be subject to the dithering. Such an implementation is shown in Fig. 5.18 with high-speed delay dithering DLY1 and low-speed delay selection DLY2. DIV1 and DIV2 are CKV clock edge dividers and might be implemented as power-of-2 numbers.

5.6.2 Regulation of Transmitting Power

The dynamic amplitude modulation method could be used in its simplest form to statically regulate the output power of the class-E power amplifier. It is done in a very efficient manner by injecting enough energy into the PA with every oscillator cycle to achieve the desired output amplitude or power. This is the main application of the pulse-width modulation for the proposed BLUETOOTH transmitter which does not require a dynamic amplitude

modulation. However, the idea of dynamic amplitude control is investigated here for other applications that would require it. Examples are 802.11b and EDGE, which is a packet data service for GSM and is based on a version of the QPSK and 8PSK modulation.

5.6.3 Tuning Word Adjustment

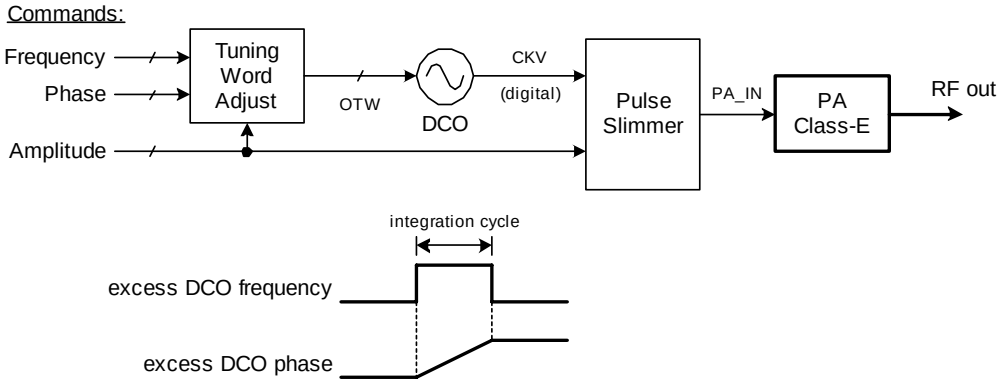

Figure 5.19. PAM modulation through tuning word adjustment

The polar coordinate representation of the PAM modulation was first illustrated in Fig. 1.12 (page 21). A block diagram of the proposed implementation is shown in Fig. 5.19. Since phase is integral of frequency

$$\phi(t) = 2\pi \int_{-\infty}^{t} f(\tau) d\tau \tag{5.14}$$

the DCO phase modulation is accomplished through a timed frequency adjustment. In a discrete-time system, the frequency control is performed only at update intervals, usually determined by the frequency reference clock edges of period T_R. Eq. 5.14 is now re-written for the discrete-time operation.

$$\phi(k) = 2\pi \sum_{-\infty}^{k} f(k) T_R \tag{5.15}$$

where k is a time index. To simplify the analysis, Eq. 5.14 and Eq. 5.15 could be interpreted as pertaining to the excess phase and amplitude quantities.

The magnitude command modulates the PA output amplitude using one of the presented above methods. However, as shown in Fig. 5.15, the side effect of the pulse slimming method is that the pulse center travels with the edge delay. Fortunately, this pulse center location is easy to predict, especially in the fully digital control environment. The proposed correction of the pulse center dislocation is to change the DCO frequency for a single clock cycle (Fig. 5.19) such that the resulting phase is equivalent to the predicted pulse center shift.

5.6.4 Advantages

The following are the benefits of the proposed method in comparison with the conventional I-Q-based transmit modulation:

- Maximally-digital implementation, very suitable for a deep submicron CMOS process implementation.
- Extremely power efficient through the utilization of a highly-nonlinear saturation-mode class-E power amplifier with regulated input duty cycle through a digital pulse sliming.
- No need for an I-Q modulator.
- No need for a digital-to-amplitude conversion of the I-Q baseband signals.

5.7 Summary

This chapter adds a phase/frequency modulating capability to the all-digital PLL frequency synthesizer proposed in Chapter 4. It is recognized that a mere adjustment of the frequency control word (FCW) would subject it to the PLL loop bandwidth, which might attenuate higher frequency components of the modulating transmit data. A predictive modulation mechanism with a closed PLL loop operation that would solve this problem is proposed. A solution to the related problem of estimating the DCO gain is proposed.

It should be noted at this point that the proposed feedforward phase/frequency adjustment capability might find useful applications in other areas that are not related to the transmit modulation. For example, a DSP processor might request the internal clock generation unit to immediately change its clock rate while entering or leaving a power-saving mode. In another example, it might be necessary in a hard-disk drive read channel to “instantly” shift the clock phase of a time-base generator by 180 degrees. Instead of multiplexing a delay line taps, one could shift the right amount of frequency within a clock cycle.

Finally, a highly-efficient class-E power amplifier (PA) and a power regulation circuitry are added to the DCO output in order to demonstrate the full transmitter for a short-range wireless communication.

This chapter also described a novel amplitude modulation method that could be conveniently implemented in a digital fashion.

CHAPTER 6

ALL-DIGITAL RF FREQUENCY SYNTHESIZER IMPLEMENTATION

6.1 Overview

In this chapter, an implementation of the full top-level transmitter core is described. First, a top-level block diagram is presented and all major constituent blocks are listed. Then, parameters of the deep-submicron CMOS process are highlighted. Chip micrographs of the testchip are shown afterwards. Finally, the IC chip evaluation board is described.

6.2 Transmitter Core Implementation

The implementation details of the testchip transmitter are shown in Fig. 6.1. The design is executed based on the Texas Instruments ASIC digital flow with special adjustment due to analog/RF content. The transmitter core (TX_CORE) is partitioned into the following blocks

- *Low-speed digital* (LSD) superblock with clocks no faster than the reference frequency of 13 MHz.
- *High-speed digital* (HSD) subchip with clocks much faster than the reference frequency. It contains variable phase accumulator PV (running at 2.4 GHz) and tracking fractional-part oscillator interface OTF (running at 600 MHz).

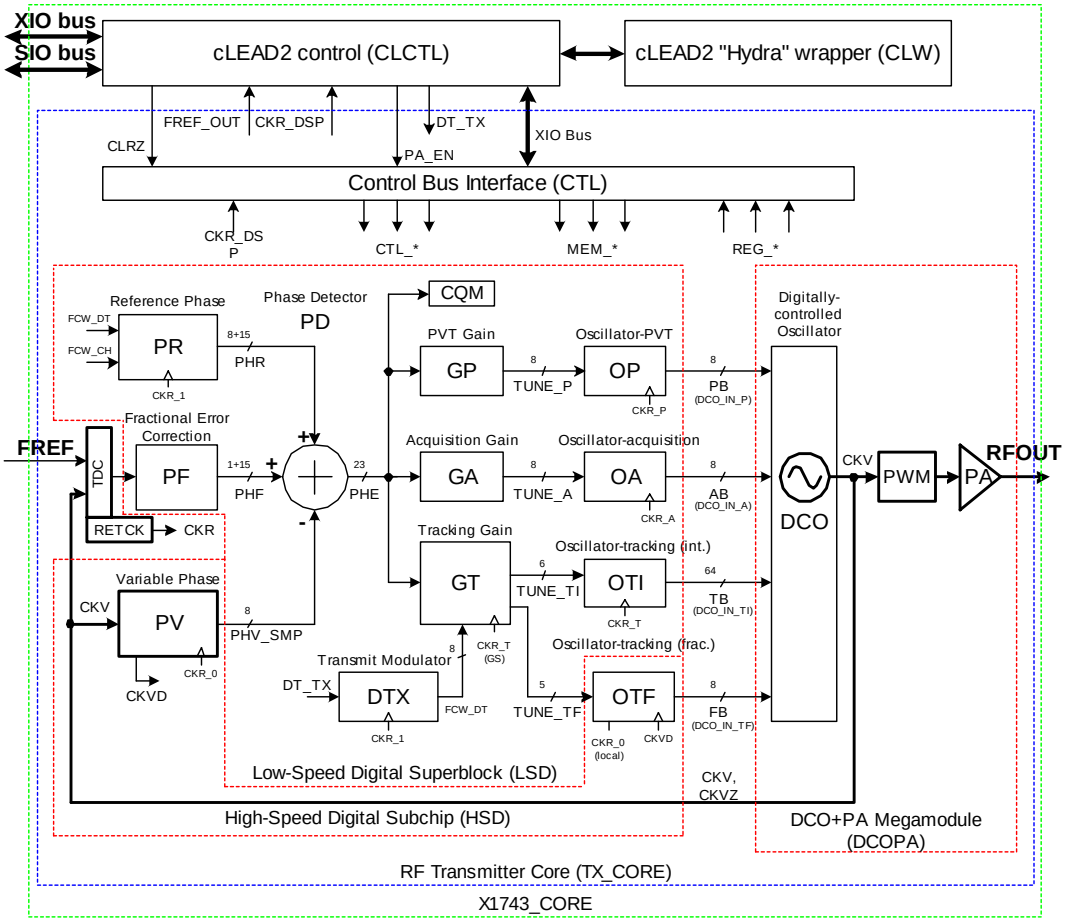

Figure 6.1. ADPLL-based “X1743” transmitter core

- *Time-to-digital converter* (TDC) megamodule. It also contains the FREF clock re-timing (RETCK) circuit. The TDC operates at FREF, but it has high timing precision requirements. RETCK operates mostly at 2.4 GHz.

- DCOPA megamodule: digitally-controlled oscillator and power amplifier are conveniently combined as a single RF/analog module.

- Control bus interface (CTL) superblock. Allows the transmitter to be controlled through an XIO parallel-port interface.

The transmitter core is encapsulated by the X1743 core, which is the highest level of hierarchy below the final chip level that contains I/O pads and I/O buffers. The X1743 core also contains an interface and wrapper circuitry for the C54X Texas Instruments DSP codenamed cLead2 “Hydra”. The control interface block (CTL) allows the transmitter to be controlled from one of several sources: external XIO bus, internal XIO bus from DSP, serial SIO bus and JTAG interface. The transmitter core has a dedicated 8-bit address space of 16-bit data registers.

All the digital blocks are synthesizable from the *register transfer level* (RTL) subset of VHDL code using SYNOPSYS Design Compiler. This includes the circuits running at the 2.4 GHz clock, such as variable phase PV. The blocks are auto-placed and auto-routed using AVANTI layout software package. The analog and RF blocks are modeled using behavioral VHDL code. Top level interconnects are described in a structural VHDL, which are synthesized into a netlist without a need to draw schematics.

The implemented transmitter is digitally-intensive. The only blocks that internally follow the established RF/analog design practices are DCO and PA, even though at their top I/O level they are characterized using the digital flow. These two blocks are vastly different from their conventional counterparts and they seamlessly fit the proposed digital design. Even though they are built of only a handful of transistors, they occupy almost half of the total RF area, as shown in Fig. 6.3.

Table 6.1 lists the top-level transmitter core building blocks with their name abbreviations and the figures (with page numbers) in which the blocks are described.

Table 6.1. Top-level transmitter core building blocks

Block Name	Block Abbrev.	Figure	Page
Reference phase accumulator	PR	4.32	172
Variable phase accumulator	PV	4.9	124
Fractional phase error estimator	PF	4.10	125
Time-to-digital converter	TDC	4.11	126
FREF clock retiming	RETCK	4.16	139
Phase detector	PD	4.4	117
Transmit modulator	DTX	5.8	189
PVT gain	GP	3.10	88
Acquisition gain	GA	3.10	88
Tracking gain	GT	3.10	88
PVT oscillator interface	OP	3.12	90
Acquisition oscillator interface	OA	3.13	91
Tracking oscillator interface	OTI & OTF	3.16	97
Digitally-controlled oscillator	DCO	2.13	69
Pulsewidth modulation	PWM	5.15	200
Power amplifier	PA	5.12	195

The sequencer used to control the operational modes of the synthesizer, such as PVT, acquisition, fast tracking and tracking, was implemented in the DSP software. It is also possible to step through the operating modes manually by writing external XIO commands.

Chip Quality Monitor (CQM) is used to gather certain statistics about the ADPLL operation. It is not described in this work.

6.3 CMOS Process Parameters

The testchip of the proposed BLUETOOTH transmitter is implemented in a latest (as of this writing) Texas Instruments' deep-submicron CMOS process named C035. Table 6.2 describes the key process technology parameters. It features the routed digital gate density of 150 equivalent gates per mm^2.

Table 6.2. Key C035 process technology parameters

interconnection material	copper
minimum metal pitch	0.35 μm
transistor nominal voltage	1.5 V
L drawn	0.11 μm
L effective	0.08 μm
gate oxide	29 A
substrate resistivity	$\leq 50\ \Omega \cdot$ cm

6.4 Chip

Fig. 6.2 is a die microphotograph of the X1743 testchip. The total silicon dimensions are 3290 μm x 3290 μm. This includes 160 μm dedicated on each side for I/O pads. The C54X DSP occupies 2430 μm x 2470 μm.

Fig. 6.3 is a die microphotograph of the RF transmitter area. It is located in the lower-left corner. Its area is only 0.54 mm^2. The inductor itself occupies a 270 μm x 270 μm square.

X1743 testchip is bonded and encapsulated in an 80-ball 5x5 mm MicroStar Junior ^{TM} ball-grid array (BGA) package.

6.5 Evaluation Board

Fig. 6.4 is a photograph of the evaluation board. The *printed circuit board* (PCB) is constructed of six layers of the standard FR4 material. The testchip is located at the center. The 2.4 GHz RF output, the 13 MHz frequency reference (FREF) input and its retimed output (BBCLK) are connected using semi-precision *subminiature "A"* (SMA) connec-

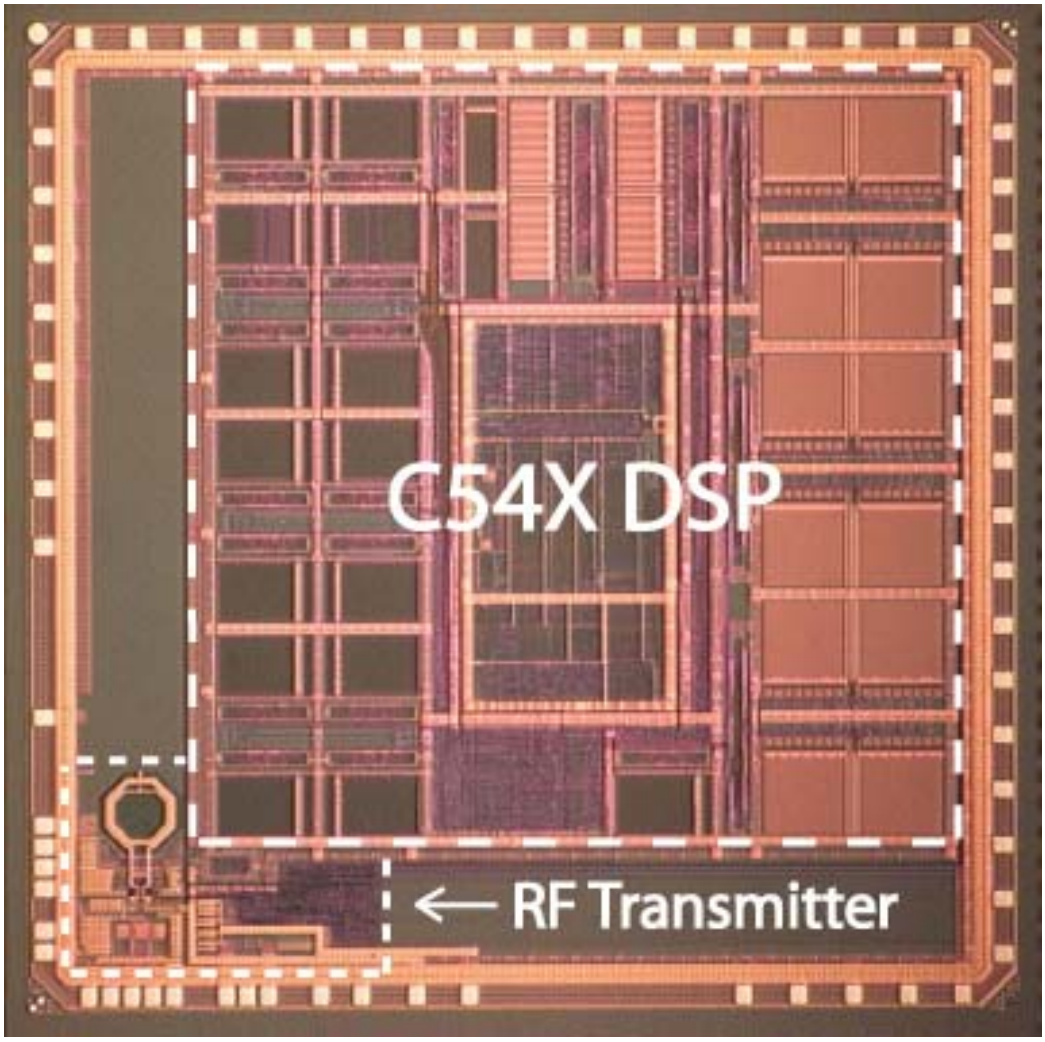

Figure 6.2. Micrograph of the X1743 testchip

tors, which provide 0–10 GHz broadband performance with low reflections and constant 50 Ω impedance. The connectors on the right-hand side attach the evaluation board to a PC interface board (not shown), whose purpose is to control the testchip by reading and writing its registers by means of a *graphical user interface* (GUI) computer program.

Fig. 6.5 shows a schematic of the evaluation board.

