CHALMERS
UNIVERSITY OF TECHNOLOGY

Coded pilot synchronization in wide band sub-THz communication systems

Master's thesis in Embedded Electronic System Design

VIJAYASRI KRISTAPARAPU

Department of Microtechnology and Nanoscience
CHALMERS UNIVERSITY OF TECHNOLOGY
Gothenburg, Sweden 2026

Master's thesis 2026

Coded pilot synchronization in wide band sub-THz communication systems

VIJAYASRI KRISTAPARAPU

Department of Microtechnology and Nanoscience
Chalmers University of Technology
Gothenburg, Sweden 2026

Coded pilot synchronization in wide band sub-THz communication systems
VIJAYASRI KRISTAPARAPU

Supervisor: Lars Svensson, Department of Microtechnology and Nanoscience
Company advisor: Sining An, Ericsson AB
Examiner: Per Larsson-Edefors , Department of Microtechnology and Nanoscience

Master's Thesis 2026
Department of Microtechnology and Nanoscience
Chalmers University of Technology
SE-412 96 Gothenburg
Telephone +46 31 772 1000

Typeset in LaTeX
Gothenburg, Sweden 2026

Coded pilot synchronization in wide band sub-THz communication systems
VIJAYASRI KRISTAPARAPU
Department of Microtechnology and Nanoscience
Chalmers University of Technology

Abstract

Future wireless communication systems operating in the sub-terahertz (sub-THz) frequency range (100–300 GHz) offer extremely large bandwidths, enabling ultra high data rates for next-generation applications such as wireless backhaul, chip-to-chip communication, and sensing. However, carrier synchronization (CS) in such systems remains a major challenge due to large carrier frequency offsets (CFO), severe phase noise, and the impracticality of using ultra-high-speed analog-to-digital converters (ADCs) for wideband signal processing.

This thesis proposes a low cost analog-digital hybrid synchronization method using a low-power pseudo-noise (PN) coded pilot that can be detected even at very low power levels using matched filtering. A PN-coded pilot is embedded within the transmitted signal (a single wideband signal) at power levels up to 30-50 dB below the data signal. At the receiver, the pilot is extracted using a narrowband filter and processed with a PN-based matched filter to achieve significant processing gain, enabling reliable detection using low-speed ADCs.

A hybrid analog–digital phase-locked loop (PLL) architecture is proposed, where carrier frequency and phase estimates obtained from PN correlation are used to control the local oscillator. The system is first validated through MATLAB simulations under realistic impairments, including large CFO and strong phase noise. Results demonstrate successful carrier recovery and reliable detection of pilot signals buried down to 30 dB below the data signal. The proposed approach is further implemented on an FPGA platform to evaluate real-time performance and hardware feasibility.

The results show that PN-coded pilot synchronization is a promising solution for enabling practical, low-complexity carrier recovery in ultra-wideband sub-THz systems without requiring high-speed data converters.

Keywords: fixed-point arithmetic, VHDL design, DPLL, NCO, signal processing, wireless communication systems

Acknowledgements

I would like to express my sincere gratitude to my company advisor, Sining An, for the technical guidance and practical insights. Regular discussions regarding progress, troubleshooting challenges, and planning subsequent steps greatly contributed to the effective execution of this project.

I am also grateful to my supervisor, Lars Svensson, for his continuous guidance, invaluable advice, and unwavering support throughout this project. Weekly meetings and discussions on updates and next steps were instrumental in shaping the direction and success of this research.

I extend my thanks to the examiner, Per Larsson-Edefors, for his time, constructive feedback on reports, and thoughtful evaluation of this work.

Finally, I would like to thank my family for their unconditional support and encouragement.

This project would not have been possible without the collective support, guidance, and encouragement from all the individuals mentioned above.

Vijayasri Kristaparapu, Gothenburg, June 2026

List of Abbreviations

ADC	Analog-to-Digital Converter
AWG	Arbitrary Waveform Generator
BER	Bit Error Rate
CFO	Carrier Frequency Offset
CORDIC	Coordinate Rotation Digital Computer
DAC	Digital-to-Analog Converter
DPLL	Digital Phase-Locked Loop
DSP	Digital Signal Processing
FIR	Finite Impulse Response
FPGA	Field-Programmable Gate Array
FP	Fixed Point (Fixed-Point Arithmetic)
GS	Gold Sequence
HSMC	High-Speed Mezzanine Card
I/Q	In-phase / Quadrature components
ISI	Inter-Symbol Interference
LFSR	Linear Feedback Shift Register
LNA	Low Noise Amplifier
LO	Local Oscillator
LUT	Look-Up Table
MATLAB	Matrix Laboratory
m-sequence	Maximal Length Sequence
NCO	Numerically Controlled Oscillator
PLL	Phase-Locked Loop
PI	Proportional–Integral controller
PN	Pseudo-Noise
PAPR	Peak-to-Average Power Ratio
PSR	Peak-to-Sidelobe Ratio
RF	Radio Frequency
ROM	Read-Only Memory
SNR	Signal-to-Noise Ratio
SSB	Single Sideband
THz	Terahertz
VCO	Voltage-Controlled Oscillator
VHDL	VHSIC Hardware Description Language
ZC	Zadoff–Chu sequence

Contents

1

Introduction

Future wireless communication systems are expected to operate in the sub-terahertz (sub-THz) frequency range (100–300 GHz), where extremely large instantaneous bandwidths, on the order of tens of gigahertz, are available. These bandwidths enable ultra-high data rates, making sub-THz communication a key technology for applications such as wireless fiber extension, chip-to-chip communication, high-capacity backhaul, and joint communication-and-sensing systems.

Despite these advantages, the realization of practical ultra-wideband sub-THz links presents significant technical challenges. One of the most critical challenges is carrier synchronization (CS), which refers to the receiver's ability to accurately track the transmitter's carrier frequency and phase. In sub-THz systems, synchronization is particularly difficult due to large carrier frequency offsets (CFOs) arising from frequency multiplication, as well as severe phase noise caused by high-frequency oscillators.

Existing approaches require high-speed ADCs or are limited to narrowband systems. Their applicability to ultra-wideband sub-THz systems therefore remains limited. For example, a communication signal occupying 20 GHz of bandwidth requires a minimum sampling rate of approximately 40 GS/s according to the Nyquist criterion, while a 40 GHz bandwidth signal would require approximately 80 GS/s. Such ADCs typically consume several watts of power and generate very large data rates that place significant demands on digital processing hardware. For instance, an 80 GS/s ADC with 8-bit resolution produces a raw data throughput of approximately 640 Gbit/s, making real-time processing and data movement extremely challenging in practical receiver implementations.

To address this challenge, recent work at millimeter-wave (mmWave) and E-band frequencies [1, 2] has proposed coded pilot-assisted synchronization techniques. In these approaches, a low-rate PN coded pilot is embedded into the transmitted signal. The receiver extracts this pilot using a narrowband low pass filter and processes it using low-speed ADCs to estimate carrier frequency and phase. These estimates are then used to correct the local oscillator (LO), enabling carrier synchronization without requiring full-bandwidth sampling.

While such techniques have been successfully demonstrated for relatively narrowband systems (below 1 GHz), their applicability to ultra-wideband sub-THz systems remains largely unexplored. Extending coded pilot synchronization to this regime introduces new challenges, including larger CFOs, stronger phase noise, stricter pilot power constraints, and increased implementation complexity.

This thesis aims to investigate and extend PN-coded pilot–assisted carrier synchronization to ultra-wideband sub-THz communication systems. The focus is on designing a system capable of reliably detecting and utilizing a very low-power pilot signal, embedded tens of decibels below the data signal, using only low-speed ADCs and DACs. The work includes system-level modeling, MATLAB-based simulation, and FPGA-based hardware implementation of key synchronization components.

The contributions of this thesis include the design of a PN-based matched filtering approach for pilot detection, development of a DPLL for carrier tracking, and evaluation of system performance under realistic sub-THz impairments. The results demonstrate that reliable carrier synchronization is achievable even when the pilot signal is significantly weaker than the data signal, making the proposed approach suitable for practical sub-THz communication systems.

1.1 Related Work

High-data-rate millimeter-wave (mmWave) communication systems have gained significant attention due to their potential for multi-gigabit wireless links [3, 4]. Synchronization is a key challenge in such systems because phase noise and timing errors degrade signal quality, particularly in wideband transmissions [5, 6].

Traditional carrier synchronization techniques have been extensively studied in digital communication theory [7, 6]. Spread spectrum approaches, utilizing PN sequences, further enhance robustness against interference and channel impairments [8, 9].

Recent research has focused on hardware-efficient implementations of baseband receivers. An et al. proposed a synchronous baseband receiver for mmWave systems, demonstrating high-speed operation [1]. This was extended with a coded-pilot-assisted architecture, improving performance under low SNR conditions [2]. Implementation on FPGA platforms, such as the Arria V GT, has enabled real-time demonstration of these advanced algorithms [10, 11, 12].

Sub-THz systems are being explored for 6G and beyond, presenting new opportunities and challenges for synchronization and phase noise compensation [4, 3]. Energy-efficient carrier phase recovery methods for higher-order modulations have also been proposed, addressing both performance and hardware constraints [13].

1.2 Purpose and Goal

The primary purpose of this thesis is to investigate and implement a PN-coded pilot–assisted carrier synchronization technique for ultra-wideband sub-THz communication systems using low-speed ADCs and DACs.

The main objectives of this work are:

- To design a PN-coded pilot signal that can be reliably detected even when embedded at very low power levels (20–50 dB below the data signal), indepen-

dent of the underlying modulation format. In this work, QPSK modulation is used as the primary validation case.

- To develop a MATLAB-based simulation framework to evaluate pilot detection, correlation performance, and carrier synchronization under realistic impairments such as large CFO and phase noise.
- To implement key synchronization blocks, including matched filtering, phase detection, loop filtering, and NCOs, on an FPGA platform.
- To design and validate a hybrid analog–digital phase-locked loop (PLL) for carrier recovery.
- To experimentally verify system performance in terms of synchronization accuracy, lock time, and robustness.

The ultimate goal is to demonstrate that reliable carrier synchronization can be achieved in ultra-wideband sub-THz systems using practical hardware constraints, without requiring full-bandwidth sampling.

1.3 Thesis Outline

This thesis is organized as follows:

- Chapter 2 presents the technical background, including carrier synchronization, PLL, PN sequences, and matched filtering concepts.
- Chapter 3 defines the system specifications, including signal parameters, pilot design, bandwidth constraints, and synchronization requirements.
- Chapter 4 describes the overall methodology, including system modeling, simulation strategy, and hardware implementation approach.
- Chapter 5 focuses on MATLAB-based system design and validation, including PN-based pilot generation, performance evaluation, and analysis under channel impairments.
- Chapter 6 presents the FPGA implementation of key subsystems, including receiver chain validation, NCO-based signal generation, filtering, and IQ signal processing.
- Chapter 7 provides simulation and experimental results, including matched filter performance, spectral analysis, IQ verification, and FPGA resource utilization.
- Chapter 8 concludes the thesis and summarizes the key findings, along with directions for future work.

2

Technical Background

This chapter presents the theoretical foundations required to understand the proposed PN-coded pilot–assisted carrier synchronization system. The description builds upon established principles in communication systems, signal processing, and control theory, with particular emphasis on synchronization techniques suitable for sub-terahertz (sub-THz) systems. Special attention is given to digital implementations, which are central to FPGA-based realizations.