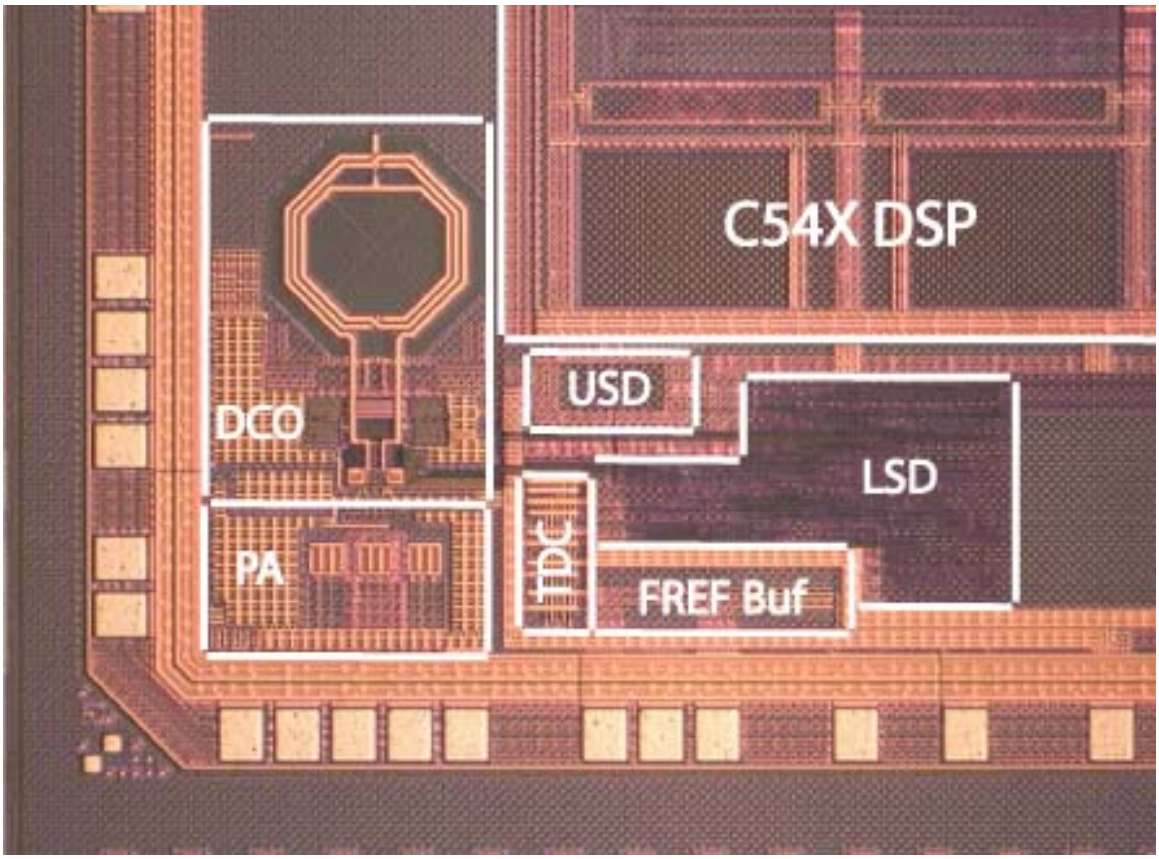

Figure 6.3. Micrograph of the RF transmitter area

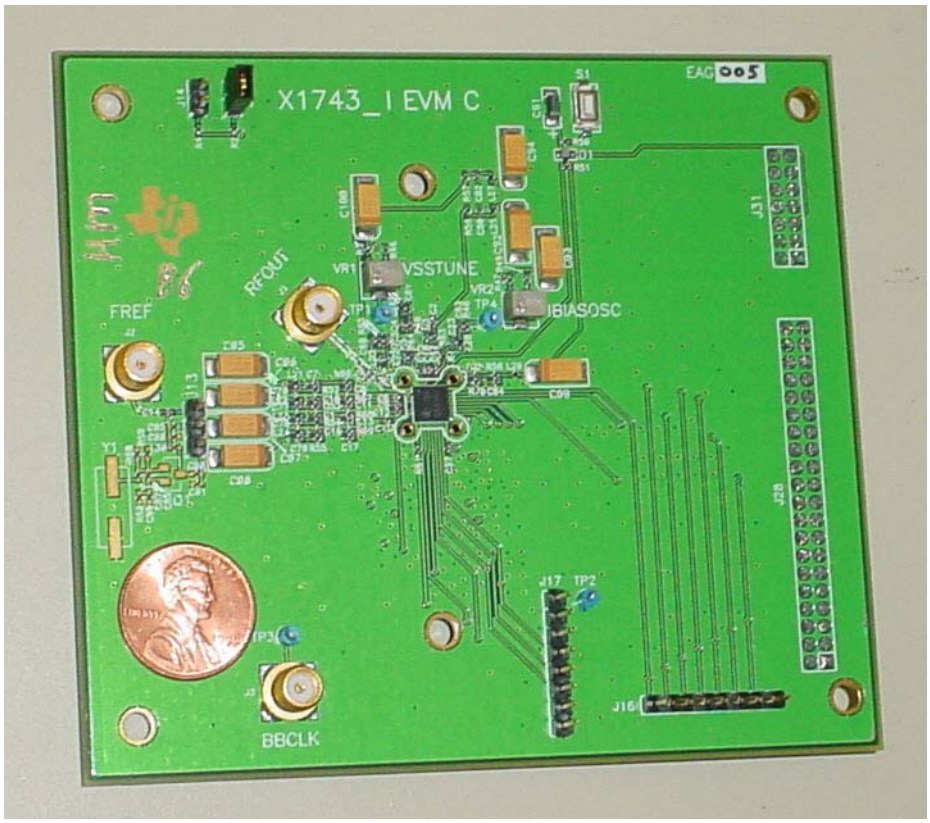

Figure 6.4. Photograph of the evaluation board

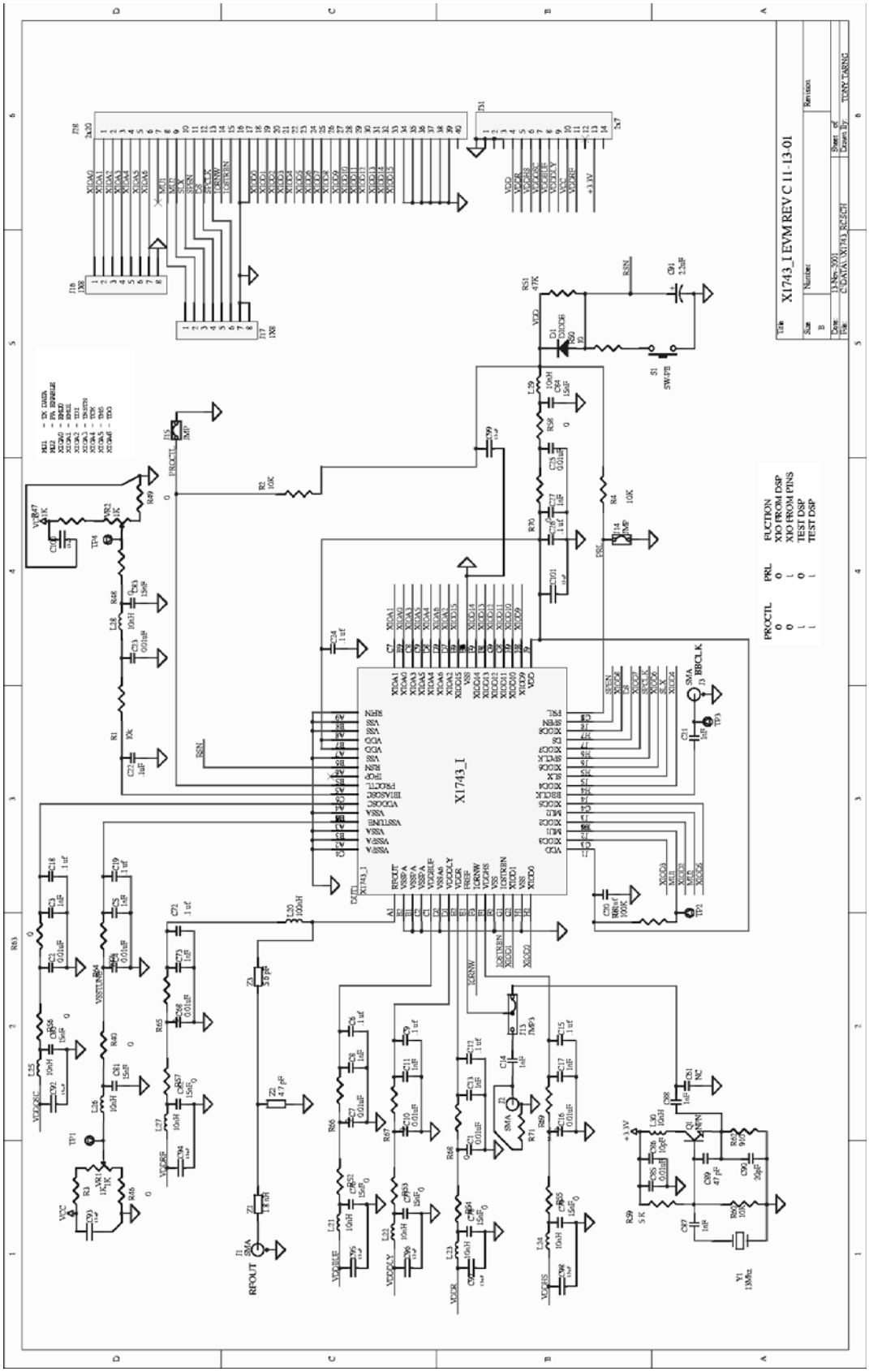

Figure 6.5. Evaluation board schematic

CHAPTER 7

BEHAVIORAL MODELING AND SIMULATION

7.1 Behavioral Modeling

The behavioral modeling and simulation methodology is based on the VHDL hardware description language and was first described by the author in [41] for a mixed-signal read channel used in magnetic recording hard-disk drives.

There have been a variety of communication channel modeling methods. At the pure system level there are C and MATLAB models, which are highly abstract and with very weak links to the actual hardware. On the opposite side of the spectrum, the system could be modeled at a very low level entirely in SPICE for analog-intensive systems or in SPICE and Verilog (or VHDL) combination, with a varying degree of link between the two disparate simulation engines (e.g., TI-SPICE and Verilog co-simulation backplane). Establishing a link to a non-event-driven engine, such as SPICE, results in a hefty price of a simulation performance, thus making it impossible to determine the very basic figure of merit of a communication channel: bit error rate.

This work describes a system and a simulation environment that are based on a standard single-core simulator. This system is well suited for digitally-intensive applications with a fair amount of analog circuit content. Extensive links to a file system for pre- and post-processing facilitate rich simulation and analysis environment. The main advantage

of the single simulation engine is that it allows a seamless integration of all hardware abstraction levels in a uniform environment. The single most important feature of a standard VHDL language, which makes it far superior than Verilog for mixed-signal designs is its support of real or floating-point type signals. Widespread simulation and synthesis support of a standard VHDL language makes it possible for a complex communication system to achieve “build what we simulate, and simulate what we build” goal.

7.2 Simulation Methodology

VHDL is based on an event-driven simulation engine. The simulator proceeds to the timestamp of the next event if all the activities associated with the current timestamp are exhausted. It is a very efficient method since the simulation activity is only spent on a per-needed basis. This is in sharp contrast with some other system-level simulators which are based on an oversampled clock, such as Simulink or SPW. In that environment, the simulation engine has to transverse all the equally-spaced timestamps that sufficiently oversample the signals.

Operating in an oversampled domain is less problematic with baseband signals and systems or in an environment with a single clock. Even two clock domains are not an issue if their frequency ratio is a small integer. In that case, the higher frequency clock is the common denominator. The operation becomes quite unwieldy, however, if the clocks are not easily related such that their common denominator clock is at a very high frequency.

Another environment exposing inefficiency of the oversampled domain simulator is a narrowband RF system. Oversampling a BLUETOOTH RF waveform of the 2.4 GHz

carrier generates an excessive amount of activity in light of the fact that the information is contained only in the 1 Mbps symbols. Considering eight samples per sinusoidal RF cycle, one 1 μs symbol would contain as many as 19200 samples! We have chosen a very efficient RF wave representation method in which only positive zero-crossing timestamps (t_k, t_{k+1}, t_{k+2} in Fig. 1.15 on page 24) are used.

7.3 Random Number Generator

A pseudo-random number generator is needed to create a long stream of the digital input stimulus, model an electronic thermal and flicker $1/f$ noise, and add jitter and wander deviations to the oscillator clock.

It became clear from the very beginning that the system-supplied pseudo-random number generators do not usually have good random properties. A typical implementation of *rand()* ANSI C function call or *uniform()* procedure call of the IEEE math_real VHDL package uses the linear congruential method which, while being very efficient and fast, suffers from sequential correlation on successive calls. There is a danger that using it might skew evaluated performance of a communication system by, for example, not exercising all possible paths through a Viterbi detector. A good uniform distributed random number generator with virtually no sequential correlation was built in VHDL based on Park and Miller algorithm with Bays-Durham shuffle as described in [66]. A random number generator with Gaussian (normal) distribution was build based on the Box-Muller method as described in [66].

7.4 Clock Jitter and Wander Effects

Phase noise of the DCO oscillator is modeled using jitter and wander constructs. The flat electronic noise of the $1/\omega^0$ region in Fig. 1.4 (page 12) is modeled as a non-accumulative jitter. The $1/\omega^2$ region of the upconverted thermal noise, on the other hand, is modeled an accumulative wander.

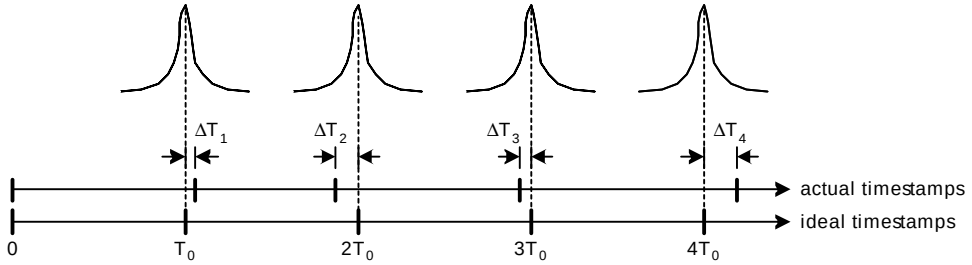

Figure 7.1. Development of a non-accumulative timing deviation

Fig. 7.1 illustrates a modeling principle of the timing jitter. The $1T_0$, $2T_0$, $3T_0$, $4T_0$ timestamps are the ideal equidistant rising-edge events of an oscillator operating at frequency $f_0 = 1/T_0$. The ideal oscillator output might pass through a physical buffer that adds random fluctuations to its delay. The actual timestamps of the physical buffer output could be described mathematically as an additive random error at each occurrence of the ideal timestamp. These timing errors do not influence one another. If the random error is due to the thermal electronic noise, then the edge timing deviations are said to be *independent and identically-distributed* (iid) and are usually modeled as an *additive white gaussian noise* (AWGN). Fig. 7.1 shows the error probability curve from each ideal timestamp.

The period deviation is statistically twice the power of the timestamp deviation. This is due to the fact that an instantaneous period is perturbed from both sides and makes the neighboring errors not entirely independent. The period deviations could be modeled by

passing the timestamp deviations through a 2-tap FIR filter with $\{1,1\}$ coefficients.

Relationship between the time and frequency domains of the jitter could be obtained as follows. The noise floor $\mathcal{L}$ is a single-sided spectral density. It needs to be multiplied by two in order to arrive at the S_ϕ [rad^2/Hz] double-sided spectral density in Eq. 1.4 (page 12). Since the flat spectrum of the jitter in the discrete-time model extends to the Nyquist frequency, S_ϕ is multiplied by $f_c/2$ to arrive at the total power [rad^2]. The rms jitter [rad] is its square root value. The radian-unit quantity is converted to the timestamp deviation [sec] by multiplying it by the normalizing factor of $T_0/2\pi$. The above results in the following equation.

$$\sigma_{\Delta t} = \frac{T_0}{2\pi} \cdot \sqrt{\mathcal{L} \cdot f_c} \tag{7.1}$$

Substituting, T_0 = 417 ps, $\mathcal{L} = 10^{-150dB/10} = 1 \cdot 10^{-15} rad^2/Hz$ and f_c = 2.4 GHz, it amounts to $\sigma_{\Delta t}$ = 103 fs. This value of Gaussian jitter is used by default in all VHDL simulations.

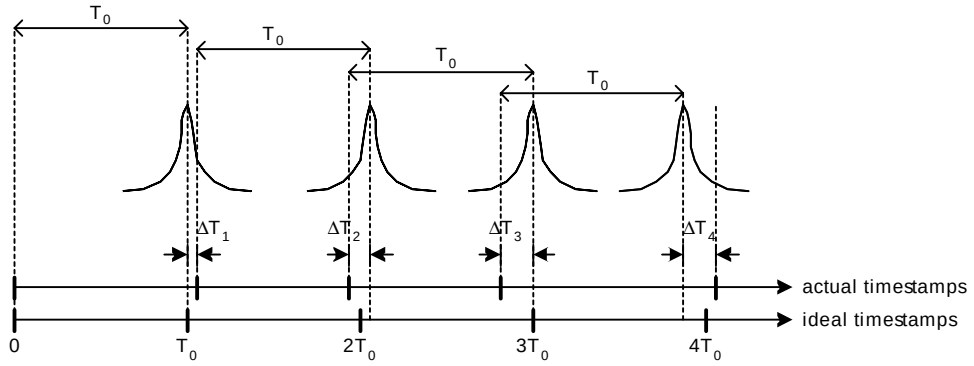

Figure 7.2. Development of an accumulative timing deviation

Fig. 7.2 illustrates a modeling principle of the timing wander, which is also called an accumulative jitter. This could be visualized by a physical oscillator of nominal period T_0, whose actual period varies slightly from one cycle to the next, due to, for example,

some thermal noise effects that are internal to the oscillator. In contrast to the jitter case, here each transition timestamp does depend on all previous period deviations. This simply acknowledges the fact that the ideal timestamps merely exist in theory and the only memory of a given transition is the previous transition timestamp. This behavior is modeled as a random walk.

Eq. 1.9 (page 15) is used to mathematically relate the wander component in the time and frequency domains. For example, based on lab measurements of a larger number of IC chips, a DCO phase noise level of -105 dBc/Hz at 500 kHz offset was conservatively assumed. Eq. 1.9 is now transformed into

$$\sigma_{\Delta T} = T_0 \cdot \Delta\omega \sqrt{\frac{\mathcal{L}\{\Delta\omega\}}{2\pi\omega_c}} \tag{7.2}$$

Substituting, T_0 = 417 ps, $\Delta\omega = 2\pi$ 500 kHz, $\mathcal{L}\{\Delta\omega\} = 10^{-105dB/10}$ = $3.16 \cdot 10^{-11} rad^2/Hz$ and $\omega_c = 2\pi \cdot 2.4GHz$, it results in $\sigma_{\Delta T}$ = 24 fs. This value of Gaussian wander is used by default in all VHDL simulations.

A diagram illustrating VHDL model of the DCO is shown in Fig. 7.3. Referring to the implementational block diagram of the DCO gain paths in Fig. 3.10 (page 88), DCO_IN_P, DCO_IN_A, DCO_IN_TI and DCO_IN_TF are the digital `std_logic_vector` inputs deviating the DCO oscillating frequency by controlling LC-tank capacitance of the PVT, acquisition, tracking-integer and tracking-fractional varactor banks, respectively. Signed-number integer representations of these inputs are multiplied by their respective unit time deviations of the "natural" period: DCO_QUANT_P, DCO_QUANT_A and DCO_QUANT_T. They are VHDL generics rounded off to the closest femptosecond and whose calculated values were shown in Table 2.2 (page 72) for the middle of the BLUETOOTH band. Their

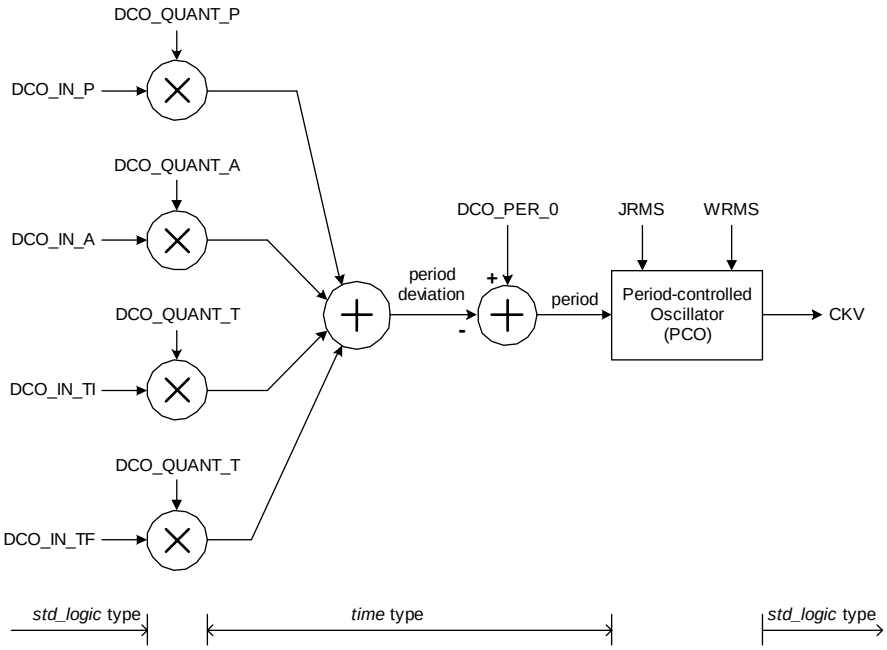

Figure 7.3. DCO time-domain model in VHDL

outputs are then summed up to create a composite period deviation signal of the VHDL `time`-type. This signal is then subtracted from the "natural" or center oscillating period DCO_PER_0[1]. This `time`-type signal controls the instantaneous period of the DCO oscillation. The actual code that implements the *period-controlled oscillator* (PCO) is listed in Sec. D.2 in Appendix D. For performance reasons, foreign C function calls are used to calculate jitter and wander perturbations which would be quite computationally expensive if implemented entirely in VHDL.

Level-2 VHDL code of the DCO oscillator is listed in Sec. D.1 in Appendix D. It uses the digital tuning input ports of `integer` type. The conversion from `std_logic_vector` to `integer` is performed at the higher level in the synthesizable connectivity-only DCO wrapper.

[1]Period deviation is of opposite sign to the frequency deviation

The $1/\omega^3$ region in Fig. 1.4 (page 12) of the upconverted flicker noise has not been modeled for this testchip.

7.5 Modeling of Metastability in Flip-Flops

Conventional synchronous digital design conveniently deals with metastability by specifying and meeting setup and hold time of the sequential components, such as flip-flops and latches. A major inconvenience of the proposed ADPLL architecture is a necessity to deal with metastability as a *normal* occurrence expected in the course of device operation. This architecture, like any other system with mutually-asynchronous clocks, requires a fair amount of attention to avoid *synchronization failures*, which are said to occur if the system attempts to use a signal while it is in a metastable state. A common method to deal with synchronization failures is to sufficiently increase *mean time between failures* (MTBF) by cascading synchronizers such that occasional errors are no longer considered a problem. However, unlike in conventional systems incorporating synchronizers in which the sufficient solution is to *resolve* metastability, this architecture further requires the metastability to be *stochastically avoided*.

The following list summarizes two ADPLL areas requiring special attention to metastability in the design, modeling and simulation phases.

1. Sampling of the delayed replica of the DCO clock by the frequency reference in the time-to-digital converter (TDC); requires metastability resolution; described in Sec. 4.8

2. The frequency reference retiming by the DCO clock in the clock retiming block

(RETCK); requires metastability avoidance; described in Sec. 4.10.3.

Attempts have been made to model the metastable behavior of flip-flops and latches in a hardware description language, such as VHDL. In [67] an exponential model of metastable behavior is described and implemented for a fundamental element of a set/reset latch. Unfortunately, the model is quite complex thus degrading the simulation performance. In this research, we have chosen a much simpler, but also effective method to model metastability.

The metastability is modeled in the critical flip-flops by continuously inspecting timing relationship between the data input (D) and clock (CLK) pins and producing an *unknown* output 'X' on the data output (Q) pin if the D-to-CLK skew falls within the forbidden metastable window. Referring to Fig. 4.14 (page 133), the metastable window is defined as an x-axis region (D-to-CLK timing skew) such that the CLK-to-Q delay on y-axis is longer by a certain amount than the nominal CLK-to-Q delay. For example, if the nominal CLK-to-Q delay is 100 ps when the D-to-CLK timing is far from critical, then the metastability window would be 20 ps if one can tolerate CLK-to-Q delay increase by 90 ps. If one can tolerate a higher CLK-to-Q increase of 170 ps, then the metastable window would drop by half to 10 ps. A question could be asked of how far this window can extend. The limitation lies in the fact that for a tight D-to-CLK skew, the noise or other statistical uncertainty, such as jitter, could arbitrarily resolve the output in a way as to miss the input data. Therefore, for the conventional definition of the setup time, not only the output has to be free from any metastable condition, but also the input data has to be correctly captured. For this reason, the setup and hold times are conservatively defined in standard-cell libraries for the output delay increase of 10 or 20% over nominal. The specific nature of the TDC vector capturing does not require this restrictive constraint. Here, any output level resolution is

satisfactory for the proper operation, as long as it is not metastable at the time of capture and, consequently, the metastable window could be made arbitrarily small. In fact, in the implemented design this timing window is made narrower than 1 fs.