2.1 Sub-THz Communication Systems

Sub-terahertz (sub-THz) communication systems operate in the frequency range between 100 and 300 GHz and offer extremely wide bandwidths, often on the order of tens of gigahertz. These large bandwidths enable very high data rates, making sub-THz technology a promising candidate for next-generation wireless systems [14].

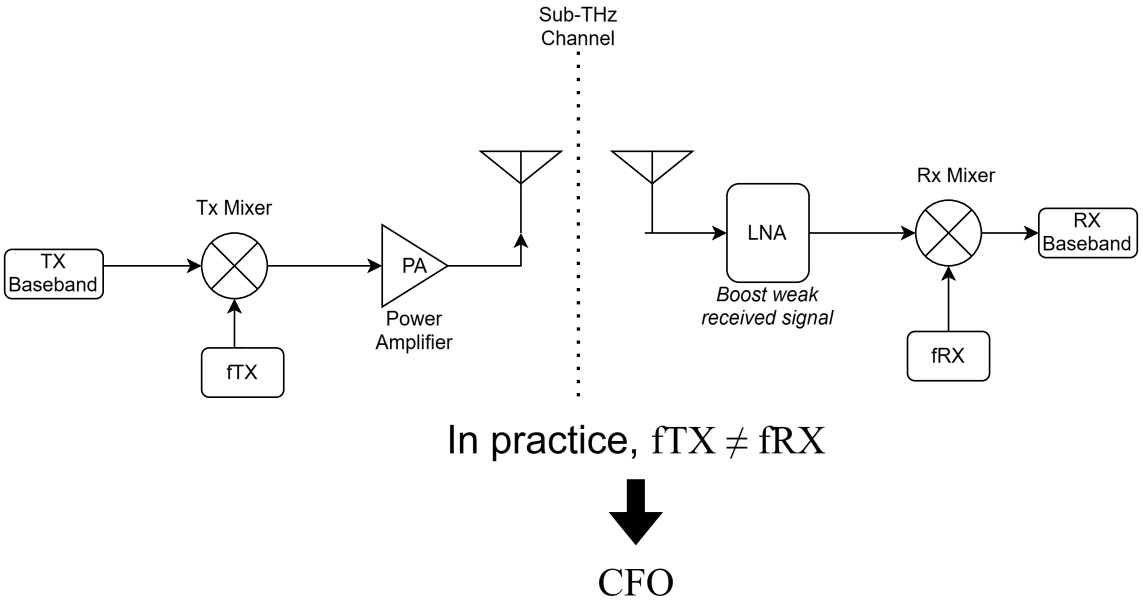


Figure 2.1: Simplified sub-THz communication receiver

However, operation at such high frequencies introduces several significant challenges in propagation, synchronization, and hardware implementation [15]. Oscillator instability leads to severe phase noise. In addition, even small mismatches between

transmitter and receiver oscillators result in large CFO due to the high carrier frequency. Hardware constraints further complicate the design, particularly in ADCs, where sampling the full bandwidth becomes impractical.

Figure 2.1 illustrates the fundamental synchronization challenge in sub-THz communication systems. Any frequency mismatch between the transmitter and receiver local oscillators introduces a carrier frequency offset (CFO), which results in continuous phase rotation of the received signal and must be estimated and compensated before reliable demodulation can be achieved.

These impairments make robust and efficient carrier synchronization techniques essential for reliable communication in sub-THz systems.

2.2 Complex Signal Representation

In modern communication systems, signals are often represented in complex form using in-phase (I) and quadrature (Q) components [16]. A complex exponential signal with constant amplitude can be expressed as

$$e^{j\omega t} = \cos(\omega t) + j\sin(\omega t)$$

This representation simplifies many signal processing operations, including frequency translation, filtering, and phase estimation. By working in the complex domain, it becomes easier to analyze and compensate for impairments such as carrier frequency offset and phase noise.

2.3 Carrier Synchronization

Carrier synchronization is a fundamental requirement in digital communication systems, where phase and frequency mismatches between the transmitter and receiver oscillators must be estimated and corrected to enable reliable symbol detection [16]. In practical systems, the received signal is affected by both carrier frequency offset and phase noise. A commonly used baseband representation of the received signal is given by

$$r(t) = s(t)e^{j(2\pi\Delta f t + \phi(t))} + n(t)$$

where $s(t)$ denotes the transmitted signal, Δf is the carrier frequency offset, $\phi(t)$ represents phase noise, and $n(t)$ is additive noise. A detailed discussion of carrier synchronization methods can be found in Chapter 5 of Proakis and Salehi [16].

The exponential term introduces a time-varying phase rotation that distorts the received constellation and significantly degrades demodulation performance. Without proper synchronization, symbol detection becomes unreliable, particularly in high-frequency systems where phase errors accumulate rapidly. Consequently, accurate estimation and compensation of both frequency and phase errors are critical.

2.4 Phase-Locked Loop (PLL)

A PLL is a feedback control system that synchronizes the phase and frequency of a local oscillator to a reference signal. It operates by continuously minimizing the phase difference between the input signal and the oscillator output.

The fundamentals of phase-locked loop operation and basic loop dynamics are introduced in Chapter 1 of Best [17]. The system consists of a phase detector, a loop filter, and a voltage-controlled oscillator (VCO) as shown in Figure 2.2. The phase detector compares the phase of the incoming signal with that of the oscillator output and generates an error signal proportional to their difference. This error signal is processed by the loop filter, which determines the dynamic behavior of the system by attenuating high-frequency components and shaping the loop response. The filtered signal is then used to control the VCO, which adjusts its frequency accordingly.

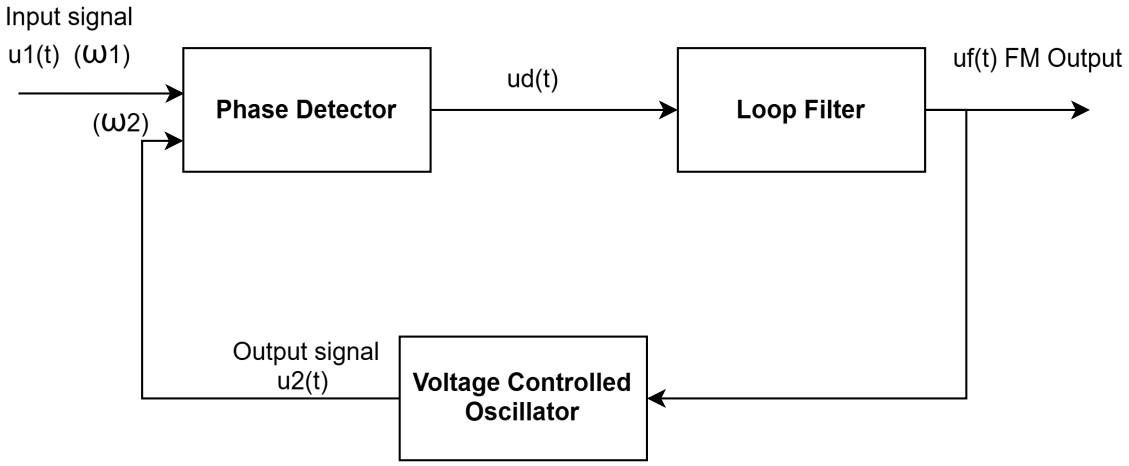


Figure 2.2: Basic block diagram of a phase-locked loop (PLL)

Through this feedback mechanism, the PLL converges to a locked state in which the output signal tracks the reference signal in both phase and frequency. The behavior of the PLL can be analyzed using linear control theory. A commonly used linearized transfer function is

$$H(s) = \frac{K_d K_v F(s)}{s + K_d K_v F(s)}$$

where K_d is the phase detector gain, K_v is the VCO gain, and $F(s)$ represents the loop filter transfer function.

The loop filter plays a central role in determining the stability, noise rejection capability, and dynamic response of the PLL. A commonly used implementation is a second-order proportional-integral (PI) loop filter, which provides a good balance between tracking performance and stability.

The loop filter parameters are typically selected based on the desired loop bandwidth and damping factor. The loop bandwidth determines how rapidly the PLL can respond to frequency and phase variations while filtering measurement noise. A

wider loop bandwidth improves tracking of carrier frequency offsets and phase fluctuations but increases sensitivity to noise. Conversely, a narrower loop bandwidth provides better noise suppression at the expense of slower convergence.

The damping factor controls the transient response of the loop. An underdamped system may exhibit oscillations and overshoot, whereas an overdamped system results in slower settling. Therefore, careful selection of the loop bandwidth and damping factor is required to achieve stable operation, fast acquisition, and reliable carrier tracking.

In the context of the proposed PN-coded pilot synchronization system, the PLL provides the fundamental mechanism for tracking carrier phase and frequency. However, due to the digital nature of the implementation and the constraints of FPGA-based systems, the classical analog PLL is replaced by a DPLL, which retains the same principles while enabling efficient discrete-time implementation.

2.5 Digital Phase-Locked Loops (DPLL)

DPLLs are widely used in communication receivers for carrier phase and frequency synchronization. DPLLs extend the concept of PLLs into the discrete-time domain [18]. Compared to analog PLLs, DPLLs offer improved robustness, precise coefficient control, reproducibility, and straightforward integration into FPGA and ASIC implementations. In a DPLL, all operations are performed in discrete time using sampled signals and fixed-point arithmetic.

The DPLL implemented in this work consists of four main components: a phase detector, a proportional-integral (PI) loop filter, an NCO, and a feedback path. The objective of the loop is to estimate and compensate the residual CFO and phase error between the received signal and the locally generated carrier reference.

2.5.1 Discrete-Time System Model

Let $\theta_i[n]$ denote the phase of the received signal and $\theta_o[n]$ denote the phase generated by the NCO. The phase detector computes the phase error

$$e[n] = \theta_i[n] - \theta_o[n]. \tag{2.1}$$

For small phase errors, the phase detector can be approximated as a linear gain element

$$u_d[n] = K_d e[n], \tag{2.2}$$

where K_d is the phase detector gain.

The phase detector output is processed by a proportional-integral (PI) loop filter. The loop filter generates a frequency correction signal according to

$$u_c[n] = K_p e[n] + K_i \sum_{k=0}^{n} e[k], \tag{2.3}$$

where K_p and K_i are the proportional and integral gains, respectively.

The proportional term provides fast response to phase variations, while the integral term accumulates residual phase error and enables long-term correction of carrier frequency offset.

2.5.2 Numerically Controlled Oscillator

The NCO replaces the VCO used in analog PLLs. The NCO acts as a discrete-time phase integrator and updates its phase according to

$$\theta_o[n] = (\theta_o[n-1] + K_0 u_c[n-1]) \tag{2.4}$$

where K_0 denotes the NCO gain. The accumulated phase is then used to generate sine and cosine reference signals for carrier correction. In the FPGA implementation, the phase accumulator is represented using fixed-point arithmetic, and only the most significant bits are used to address the sine and cosine lookup tables in order to reduce memory requirements.

2.5.3 Loop Bandwidth and Stability

The dynamic behavior of the DPLL is primarily determined by the loop bandwidth and damping factor. A wider loop bandwidth allows the loop to track larger frequency variations and faster phase changes but increases sensitivity to noise. Conversely, a narrower loop bandwidth improves noise rejection at the expense of slower convergence.