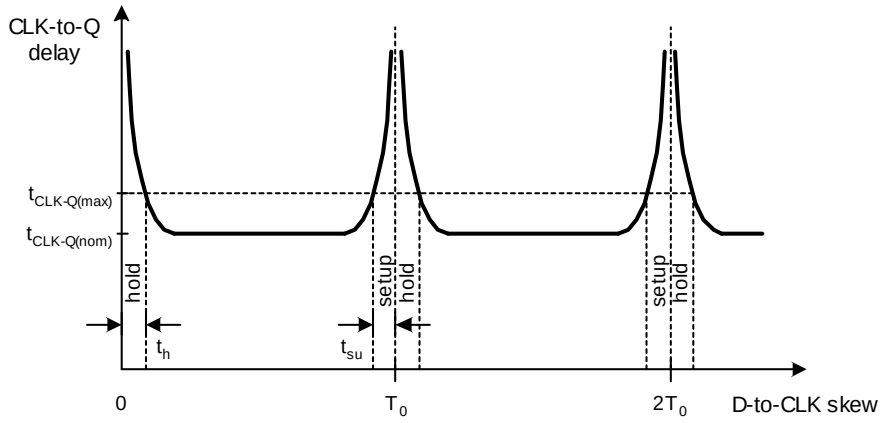

Figure 7.4. Timing diagram of a flip-flop

The timing diagram of a flip-flop is redrawn in Fig. 7.4 to further expound on the idea. The x-axis denotes the periodic D-to-CLK relationship with the repetition period equal to the CLK clock period of T_0. The y-axis denotes the CLK-to-Q delay, which nominally equals $t_{CLK-Q(nom)}$, but exponentially increases as data transitions closer to the capturing clock. Here, for simplicity, it is shown that the exact metastable point, where the flip-flop delay becomes very large, corresponds to the perfect alignment of the clock and data. In practical circuits, this need not be the case and, as Fig. 4.14 (page 133) of the tactical "Bora" flip-flop reveals, the metastable point is skewed by 47 ps with the data lagging the clock. This particular condition is subject to the specific circuit implementation. Just to demonstrate this point, the standard high-performance flip-flop in Fig. 4.13 shows that the metastable window lies on the opposite side, where the data leads the clock. This plot actually reveals two metastable windows, separated by about 65 ps, due to the asymme-

try between rise and fall transitions of the conventional master-slave fully-static CMOS topology. This single aspect of the traditional flip-flop makes it unusable for the TDC implementation which requires resolution in the range of 20-40 ps while maintaining a good linearity. Consequently, a symmetric sense-amplifier-based flip-flop appears to be a better choice.

The setup and hold violation windows (t_{su} and t_h, respectively) are centered around the clock and are defined for the D-to-CLK conditions, such that $t_{CLK-Q} > t_{CLK-Q(max)}$, where $t_{CLK-Q(max)}$ is a maximum allowed flip-flop output delay.

The timing periodicity shown in Fig. 7.4 is a general case where the clock multiplicity factor, i.e., the distance in number of edges between the launching and capturing clocks, could be greater than one. Generally, any timing relationship is valid as long as the D-to-CLK skew does not fall into the forbidden regions.

In our system, an advantage is taken of the fact that VHDL already supports a 9-valued digital bit type `std_logic`, which is an IEEE standard, and one of its levels is 'X', defined as "forcing unknown". Referring to Fig. 4.11 (page 126), the TDC flip-flops are modeled such that an 'X' is generated on a Q output to indicate metastable region of its D-to-CLK timing. The 'X' could then be detected in the pseudo-thermometer-code edge detector and replaced with a randomly picked '0' or '1'. It is the nature of this circuit that the metastable condition will be resolved within one full FREF clock cycle. However, due to noise, the resolution outcome is not known at the time of sampling. Therefore, a statistically probable binary result appears to be a good modeling choice for this phenomenon. For example, if the Q vector is "00111X0000...", then there is a 50% chance of resolving it to either "0011100000..." or "0011110000" with the respective decoded TDC_RISE outcome of 5

or 6. The measurement error is thus contained to a single LSB.

Metastability modeling of the clock retiming circuit of Fig. 4.16 (page 139) serves another purpose. These registers also generate an 'X' on the Q output whenever D-to-CLK timing relationship is in the forbidden region. However, no resolution attempt is carried out. This is in accordance with the intended operation of the clock retiming circuit that should avoid the metastable candidate altogether. At the bottom of Fig. 4.16 are shown cases of two potential metastable events and how the circuit keeps away from them. Of course, the mux select signal that originates in TDC needs to be resolved in the same manner as with the pseudo-thermometer-code detector.

Listing of the VHDL code for the tactical flip-flop is presented in Sec. D.3 in Appendix D. The TDC output decoder capable of handling metastable inputs appears in Sec. D.4.

7.6 Digital Blocks

The digital blocks are modeled at various abstraction levels, as shown in Table 7.1. As an example, level-1 representation of an GFSK transmit filter might describe the behavior with a simple direct-form finite-impulse response (FIR) filter equation using real numbers for input, output, coefficients and all the intermediate signals. Level-2 representation of the actual implementation with accumulated-coefficients might show the top-level structure of major building blocks, which are then modeled behaviorally on the bit level using integers. Second-order effects, such as LSB truncation and rounding as well as MSB clipping, are included. Level-3 representation has the same I/O behavior as level-2, but its RTL represen-

Table 7.1. VHDL modeling abstraction levels

Level-1	Mathematical equations and high-level behavioral description. Parameterized for easy analytic "what-if" questions. Optimized for simulation speed and flexibility. Fast enough to replace MATLAB for a bit error rate analysis. Includes important hardware-related non-idealities and second order effects. "_l1.vhd" file name suffix.
Level-2	Mathematical equations in integer domain. Implying the underlying architectural structure. 100% pin compatible with Level-3. Can be used for top-level connectivity verification. "_l2.vhd" file name suffix.
Level-3	Synthesizable register transfer level (RTL).
Level-4	Gate-level netlist produced by a synthesis tools. It could also be the actual gate-level netlist extracted from an auto-place and route tool (APR). Accompanied by the cell and wire timing information (Vital or SDF).

tation drives the synthesis of gate connectivity. Each higher level of modeling is expected to improve the simulation time by an order of magnitude.

The unified approach ensures interoperability of various abstraction levels within a single simulation environment. This allows simulating a system with a mixture of synthesized blocks and those that are still at the mathematical description levels.

7.7 Support of Digital Stream Processing

Digital communication channel evaluation usually requires processing of a long stream of digital data. Some measurements, such as bit error rate, require as much as tens of millions of data bits, therefore, efficient and fast algorithms to store, retrieve and access the data are necessary. Throughout this system, a time-causal operation and a linear complexity order is forced on the processing algorithms. As a result, the temporary storage requirement (RAM or disk swap space) is constant and the simulation time is only linearly proportional to the

overall digital stream length.

7.8 Simulation Results

7.8.1 Time-domain Simulations

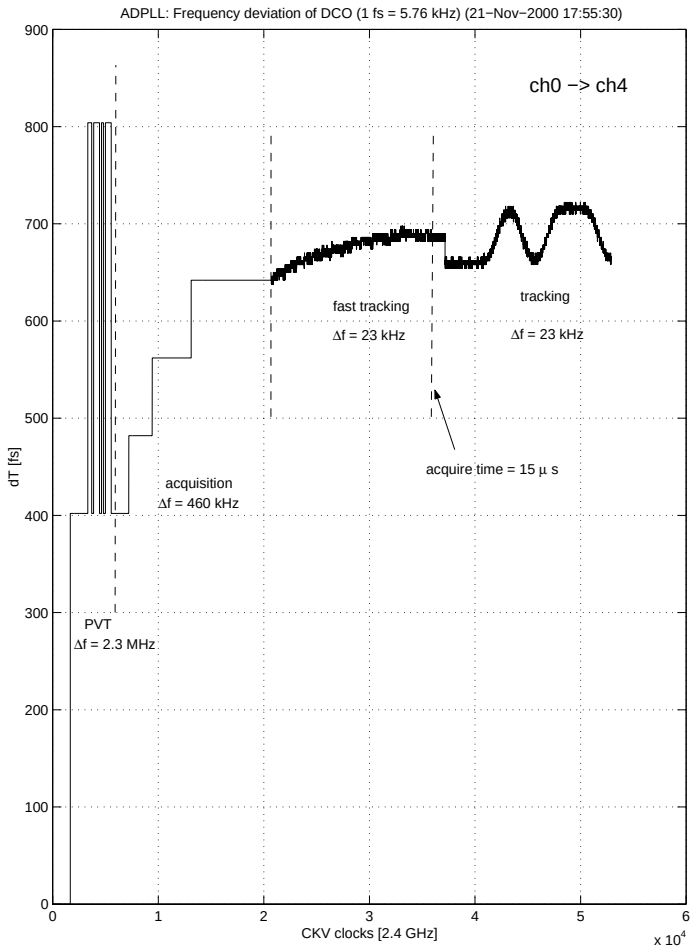

Figure 7.5. Simulation plot of the transmit modulation at @2.4 GHz RF output; x-axis: Δf in femptosecond time units (1 fs = 5.75 kHz); y-axis: time in 417 ps RF clock periods

Fig. 7.5 shows the composite trajectory plot of the instantaneous frequency deviation while illustrating operation of various PLL modes. The x-axis is the time evolution in CKV clock units (about 417 ps). The y-axis is the frequency deviation from an initial value of

2402 MHz (channel 0) expressed in femptosecond time units, where 1 fs corresponds to 5.77 kHz (see Table 2.1 on page 71).

The initial starting point is the center frequency set to channel zero. At power-up, a "cold start" to channel four at 4 MHz away is initiated. The ADPLL operates first in the PVT mode by enabling the *PVT oscillator controller* (OP). This controller makes very coarse (2.3 MHz) adjustments to the frequency. Next, the output of the PVT controller is put on hold and the *acquisition oscillator controller* (OA) is enabled. The acquisition controller quickly brings the frequency near the selected channel in 460 kHz steps. After acquisition of the selected channel is complete, the output of the OA controller is put on hold and the *integer tracking oscillator controller* OTI and *fractional tracking oscillator controller* OTF are enabled. The finest selection of the requested channel can only be accomplished using the tracking bank varactors with all the resolution enhancement techniques possible for this capacitor bank. The dynamic range of this mode has to cover the frequency resolution grid of the preceding acquisition mode. In the fast tracking mode, the frequency steps are the finest (less than 1 kHz) but the loop bandwidth could be as fast as in the acquisition mode. The tracking mode featuring the narrow loop bandwidth completes the channel acquisition and frequency locking.

The locking process takes altogether 15 μs with the reference frequency of 13 MHz (about 36 thousand CKV cycles or 196 FREF cycles). Upon reaching the steady state of the acquisition, the data GFSK modulation takes place.

The plot also shows how the PLL loop naturally deals with the frequency quantization effects of the oscillator in the PVT mode by dithering between the allowed upper and lower frequency levels from the desired frequency. This phenomenon was first described

in Sec. 2.4. As the simulations reveal, the closed-loop PLL system performs an inherent time-dithering operation if the DCO frequency granularity is finite. The mechanism is as follows: If the long-term average oscillating frequency lies between the two neighboring steps, the PLL will operate at lower frequency for some time until the accumulated phase error exceeds the resolution threshold. At that point the PLL loop will switch to the higher frequency until the accumulated negative phase error forces it to go back to the first lower frequency. This type of dithering of a single quantized level is a slow process, on the order of FREF clock cycles, and it is not observed in the acquisition mode. It is because the synthesizer does not stay there sufficiently long for the loop to accumulate enough phase to trigger the phase detector output change that would make a corrective action.

The above closed-loop low-speed dithering mechanism should be distinguished from the open-loop $\Sigma\Delta$ high-speed dithering in the tracking mode.

7.8.2 "Frequency-deviation" Simulations

Fig. 7.6 shows a time-domain plot of an instantaneous frequency deviation at the transmitter RF output when the $1/f^2$ noise (wander) of the oscillator is turned off. However, the DCO still contains the electronic noise floor (jitter) of -150 dBc. Since the VHDL simulator is foreign to the frequency concept, the y-axis represents the related period deviation in units of a femptosecond. As per Table 2.1 (page 71), 1 fs of the DCO period deviation is equivalent to 5.77 kHz of the DCO frequency deviation at the beginning of the BLUETOOTH band. Full symbol deviation of 160 kHz translates to about 28 fs. It might seem as a surprise that signal content is completely sunk within the noise whose peaks are order-of-magnitude larger than the signal peaks. Fortunately, performing a simple low-pass filtering through

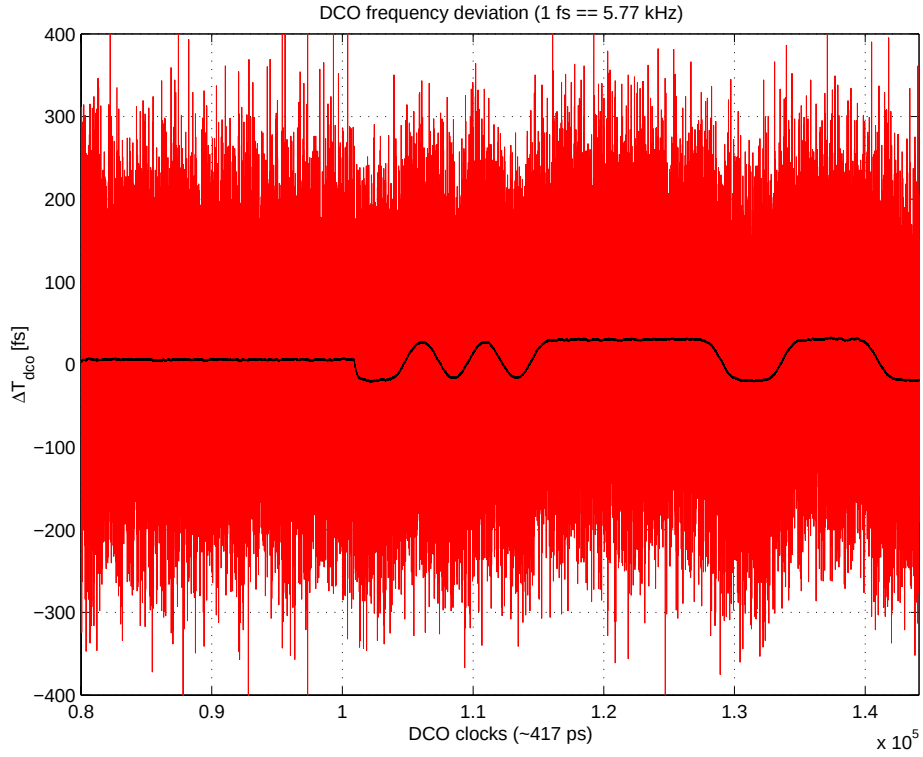

Figure 7.6. Instantaneous period deviation of CKV for $\alpha = 1/2^8$ with only the phase noise floor of -150 dBc; black thick line is a "leaky" integration

a "leaky" integration reveals a very clean modulating signal. The reason is simple: the white electronic noise has most of its energy contained in higher frequencies. Due to the sampling nature of the discrete-time oscillator, its noise extends from dc to one-half of the RF frequency of 2.4 GHz. Since 99.9% of the signal energy is contained in the 1 MHz band, noise portion that cannot be distinguished from the signal by virtue of falling into the signal band is

$$\frac{1MHz}{2400MHz/2} = 0.00083 = 0.083\% \tag{7.3}$$

The out-of-band components are easily filtered out.

Fig. 7.7 shows a similar time-domain plot but with the $1/f^2$ noise now turned on. The maximum frequency deviation peaks due to the noise are about the same, but the filtered

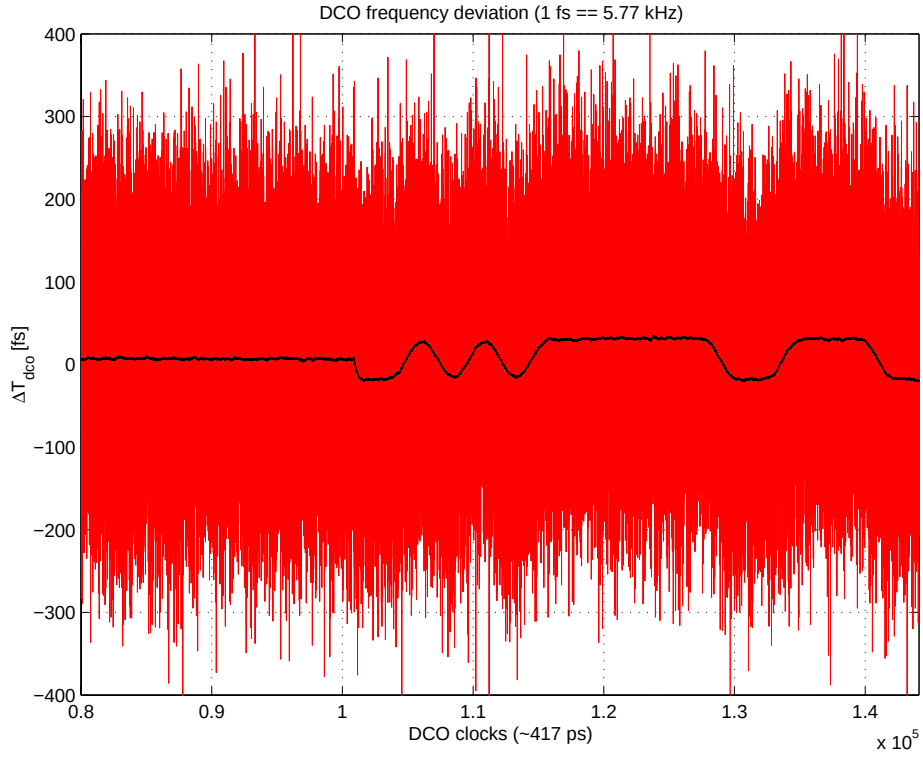

Figure 7.7. Instantaneous period deviation of CKV for $\alpha = 1/2^8$ with the phase noise floor of -150 dBc and $1/f^2$ noise of -105 dBc at 500 kHz; black thick line is a "leaky" integration

component shows some distortion. In this case, the $1/f^2$ component contains large energy at lower frequencies which cannot be separated through a linear filtering.

The DCO jitter source is turned off in Fig. 7.8, leaving only the wander component, which is quite small in amplitude term, but distorts the signal mainly through lower frequency components.

7.8.3 Phase-domain Simulations of the Transmitter

Fig. 7.9 shows a power spectral density of the GFSK digital filter output. The filter coefficients were designed for the nominal BLUETOOTH bandwidth–symbol period product of $BT = 0.32$. The filter output is in *frequency control word* (FCW) format (defined in

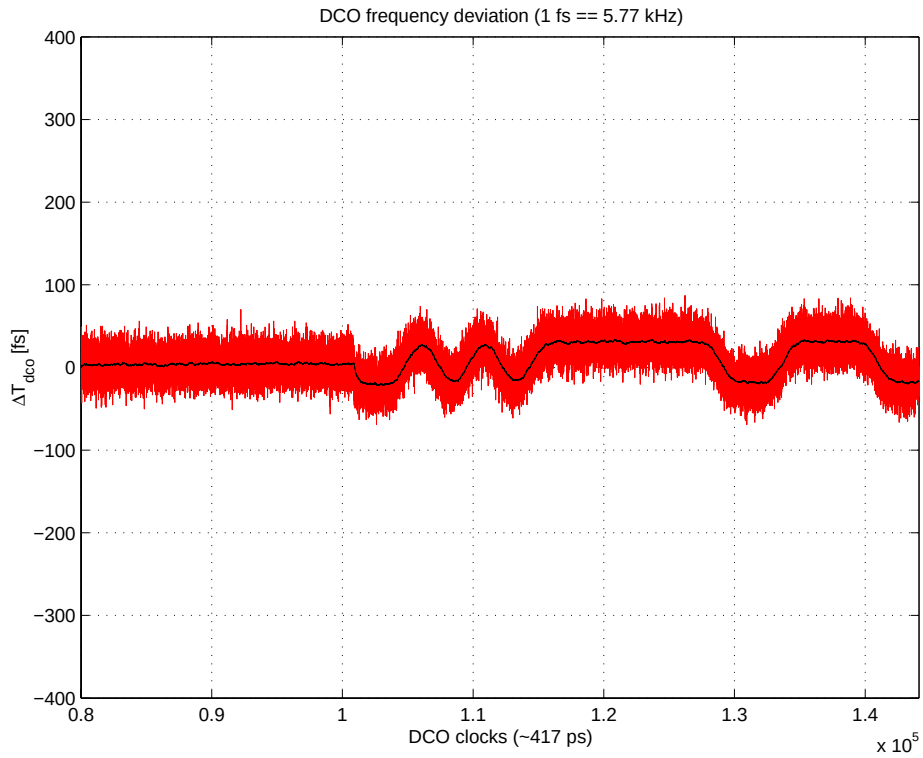

Figure 7.8. Instantaneous period deviation of CKV for $\alpha = 1/2^8$ with only the $1/f^2$ noise of -105 dBc at 500 kHz; black thick line is a "leaky" integration

Eq. 4.9 on page 109), which is interpreted as a carrier frequency deviation. A nominal modulation index of $h = 0.5$ is used for the FCW construction.

Fig. 7.10 shows a power spectral density of a GFSK-modulated RF carrier. Its center lies at the carrier frequency.

7.8.4 Synthesizer Phase Noise Simulations

Fig. 7.11 shows the synthesizer phase noise spectrum for the proportional loop gain setting of $\alpha = 2^{-8}$. The reference frequency is 13 MHz, so the PLL loop bandwidth is calculated as

$$13MHz \cdot \frac{2^{-8}}{2\pi} = 8086Hz \tag{7.4}$$

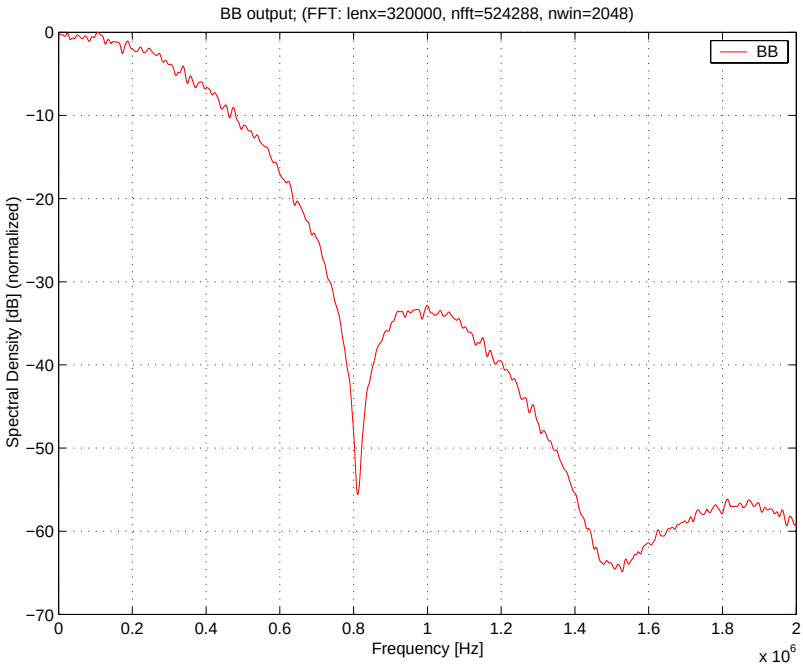

Figure 7.9. Power spectrum of the GFSK filter output

The FREF phase noise is set to -130 dBc/Hz, which roughly corresponds to an inexpensive, readily available crystal oscillator.