For a second-order loop, the relationship between the loop bandwidth B_L, natural frequency ω_n, and damping ratio ξ is given by

$$B_L = \frac{\omega_n}{2}\left(\xi + \frac{1}{4\xi}\right). \tag{2.5}$$

A damping factor of $\xi = 0.707$ is frequently chosen because it provides a near-Butterworth response with a good compromise between stability and settling time.

The loop filter constants τ_1 and τ_2 can be determined from

$$\omega_n = \sqrt{\frac{K_0 K_d}{\tau_1}}, \tag{2.6}$$

and

$$\xi = \frac{\omega_n \tau_2}{2}, \tag{2.7}$$

where K_d is the phase detector gain and K_0 is the NCO (or VCO) gain.

For stable loop operation, the lock range $\Delta\omega_L$ should satisfy

$$\Delta\omega_L \leq K_0 K_d. \tag{2.8}$$

The transfer function of the proportional-integral (PI) loop filter is

$$F(s) = \frac{1 + s\tau_2}{s\tau_1}. \tag{2.9}$$

To implement the loop filter digitally, the continuous-time transfer function is converted into the z-domain using the bilinear transformation

$$s = \frac{2}{T}\left(\frac{1 - z^{-1}}{1 + z^{-1}}\right), \tag{2.10}$$

where T denotes the sampling period.

Substituting (2.10) into (2.9) yields the digital loop filter implementation used in the DPLL. The detailed derivations of the discrete-time Costas loop and carrier recovery equations can be found in [19].

2.6 Numerically Controlled Oscillator (NCO)

The NCO is a key component of the DPLL [20] and in digital frequency synthesis [21, 22, 23] and is responsible for generating the phase and frequency correction signals required for carrier synchronization. Unlike an analog Voltage-Controlled Oscillator (VCO), the NCO operates entirely in the digital domain and is implemented using arithmetic operations and memory resources, making it well suited for FPGA realization.

The NCO architecture is based on a phase accumulator and LUT/CORDIC-based sine generation shown in Figure 2.3, as described in the Intel FPGA NCO IP Core User Guide [21]. An NCO generates discrete-time sinusoidal signals by accumulating phase increments and mapping the resulting phase to amplitude values [24, 25, 20].

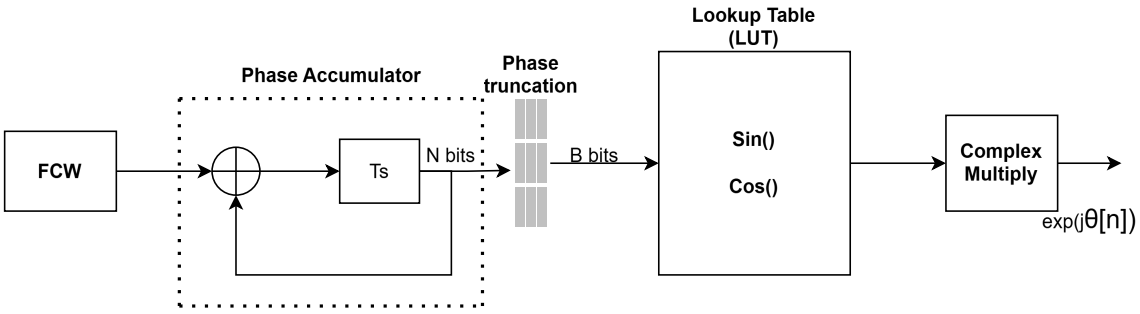


Figure 2.3: NCO architecture used in the DPLL [20]

Figure 2.3 illustrates the operation of the NCO within the carrier synchronization loop. The frequency control signal generated by the loop filter is accumulated to

produce the NCO phase. This phase accumulator acts as a discrete-time integrator, continuously updating the phase according to the estimated carrier frequency offset and phase error. The accumulated phase is subsequently used to generate cosine and sine reference signals through a lookup table (LUT)-based Direct Digital Synthesizer (DDS) architecture. These reference signals drive the complex rotator, which compensates for carrier impairments by multiplying the received signal with the complex conjugate of the NCO output. As a result, the estimated carrier frequency offset and phase error are removed, enabling continuous carrier tracking and maintaining synchronization in the presence of frequency and phase disturbances.

To reduce memory consumption and hardware complexity, the lookup table address is generated using only the most significant bits of the NCO phase accumulator. The least significant bits are omitted when addressing the sine and cosine lookup tables, allowing a smaller LUT implementation while maintaining sufficient phase resolution for carrier synchronization. The truncation process introduces a small quantization error resulting spurious tones in the frequency spectrum [20]. Techniques such as dithering and increased phase resolution are often employed to mitigate these effects. However, truncation's impact is negligible compared to the substantial reduction in memory utilization and FPGA resource consumption. Consequently, this truncation process provides an efficient trade-off between implementation complexity and synchronization performance.

The operation of the NCO is based on the fundamental relationship between frequency and phase:

$$\omega(t) = \frac{d\theta(t)}{dt} \tag{2.11}$$

where $\omega(t)$ denotes the instantaneous angular frequency and $\theta(t)$ represents the signal phase. Integrating the frequency over time yields the phase:

$$\theta(t) = \int_{-\infty}^{t} \omega(\tau) d\tau \tag{2.12}$$

In discrete-time systems, this integration is implemented as an accumulation operation. Therefore, the NCO phase accumulator can be expressed as

$$\theta[n] = \theta[n-1] + \omega[n] \tag{2.13}$$

where $\omega[n]$ is the frequency control word generated by the loop filter and $\theta[n]$ is the accumulated NCO phase. Equation (2.13) shows that the NCO acts as a discrete-time integrator, converting frequency corrections into phase corrections.

The core of the NCO is a phase accumulator, which updates its phase value at each clock cycle according to a frequency control word (FCW). The relationship between the output frequency and the control word is given by

$$f_{out} = \frac{f_{clk} \cdot \mathrm{FCW}}{2^N}$$

where f_{clk} is the system clock frequency and N is the bitwidth of the phase accumulator. This formulation allows very fine frequency resolution, determined by the number of bits in the accumulator.

The accumulation process is essential because frequency corresponds to the rate of change of phase. Any residual carrier frequency offset (CFO) estimated by the DPLL is continuously accumulated within the NCO, enabling accurate carrier tracking and phase correction.

Complex Rotation for Carrier Correction

The accumulated phase generated by the NCO is used to perform complex rotation of the received baseband signal. Let the received complex signal be represented as

$$r[n] = I[n] + jQ[n] \tag{2.14}$$

where $I[n]$ and $Q[n]$ denote the in-phase and quadrature components, respectively.

To compensate for carrier frequency offset and phase errors, the received signal is multiplied by the complex conjugate of the NCO output:

$$y[n] = r[n]e^{-j\theta[n]} \tag{2.15}$$

Expanding (2.15) using Euler's identity gives

$$e^{-j\theta[n]} = \cos(\theta[n]) - j\sin(\theta[n]) \tag{2.16}$$

which results in the following in-phase and quadrature outputs:

$$I_{\text{corr}}[n] = I[n]\cos(\theta[n]) + Q[n]\sin(\theta[n]) \tag{2.17}$$

$$Q_{\text{corr}}[n] = Q[n]\cos(\theta[n]) - I[n]\sin(\theta[n]) \tag{2.18}$$

These equations represent a rotation of the received signal constellation by the estimated phase angle. As the DPLL converges, the phase error approaches zero and the received constellation becomes aligned with its ideal reference position.

In practical implementations [26], the phase accumulator output is converted into sinusoidal values using either a lookup table (LUT) or an algorithmic approach such as the CORDIC algorithm [19]. The lookup table method stores precomputed sine and cosine values, providing fast and deterministic output. However, it requires memory resources, which can become significant for high-resolution systems. Alternatively, CORDIC-based implementations [27, 28, 29] compute trigonometric functions iteratively, reducing memory usage at the cost of increased computational latency.

Modern FPGA implementations, such as those described in Intel's NCO IP core documentation [21], support multiple architectures including large ROM, small ROM,

and CORDIC-based designs [19]. The small ROM architecture reduces memory requirements by exploiting waveform symmetry, while maintaining acceptable spectral purity. These design trade-offs are critical in hardware-constrained systems.

The flexibility, precision, and digital nature of the NCO make it highly suitable for carrier synchronization in FPGA-based receivers. In the proposed system, the NCO serves as the digitally controlled oscillator within the DPLL, enabling accurate frequency and phase tracking. The performance of the NCO is influenced by factors such as phase resolution, quantization error, and spectral purity. Phase truncation can introduce spurious tones, which must be minimized through proper design techniques. Advanced architectures improve performance by optimizing phase accumulation and waveform generation.

2.7 Digital Filtering in DPLL : FIR Filters

Finite Impulse Response (FIR) filters are commonly used in digital systems due to their inherent stability and linear phase characteristics [30]. An FIR filter computes its output as a weighted sum of current and past input samples:

$$y[n] = \sum_{k=0}^{M} b_k x[n-k] \tag{2.19}$$

where b_k are the filter coefficients and M is the filter order.

A simple two-tap FIR filter can be expressed as:

$$y[n] = \frac{1}{2}(x[n] + x[n-1]) \tag{2.20}$$

This filter provides basic low-pass filtering and is useful for reducing high-frequency noise within the DPLL loop. Unlike infinite impulse response (IIR) filters, FIR filters do not rely on feedback, which guarantees stability [31]. Their linear phase response ensures that all frequency components experience the same delay, preserving signal waveform characteristics. These properties make FIR filters particularly suitable for applications such as matched filtering and digital loop filtering in DPLL systems.

FIR filters can be integrated into the loop to improve noise suppression and stability. However, they introduce additional delay, which can negatively impact stability margins. Therefore, a balance must be achieved between filtering performance and loop dynamics.

2.8 PN Sequences and Coded Pilots

PN sequences are deterministic sequences that exhibit noise-like statistical properties [32, 7]. Despite being generated algorithmically, they closely resemble random signals in terms of spectral distribution and correlation behavior. These properties make them highly suitable for synchronization, channel estimation, and spread-spectrum communication systems.

2.8.1 Maximal-Length Sequences (m-Sequences)

Maximal-length sequences (m-sequences) are a class of PN sequences generated using linear feedback shift registers (LFSRs) [7]. For a register of length m, the sequence length is:

$$N = 2^m - 1 \tag{2.21}$$

m-sequences are widely used due to their good autocorrelation properties. The autocorrelation function is given by:

$$R_{xx}(\tau) = \begin{cases} 1, & \tau = 0 \\ -\frac{1}{N}, & \tau \neq 0 \end{cases} \tag{2.22}$$

This results in a sharp peak at zero lag and very low sidelobes elsewhere, enabling reliable detection even under low signal-to-noise ratio (SNR) conditions.

2.8.2 Gold Sequences

Gold sequences are constructed by combining two preferred m-sequences. They provide a larger set of sequences with bounded cross-correlation properties, making them suitable for multi-user environments such as CDMA systems.

The cross-correlation of Gold sequences is three-valued:

$$R_{xy}(\tau) \in \{-1, -t(m), t(m) - 2\} \tag{2.23}$$

where $t(m)$ depends on the register length m.

Gold sequences offer a trade-off between autocorrelation and cross-correlation performance, making them useful for systems requiring multiple orthogonal or -orthogonal codes.

2.8.3 Zadoff–Chu (ZC) Sequences

Zadoff–Chu sequences are complex-valued sequences with constant amplitude and ideal periodic autocorrelation properties [33]. They are defined as:

$$x(n) = \exp\left(-j\frac{\pi q n(n+1)}{N}\right), \quad 0 \leq n < N \tag{2.24}$$

where N denotes the sequence length (typically chosen as a prime number) and q represents the root index, which is selected to be relatively prime to N.

ZC sequences exhibit several important properties that make them highly suitable for synchronization applications. First, they have constant amplitude in the time domain, which reduces the peak-to-average power ratio (PAPR) and improves power

efficiency in transmission. Second, they possess ideal cyclic autocorrelation characteristics, meaning that the autocorrelation is zero for all non-zero cyclic shifts, resulting in excellent detection performance in noisy environments. Finally, cyclically shifted versions of a ZC sequence remain orthogonal to each other, enabling reliable multi-user or multi-pilot scenarios without mutual interference.

The cyclic autocorrelation is:

$$R_{xx}(\tau) = \begin{cases} 1, & \tau = 0 \\ 0, & \tau \neq 0 \end{cases} \tag{2.25}$$

Due to these properties, ZC sequences are extensively used in modern communication systems such as LTE and 5G for synchronization signals (e.g., Primary Synchronization Signal, PSS) and random access preambles.

2.8.4 PN Sequences as Pilot Signals

In communication systems, PN sequences are widely used as pilot signals for several critical receiver functions, including frame synchronization, channel estimation, timing recovery, and frequency offset estimation [34, 35]. Their deterministic structure enables reliable correlation-based detection at the receiver, where the incoming signal is matched against a locally generated reference sequence. This operation can be expressed mathematically as

$$y(\tau) = \sum_{n=0}^{N-1} r(n)\, x(n-\tau) \tag{2.26}$$

where $r(n)$ represents the received signal and $x(n)$ denotes the known pilot sequence. The resulting correlation output $y(\tau)$ exhibits a pronounced peak at the correct alignment, which is used to estimate timing and synchronization parameters in the system.

The strong autocorrelation peak enables precise timing acquisition even in noisy environments. Therefore, PN sequences and coded pilots play a critical role in modern digital communication systems. While m-sequences provide excellent autocorrelation, advanced sequences such as ZC offer ideal correlation and constant amplitude properties, making them indispensable in contemporary wireless standards. The choice of sequence depends on system requirements, including synchronization accuracy, multi-user interference, and implementation complexity.

2.9 Matched Filtering and Correlation

Matched filtering is a signal processing technique used to maximize the output signal-to-noise ratio for a known signal in the presence of noise. The impulse response of a matched filter is defined as the complex-conjugated version of the transmitted signal [36], such that the filtering operation effectively performs correlation with

the received signal, producing a distinct peak at the correct detection instant and maximizing the signal-to-noise ratio.

$$h(t) = s^*(-t)$$

When the received signal is passed through the matched filter, the output exhibits a peak when the signal aligns with the reference sequence. This property is particularly useful for detecting PN sequences used as pilot signals. Matched filtering is therefore a fundamental component in synchronization systems that rely on known reference signals.

2.10 Low-Speed ADC-Based Synchronization

In ultra-wideband systems, sampling the entire signal bandwidth requires ADC sampling rates on the order of tens of gigasamples per second. While such ADCs are available for laboratory and specialized applications, their high power consumption and the resulting digital data rates present significant challenges for practical wideband receiver implementations. An alternative approach is to extract a narrowband pilot signal and process it using a low-speed ADC.

This approach significantly reduces the ADC sampling rate requirements and digital processing complexity while still enabling carrier synchronization. By focusing on the pilot signal, the receiver can estimate frequency and phase offsets without requiring full-bandwidth sampling. This concept is particularly relevant for the proposed PN-coded pilot system, where synchronization is achieved efficiently using a reduced sampling rate. However, the synchronization performance depends on reliable detection of the low-power PN-coded pilot signal, which becomes increasingly challenging in the presence of strong noise, phase noise, and carrier frequency offsets.

3

System Specification

The proposed thesis system targets a sub-THz wideband communication link with bandwidths ranging from 10 to 40 GHz, where a low-power PN-coded pilot is embedded beneath the data signal to enable carrier synchronization. To avoid digitizing the ultra-wideband data signal, a hybrid analog–digital architecture is adopted [1, 2]. The wideband data path remains analog, while only the narrowband pilot is extracted, digitized using a low-speed ADC, and processed on an FPGA.

The FPGA performs PN correlation, carrier frequency and phase estimation, and the resulting correction signals are used to control the receiver local oscillator. The system will be validated through both simulations and hardware experiments, demonstrating reliable synchronization without digitizing the full wideband signal.

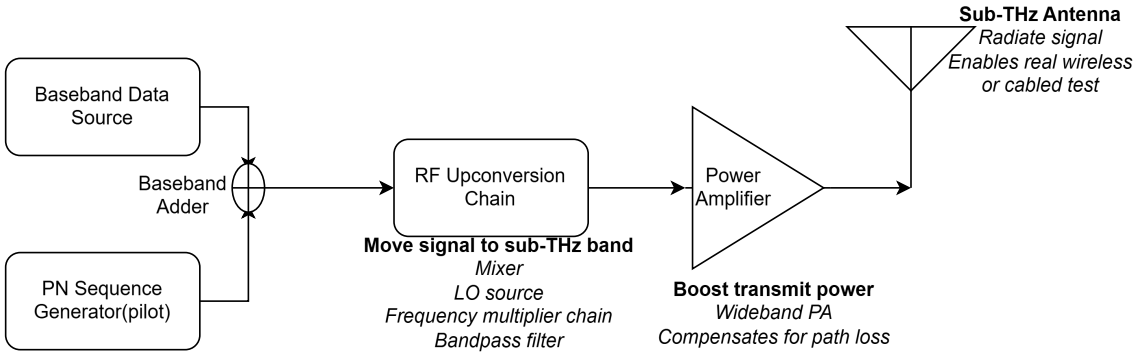


Figure 3.1: Transmitter architecture

Figure 3.1 and Figure 3.2 show the transmitter and receiver architectures used in this thesis. At the transmitter, a high-speed baseband data signal is combined with a low-power PN-coded pilot generated by a PN sequence generator. The combined signal is then upconverted to the sub-THz frequency band through an RF upconversion chain consisting of a mixer, a local oscillator source, frequency multiplier chain, and bandpass filtering. A wideband power amplifier boosts the signal to compensate for high propagation loss, and the signal is radiated using a sub-THz antenna.

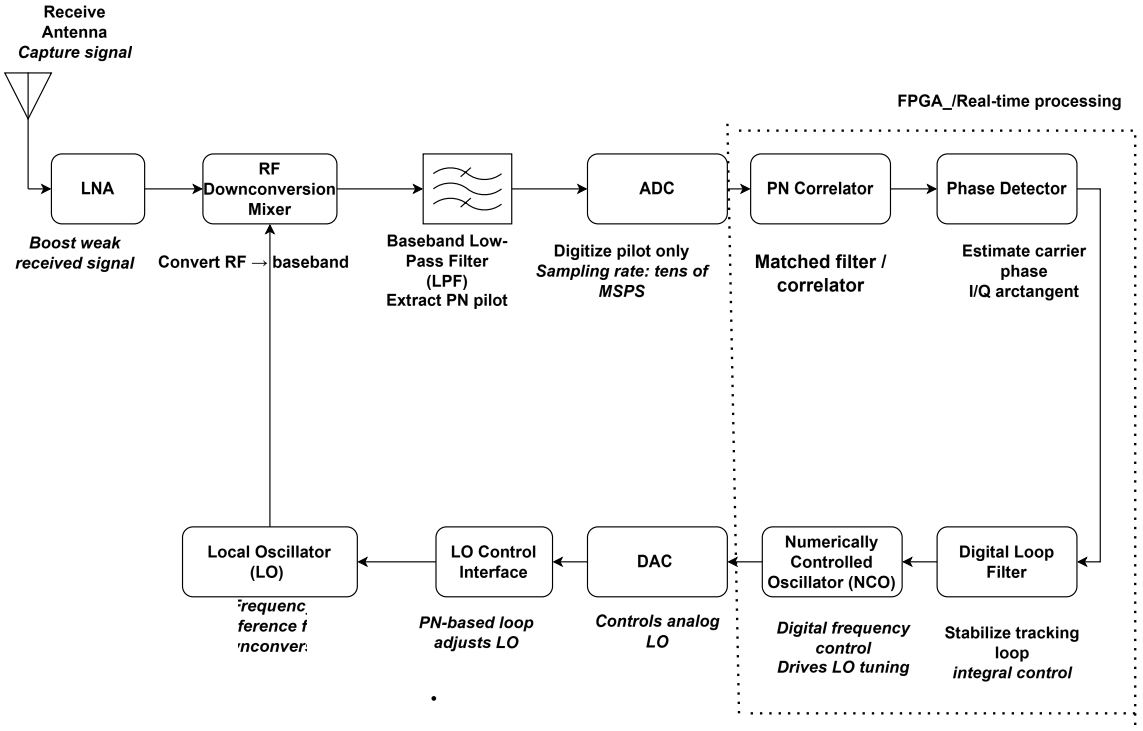


Figure 3.2: Receiver architecture

At the receiver, the incoming signal is captured by the antenna, amplified by a low-noise amplifier, and downconverted to baseband using a single-sideband mixer driven by a controllable local oscillator. The wideband baseband signal is then split, with a narrowband low-pass filter extracting the PN-coded pilot, which is digitized using a low-speed ADC. The digitized pilot is processed in real time on an FPGA using PN correlation, phase estimation, and carrier frequency offset estimation. A digital tracking loop generates correction signals that are converted to analog and applied to the local oscillator, enabling hybrid analog–digital carrier synchronization without digitizing the wideband data signal.

The component list for the complete transceiver chain is divided into four sections as explained in below Table 3.1 and Table 3.2: transmitter components for signal generation and modulation, receiver RF and analog components for signal capture and conditioning, digital/FPGA components for processing and control, and measurement and support components for testing, calibration, and system support.

Table 3.1: Transmitter and Receiver Component List

Component	Domain	Purpose / Function
Transmitter (TX)		
Baseband Data Source	FPGA	Generates high-speed data stream for transmission and BER evaluation.
PN Sequence Generator	FPGA	Generates low-power PN-coded pilot for synchronization.
Baseband Adder	FPGA	Combines pilot and data signal below data power level.
High-Speed DAC	Mixed-Signal	Converts digital baseband to analog RF signal.
Analog RF Front-End	RF	Performs upconversion, filtering, and amplification to sub-THz band.
Channel Interface (TX)	RF	Interfaces with transmission medium using waveguides and attenuators.
Receiver (RX)		
Channel Interface (RX)	RF	Receives signal from controlled channel environment.
Analog RF Front-End	RF	Performs LNA, filtering, and SSB downconversion.
Local Oscillator (LO)	RF	Provides reference for downconversion and synchronization.
Bandpass Filter (Pilot)	Analog	Extracts PN pilot and suppresses interference.
Low-Speed ADC	Mixed-Signal	Digitizes pilot signal at reduced sampling rate.