This plot shows two distinct regions of the upconverted thermal noise of -20 dB/dec slope, which was first introduced in Fig. 1.4 (page 12). The first region contains noise frequency components within the loop bandwidth that are corrected by the PLL loop. Since the loop is of type-I with slope of 20 dB/dec (see Eq. 4.43 on page 156), the noise characteristic becomes flat. In the second region lie frequency components outside of the loop that could not be corrected and which exhibit the original -20 dB/dec slope. The electronic noise floor of -150 dBc/Hz at high-frequencies confirms Eq. 7.1 (page 221). The phase noise readout of -105 dBc/Hz at 500 kHz offset confirms Eq. 7.2 (page 222).

Fig. 7.12 shows the simulated phase noise for $\alpha = 2^{-6}$. To be noted is the wider and lower flat region resulting from wider PLL loop bandwidth.

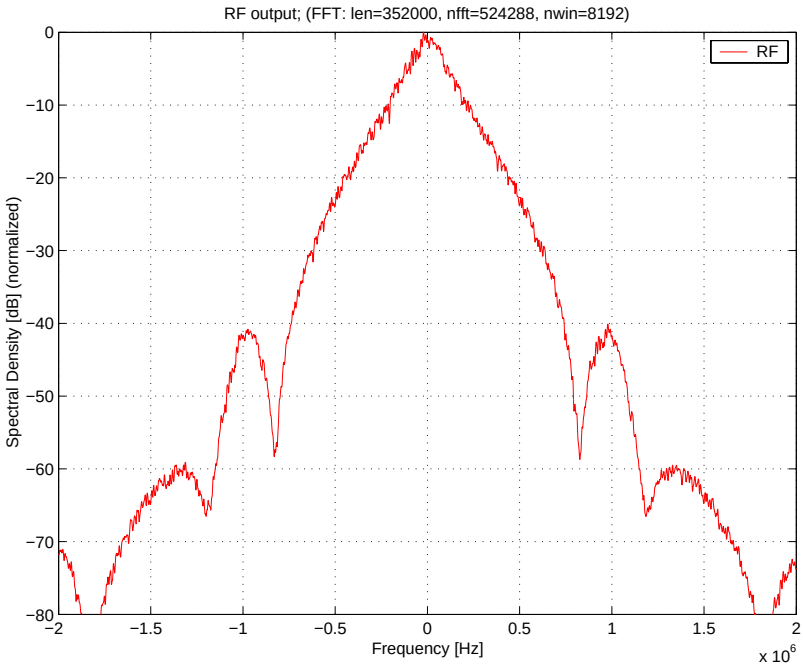

Figure 7.10. Output power spectrum of GFSK-modulated RF carrier; center is at the carrier frequency

Fig. 7.13 shows the timing deviation TDEV (random walk from the ideal timing instances) evolution for $\alpha = 2^{-8}$ and $\alpha = 2^{-6}$, respectively. As demonstrated above in the frequency-domain plots, the wider PLL loop bandwidth removes more lower-frequency DCO phase noise components.

Fig. 7.14 shows the phase noise spectrum when the FREF noise is turned off.

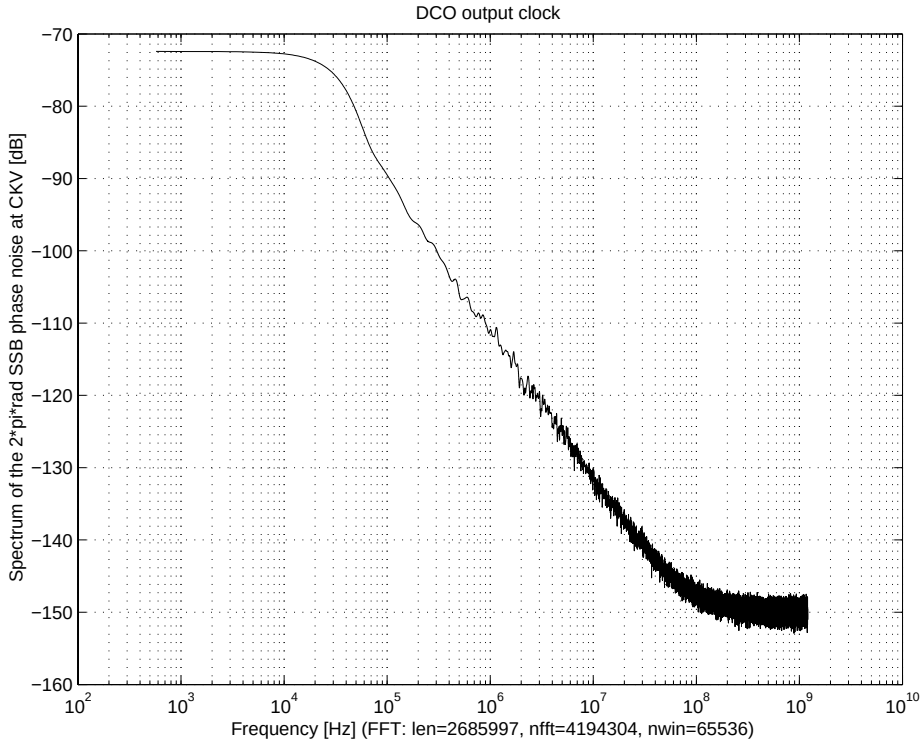

Figure 7.11. Simulated spectrum of the CKV clock for $\alpha = 1/2^8$

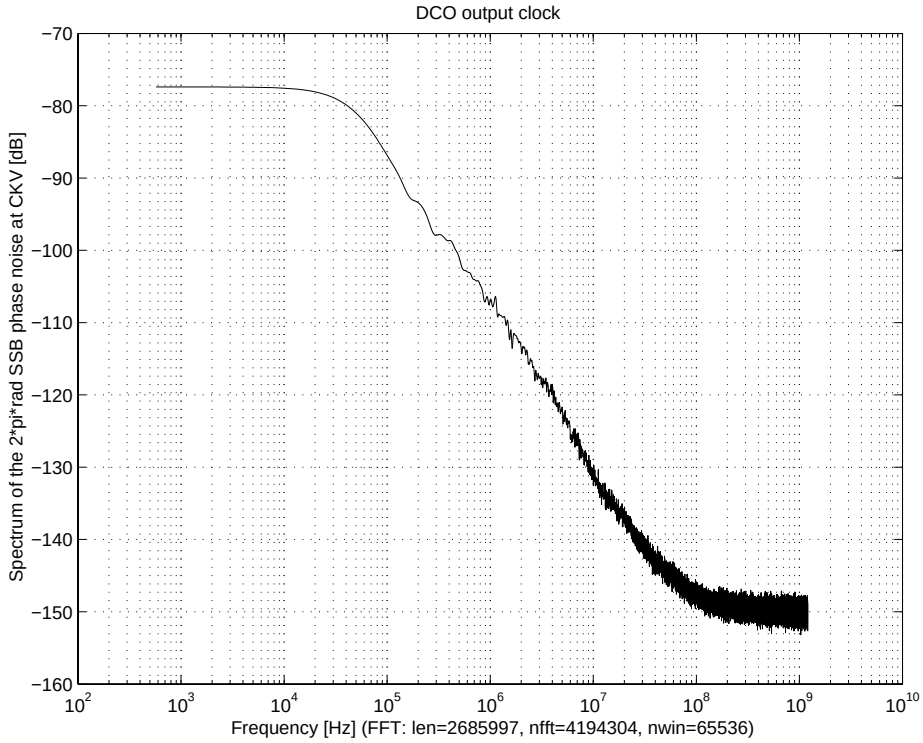

Figure 7.12. Simulated spectrum of the CKV clock for $\alpha = 1/2^6$

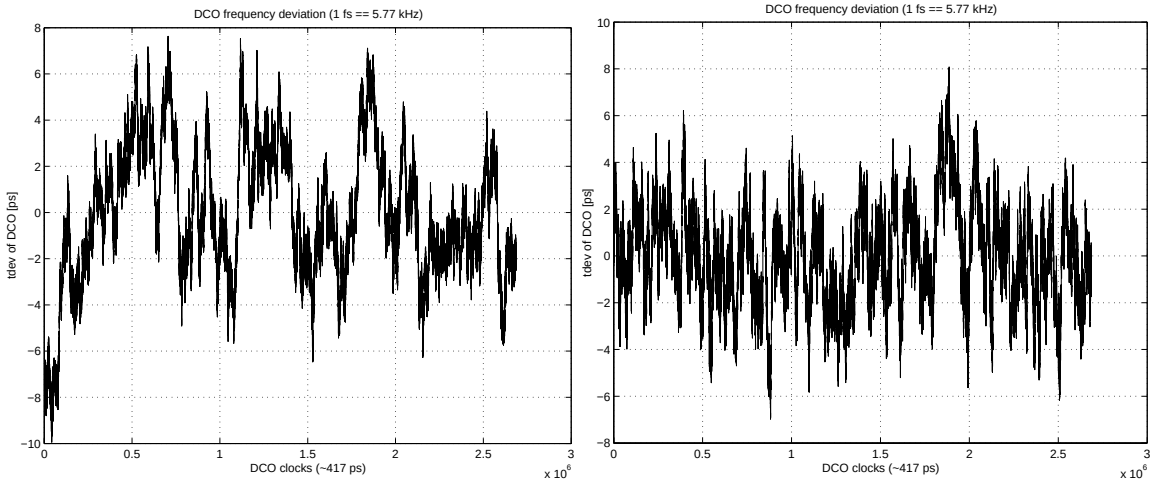

Figure 7.13. Simulation plot of TDEV for $\alpha = 1/2^8$ (left) and $\alpha = 1/2^6$ (right)

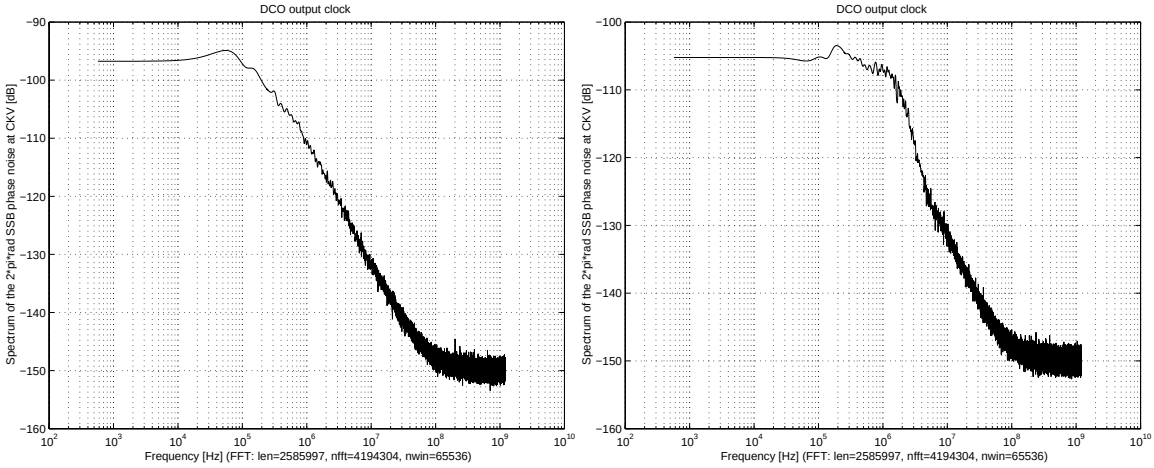

Figure 7.14. Simulated spectrum of the CKV clock for $\alpha = 1/2^8$ and $\alpha = 1/2^6$ (right) when FREF noise is turned off

CHAPTER 8

EXPERIMENTAL RESULTS

8.1 Overview

Following the design and fabrication of the experimental prototype, the BLUETOOTH transmitter IC chip based on the proposed all-digital RF frequency synthesizer architecture has been characterized for its performance. The key performance measures for the transmitter are demodulated eye diagram and spectrum. The key performance measures for the unmodulated synthesizer are its phase noise and spurious tone output. The synthesizer without the frequency modulation capability could also be used as a *local oscillator* (LO) to perform frequency translation in a receiver path.

8.2 Measurement Equipment

The RF output and the frequency reference input are connected using semi-precision *subminiature "A"* (SMA) connectors with the characteristic impedance of 50 Ω. The reference input is provided using an HP8662A synthesizer signal generator which has phase noise of about -140 dBc/Hz at a 20-kHz offset. The RF output port is connected to an HP8563E spectrum analyzer. The phase noise of the closed-loop PLL are measured using an HP85671A phase noise measurement utility. The eye-diagram measurement and the modulated spectrum were provided using a Rohde&Schwarz (R&S) FSIQ-7 signal ana-

lyzer, which downconverts the RF signal.

The measurement system is set up inside a Faraday cage which blocks virtually all RF signals that are present in the ambient environment.

8.3 GFSK Transmitter Performance

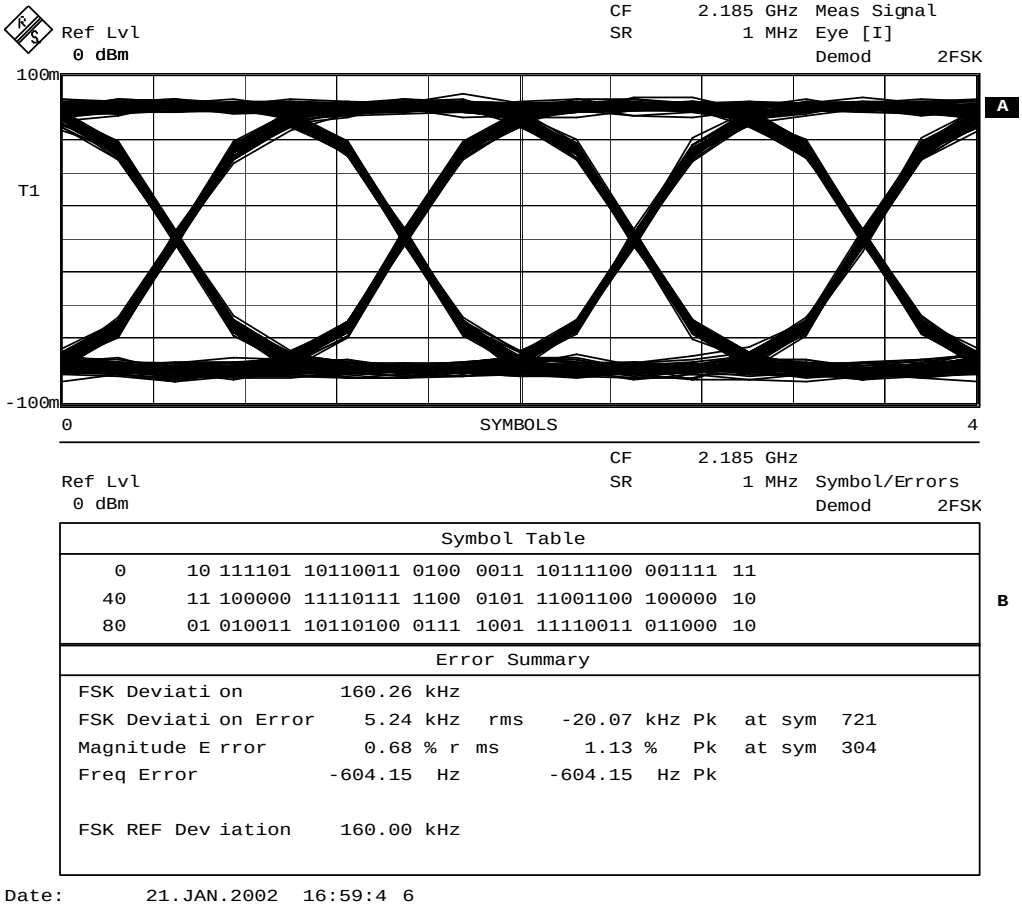

Figure 8.1. Eye diagram of a PN9 pseudo-random data at 2185 MHz, BW=4 kHz and room temperature with the measured statistics

Eye-diagram of the pseudo-random modulated data is shown in Fig. 8.1. It was taken with the R&S signal analyzer, which downconverts, FM demodulates, and then plots it and uses it to calculate various statistics. The x-axis is the time evolution in 1 μs symbols.

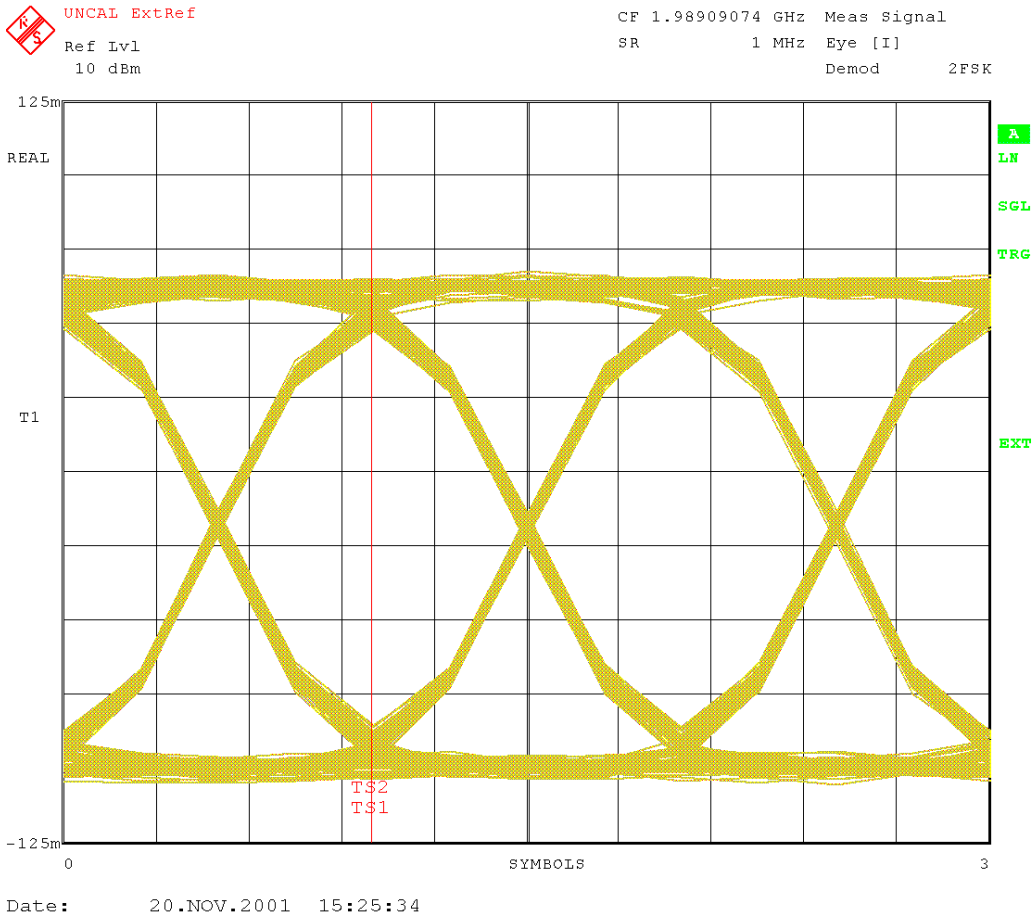

Figure 8.2. Eye diagram of a pseudo-random data. Measured peak zero crossing error is ±2.6% or 4.8 deg; RMS phase error is 2.06 deg.

Time-base of the instrument is externally synchronized to the symbol generator in order to avoid so-called cycle slipping. The peak-to-peak frequency deviation was measured to be 320.52 kHz, which is very close to the theoretical value of 320 kHz, calculated as the modulation index of 0.32 times the 1 Mbps data rate frequency. It also demonstrates a very wide eye opening ratio of 86-87% (the BLUETOOTH spec is ≥80%), which is desired for error-free symbol detection and very narrow zero crossing, which is desired for symbol timing and synchronization. Another eye-diagram of a different chip, taken at a different setting, is shown in Fig. 8.2. The R&S-calculated peak zero crossing error is only 4.8 deg

and the RMS phase error is merely 2.06 deg.

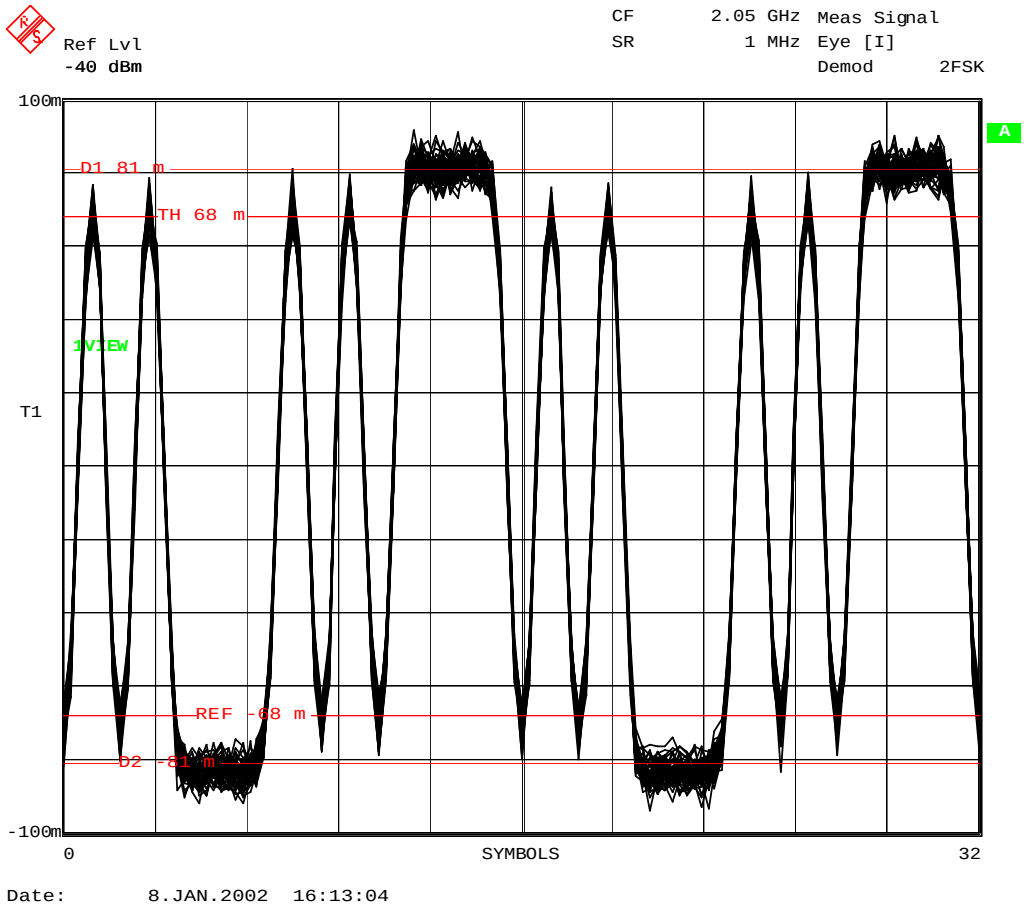

Figure 8.3. Demodulated diagram of the 111101010000 repetitive pattern. The ratio of $f_{2,avg}$ and $f_{1,avg}$ = 84% $\geq$ 80%.

Fig. 8.3 shows a similar plot but with a deterministic bit pattern of “111101010000”, instead of a pseudo-random sequence. It demonstrates higher frequency deviation for regions with little ISI (“1111” and “0000”) and less frequency deviation for regions with most ISI (“0101”). The ratio of average frequency deviations in those regions is part of the Bluetooth qualification tests and should be greater than 80%. The diagram shows that it is easily met with the 84% value.

The PSD of the pseudo-random data is shown in Fig. 8.4. A similar measurement

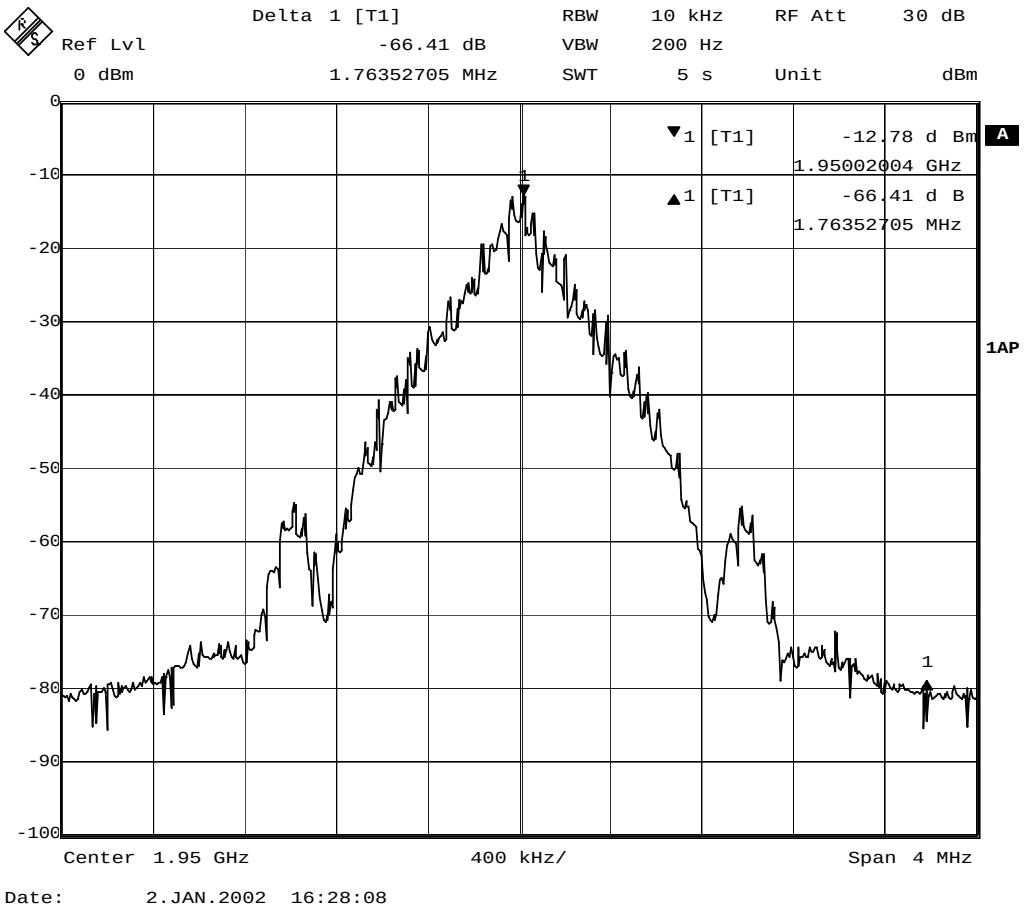

Figure 8.4. Measured output power spectrum of GFSK modulated pseudo-random data with a 1950.0 MHz carrier frequency

was taken with longer averaging (hence smoother curve) and is shown in Fig. 8.5. An instrumentation-quality internal BLUETOOTH reference source of the Rohde&Schwarz signal analyzer was superimposed, which turned out to have identical shape, except at high $\geq$1.5 MHz frequency offsets, where it was actually worse! It indicates that the far-out phase noise of the demonstrated oscillator is better than that used in R&S. The BLUETOOTH specification of power level $\geq$20 dB below the peak at $\pm$500 kHz offsets is easily met.

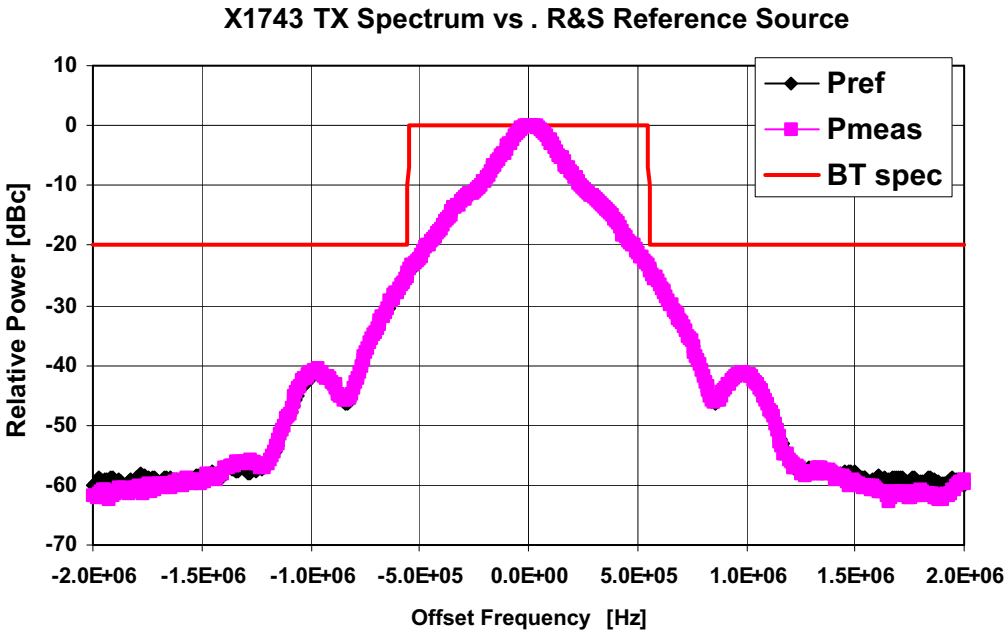

Figure 8.5. Measured GFSK spectrum superimposed on the Rohde&Schwarz FSIQ-7 internal source reference and the BLUETOOTH spec

8.4 Synthesizer Performance

Synthesizer performance pertains to the GFSK transmitter operating in its unmodulated state such that only a carrier is present. In this mode, the transmitter noise performance can be evaluated to ensure it is low enough to maintain an adequate SNR in-band during modulation and prevent any interference out-of-band. This mode is also active when the channel frequency is changed (either user initiated or during frequency hopping), since transmission of data is disabled. The unmodulated state would also be used when the frequency synthesizer acts as a receiver LO.

The closed-loop oscillator output voltage power spectral density $S_x(f)$, as defined in Sec. 1.2.1, is shown in Fig. 8.6 for the wide frequency span of 1 MHz and in Fig. 8.7 for the narrower frequency span of 100 kHz. From Fig. 8.7, the suppression of the close-in phase

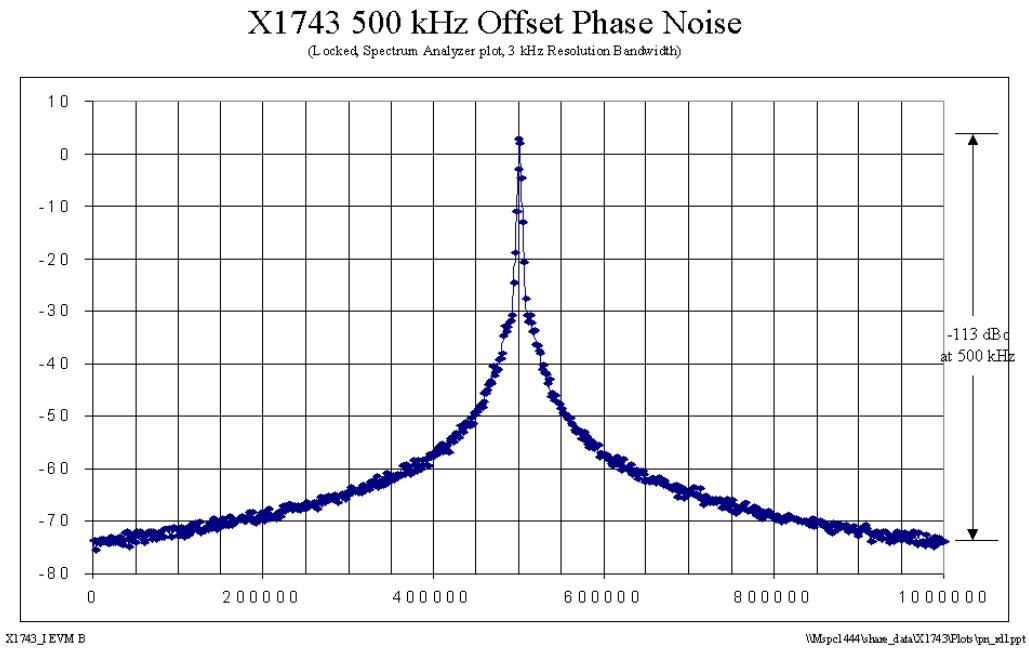

Figure 8.6. Measured synthesizer output spectrum (wide), resolution bandwidth RBW=3 kHz; center at 500 kHz offset mark

noise up to about the 10 kHz closed-loop bandwidth is readily visible. The actual phase noise can be estimated, provided the small angle criteria is not violated, my measuring the relative noise level with respect to the carrier and compensating for the finite resolution bandwidth (RBW) of the spectrum analyzer. In this case, the close-in noise level is at -35 dBc and the RBW is 1 kHz so the estimated phase noise is

$$\mathcal{L}\{\Delta f\} = -35dBc - 10 \cdot log(1kHz) = -65dBc/Hz \tag{8.1}$$

The synthesizer output spectrum is shown in Fig. 8.8 for the narrow bandwidth of 1 kHz and in Fig. 8.9 for the wide bandwidth of 8 kHz. The reference frequency is 13 MHz and the output frequency is 13x153=1989 MHz. The RF power level is 4 dBm (2.5 mW). The data file of the phase noise measurement utility was post-processed and displayed in

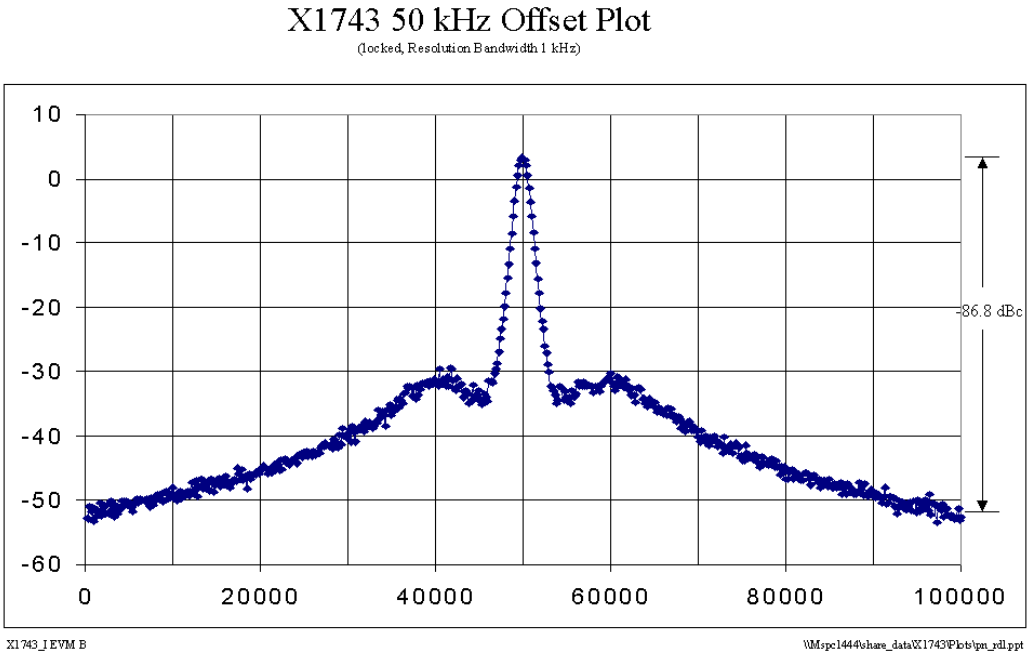

Figure 8.7. Measured synthesizer output spectrum (close-in); resolution bandwidth RBW=1 kHz; center at 50 kHz offset mark

MATLAB. Fig. 8.9 of the default PLL loop bandwidth of 8 kHz is to be compared with the VHDL simulation results in Fig. 7.11 (page 238) with the same loop setting. It reveals accurate modeling of the proposed methodology for the upconverted thermal noise in the loop-uncorrected and loop-corrected regions. The 10 dB/dec 1/f noise contribution revealed in the measure plot is not modeled.

Fig. 8.10 reveals the out-of-band spurious tone level at the RF output without and with a standard 2-pole antenna filter, which is normally used in an RF system as a duplexer. With filtering, the largest spur is at -62.5 dBm power level $f_{RF}/4$ away from the carrier, which easily meets the BLUETOOTH spec. The specific antenna filter is used for PCS applications and attenuates the signal by 1 dB.

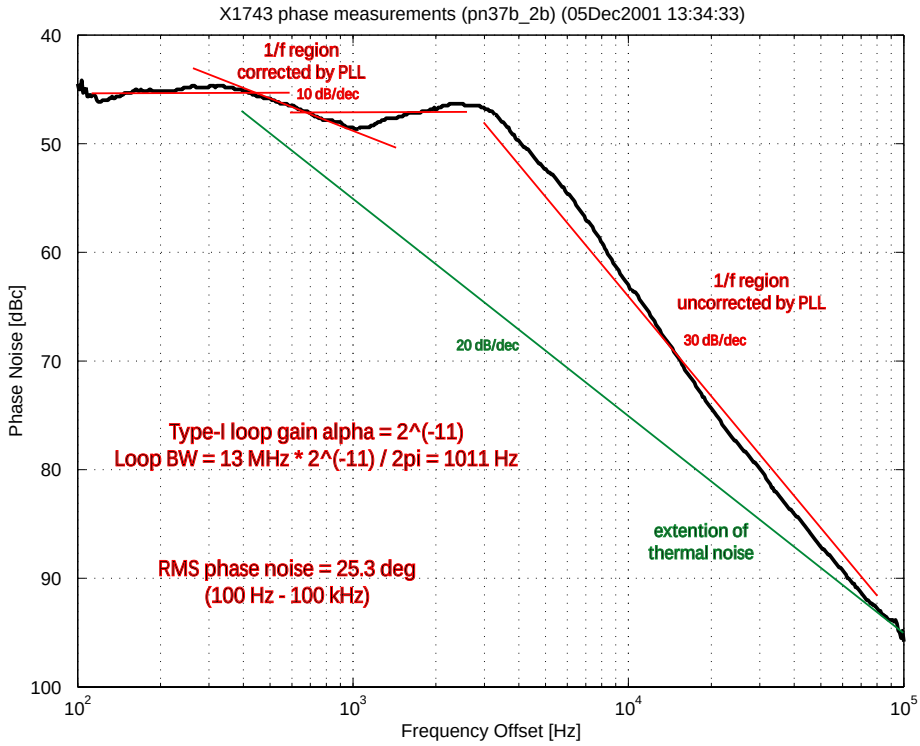

Figure 8.8. Measured synthesizer phase noise: narrow bandwidth

8.5 Synthesizer Switching Transients

Fig. 8.11 gives insight into the time-domain operation of the ADPLL. A simple DSP program was written to step the ADPLL through the three operational modes while observing the actual frequency deviation at the RF output. The observation was performed with the Rohde&Schwarz FSIQ-7 signal analyzer which demodulates the BLUETOOTH RF signal into baseband and plots the detected instantaneous frequency deviation on the y-axis. The y-axis scale is 300 kHz per grid. The x-axis on the plot is a time progression in units of 1 μs symbols, with the scale of about 20 μs per grid.

The initial frequency deviation was 2 MHz. The following is the predetermined timing sequence: (1) PVT mode was operational for 5 μs, (2) the acquisition mode was then

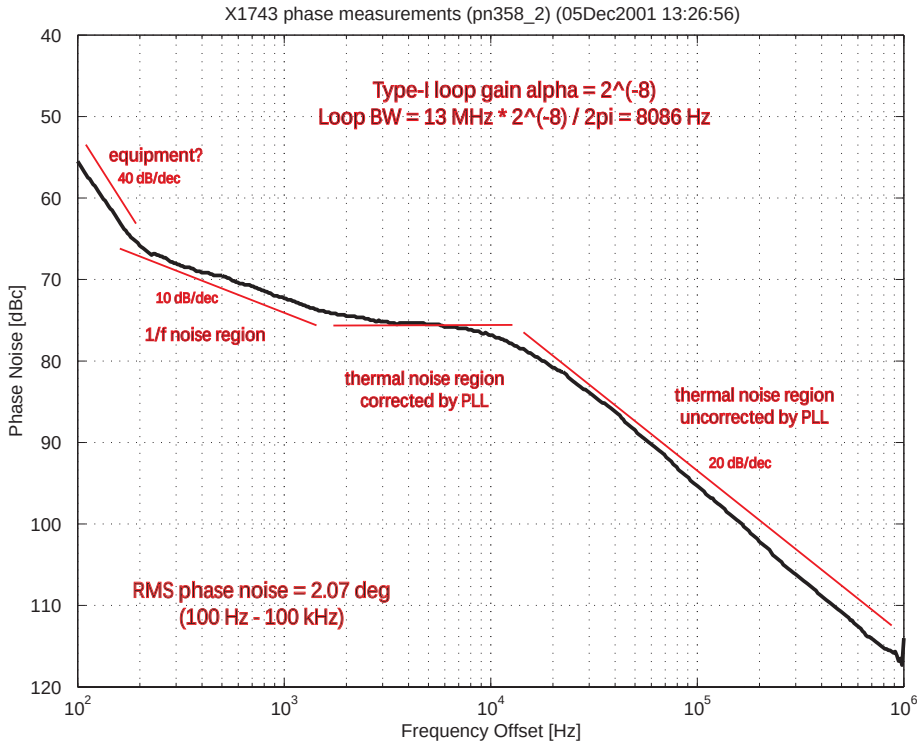

Figure 8.9. Measured synthesizer phase noise: wide bandwidth

operational for 25 μs, (3) the fast tracking mode was then operational for 18 μs, and (4) the regular tracking mode was finally turned on for the remaining time. The plot shows that in the above configuration, the settling time is about 65 μs for a 2 MHz jump. The DSP program was not optimized for performance and it is expected that a similar acquisition speed to that indicated by simulation in Fig. 7.5 (page 230) could be achieved.