Table 3.2: FPGA, Simulation, and Measurement Components

Component	Domain	Purpose / Function
FPGA / Digital Processing		
FPGA Platform	Digital	Performs real-time processing for synchronization and tracking.
PN Correlator	Digital	Extracts pilot using correlation/matched filtering.
Phase Estimator	Digital	Estimates carrier phase.
CFO Estimator	Digital	Tracks carrier frequency offset over time.
Digital Loop Filter	Digital	Ensures stable and fast loop convergence.
NCO	Digital	Generates frequency/phase correction signal.
DAC (LO Control)	Mixed-Signal	Converts digital control to analog LO tuning voltage.
Simulation and Measurement		
MATLAB	Simulation	Models PN correlation, CFO, and phase noise behavior.
Spectrum Analyzer	Measurement	Verifies spectrum and pilot power.
Oscilloscope	Measurement	Observes baseband and pilot signals.
BER Tester	Measurement	Evaluates system performance (BER $< 10^{-3}$).

4

Method

This chapter describes the overall methodology used to design, simulate, and implement the proposed PN-coded pilot–assisted carrier synchronization system. The approach follows a systematic progression from system-level modeling in MATLAB to real-time hardware implementation on an FPGA platform integrated with RF front-end components.

The methodology consists of three main stages: MATLAB-based system modeling and verification, FPGA-based digital implementation, and hardware integration with real-time experimental validation. Each stage is designed to validate specific aspects of the system, ensuring correctness before proceeding to the next level of complexity. This structured design approach enables early identification of design issues and facilitates a reliable transition from theoretical modeling to practical realization.

4.1 MATLAB-Based System Modeling

The initial phase of this work focused on developing a comprehensive MATLAB simulation framework to model and evaluate the proposed PN-coded pilot synchronization architecture. The objective of the simulation environment was to establish a reference model for algorithm development, investigate synchronization behavior under realistic operating conditions, and support the transition to fixed-point FPGA implementation.

The transmitter model generates a wideband communication signal consisting of a data-bearing waveform combined with a low-power PN-coded pilot sequence. The pilot is embedded within the transmitted signal to provide synchronization information while minimizing its impact on the overall signal power. Several PN sequence configurations, including different sequence lengths and pilot power levels, were considered to study their influence on synchronization performance.

To emulate practical wireless communication scenarios, channel impairments such as CFO, phase offset, phase noise, and additive noise were incorporated into the simulation model. These impairments represent the primary synchronization challenges encountered in wideband and sub-THz communication systems.

At the receiver, the synchronization process begins with matched filtering using a locally stored replica of the PN sequence. Correlation between the received signal and the known pilot sequence is used to identify synchronization peaks and recover

the pilot component from the received waveform. The extracted pilot signal is subsequently utilized for carrier phase estimation, carrier frequency offset estimation, and carrier recovery through a DPLL.

The MATLAB framework was also used to investigate the influence of key system parameters, including PN sequence length, pilot-to-signal power ratio, loop filter characteristics, sampling rate, and fixed-point word-length selection. In addition, the simulation environment served as a reference platform for validating synchronization algorithms prior to hardware implementation.

To facilitate FPGA realization, fixed-point models of the matched filter, phase estimator, carrier recovery loop, and NCO were developed and verified in MATLAB. This enabled early evaluation of quantization effects, finite word-length constraints, scaling requirements, and numerical precision before deployment on FPGA hardware. The resulting simulation framework therefore provided a consistent development and verification platform spanning both algorithm design and hardware implementation stages.

4.2 FPGA-Based Digital Implementation

Following MATLAB-based verification, the synchronization system is implemented on an FPGA platform using the Intel Arria V device [37]. The FPGA pin configuration and ADC/DAC interface mapping are based on the Arria V GX Starter Kit documentation [38], together with the HSMC data conversion reference manual [39, 40]. Additional board-level interface details are provided in the Terasic development board documentation [41]. The design is developed using *Quartus Prime Standard Edition 18.0*, and functional verification is carried out using ModelSim-Altera.

The FPGA architecture comprises several key processing blocks as shown in Figure 4.1. A NCO is used for signal generation, while a PN sequence generator produces the pilot signal. These signals are combined and processed through a digital mixer to perform I/Q modulation. A Finite Impulse Response (FIR) filter is employed for matched filtering, enabling reliable pilot extraction. Carrier phase estimation is achieved using a phase detector, and the resulting phase error is smoothed using a digital loop filter to ensure stable operation.

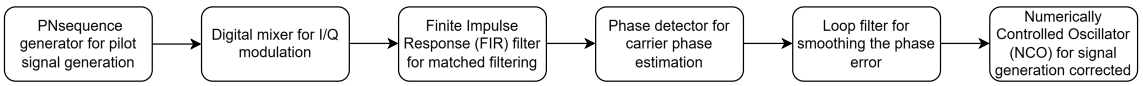


Figure 4.1: FPGA-Based Digital Implementation Block Diagram

The overall design follows a modular approach, where each functional block is individually verified prior to system-level integration. Fixed-point arithmetic is adopted throughout the implementation to optimize resource utilization and ensure efficient real-time processing.

The FPGA interfaces with external data converters through a Terasic development board equipped with an AD9254, a 14-bit 150 MS/s Analog-to-Digital Converter

(ADC), and a DAC5672, a 14-bit 275 MS/s Digital-to-Analog Converter (DAC). The ADC interface is implemented using the AD9254 high-speed converter [42], while the DAC stage is based on the DAC5672 14-bit high-speed digital-to-analog converter [43]. The pin-level configuration and HSMC mapping follow the Terasic data conversion interface documentation [44]. The ADC is used to capture the down-converted baseband signal containing the embedded pilot, while the DAC outputs the digitally generated correction signal from the FPGA.

The interface design incorporates data alignment and synchronization mechanisms to ensure accurate sampling, along with proper clock domain management to handle different timing requirements. In addition, buffering and pipeline registers are implemented to maintain continuous data flow and support high-speed operation.

4.3 Integration with RF Hardware

For real-time validation, the FPGA system is integrated with RF front-end modules, including mixers, filters, and signal generation equipment. The experimental setup utilizes a Keysight Arbitrary Waveform Generator (AWG) to generate wideband test signals with embedded pilot sequences, while a Keysight digitizer is used to capture the resulting baseband I/Q signals.

This integrated setup enables end-to-end validation of the proposed synchronization method under realistic conditions, allowing the performance of the system to be evaluated in terms of carrier recovery accuracy and robustness against channel impairments.

5

System Design and Validation

A MATLAB-based system model was developed to evaluate the proposed PN-coded pilot synchronization approach under realistic channel conditions. The model includes pilot generation, signal embedding, channel impairments, and matched filtering for pilot detection.

The objective of the simulation is to determine suitable design parameters and validate the feasibility of detecting a low-power pilot signal. The main components of the MATLAB simulation model are summarized below:

- **PN-Based Pilot Signal Design:**

  A pseudo-noise (PN) sequence was generated using a linear feedback shift register (LFSR) and mapped to a binary phase-shift keying (BPSK) representation, resulting in a sequence of± 1 values.

  This PN sequence was used as a pilot signal due to its favorable autocorrelation properties, which enable reliable detection through matched filtering. The pilot signal was scaled and embedded into the transmitted signal at controlled power levels.

- **Pilot Embedding and Power Scaling:**

  The PN-coded pilot was superimposed onto the transmitted signal to form a composite signal consisting of both data and pilot components. The relative power of the pilot signal was adjusted to evaluate detection performance under low signal-to-noise ratio conditions.

  This approach reflects practical scenarios where the pilot signal must remain significantly weaker than the data signal to maintain spectral efficiency.

- **Channel Impairments:**

  To emulate realistic sub-THz communication conditions, the received signal was impaired with CFO and phase noise. These impairments introduce phase rotation and frequency drift, which degrade synchronization performance.

  The inclusion of these effects allows for evaluation of the robustness of the proposed synchronization method under practical conditions.

- **Matched Filter for Pilot Detection:**

  Pilot detection is performed using a matched filter, which maximizes the signal-to-noise ratio for a known sequence. The matched filter is implemented using the time-reversed PN sequence as filter coefficients.

The output of the matched filter produces a correlation peak when the received signal aligns with the PN sequence, enabling reliable detection of the pilot signal even at low power levels.

5.1 Performance Metrics

The performance of the proposed synchronization system is evaluated using several key metrics that quantify detection accuracy, robustness, and reliability under practical impairments.

First, the *correlation peak amplitude* is used to measure the strength of the detected pilot signal after matched filtering. A higher peak indicates stronger alignment between the received signal and the locally generated PN sequence, reflecting improved synchronization performance. Closely related to this is the *peak-to-sidelobe ratio (PSR)*, which represents the ratio between the main correlation peak and the largest sidelobe. A high PSR ensures that the pilot can be clearly distinguished from noise and interference, thereby reducing the probability of false detection.

In addition, the *detection capability at low pilot power* is evaluated to assess the sensitivity of the system. Since the PN-coded pilot is intentionally embedded at a power level below the data signal, reliable detection under low signal-to-noise ratio conditions is critical for practical implementation.

Finally, the *robustness to CFO and phase noise* is analyzed to determine the effectiveness of the synchronization loop in realistic scenarios. CFO introduces a time-varying phase rotation, while phase noise causes random fluctuations in the carrier phase. The ability of the system to accurately estimate and compensate for these impairments directly impacts overall synchronization accuracy and system stability.

- **Evaluation of PN Sequence Polynomials:**

  Different primitive polynomials were tested for generating m-sequences of fixed length. The results showed that altering the polynomial does not change the inherent autocorrelation characteristics of the sequence.

  Thus, for a fixed sequence length, all m-sequences exhibit identical peak-to-sidelobe performance, confirming that polynomial selection does not provide performance gains in this context.

- **Comparison of Pilot Sequence Types:**

  For a fixed sequence length of $N = 31$, multiple candidate sequences were evaluated, including m-sequences, Gold codes, and ZC sequences. Among these options, the ZC sequence demonstrated the best overall performance due to its ideal periodic autocorrelation properties and constant amplitude characteristics, which make it particularly robust for synchronization applications and reduce sensitivity to interference and timing offsets compared to the other sequence types.

- **ZC Sequence Performance:**

  The ZC sequence achieved a significantly higher peak-to-sidelobe ratio of approximately 400 when compared to m-sequences and Gold codes. This improvement is primarily attributed to its ideal correlation properties, which produce a dominant and sharply defined correlation peak with near-zero sidelobes. As a result, the sequence exhibits improved synchronization robustness, as the clear peak structure reduces the probability of false detection in noisy environments.

  In addition, the low sidelobe levels contribute to enhanced CFO estimation stability, since the correlation output is less sensitive to interference and noise fluctuations. These characteristics make ZC sequences particularly suitable for pilot-based synchronization in wideband communication systems, especially in scenarios where the pilot signal is transmitted at low power relative to the data signal.

5.2 Pilot-Based Synchronization Power study

The robustness of the ZC sequence was further evaluated under reduced pilot power levels for a fixed sequence length of $N = 31$. The results are summarized in Table 5.1. At –10 dB, the pilot remained dominant and CFO estimation was highly stable. At –20 dB, acceptable performance was observed with moderate degradation. At –30 dB, the pilot signal was significantly buried under data, resulting in unstable CFO estimates.