8.6 DCO Tuning Characteristic

Fig. 8.12 shows an open-loop DCO frequency tuning characteristic in the PVT mode for three different temperature/voltage corners. The 8-bit tuning register is scanned from the smallest to the largest setting while the output frequency is measured. It shows a very large tuning range of about 400 MHz. Unfortunately, the center frequency lies at about 2.1 GHz

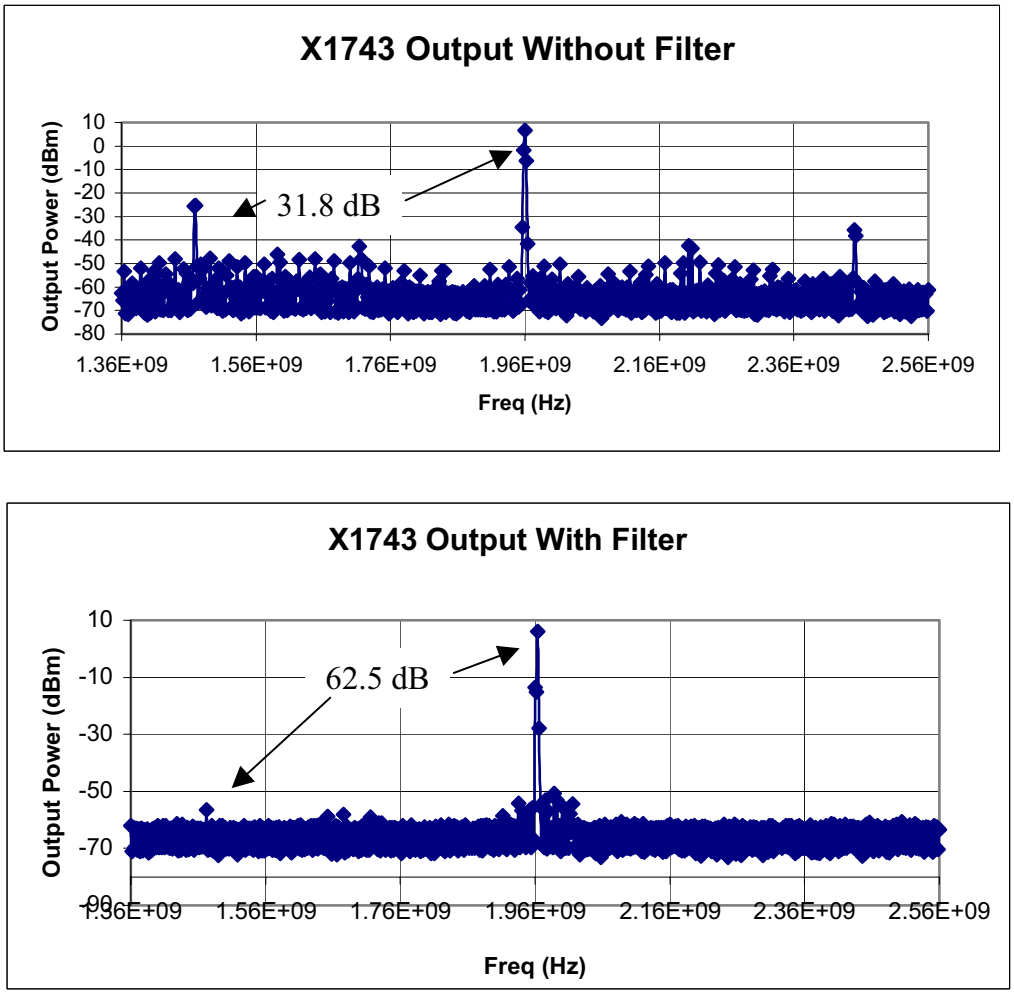

Figure 8.10. Measured spurious tone level at the RF output without (top) and with a standard 2-pole antenna filter (bottom)

instead of 2.45 GHz, which is the middle of the BLUETOOTH band. The problem was traced to an incorrect varactor model which was derived from an early testchip. It has been fixed in a subsequent testchip.

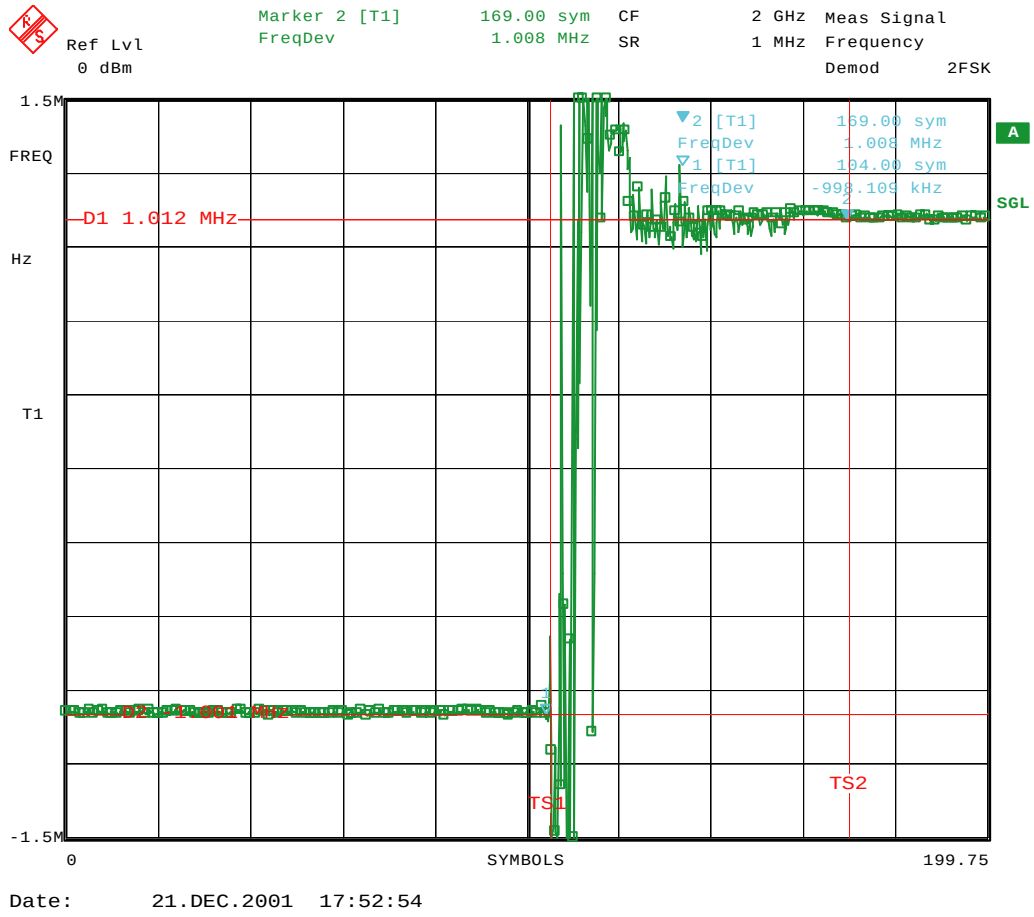

Figure 8.11. Observed settling time of the ADPLL

8.7 DSP-driven Modulation

As a demonstration of the tight operational integration between the DSP and the digital RF, a program was written to perform a software GSM modulation of the transmitter, instead of using a dedicated hardware for the BLUETOOTH modulation. The GSM data rate is 270.833 kbps. At the reference frequency of 13 MHz, the symbol oversampling ratio is exactly 48 (= 13 MHz / 270.833 kHz), meaning there are 48 FREF samples representing a symbol. A pseudo-random data sequence is being precalculated in software and it resides in the RAM. During the transmit modulation, the DSP fetches and adds to the FCW a new sample

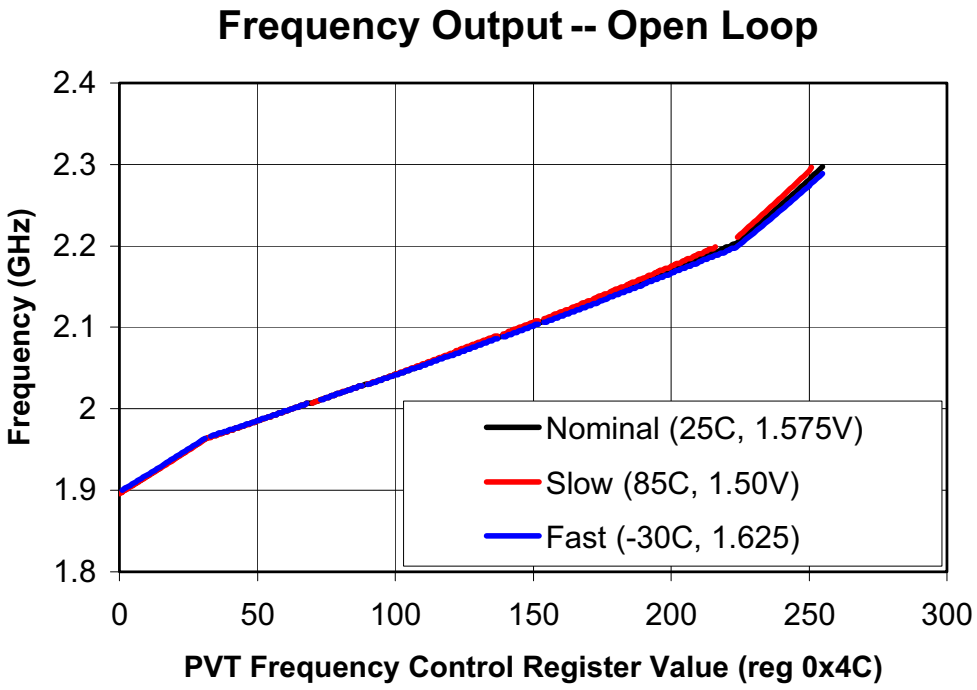

Figure 8.12. Open-loop frequency transfer of the oscillator

every 6 FREF clock cycles, so the actual oversampling ratio is 8, which is quite adequate. Unfortunately, because the data cannot be fed into the ADPLL in a two-point modulation manner, the higher-frequency spectrum components get filtered by the single-pole loop attenuation of 8 kHz 3 dB bandwidth. It would be quite feasible to accurately predistort the input data by preemphasizing the higher frequencies (the ADPLL loop transfer function is almost exact due to its digital nature) in order to extend the flat part of the transfer function, as reported in [33] [34]. However this task is beyond the scope of this research. The PSD of the GMSK-modulated data for GSM is shown in Fig. 8.13.

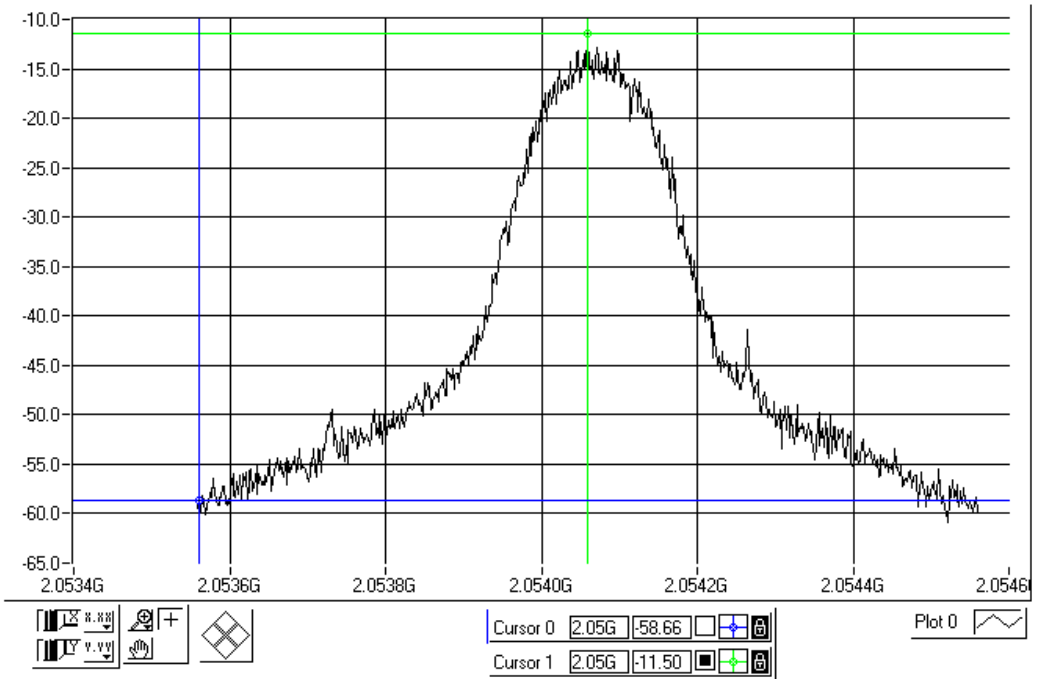

Figure 8.13. Measured output power spectrum of the DSP-driven GSM modulation, but with loop attenuation of higher-frequency components

8.8 Performance Summary

Table 8.1 summarizes the key measured transmitter performance parameters, some of which are based on the presented information.

The total current consumption is 49.5 mA, as shown in Table 8.2. It also includes the DSP under light-to-moderate loading.

Table 8.1. Measured key transmitter performance

Phase noise	$\leq$-114 dBc/Hz @500 kHz
Spurious noise	$\leq$-62.5 dBm (with antenna filter)
DCO frequency pushing	600 kHz/V
PA output power	4 mW @50 Ω load
RMS phase error	2.06 deg
ADPLL settling time	$\leq$50 μs

Table 8.2. Current consumption at 1.55 V supply

Circuit	Supply	Current
Low-speed digital + DSP	VDD	12.7 mA
High-speed digital	VDD-HS	12.8 mA
Oscillator	VDD-OSC	2.8 mA
RF w/o the oscillator	VDD-RF	21.2 mA
Total		49.5 mA

8.9 Summary

This chapter presents experimental results and shows suitability of the proposed RF frequency synthesizer in the targeted application of a BLUETOOTH transmitter. The eye diagram and the modulation spectrum indicate an excellent performance of the transmitter, while phase noise and spurious tones of the synthesizer would make it a good choice for the local oscillator of a receiver. The switching speed is extremely fast making it suitable for modern communication devices utilizing channel hopping. The power consumption is very low, thus making it ideal for battery operated mobile units.

CHAPTER 9

CONCLUSION AND FUTURE RESEARCH

9.1 Conclusion

The conventional approach for the RF synthesizer design is analog intensive. It requires large loop filter capacitors and does not suppress the poor phase noise performance of a monolithic oscillator. The phase detector is correlative in nature and generates reference spurs that could only be filtered out by making the loop filter narrow at the expense of slow transient responses. Moreover, the analog intensive architectures do not integrate well with the digital baseband making the goal of a single-chip radio difficult to achieve.

New frequency synthesizer architecture was presented in this dissertation that is suitable to be implemented in a digital deep-submicron CMOS process technology. In fact, the techniques employed take advantage of the technology's underlying strengths, while de-emphasizing its weaknesses. In order to demonstrate that this digital CMOS synthesizer is feasible for multi-GHz RF applications, a transmitter for BLUETOOTH short-range wireless communication devices was built. The resulting synthesizer is primarily digital in nature and it is integrated into a large digital baseband that includes a digital signal processor (DSP).

The proposed synthesizer is based on an *all-digital PLL* (ADPLL) architecture that uses digital techniques from the ground up. The only block that uses analog/RF design

techniques at its core is a *digitally-controlled oscillator* (DCO). However, it I/O is completely digital and it stops the analog nature from propagating up in the hierarchy level. This allows the system to be modeled in the discrete-time domain. A standard VHDL, which is a timing-driven simulation engine with real-valued signal support, is used as a complete system simulator. This is in contrast with conventional designs that use analog and RF techniques that are time consuming and require multiple design and layout iteration cycles. Moreover, the best choice of the design methodology now is a digital flow with its numerous advantages of time, cost and accuracy. Especially important in a mass production setting of a commercial product are the self-testing and self-calibration capabilities of the all-digital implementation.

There are only two phase noise sources in the proposed architecture: the DCO oscillator and the time-to-digital converter (TDC). While the former phase noise mechanism is substantially equivalent to the conventional designs, the latter is destined to become better with every process node providing better time-domain resolution. The rest of the synthesizer is digital in nature and, as such, is does not introduce any noise. This is in sharp contrast with conventional synthesizers where every major component (frequency divider, phase detector, charge pump, filter, etc.) is a noise source contributor.

This architecture appears to be the first published report on an all-digital phase-locked loop for wireless RF applications, which generally feature stringent requirements on phase noise purity and low level of spurious tone content. This is also a first report on RF circuits that are integrated with a DSP in a deep-submicron CMOS process. This is also a first report on using a DCO for a wireless RF application.

9.2 Future Research

The results of this work verified the feasibility of new digitally-intensive architecture for RF synthesis, but also open up opportunities for improved performance and future research areas.

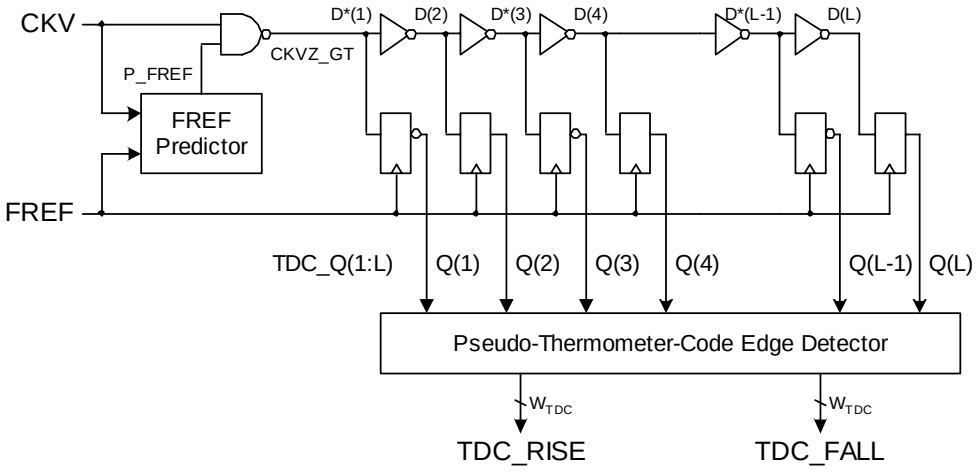

Figure 9.1. Time-to-digital Converter (TDC) with FREF prediction

Power consumption is a very crucial parameter for the battery-operated wireless terminals. Even though, the demonstrated transmitter is very competitive in this regard with other devices available on the market, dramatically increasing the battery life would open up new range of applications. Reducing the current consumption through an intelligent power management scheme suggests an area for future research. Fig. 9.1 shows an example of periodic gating of the oscillator clock inside the TDC (originally shown in Fig. 4.11 on page 126) by predicting where the next FREF edge might lie. The timing information lies in the edges of the FREF clock and thus operating the fractional phase detector in between its edges unnecessarily propagates the gate switching transitions. Statistics could be made about the past edges and hit/miss ratios in order to allow the prediction window to be

as narrow as practically possible.

The phase information of the frequency reference clock lies in both of its edges. In this architecture only the rising edge is utilized. Exploration of this possibility is an area of future research. This would allow to effective double the “sampling” rate of the PLL operation. Obviously, there is no assumption about the 50% duty cycle of the reference clock. However, the timing consistency between rising-to-rising and falling-to-falling edge separation could be exploited at the cost of increasing the computational and hardware complexity.

The transmit filter requires an integer ratio of the reference frequency to the symbol data rate. Most wireless systems are built on this assumption and specify that the crystal oscillator is a multiple of the data rate. However, relaxing this constraint by allowing the use of frequency reference from any crystal would make the architecture very portable. An interpolative transmit filter that would made the goal feasible is another area to investigate.

Another area for future research deals with increasing randomness of the $\Sigma\Delta$ digital modulator. Its chief purpose is to randomize the stream of the fractional tuning word. However, a small fractional value might produce low frequency oscillations. Randomizing the LSB bit of the first accumulator stage would effectively break any long periodic sequences.

APPENDIX A

SPURS DUE TO DCO SWITCHING

Let's assume a sinusoidal modulating signal,

$$f(t) = a \cos w_m t \tag{A.1}$$

For a frequency modulation (FM), the instantaneous frequency ω_i is the carrier frequency ω_c modified by the modulating signal times the FM constant k_f.

$$\omega_i(t) = \omega_c + k_f f(t) = \omega_c + a k_f \cos w_m t \tag{A.2}$$

Defining a new constant called the *peak frequency deviation*,

$$\Delta\omega = a k_f \tag{A.3}$$

we can rewrite Eq. A.2 as

$$\omega_i(t) = \omega_c + \Delta\omega \cos w_m t \tag{A.4}$$

The phase of this FM signal is

$$\theta(t) = \omega_c t + \frac{\Delta\omega}{\omega_m} \sin w_m t = \omega_c t + \beta \sin \omega_m t \tag{A.5}$$

where

$$\beta = \frac{\Delta\omega}{\omega_m} \tag{A.6}$$

is a dimensionless ratio of the peak frequency deviation to the modulating frequency.

The resulting signal is

$$\phi_{FM}(t) = A\cos(\omega_c t + \beta \sin \omega_m t) \tag{A.7}$$

Eq. A.7 could be rewritten after some trigonometric expansion

$$\phi_{FM}(t) = A\cos(\omega_c t)\cos(\beta \sin \omega_m t) - A\sin(\omega_c t)\sin(\beta \sin \omega_m t) \tag{A.8}$$

For small values of β we can make the following approximations

$$\cos(\beta \sin \omega_m t) \approx 1 \tag{A.9}$$

$$\sin(\beta \sin \omega_m t) \approx \beta \sin \omega_m t \tag{A.10}$$

Substituting these into Eq. A.8 we obtain an approximate solution for small β.

$$\phi_{NBFM}(t) = A\cos(\omega_c t) - \beta A \sin(\omega_m t)\sin(\omega_c t) \tag{A.11}$$

Expanding Eq. A.11 into phasor form, we have

$$\begin{aligned} \phi_{NBFM}(t) &= \Re\{Ae^{j(\omega_c t)}(1 + j\beta \sin \omega_m t\} & \text{(A.12)} \\ &= \Re\{Ae^{j(\omega_c t)}(1 + \frac{1}{2}\beta e^{j\omega_m t} - \frac{1}{2}\beta e^{-j\omega_m t})\} & \text{(A.13)} \end{aligned}$$

From Eq. A.12 it is clearly seen that the power spectral density of a narrow-band frequency modulated signal is the continuous-wave carrier frequency plus two side-bands ω_m away and $20\log(\frac{1}{2}\beta)$ dB below the carrier.

A.1 Spurs Due to DCO Modulation

Let's apply the above analysis of a narrowband FM modulation to our specific case of dithering the oscillating frequency of an LC-tank by switching a unit capacitor value through a digital control.