Table 5.1: Performance of Zadoff–Chu Pilot Under Different Pilot Power Levels

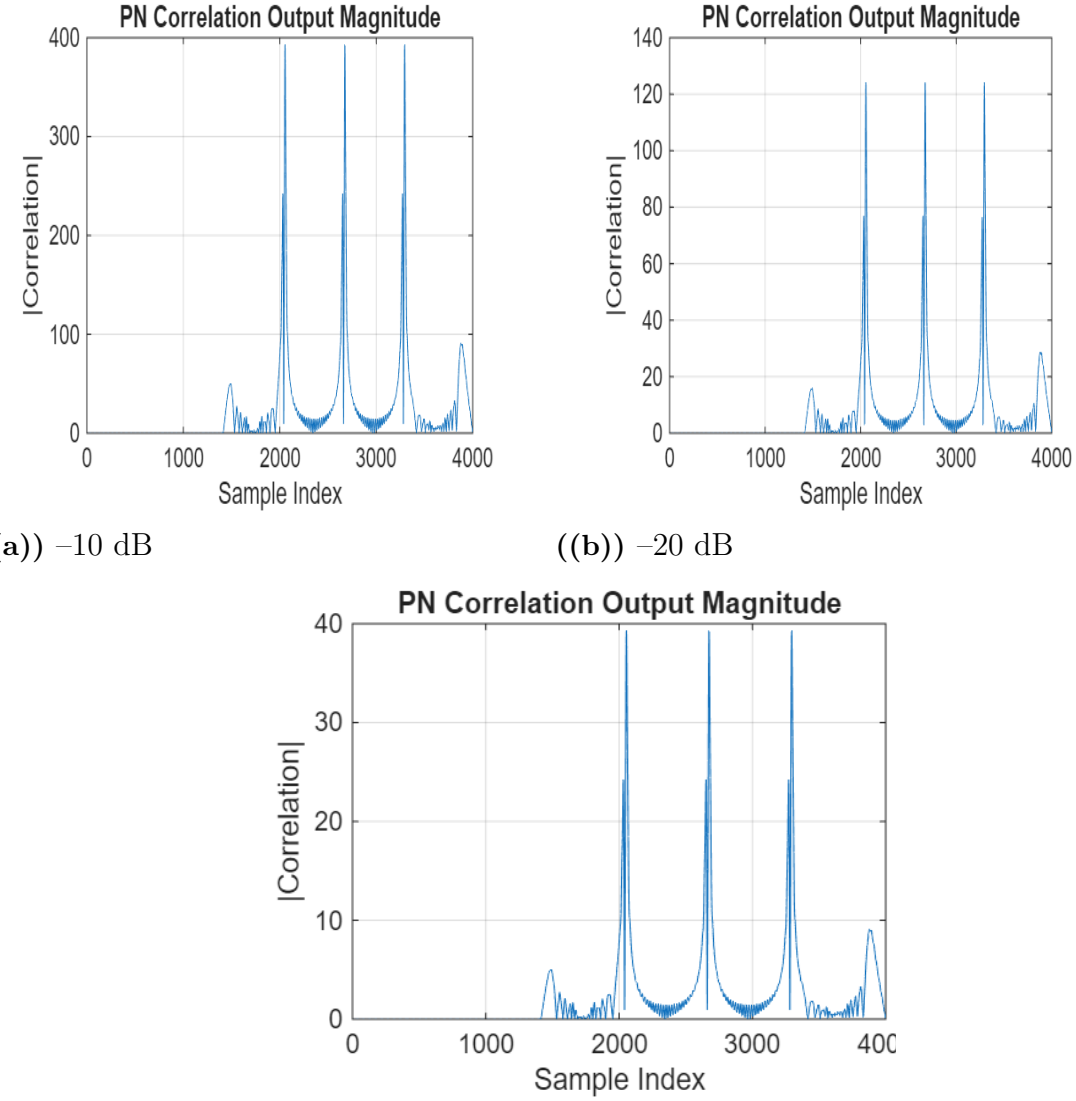
Pilot Power	Peak Ratio	CFO Est. 1	CFO Est. 2	Observation
–10 dB	∼400	99.38	98.98	Strong peak and highly stable estimation.
–20 dB	∼120	95.60	105.23	Moderate degradation while maintaining reliable detection.
–30 dB	∼40	117.53	55.08	Significant degradation due to low pilot power.

Figure 5.1 illustrates the matched filter output for different pilot power levels. As the pilot power decreases, the amplitude of the correlation peak is reduced. However, the peak remains distinguishable from the sidelobes even at low power levels. This demonstrates that the PN sequence provides sufficient processing gain to enable detection of weak pilot signals.

However, at very low pilot power levels (e.g., –30 dB), the pilot signal becomes significantly masked by the data signal, which can degrade correlation performance and lead to unstable frequency estimation when short PN sequences($N = 31$) are used. This limitation can be effectively mitigated by increasing the length of the PN sequence($N = 511$), thereby improving the processing gain and enhancing the effective signal-to-noise ratio at the output of the matched filter. Consequently,

reliable detection of low-power pilots is achievable even at power levels below –30 dB.

It is important to note that the example shown in Figure 5.2 with a –10 dB pilot power is primarily for visualization purposes, as such a relatively strong pilot introduces noticeable interference to the baseband data signal. In practical implementations, significantly lower pilot power levels (e.g., below –30 dB) combined with longer PN sequences are preferred to minimize data distortion while maintaining robust synchronization performance.

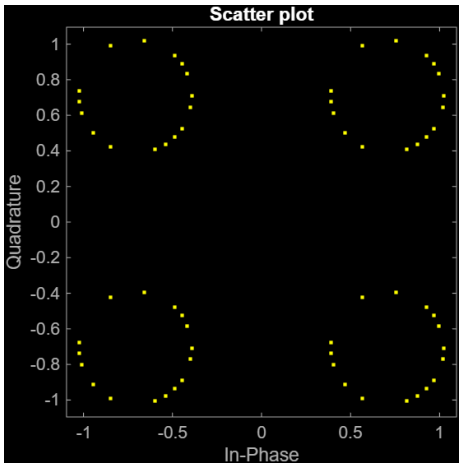


((a)) –10 dB

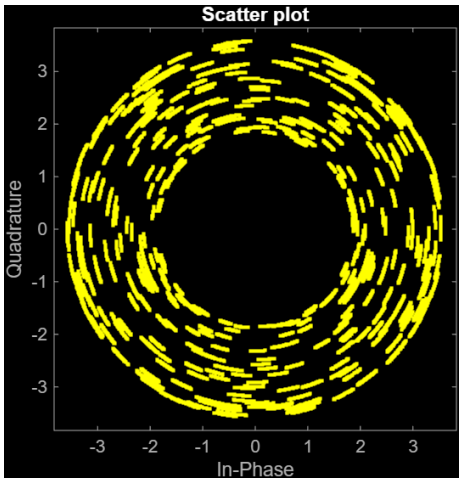
((b)) –20 dB

((c)) –30 dB

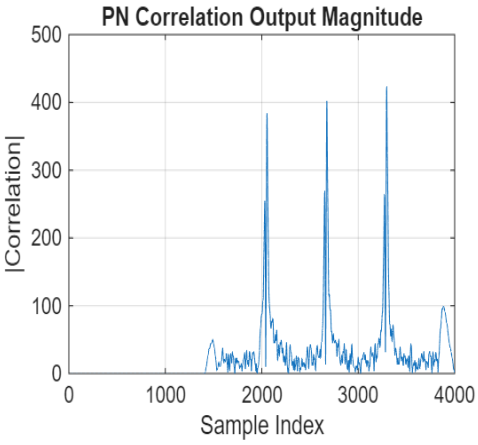
Figure 5.1: Matched filter output showing correlation peak for Zadoff–Chu sequence under different pilot power levels



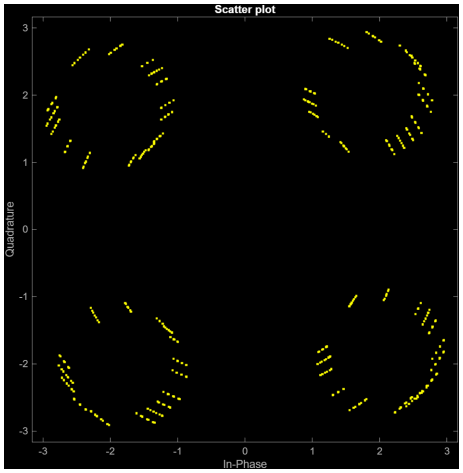
((a)) Tx



((b)) Rx



((c)) Matched Filter output



((d)) Rx corrected

Figure 5.2: Cross-correlation, tx, rx performance using ZC sequence for a given pilot power of –10 and PN-sequence length of 31

Carrier Frequency and Phase Estimation

The impact of CFO and phase noise on the matched filter output is shown in Figure 5.2. These impairments introduce distortion in the correlation peak and reduce detection accuracy.

Despite these effects, the pilot signal remains detectable within a certain range of impairments, indicating robustness of the proposed method. This also highlights the need for a carrier synchronization mechanism, such as a DPLL.

5.3 Residual Phase Tracking

After initial frequency and phase correction, a PLL was used to track residual phase variations.

This loop allows for the continuous correction of phase drift caused by channel impairments or oscillator mismatches.

To enable reliable detection of low-power pilots (e.g., –30 dB), the impact of PN sequence length on CFO estimation was investigated. Increasing the PN length improves processing gain, which enhances detection performance. However, it also introduces limitations in the maximum unambiguous CFO range. For a PN sequence length of $N = 127$, a CFO of 10 kHz was successfully estimated, and the system exhibited stable convergence. In contrast, when a longer PN sequence of $N = 511$ was used, the same CFO of 10 kHz could not be correctly estimated. Instead, the estimated CFO was approximately 200 Hz, indicating aliasing due to the limited unambiguous frequency range.

The maximum unambiguous CFO ($f_{\max}$) that can be reliably estimated is given by

$$f_{\max} = \frac{R_{pn}}{2 \cdot N_{pn}}, \tag{5.1}$$

where R_{pn} is the PN chip rate and N_{pn} is the sequence length. For $R_{pn} = 5$ MHz and $N_{pn} = 511$, the maximum unambiguous CFO is approximately 4.9 kHz. Since the applied CFO of 10 kHz exceeds this limit, ambiguity arises, resulting in incorrect estimation.

To address this limitation, the CFO was reduced to 2 kHz while maintaining the PN length at $N = 511$. Under these conditions, the system successfully estimated the CFO and achieved stable convergence, while also reliably detecting the pilot signal at -30 dB. This demonstrates the fundamental trade-off between processing gain and CFO estimation range: longer PN sequences improve detection performance at low pilot power levels but reduce the allowable CFO range. Therefore, in practical system design, a balance must be achieved between PN length, pilot power, and expected CFO to ensure robust synchronization performance.

The performance of the proposed synchronization system is governed by several interdependent design parameters, including PN sequence length, pilot power, and loop bandwidth. The length and power of the PN-coded pilot jointly determine the effective signal-to-noise ratio (SNR) after low-pass filtering and matched filter correlation. Specifically, increasing the PN sequence length improves the processing gain, thereby enabling reliable detection of very low-power pilots. However, this improvement comes at the cost of reduced update rate, as longer sequences require more time for correlation, which in turn limits the speed of phase and frequency estimation.