When the control signal is high, the capacitor value is C_{on} and the oscillating frequency is

$$f_{osc,h} = \frac{1}{2\pi\sqrt{LC_{on}}} \tag{A.14}$$

Similarly, when the control signal is low, the capacitor value is C_{off} and the oscillating frequency is

$$f_{osc,l} = \frac{1}{2\pi\sqrt{LC_{off}}} \tag{A.15}$$

The difference between the high and low oscillation frequencies is

$$\Delta f_{osc} = f_{osc,h} - f_{osc,l} \tag{A.16}$$

The peak frequency deviation is

$$\Delta\omega = 2\pi\frac{\Delta f_{osc}}{2} = 2\pi\frac{f_{osc,h} - f_{osc,l}}{2} \tag{A.17}$$

The modulating signal frequency is ω_m.

A.1.1 Example I

Let's consider an example of switching the oscillating frequency by Δf_{osc} = 23 kHz at the rate of 600 MHz. The narrowband FM analysis assumed a sinusoidal modulating signal $f(t)$ in Eq. A.1. However, the frequency modulation function of turning on and off a small unit-size capacitor is obviously a square wave. Let's consider first harmonic in the Fourier series decomposition whose amplitude is $4/\pi$ times the square wave. The modulating frequency is one half of the switching rate, i.e., $\omega_m/2\pi$ = 300 MHz.

$$\beta = \frac{4}{\pi} \cdot \frac{23\ kHz/2}{600\ MHz/2} = 4.88 \cdot 10^{-5}$$

This gives rise to spurs 600 MHz away on both sides from the oscillating frequency. Their power level is at

$$20\log(\frac{\beta}{2}) = -92.3\ dB$$

relative to the carrier.

The above -92.3 dBc ("c" in dBc stands for "relative to the carrier") spur level corresponds to the situation when the DCO is dithered continuously at the highest possible rate. Obviously, this is not a practical case. In fact, the $\Sigma\Delta$ dithering will randomize the spurious energy and blur it into the background.

A.1.2 Example II

Let's consider now the case of performing the same switching but at the 13 MHz of FREF frequency.

$$\beta = \frac{4}{\pi} \cdot \frac{23\ kHz/2}{13\ MHz/2} = 1.77 \cdot 10^{-3}$$

and the spur power level is much higher now at

$$20\log(\frac{\beta}{2}) = -61.1\ dB$$

APPENDIX B

OPERATION OF THE SYNCHRONOUS PHASE-DOMAIN ALL-DIGITAL PLL SYNTHESIZER

B.1 Basic definitions

Let's define the actual and desired clock period of the variable (VCO, DCO or a generally controllable oscillator) output CKV as T_V and T_V^0, respectively. Similarly, let's denote the actual and ideal clock period of the frequency reference FREF as T_R and T_R^0, respectively. Since T_V^0 and T_R^0 are ideal, they are known exactly. On the other hand, the physical entities T_V and T_R are not known exactly and could only be estimated. Let's further assume, in order to simplify the analysis, that the actual clock periods are constant or time-invariant, and that the oscillator runs appreciably faster than the available reference clock, $T_V \ll T_R$.

In practice, the reference clock is of excellent long-term stability in comparison with the CKV clock. This implies that the reference clock transitions are considered absolutely precise in the long term. However, the poor short-term timing stability of the reference oscillator could be quite poor.

The CKV and FREF clock transition times t_V and t_R, respectively, are governed by the following equations:

$$t_V = i \cdot T_V \tag{B.1}$$

$$t_R = k \cdot T_R + t_0 \tag{B.2}$$

where $i = 1, 2, \ldots$ and $k = 1, 2, \ldots$ are the CKV and FREF clock transition index numbers, respectively, and t_0 is some initial time offset between the two clocks, which is absorbed into the FREF clock. The ideal (or desired) CKV transition times are

$$t_V^0 = i \cdot T_V^0 \tag{B.3}$$

The actual frequency ratio $N = T_R/T_V$ and the ideal frequency ratio $N^0 = T_R/T_V^0$ are generally real numbers.

It is convenient in practice to normalize the transition times in terms of T_V units (instead of T_V^0) since it is easy to observe and operate on the actual CKV clock events. Let's define dimensionless variable and reference "phase"

$$\theta_V \equiv \frac{t_V}{T_V} \tag{B.4}$$

$$\theta_R \equiv \frac{t_R}{T_V} \tag{B.5}$$

The term θ_V is only defined at CKV transitions and indexed by i. Similarly, θ_R is only defined at FREF transitions and indexed by k. This results in

$$\theta_V(i) = i \tag{B.6}$$

$$\theta_R(k) = k \cdot \frac{T_R}{T_V} + \frac{t_0}{T_V} = k \cdot N + \theta_0 \tag{B.7}$$

The ideal normalized CKV transition times are

$$\theta_V^0 \equiv \frac{t_V^0}{T_V} \tag{B.8}$$

$$\theta_V^0 = i \cdot \frac{T_V^0}{T_V} = i \cdot \frac{N}{N^0} = i(1 + \nu) \tag{B.9}$$

where $\nu = 1 - f_v/f_v^o$ is the normalized frequency departure of the oscillator. If $\theta_V^0(i)$ were known, the normalized timing error $\phi_e(i)$ could ideally be defined as

$$\phi_e(i) \equiv \theta_V(i) - \theta_V^0(i) = -i \cdot \nu \tag{B.10}$$

Thus, for a positive frequency offset, $\nu > 0$, the oscillator transition times would occur progressively earlier, or with less delay, than expected, hence larger and larger negative timing error. Note that the *negative timing error* corresponds to a *positive phase error*, as conventionally defined.

B.2 Integer phase estimator

Unlike with a sinusoidal signal, the only timing information available in the digital clock lies with its transition edge times. In order to compare the timing of two clocks, the lower-frequency clock sets the comparison rate. Therefore, in our case, the lower frequency reference clock with edge index k will drive the timing comparison process with the high-frequency oscillator clock. $\phi_e(i)$ as defined in Eq. B.10 is not physical and could not be even measured for all values of i. It could only serve as an approximation guide for another, physically-realizable estimate (later defined) of the timing error.

$$\widehat{\phi}_e(k) \cong \phi_e(i) \tag{B.11}$$

B.2.1 Approximation bound of the timing interpolation error

In Eq. B.11, index i should be chosen such that the $t_V(i)$ timing instance lies the closest to $t_R(k)$. This sets the limit on the time interpolation error $\widehat{\phi}_e^0(k)$ as expressed by deviation

of the reference timing from the corresponding ideal oscillator timing to be one half of the oscillator clock period. In other words,

$$|\widehat{\phi}_e^0(k)| \equiv |\theta_V^0(i) - \theta_R^0(k)| \leq 1/2 \tag{B.12}$$

and the timing error uncertainty of Eq. B.11 will be small and bounded by $\nu/2$. Eq. B.12 represents the time interpolation error of $\widehat{\phi}_e(k)$ due to the use of a system with two asynchronous clocks. Note that $\widehat{\phi}_e^0(k)$ could not be known precisely since $\theta_V^0(i)$ is not physical.

It should be emphasized that, unlike the FREF clock transition times in Eq. B.2, the normalized clock transition times of Eq. B.7 contain the actual CKV frequency information.

Let's define the physically realizable estimate of the timing error as

$$\widehat{\phi}_e(k) \equiv \theta_V(i_{opt}) - \theta_R(k) \tag{B.13}$$

with index i_{opt} meeting the constraint defined in Eq. B.12.

Using a digital implementation, it is rather more convenient to use a one-sided delay estimator, rather a two-sided advance/delay estimator of Eq. B.13. For each value of index k, there are two index values i such that the CKV transition times are within a fractional T_V distance from the reference transitions.

$$\theta_V(i_l) \leq \theta_R(k) \leq \theta_V(i_h) \qquad i_h = i_l + 1 \tag{B.14}$$

In other words, the reference timing is "sandwiched" between the two CKV transitions and either only i_l or i_h should be found – the other could be easily obtained by simply incrementing or decrementing. The resulting timing error has the following bounds

$$\widehat{\phi}_e^l(k) \equiv \theta_V^0(i_l) - \theta_R^0(k) \in (-1, 0] \tag{B.15}$$

$$\widehat{\phi}_e^h(k) \equiv \theta_V^0(i_h) - \theta_R^0(k) \in [0, 1) \tag{B.16}$$

For time-causal clock edge operations, it is easier to find i_h, simply obtained as the next CKV edge after the FREF transition. This is also preferred from a low power standpoint to "sleep" until FREF activity before starting to search for the next edge of CKV.

B.3 Fractional error correction

The approximation of the timing error of Eq. B.10 could be further improved with the *real-valued* measurements of the actual time delay between $t_R(k)$ and $t_V(i_{opt})$.

B.4 Hardware implementation

The previously stated timing equations could be modeled in hardware in the following way:

$$R_V \equiv i \tag{B.17}$$

$$R_R \equiv k \cdot N^0 \tag{B.18}$$

$$R_E(k) \equiv R_V(i_h) - R_R(k) = i_h - k \cdot N^0 \tag{B.19}$$

Equation Eq. B.17 is the hardware-equivalent of Eq. B.6 and could be realized as an incrementer or counter of the CKV clock edges. Equation Eq. B.18 is the hardware-equivalent of Eq. B.7 and could be realized as an accumulator of the N^0 digital word on FREF clock edges.

APPENDIX C

GAUSSIAN PULSE-SHAPING FILTER

The Gaussian low-pass filter has a transfer function given by

$$H(f) = \exp(-\alpha^2 \cdot f^2) \tag{C.1}$$

The parameter α is related to B, the 3-dB bandwidth of the baseband Gaussian shaping filter. It is commonly expressed in terms of a normalized 3-dB bandwidth-symbol time product (BT_s).

$$\alpha = \frac{\sqrt{ln(2)}}{\sqrt{(2)}} \cdot \frac{T_s}{BT_s} \tag{C.2}$$

As α increases, the spectral occupancy of the Gaussian filter decreases and the impulse response spreads over adjacent symbols leading to increased ISI at the receiver. The impulse response of the Gaussian filter in the continuous-time domain is given by

$$h(t) = \frac{\sqrt{\pi}}{\alpha} \cdot \exp\left(-\frac{\pi}{\alpha} \cdot t\right)^2 \tag{C.3}$$

Let us now express the Gaussian filter in the discrete-time domain. Let $t_0 = \frac{T_s}{OSR}$ be an integer oversample of the symbol duration and $t = k \cdot t_0$, k being the sample index. Discrete-time impulse response becomes:

$$h(kt_0) = \frac{\sqrt{\pi}}{\alpha} \cdot \exp\left(-\frac{\pi}{\alpha} \cdot kt_0\right)^2 \tag{C.4}$$

Substituting Eq. C.2 and dropping explicit dependence on t_0 results in

$$h(k) = \underbrace{\frac{\sqrt{2\pi}}{\sqrt{ln(2)}} \cdot \frac{BT_s}{T_s}}_{h_{max}} \cdot \exp\left(-\frac{\sqrt{2}\pi}{\sqrt{ln(2)}} \cdot BT_s \cdot \frac{k}{OSR}\right)^2 \tag{C.5}$$

The first factor in Eq. C.5 is the peak of the impulse frequency response.

$$h_{max} = \frac{\sqrt{2\pi}}{\sqrt{ln(2)}} \cdot B \tag{C.6}$$

For BLUETOOTH, with BT_s = 0.5 and T_s = 1 μs we obtain h_{max} = 1.5054 MHz.

Due to reasons described in Chapter 5, it is more efficient to operate on the *cumulative coefficients*

$$c(k) = \sum_{k=0}^{OSR-1} h(k) \tag{C.7}$$

which could be precalculated and stored in a lookup table with k being the index.

APPENDIX D

EXAMPLE VHDL SOURCE CODE

D.1 DCO Level-2

This VHDL code pertains to the DCO model description in Sec. 7.4.

The entity declaration of the level-2 DCO is between lines 18 and 39. The VHDL generics or elaboration-phase parameter constants are declared between lines 19 and 30. The port declaration of the block's I/O signals is between lines 31 and 39. The behavioral architecture describing the block starts at line 42. Lines 43 through 49 declare internal signals. Lines 56 through 61 describe the DCO varactor merging operation on the left of Fig. 7.3 (page 223). The period is calculated and checked for upper and lower bounds between lines 67 and 81. Finally, the period-controlled oscillator (PCO) engine is instantiated between lines 87 and 98.

```
1  ------------------------------------------------------------------
2  -- $Id: dco_l2.vhd,v 1.11 2001/03/19 21:10:43 a196312 Exp $
3  --
4  -- Digitally-controlled Oscillator, dco_l2.vhd
5  --
6  -- Note: actual OP, OA and OT inputs are unsigned
7  --
8  ------------------------------------------------------------------
```

```
 9  -- (C) Robert B. Staszewski, Texas Instruments Inc, 2001/01/11
10  ------------------------------------------------------------------------
11
12  library ieee;
13    use ieee.std_logic_1164.all;
14
15  library rf;
16    use rf.components_common.src_pco_c;
17
18  entity dco_l2 is
19    generic (
20      DCO_PER_0      : time := 417 ps;       -- period of center freq.
21      DCO_PEROFF_LIM : time := 83 ps;      -- maximum period deviation
22      DCO_TIME_RES   : time := 1 fs;         -- finest time resolution
23      DCO_QUANT_P    : positive := 402;
24      DCO_QUANT_A    : positive := 80;
25      DCO_QUANT_T    : positive := 4;          -- (DCO_TIME_RES units)
26      DCO_INIT_DLY   : time := 0 ns;      -- initial oscillation delay
27      DCO_WRMS       : time := 0 ps;   -- accumulative jitter (wander)
28      DCO_JRMS       : time := 0 ps;        -- non-accumulative jitter
29      SEED           : integer := -1             -- time-based, if < 0
30    );
31    port (
32      dco_in_p  : in integer;
33      dco_in_a  : in integer;
34      dco_in_ti : in integer;
35      dco_in_tf : in integer;
36      mat_pdev  : out time := 0 ns;            -- DCO period deviation
```

```
37      ckv       : out std_logic          -- digitized DCO clock output
38    );
39  end;
40  ----------------------------------------------------------------------
41
42  architecture behav of dco_l2 is
43    signal mat_quant_p: integer;
44    signal mat_quant_a: integer;
45    signal mat_quant_ti: integer;
46    signal mat_quant_tf: integer;
47    signal mat_quant_ls: integer;
48    --
49    signal mat_per: time := DCO_PER_0;        -- period of oscillation
50  begin
51
52    ----------------------------------------------------------------------
53    -- Calculate the tuning input
54    ----------------------------------------------------------------------
55
56    mat_quant_p <= dco_in_p * DCO_QUANT_P;
57    mat_quant_a <= dco_in_a * DCO_QUANT_A;
58    mat_quant_ti <= dco_in_ti * DCO_QUANT_T;
59    mat_quant_tf <= dco_in_tf * DCO_QUANT_T;
60    mat_quant_ls <= mat_quant_p + mat_quant_a + mat_quant_ti;
61
62    ----------------------------------------------------------------------
63    -- Calculate the period
64    ----------------------------------------------------------------------
```

```
65
66    process (mat_quant_tf, mat_quant_ls)
67      variable mat_pdev_var: time;
68    begin
69      mat_pdev_var := DCO_TIME_RES * (mat_quant_tf + mat_quant_ls);
70      --
71      -- limit the oscillation period
72      if mat_pdev_var > DCO_PEROFF_LIM then
73        mat_pdev_var := DCO_PEROFF_LIM;
74      elsif mat_pdev_var < -DCO_PEROFF_LIM then
75        mat_pdev_var := -DCO_PEROFF_LIM;
76      end if;
77      --
78      mat_pdev <= mat_pdev_var;
79      mat_per <= DCO_PER_0 - mat_pdev_var;
80    end process;
81
82    ------------------------------------------------------------------
83    -- Computationally-efficient period-controled oscillator
84    ------------------------------------------------------------------
85
86    xpco: src_pco_c
87      generic map (
88        INIT_DELAY => DCO_INIT_DLY,
89        WANDER_RMS => DCO_WRMS,
90        JITTER_RMS => DCO_JRMS,
91        SEED       => SEED
92      )
```

```
93      port map (
94        period0 => mat_per,
95        clk     => ckv,
96        clk2    => open
97      );
98
99  end;
100 ------------------------------------------------------------------------
101 -- end of dco_l2.vhd
```

D.2 Period-controlled Oscillator (PCO)

This VHDL code pertains to the PCO referred to in the previous section.

Input port “period0” of type `time` (line 28) controls the oscillator period during the next cycle. The “smp” internal signal is used to control event activity and to establish the next time-stamp (line 82). It is also used to schedule rise and fall times of the “clk” clock (line 77). With each timestamp, a jitter and wander values are added at lines 58–67 and 68–76, respectively.

```
1   ------------------------------------------------------------------------
2   -- $Id: src_pco_c.vhd,v 1.3 2001/04/07 00:32:00 a196312 Exp $
3   --
4   -- Period-controlled Oscillator, src_pco_c.vhd
5   --
6   -- Includes jitter.
7   -- Includes wander (random walk).
8   -- Uses foreign-C subroutine for Gaussian random number generator.
```

```
 9  --
10  ------------------------------------------------------------------
11  -- (C) Robert B. Staszewski, Texas Instruments Inc, 2000/02/02
12  ------------------------------------------------------------------
13
14  library ieee;
15    use ieee.std_logic_1164.all;
16    use ieee.math_real.all;
17  library rf;
18    use rf.sub_rand.all;
19
20  entity src_pco_c is
21    generic (
22      INIT_DELAY : time := 0 ns;
23      WANDER_RMS : time := 0 ps;
24      JITTER_RMS : time := 0 ps;
25      SEED       : integer := -1            -- time-based seed, if < 0
26    );
27    port (
28      period0 : in time;
29      en      : in std_logic := '1';
30      clk     : out std_logic := '0';
31      clk2    : out std_logic := '0'
32    );
33  end entity;
34  ------------------------------------------------------------------
35
36  architecture behav of src_pco_c is
```

```
37    signal smp: bit := '0';                                    -- tick
38    -- internal measurement signals
39    signal period_s: time := 0 ns;
40    signal tdiff_s: time := 0 ns;-- diff between actual and ref sample
41  begin
42    process (smp, en) is
43      variable initial: boolean := true;
44      variable tref: time := 0 ns;            -- reference sample time
45      variable jitter: time := 0 ns;     -- instantaneous jitter value
46      variable jitter_prev: time := 0 ns;
47      variable wander: time := 0 ns;     -- instantaneous wander value
48      variable period: time := 0 ns;       -- the current clock period
49      variable s1: integer := SEED;
50      variable randvar: real;
51    begin
52      if not initial and en='1' then
53        -- find time difference between actual and referenced samples
54        tdiff_s <= now - tref;
55        tref := tref + period0;
56        -- adjust the next VCO period
57        period := period0;
58        if JITTER_RMS /= 0 ns then
59          -- add Gaussian-distributed jitter
60          sub_randn(randvar);
61          jitter := randvar * JITTER_RMS;
62          if abs(jitter) >= period/2 then       -- check if an outlier
63            jitter := 0 ns;
64          end if;
```

```
65          period := period + jitter - jitter_prev;
66          jitter_prev := jitter;
67        end if;
68        if WANDER_RMS /= 0 ns then
69          -- add Gaussian-distributed wander
70          sub_randn(randvar);
71          wander := randvar * WANDER_RMS;
72          if abs(wander) >= period/2 then       -- check if an outlier
73            wander := 0 ns;
74          end if;
75          period := period + wander;
76        end if;
77        clk <= '1', '0' after period/2;   -- clock with 50% duty cycle
78        -- half-rate clock
79        if smp = '0' then  clk2 <= '0';
80        else               clk2 <= '1';
81        end if;
82        smp <= not smp after period;
83        period_s <= period;
84      else
85        sub_randomize_i(s1);
86        sub_randomize_o(s1);
87        period := period0;       -- the initial time period is period0
88        tref := INIT_DELAY;  -- mark the first transition as reference
89        clk <= '0';                -- the initial clock output is low
90        clk2 <= '0';              -- the initial clock/2 output is low
91        smp <= transport '1' after INIT_DELAY;     -- first transition
92        initial := false;
```

```
93         period_s <= period;
94       end if;
95     end process;
96   end architecture;
97   ------------------------------------------------------------------
98   -- end of src_pco_c.vhd --
```

D.3 Tactical Flip-Flop

This VHDL code pertains to the flip-flop model description in Sec. 7.5.

The flip-flow is powered-up at a random logic low or high state with equal probaI also appreciate your being on my committee and taking time to read my dissertation.bility. This is performed between lines 53 and 62. In the course of normal operation, upon rising edge of the CLK clock (line 65), the input level is sampled (line 69) and activated on the complementary output ports (lines 81 and 87) after an appropriate delay of TD_Q. Metastability detection between the D data input and CLK clock is performed on line 67. If the data is changing within a certain window T_SU before the clock, an 'X' will be produced at the output.

The "translate_on" and "translate_off" pragma directives hide the non-synthesizable behavioral code from synthesis tools, but keep it visible forI also appreciate your being on my committee and taking time to read my dissertation. the VHDL simulator. The "dc_script_begin" and "dc_script_end" pragma directives convey special non-VHDL information to the SYNOPSYS Design Compile. In this case, they specify the exact flip-flop and its library location to be used.

```
1  ------------------------------------------------------------------
2  -- $Id: DTT01.vhd,v 1.2 2001/03/19 23:01:52 a196312 Exp $
3  --
4  -- Single-bit register with complementary outputs.
5  -- (Detects metastability.)
6  -- (Randomizes the power-up value of the registers.)
7  --
8  ------------------------------------------------------------------
9  -- (C) Robert B. Staszewski, Texas Instruments Inc, 2001/03/15
10 ------------------------------------------------------------------
11
12 library ieee;
13   use ieee.std_logic_1164.all;
14
15 -- pragma translate_off
16 library rf;
17   use rf.sub_rand.all;
18 -- pragma translate_on
19
20 entity DTT01 is
21   -- pragma translate_off
22   generic (
23     SEED : integer := -1;                    -- time-based, if < 0
24     T_SU : time := 0 ps;
25     TD_Q : time := 0 ps
26   );
27   -- pragma translate_on
28   port (
```

```
29      D   : in std_logic;
30      CLK : in std_logic;
31      Q   : out std_logic;
32      QZ  : out std_logic
33    );
34    -- pragma dc_script_begin
35    -- set_register_type -exact -flip_flop DTT01
36    -- remove_attribute GS40_DTT01_W_115_1.35_CORE.db/DTT01 dont_use
37    -- pragma dc_script_end
38  end;
39  ------------------------------------------------------------------------
40
41  architecture rtl of DTT01 is
42    signal qq: std_logic;
43  begin
44
45    process (CLK)
46      -- pragma translate_off
47      variable s1: integer := SEED;
48      variable initial: boolean := true;
49      variable randvar: natural;
50      -- pragma translate_on
51    begin
52      -- pragma translate_off
53      if initial then
54        sub_randomize_i(s1);
55        sub_randomize_o(s1);
56        sub_randb(randvar);
```

```
57        if randvar = 0 then
58          qq <= '0';
59        else
60          qq <= '1';
61        end if;
62        initial := false;
63      elsif now > 0 ns then
64        -- pragma translate_on
65        if CLK'event and CLK='1' then
66          -- pragma translate_off
67          if D'last_event >= T_SU then
68            -- pragma translate_on
69            qq <= D;
70            -- pragma translate_off
71          else
72            qq <= 'X';
73          end if;
74          -- pragma translate_on
75        end if;
76      -- pragma translate_off
77      end if;
78      -- pragma translate_on
79    end process;
80
81    Q <= qq
82    -- pragma translate_off
83    after TD_Q
84    -- pragma translate_on
```

```
85     ;
86
87     QZ <= not qq
88     -- pragma translate_off
89     after TD_Q
90     -- pragma translate_on
91     ;
92
93   end;
94   ---------------------------------------------------------------------
95   -- end of DTT01.vhd
```

D.4 TDC Pseudo-thermometer Output Decoder

This VHDL code pertains to the TDC output vector decoder model described in Sec. 7.5.

The 48-bit input vector from the TDC is inspected bit-by-bit for “forced unknown” (’X’) on line 76. If detected, a logic level zero (line 78) or one (line 79) is used instead with equal probability. Thus resolved input vector is used by the main process (lines 99 through 152). Lines 107 through 123 perform a combinatorial logic (terminating on a latch – line 105) of priority decoding by detecting the earliest occurrence of rising and falling digital edge transitions. The half-period calculation for the period inversion estimation is done between the lines of 140 and 144.

```
1   ---------------------------------------------------------------------
2   -- $Id: pf_dec.vhd,v 1.4 2001/03/22 00:29:25 a196312 Exp $
3   --
```

```
4  -- Thermometer-code decoder of time-to-digital converter, pf_dec.vhd
5  --
6  -- (Combinatorial logic)
7  -- (Includes edge-skipping compensation)
8  --
9  ------------------------------------------------------------------------
10 -- (C) Robert B. Staszewski, Texas Instruments Inc, 2001/01/29
11 ------------------------------------------------------------------------
12
13 library ieee;
14   use ieee.std_logic_1164.all;
15   use ieee.numeric_std.all;
16
17 -- pragma translate_off
18 library rf;
19   use rf.sub_rand.all;
20 -- pragma translate_on
21
22 entity pf_dec is
23   generic (
24     -- pragma translate_off
25     TD_Q : time := 0 ps;
26     SEED : integer := -1;                    -- time-based, if < 0
27     -- pragma translate_on
28     SELQ : integer := 4;      -- TDC_Q index used for edge selection
29     DTDC : integer := 48;             -- latched TDC array bus width
30     WTDC : integer := 6              -- decoded TDC output bus width
31   );
```

```
32    port (
33      tdc_q    : in std_logic_vector (DTDC downto 1);
34      ckr      : in std_logic;                -- level-sensitive latch
35      tdc_rise : out unsigned (WTDC-1 downto 0);  -- quant. rise delta
36      tdc_skip : out std_logic;             -- skip one full CKV cycle
37      tdc_hper : out unsigned (WTDC-1 downto 0)  -- quant. half-period
38    );
39    -- pragma dc_script_begin
40    -- set_driving_cell -cell IV120 all_inputs()
41    -- set_max_fanout 1 {tdc_q}
42    -- create_clock -name CKR -period 20.0 find(port,ckr)
43    -- set_input_delay 0.3 -clock CKR {tdc_q}
44    -- set_fix_multiple_port_nets -all
45    -- pragma dc_script_end
46  end;
47  ------------------------------------------------------------------------
48
49  architecture rtl of pf_dec is
50    signal q1: std_logic_vector (DTDC downto 1);
51    constant SLV_0: std_logic_vector (SELQ downto 1) := (others=>'0');
52  begin
53
54    ------------------------------------------------------------------
55    -- Resolve the TDC X'es
56    ------------------------------------------------------------------
57
58    process (tdc_q)
59      -- pragma translate_off
```

```
60      variable s1: integer := SEED;
61      variable initial: boolean := true;
62      variable randvar: natural;
63      -- pragma translate_on
64      variable q: std_logic_vector (DTDC downto 1);
65    begin
66      -- pragma translate_off
67      if initial then
68        sub_randomize_i(s1);
69        sub_randomize_o(s1);
70        initial := false;
71      else
72      -- pragma translate_on
73        -- resolve X'es
74        for k in 1 to DTDC loop
75          -- pragma translate_off
76          if tdc_q(k) = 'X' then
77            sub_randb(randvar);
78            if randvar = 0 then     q(k) := '0';
79            elsif randvar = 1 then  q(k) := '1';
80            end if;
81          else
82          -- pragma translate_on
83            q(k) := tdc_q(k);
84          -- pragma translate_off
85          end if;
86          -- pragma translate_on
87        end loop;
```

```
88        --
89        q1 <= q;
90      -- pragma translate_off
91      end if;
92      -- pragma translate_on
93    end process;
94
95    ------------------------------------------------------------------
96    -- Decode and latch
97    ------------------------------------------------------------------
98
99    process (ckr, q1)
100     variable rise: integer range DTDC-1 downto 0;
101     variable fall: integer range DTDC-1 downto 0;
102     variable half_period: integer range DTDC-1 downto 0;
103     variable skip: std_logic;
104   begin
105     if ckr = '1' then
106       -- digital rising transition detector
107       rise := 0;   -- in case not found
108       for k in 2 to DTDC loop
109         if q1(k-1)='1' and q1(k)='0' then
110           rise := k-1;
111           exit;
112         end if;
113       end loop;
114       -- digital falling transition detector
115       fall := 0;   -- in case not found
```

```
116         for k in 2 to DTDC loop
117           if q1(k-1)='0' and q1(k)='1' then
118             fall := k-1;
119             exit;
120           end if;
121         end loop;
122         --
123         tdc_rise <= to_unsigned(rise, WTDC)
124         -- pragma translate_off
125         after TD_Q
126         -- pragma translate_on
127         ;
128         if q1(SELQ downto 1) = SLV_0 then
129           skip := '1';
130         else
131           skip := '0';
132         end if;
133         --
134         tdc_skip <= skip
135         -- pragma translate_off
136         after TD_Q
137         -- pragma translate_on
138         ;
139         -- calculate the oscillator clock instantaneous half-period
140         if rise > fall then
141           half_period := rise - fall;
142         else
143           half_period := fall - rise;
```

```
144         end if;
145         --
146         tdc_hper <= to_unsigned(half_period, WTDC)
147         -- pragma translate_off
148         after TD_Q
149         -- pragma translate_on
150         ;
151       end if;
152     end process;
153   end;
154   ---------------------------------------------------------------------
155   -- end of pf_dec.vhd
```

REFERENCES

[1] T. H. Lee, “CMOS RF: (Still) no longer an oxymoron,” *Proc. of Symposium on Radio Frequency Integrated Circuits*, pp. 3–6, 1999.

[2] A. A. Abidi, “Wireless transceivers in CMOS IC technology. The new wave,” *Proc. of Symposium on VLSI Technology*, pp. 151–158, 2000.

[3] J. N. Burghartz, M. Hargrove, C. S. Webster, et al., “RF potential of a 0.18-μm CMOS logic device technology,” *IEEE Trans. on Electron Devices*, vol. 47, no. 4, pp. 864–870, April 2000.

[4] H. Iwai, “CMOS technology for RF applications,” *Proc. of 22nd International Conf. on Microelectronics*, vol. 1, pp. 27–34, May 2000.

[5] J. T. Wu, M. J. Chen, C. C. Hsu, “A 2 V 900 MHz CMOS phase-locked loop,” *Proc. of IEEE Symposium on VLSI Circuits*, pp. 52–53, June 1998.

[6] Q. Huang, P. Orsatti, F. Piazza, “GSM transceiver front-end circuits in 0.25-mum CMOS,” *IEEE Journal of Solid-State Circuits*, vol. 34, no. 3, pp. 292–303, March 1999.

[7] J. L. Tham, M. A. Margarit, B. Pregardier, et al., “A 2.7-V 900-MHz/1.9-GHz dual-band transceiver IC for digital wireless communication,” *IEEE Journal of Solid-State Circuits*, vol. 34, no. 3, pp. 286–291, March 1999.

[8] "GS40 0.11-μm CMOS Standard Cell/Gate Array," *Texas Instruments Application Specific Integrated Circuits Macro Library Summary, Version 1.0*, Jan. 2001.

[9] *Specification of the Bluetooth System, Version 1.1*, www.bluetooth.com, 22 Feb. 2001.

[10] G. K. Dehng, C. Y. Yang, J. M. Hsu, et al., "A 900-MHz 1-V CMOS frequency synthesizer," *IEEE Journal of Solid-State Circuits*, vol. 35, no. 8, pp. 1211–1214, Aug 2000.

[11] *The national technology roadmap for semiconductors*, Semiconductor Industries Association, 1997.

[12] N. K. Verghese, T. J. Schmerbeck, D. J. Allstot, *Simulation techniques and solutions for mixed-signal coupling in integrated circuits*, Boston, MA: Kluwer Academic, 1995.

[13] B. Razavi, *RF Microelectronics*, Upper Saddle River, NJ: Prentice Hall, 1998.

[14] J. Craninckx, M. Steyaert, *Wireless CMOS Frequency Synthesizer Design*, Boston, MA: Kluwer Academic, 1998.

[15] T. C. Weigandt, B. Kim, P. R. Gray, "Analysis of timing jitter in CMOS ring oscillators," *Proc. of IEEE Symposium on Circuits and Systems.*, pp. 27–30, 1994.

[16] B. Razavi, "Challenges in the design of frequency synthesizers for wireless applications," *Proc. of the Custom Integrated Circuits Conference*, pp. 395-402, 1997.

[17] K. Muhammad, R. B. Staszewski, P. T. Balsara, "Challenges in integrated CMOS transceivers for short distance wireless" *Proc. of Great Lakes Symposium on VLSI*,

March 2001.

[18] T. S. Rappaport, *Wireless Communications: Principles and Practice*, Prentice Hall, 1996.

[19] M. Bopp et al., "A DECT transceiver chip set using SiGe technology," *Proc. of IEEE Solid-State Circuits Conf.*, sec. MP4.2, pp. 68–69, 447, Feb. 1999.

[20] B. Zhang, P. Allen, "Feed-forward compensated high switching speed digital phase-locked loop frequency synthesizer," *Proc. of IEEE Symposium on Circuits and Systems.*, vol. 4, pp. 371–374, 1999.

[21] V. Reinhardt, K. Gould, K. McNab et al., "A short survey of frequency synthesizer techniques," *Proc. 40th Annual Frequency Control Symposium*, pp. 355–365, May 1986

[22] J. Tierney, C. M. Radar, B. Gold, "A digital frequency synthesizer," *IEEE Trans. on Audio Electroaccoust.*, vol. AU-19, pp. 48–57, Mar 1971.

[23] L. K. Tan, H. Samueli, "A 200MHz quadrature digital synthesizer/mixer in 0.8 μm CMOS," *IEEE Journal of Solid-State Circuits*, vol. 30, no. 3, pp. 193–200, March 1995

[24] A. N. Hafez, M. I. Elmasry, "A low power monolithic subsampled phase-locked loop architecture for wireless transceivers," *Proc. of IEEE Symposium on Circuits and Systems*, vol. 2, pp. 549–552, May–June 1999.

[25] F. M. Gardner, "Charge-pump phase-locked loops," *IEEE Trans. on Communications*, vol. COMM-28, pp. 1849–1858, Nov. 1980.

[26] D. H. Wolaver, *Phase-locked Loop Circuit Design*, Englewood Cliffs, NJ: Prentice Hall, 1993.

[27] W. B. Wilson, U. K. Moon, K. R. Lakshmikumar, et al., "A CMOS self-calibrating frequency synthesizer," *IEEE Journal of Solid-State Circuits*, vol. 35, no. 10, pp. 1437–1444, Oct. 2000.

[28] I. C. Hwang, S. H. Song, S. W. Kim, "A digitally controlled phase-locked loop with a digital phase-frequency detection for fast acquisition," *IEEE Journal of Solid-State Circuits*, vol. 36, no. 10, pp. 1574–1581, Oct. 2001.

[29] T. P. Kenny, T. A. Riley, N. M. Filiol, M. A. Copeland, "Design and realization of a digital Delta-sigma modulator for fractional-N frequency synthesis," *IEEE Journal of Solid-State Circuits*, vol. 34, no. 3, pp. 292–303, March 1999.

[30] B. Miller, R. J. Conley, "A multiple modulator fractional divider," *IEEE Trans. on Instrumentation and Measurement*, vol. 40, no. 3, pp. 578–583, June 1991.

[31] T. Riley, M. Copeland, T. Kwasniewski, "Delta-sigma modulation in fractional-N frequency synthesis," *IEEE Journal of Solid-State Circuits*, vol. 28, no. 5, pp. 553–559, May 1993.

[32] Y. Matsua et al., "A 16-bit oversampling A/D conversion technology using triple integration noise shaping," *IEEE Journal of Solid-State Circuits*, vol. SC-22, pp. 921–929, Dec. 1987.

[33] M. H. Perrot et. al., "A 27-mW CMOS fractional-N synthesizer using digital compensation for 2.5-Mb/s GFSK modulation," *IEEE Journal of Solid-State Circuits*, vol. 32,

no. 12, pp. 2048–2060, Dec 1993.

[34] W. T. Bax, M. A. Copeland, “A GMSK modulator using a $\Delta\Sigma$ frequency discriminator-based synthesizer,” *IEEE Journal of Solid-State Circuits*, vol. 36, no. 8, pp. 1218–1227, Aug. 2001.

[35] H. Brugel, P. F. Driessen, “Variable bandwidth DPLL bit synchronizer with rapid acquisition implemented as a finite state machine,” *IEEE Trans. on Communications*, vol. 42, pp. 2751–2759, Sept. 1994.

[36] J. Dunning, G. Garcia, J. Lundberg, E. Nuckolls, “An all-digital phase-locked loop with 50-cycle lock time suitable for high performance microprocessors,” *Journal of Solid-State Circuits*, vol. 30, pp. 412–422, Apr. 1995.

[37] R. E. Best, *Phase Locked Loops: Design, Simulation and Applications, third edition*, New York, NY: McGraw-Hill, 1997.

[38] T. Y. Hsu, B. J. Shieh, C. Y. Lee, “An all-digital phase-locked loop (ADPLL)-based clock recovery circuit,” *Journal of Solid-State Circuits*, vol. 34, pp. 1063–1073, Aug. 1999.

[39] M. Olivieri, A. Trifiletti, “An all-digital clock generator firm-core based on differential fine-tuned delay for reusable microprocessor cores,” *Proc. of IEEE Symposium on Circuits and Systems*, vol. 4, pp. 638–641, 2001.

[40] A. Kajiwara, M. Nakagawa, “A new PLL frequency synthesizer with high switching speed,” *IEEE Trans. on Vehicular Technology*, vol. 41, no. 4, pp. 407–413, Nov 1992.

[41] R. B. Staszewski and S. Kiriaki, “Top-down simulation methodology of a 500 MHz mixed-signal magnetic recording read channel using standard VHDL,” *Proc. of Behavioral Modeling and Simulation Conf.*, sec. 3.2, Oct. 1999.

[42] D. G. Chinnery, B. Nikolic and K. Keutzer, “Achieving 550 MHz in an ASIC Methodology,” *Proc. of Design Automation Conference*, pp. 420–425, June 2001

[43] R. B. Staszewski, K. Muhammad and P. Balsara, “A 550-MSample/s 8-tap FIR digital filter for magnetic recording read channels,” *IEEE Journal of Solid-State Circuits*, vol. 35, pp. 1205–1210, Aug. 2000.

[44] K. Muhammad, R. B. Staszewski and P. T. Balsara, “Speed, power, area, and latency tradeoffs in adaptive FIR filtering for PRML read channels,” *IEEE Transactions on VLSI Systems*, vol. 9, iss. 1, pp. 42–51, Feb. 2001

[45] R. B. Staszewski, K. Muhammad and P. T. Balsara, “A constrained asymmetry LMS algorithm for PRML disk drive read channels,” *IEEE Transactions on Circuits and Systems II*, vol. 48, pp. 793–798, Aug. 2001

[46] www.homerf.org.

[47] www.ieee802.org/11

[48] C. L. Huang, N. D. Arora, “Measurements and modeling of MOSFET I-V characteristics with polysilicon depletion effect,” *IEEE Trans. on Electron Devices*, vol. 40, no. 12, pp. 2330–2337, Dec 1993.

[49] J. B. Shyu, G. C. Temes, F. Krummenacher, “Random error effects in matched MOS

capacitors and current sources," *Journal of Solid-State Circuits*, vol. SC-19, pp. 948–955, Dec. 1984.

[50] S. H. Lee, B. S. Song, "Digital-domain calibration of multistep analog-to-digital converters," *Journal of Solid-State Circuits*, vol. 27, pp. 1679–1688, Dec. 1992.

[51] T. H. Lee, A. Hajimiri, "Oscillator Phase Noise: A Tutorial," *IEEE Journal of Solid-State Circuits*, vol. 35, no. 3, pp. 326–336, March 2000.

[52] A. Hajimiri, T. H. Lee, "A general theory of phase noise in electrical oscillators," *IEEE Journal of Solid-State Circuits*, vol. 35, no. 3, pp. 326–336, Feb. 1998.

[53] A. Hajimiri, T. H. Lee, *The Design of Low Noise Oscillators*, Norwell, MA: Kluwer Academic, 1999.

[54] J. C. Candy, G. C. Temes, "Oversampling methods for A/D and D/A conversion," *Oversampling Delta-Sigma Data Converters*, New York: IEEE Press, 1991.

[55] R. E. Radke, A. Eshraghi, T. S. Fiez, "A 14-bit current-mode $\Sigma - \Delta$ DAC based upon rotated data weighted averaging," *IEEE Journal of Solid-State Circuits*, vol. 35, no. 8, pp. 1074–1084, Aug. 2000

[56] P. Dudek, S. Szczepanski, J. Hatfield, "A high-resolution CMOS time-to-digital converter utilizing a Vernier delay line," *IEEE Journal of Solid-State Circuits*, vol. 35, no. 2, pp. 240–247, Feb. 2000.

[57] B. N. Nikolic, V. G. Oklobdzija, V. Stajonovic et al., "Improved sense-amplifier-based flip-flop: design and measurements," *IEEE Journal of Solid-State Circuits*, vol. 35, no. 6, pp. 876–884, June 2000.

[58] T. J. Gabara, G. J. Cyr, C. E. Stroud, “Metastability of CMOS master/slave flip-flops,” *IEEE Trans. on Circuits and Systems–II*, vol. 39, no. 10, pp. 734–740, Oct. 1992.

[59] C. Brown, K. Feher, “Measuring metastability and its effect on communication signal processing systems,” *IEEE Trans. on Instrumentation and Measurement*, vol. 46, no. 1, pp. 61–64, Feb. 1997.

[60] B. Razavi, “Design of monolithic phase-locked loops and clock recovery circuits – a tutorial,” in *Monolithic phase-locked loops and clock recovery circuits: theory and design*, New York, NY: IEEE Press, 1996.

[61] F. Spagna, “Phase locked loop using delay compensation techniques,” *Proc. of IEEE Symposium on Computers and Communications.*, pp. 417–423, 2000.

[62] F. M. Gardner, *Phaselock Techniques*, John Wiley Sons, 1979.

[63] T. M. Almeida, M. S. Piedade, “High performance analog and digital PLL design,” *Proc. of IEEE Symposium on Circuits and Systems*, vol. 4, pp. 394–397, 1999.

[64] J. Lee, B. Kim, “A 200MHz Low Jitter Adaptive Bandwidth PLL,” *Proc. of IEEE Solid-State Circuits Conf.*, sec. WA20.1, pp. 346–347, 477, Feb. 1999.

[65] H. Sato, K. Kato, T. Sase, “A fast pull-in PLL IC using two-mode pull-in technique,” *Electronics and Communications in Japan*, part 2, vol. 75, no. 3, pp. 41–50, 1992.

[66] W. H. Press, S. A. Teukolsky, W. T. Vetterling et al., Numerical Recipes in C, second edition, Cambridge University Press, 1994

[67] A. J. Acosta, A. Barriga, M. Valencia et al., “Modeling of real bistables in VHDL,” *Proc. of Design Automation Conference*, pp. 460–465, Feb. 1993.

[68] T. H. Lee, *The Design of CMOS Radio-Frequency Integrated Circuits*, Cambridge, UK: Cambridge University Press, 1998.

[69] T. Sowlati, C. A. Salama, J. Sitch et al., “Low voltage, high efficiency GaAs class E power amplifiers for wireless transmitters,” *IEEE Journal of Solid-State Circuits*, vol. 30, no. 10, pp. 1074–1080, Oct. 1995.