The PN sequence rate (chip rate) and sequence length together define how frequently phase updates can be obtained. This update rate directly constrains the achievable loop bandwidth of the carrier recovery loop, since the loop must operate within the rate at which new phase estimates are available. At the same time, the PN chip rate determines the required bandwidth of the low-pass filter for pilot extraction. A wider filter bandwidth allows faster tracking but admits more noise, reducing the effective SNR, while a narrower bandwidth improves noise suppression but limits the tracking speed.

Consequently, increasing the PN sequence length enhances detection performance at low SNR but reduces the maximum unambiguous CFO range and slows loop response. Similarly, pilot power must be carefully chosen to balance detectability and interference with the data signal. The MATLAB analysis confirms that sequences with ideal correlation properties, such as Zadoff–Chu sequences, provide superior performance for pilot-based synchronization. Overall, the system requires joint optimization of PN sequence length, pilot power, chip rate, and loop bandwidth to achieve robust and stable carrier synchronization under practical operating conditions.

5.4 Fixed Point DPLL Implementation

The DPLL is used to estimate and compensate for residual CFO and phase noise. The loop consists of a phase detector, a proportional-integral (PI) loop filter, and a NCO as shown in Figure 5.3. The phase detector computes the phase error between the received signal and the locally generated reference signal.

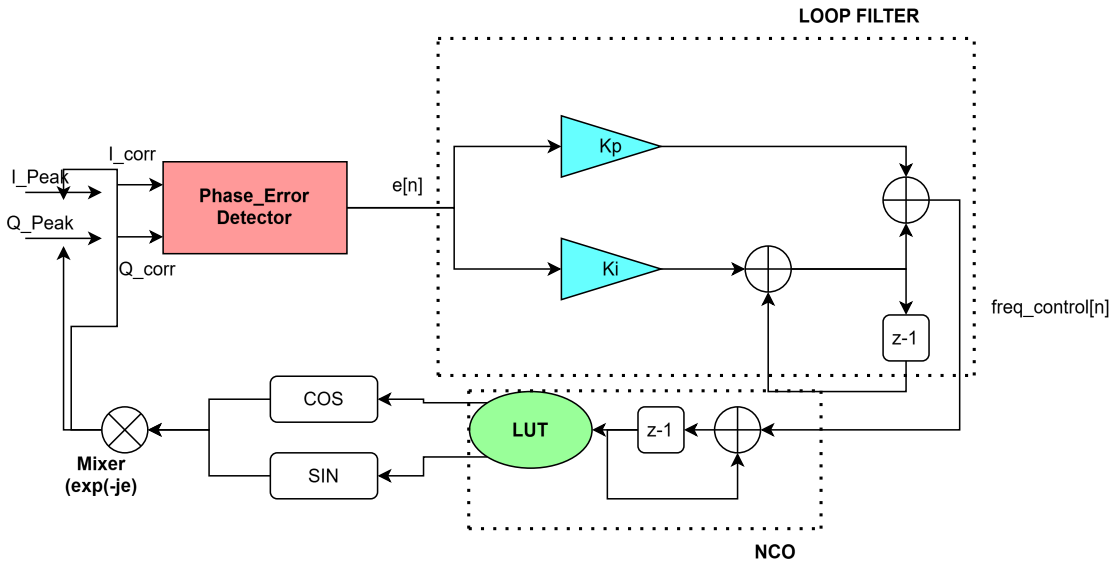


Figure 5.3: Structure of a DPLL with proportional-integral (PI) loop filter and Numerically Controlled Oscillator (NCO)

The proportional path, controlled by gain K_p, provides a fast response to instantaneous phase errors. The integral path, controlled by gain K_i, accumulates phase errors over time and eliminates steady-state frequency offsets. The output of the PI loop filter generates a frequency control signal that drives the NCO. The NCO acts as a digital oscillator and continuously updates the local carrier phase, enabling the DPLL to maintain synchronization with the received signal.

The phase detector generates a phase error signal $e[n]$, which is processed by the PI controller according to

$$i_{\text{term}}[n] = i_{\text{term}}[n-1] + K_i e[n] \tag{5.2}$$

where K_i is the integral gain and i_{term}[n] represents the accumulated phase error.

The loop filter output is then computed as

$$f_{\text{control}}[n] = K_p e[n] + i_{\text{term}}[n] \tag{5.3}$$

where K_p is the proportional gain. The proportional path improves the transient response and lock acquisition speed, whereas the integral path compensates for constant frequency offsets and f_{control} represents NCO frequency control signal.

The resulting control signal drives the NCO, which acts as a discrete-time phase accumulator. The NCO phase is updated according to

$$\phi_{\text{NCO}}[n] = \phi_{\text{NCO}}[n-1] + f_{\text{control}}[n] \tag{5.4}$$

where $\phi_{\text{NCO}}[n]$ denotes the current NCO phase accumulator value.

The accumulation process is essential because frequency corresponds to the rate of change of phase. By integrating the frequency control signal over time, the NCO generates the phase and frequency corrections required for carrier synchronization. The corrected phase is then fed back to the phase detector, forming a closed-loop tracking system capable of compensating both CFO and phase noise.

The DPLL architecture described in this chapter forms the basis of the FPGA implementation presented in Chapter 6. The phase detector, PI loop filter, and NCO were implemented using fixed-point arithmetic in Quartus Prime. The proportional and integral gains were selected to achieve stable loop operation while maintaining adequate tracking performance under carrier frequency offset and phase-noise impairments.

6

FPGA Implementation

This chapter presents the FPGA-based implementation of the proposed receiver architecture and the validation of key signal processing blocks. The FPGA design was implemented using Intel Quartus Prime software with VHDL-based development flow [45]. Debugging and real-time signal observation were performed using the SignalTap Logic Analyzer integrated within the Quartus environment, with a focus on verifying the functionality of the signal chain prior to integrating synchronization algorithms.

A step-by-step validation approach is adopted, beginning with basic signal generation and loopback tests, followed by frequency-selective filtering and complex signal processing. These experiments establish a reliable foundation for the subsequent implementation of PN-based matched filtering and carrier synchronization.

6.1 Receiver Chain Validation

Before interfacing external analog components such as ADC and DAC, an internal loopback test was performed on the FPGA to validate the correctness of the digital signal path. This step ensures that the core logic, clocking architecture, and memory interfacing are functioning correctly in hardware.

The design was implemented and programmed onto the FPGA device, with internal signals monitored using the SignalTap Logic Analyzer, which enables real-time observation of system behavior without requiring external I/O connections.

The test setup included several key functional components: a Phase-Locked Loop (PLL) for stable clock generation, a ROM initialized using a Memory Initialization File (.mif) to store predefined test data, an address counter for sequential ROM access, and dedicated transmit (tx_data) and receive (rx_data) signal paths for verifying internal data flow and system integrity.

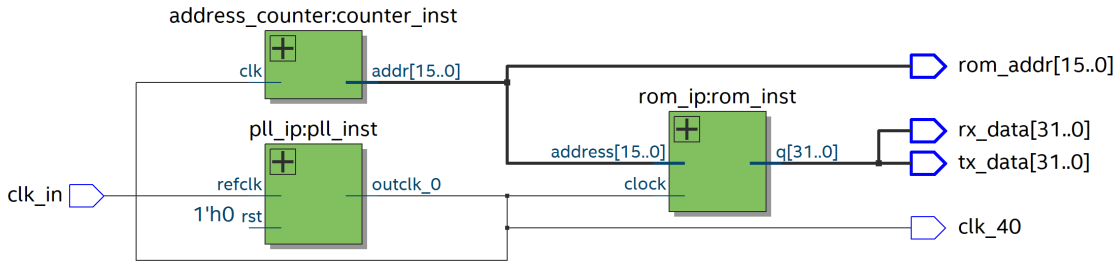


Figure 6.1: Internal Loopback Block Diagram

Figure 6.1 illustrates the architecture of the internal loopback test implemented on the FPGA. The design consists of a PLL, a ROM-based data source, an address generation unit, and internal transmit and receive data paths. The PLL generates a stable 40 MHz clock (clk_40), which drives the entire system. An address counter increments sequentially at each clock cycle and is used to access data stored in the ROM. The ROM is pre-initialized using a Memory Initialization File (.mif), which contains predefined test data.

The output of the ROM is assigned to the transmit data signal (tx_data), which is internally looped back to the receive path (rx_data) without passing through any external interfaces. This configuration allows verification of the internal data path integrity within the FPGA.

A SignalTap Logic Analyzer is connected to key internal nodes, enabling real-time observation of signals such as clk_40, rom_addr, tx_data, and rx_data. This facilitates debugging and validation directly on hardware. The block diagram therefore represents a closed-loop system used to validate clock generation, memory access, and data propagation before integrating external ADC and DAC components.

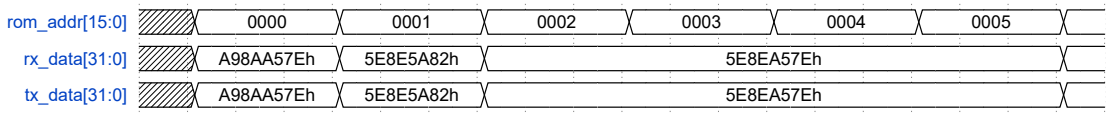


Figure 6.2: SignalTap Capture of Internal Loopback Signals

Figure 6.2 shows the internal FPGA signals captured using the SignalTap Logic Analyzer. The key observations confirm correct system behavior at the digital level. The ROM address (*rom_addr*) increments sequentially, indicating proper operation of the address counter. The transmitted data (*tx_data*) follows the expected waveform defined in the ROM contents, while the received data (*rx_data*) matches the transmitted data exactly, confirming correct internal loopback functionality without data corruption.

Overall, the internal loopback validation confirms that the FPGA design operates correctly at the digital level. The successful verification of clock generation, memory access, and data routing establishes a reliable foundation for subsequent integration with external ADC and DAC components in the full hardware system.

6.2 Signal Generation and Verification

To verify the functionality of the digital signal chain, an internal loopback test was performed using a NCO. The NCO was configured to generate sinusoidal signals at frequencies of 1 MHz.

These signals were transmitted through the DAC and captured using the ADC in a loopback configuration. The received signals were analyzed using SignalTap and external measurement equipment. The results confirmed accurate frequency generation, correct DAC–ADC interfacing, and proper timing alignment within the FPGA. This step validates the functionality of the fundamental signal generation and acquisition chain.

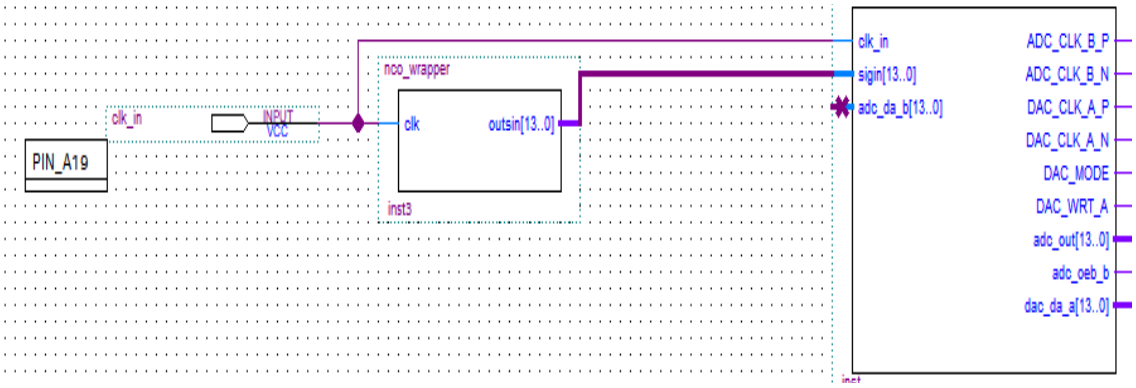


Figure 6.3: Block diagram of NCO-based ADC-DAC loopback system

Figure 6.3 shows the block diagram of the implemented system. The input clock is fed into the NCO wrapper module, which generates a 14-bit sinusoidal output signal at a frequency of 1 MHz based on the FPGA system clock. This signal is then passed to the ADC/DAC interface module, which manages data transfer between the digital and analog domains. The interface is responsible for transmitting the generated digital signal to the DAC and receiving the corresponding feedback from the ADC.

A key aspect of the setup is the loopback configuration, where the DAC output is physically connected back to the ADC input, enabling verification of end-to-end signal integrity. The generated sinusoidal signal (`outsin[13:0]`) is sent to the DAC, while the resulting ADC output (`adc_da_b[13:0]`) is captured and analyzed to confirm correct system operation and validate the integrity of the signal path.

SignalTap was used to capture and observe the transmitted and received signals, as shown in Figure 6.4. The recorded waveforms suggest that the system is behaving as expected across the signal chain. The NCO output shows a clean sinusoidal

waveform in digital form, while the DAC input reflects the MSB-adjusted (unsigned) representation of the generated sine wave. The ADC output closely resembles the transmitted waveform, indicating that loopback operation is functioning as intended. In addition, the observed timing alignment between the transmitted (TX) and received (RX) data suggests that clocking and synchronization are correctly configured throughout the system. Overall, the waveform observations indicate consistent signal propagation through the ADC–DAC path and support the conclusion that the system is operating correctly at a functional level, although full validation would require additional quantitative verification.

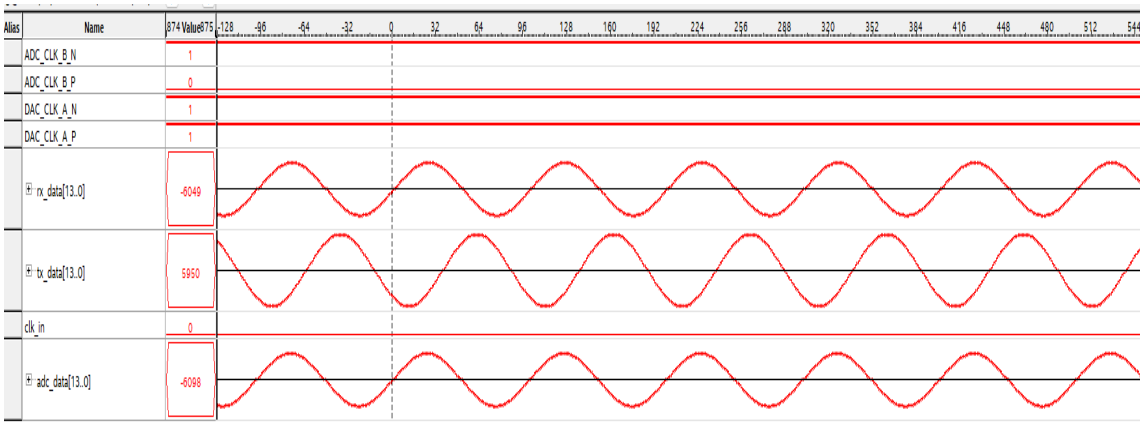


Figure 6.4: Signal Tap view of NCO-based ADC-DAC loopback system

The implementation successfully demonstrates the generation of a 1 MHz sine wave using the NCO IP core, its proper transmission through the DAC and reception via the ADC, and correct handling of signed-to-unsigned data conversion within the digital processing chain. Functional verification using SignalTap further confirms correct system operation. These results validate the end-to-end functionality of the ADC–DAC interface and confirm correct clock configuration, pin mapping, and data alignment across the FPGA-based implementation.

6.3 Frequency Selection via Filtering

To emulate the extraction of a narrowband pilot signal from a composite wideband signal, multiple frequency components were generated and passed through the signal chain. A digital filter was then applied to isolate a lower-frequency component while suppressing higher-frequency components.

This experiment demonstrates the feasibility of separating a low-power pilot signal from a stronger data signal using frequency-selective filtering. Such functionality is essential for the proposed synchronization architecture, where the pilot is embedded at significantly lower power levels than the data.

Figure 6.5 illustrates the complete system architecture implemented on the FPGA. The design is built around the HSMC-based data conversion interface and integrates

multiple functional blocks to enable end-to-end signal generation, conversion, and processing. A PLL block generates a 50 MHz clock domain from a 100 MHz reference clock, which is then used to drive all system modules to ensure synchronous operation and reliable data reception. The NCO modules are configured to generate sinusoidal signals at 1 MHz and 10 MHz, which are subsequently combined using an adder to form a composite test signal. This signal is passed through the DAC interface, where it is converted into an analog waveform, and then routed through the ADC interface, which samples the analog signal and converts it back into digital form.

The digitized output is captured using a data input latch to ensure proper synchronization, followed by a Finite Impulse Response (FIR) filter with a 3 MHz bandwidth for further signal conditioning and validation. The HSMC connector provides the high-speed physical interface between the FPGA and the external ADC/DAC hardware, enabling reliable data transfer across the system. Overall, the architecture demonstrates a fully integrated loopback system for verifying signal generation, conversion accuracy, and digital processing functionality.

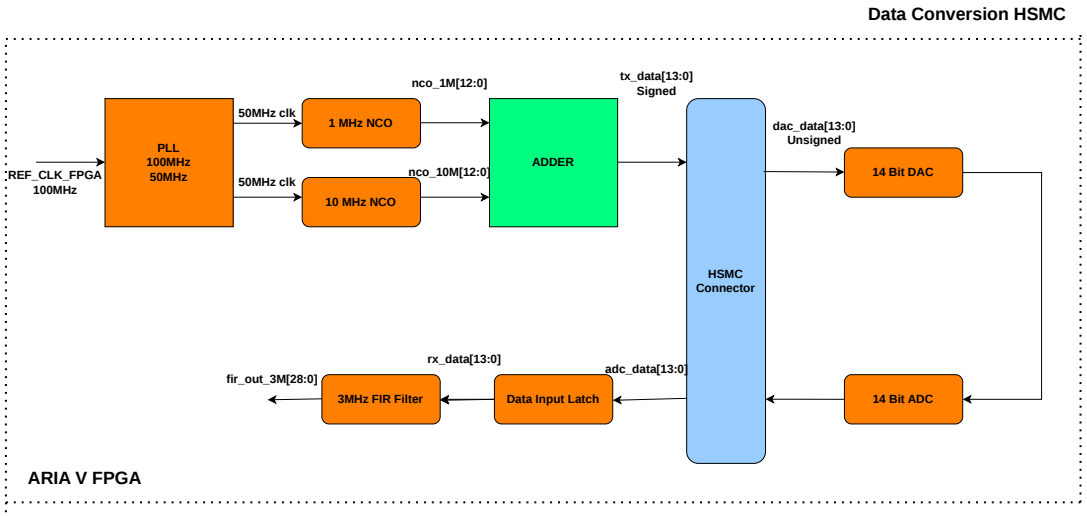


Figure 6.5: System-level loopback architecture for ADC-DAC verification

Functional Description

The PLL generates stable clock signal required for the operation of all modules. The NCO blocks generate two sinusoidal signals at different frequencies (1 MHz and 10 MHz), which are combined using an adder to create a multi-tone test signal. This signal is then sent to the DAC, where it is converted into an analog waveform.

The analog output of the DAC is looped back into the ADC input through the HSMC interface described in [39], while hardware connectivity follows the Arria V GX board schematic [38]. The ADC samples this signal and converts it back into digital form. The data is then latched and passed through an FIR filter for analysis.

A low-pass FIR filter was designed with a sampling frequency of $F_s = 50$ MHz, a cutoff frequency of $F_c = 3$ MHz, and a filter length of $N = 37$ taps. The filter

operation is described by the convolution equation

$$y[n] = \sum_{k=0}^{N-1} h[k] \cdot x[n-k], \tag{6.1}$$

where $h[k]$ represents the filter coefficients and $x[n]$ is the input signal. The system processes a 14-bit ADC input, and due to internal accumulation within the filter structure, the output word length increases to 29 bits, requiring appropriate MSB selection and scaling to maintain numerical stability and prevent overflow [30].

The NCO is used to generate a digitally synthesized sine wave, where the output frequency is defined by

$$f_{out} = \frac{\text{phaseinc} \cdot f_{clk}}{2^N}. \tag{6.2}$$

The NCO architecture is based on a phase accumulator, a lookup table (LUT), and sine/cosine output generation. The phase accumulator increments at each clock cycle according to the frequency control word, while the LUT converts the accumulated phase into corresponding sinusoidal amplitude values. This structure enables precise and flexible digital frequency generation suitable for FPGA-based signal processing applications.

6.4 IQ Signal Generation and Verification

To validate complex signal processing capabilities, in-phase (I) and quadrature (Q) components were generated using the NCOs at frequencies of 1 MHz and 10 MHz. These components were combined to form a complex baseband signal at a frequency of 11 MHz. The generated IQ signals were verified using SignalTap and external measurement tools, confirming correct amplitude, phase relationship, and frequency characteristics.

This step ensures that the FPGA implementation supports complex-valued signal processing and the receiver front-end can correctly process complex-valued signals, which is essential for subsequent carrier synchronization and phase tracking.

Design Architecture

The 11 MHz signal is generated by combining two Numerically Controlled Oscillator (NCO) outputs, where NCO_1M produces a 1 MHz signal and NCO_10M generates a 10 MHz signal. These signals are combined using complex exponential (IQ) modulation, where the individual sinusoidal components are added in the complex baseband domain to form the resulting composite waveform. This approach enables flexible frequency synthesis by exploiting linear superposition of digitally generated tones, allowing the final 11 MHz signal to be accurately produced within the FPGA-based signal generation framework.

$$e^{j(2\pi \cdot 11\text{MHz} \cdot t)} = e^{j(2\pi \cdot 10\text{MHz} \cdot t)} \cdot e^{j(2\pi \cdot 1\text{MHz} \cdot t)} \tag{6.3}$$

IQ Modulation and Complex Signal Combination

To generate the 11 MHz signal using two NCOs (1 MHz and 10 MHz), the design follows a complex exponential representation using in-phase (I) and quadrature (Q) components.

Each NCO generates a complex sinusoidal signal:

$$S_{1M}(t) = \cos(2\pi \cdot f_1 \cdot t) + j \sin(2\pi \cdot f_1 \cdot t) \tag{6.4}$$
$$S_{10M}(t) = \cos(2\pi \cdot f_{10} \cdot t) + j \sin(2\pi \cdot f_{10} \cdot t) \tag{6.5}$$

The combined signal is obtained by complex multiplication:

$$\begin{aligned} S_{11M}(t) &= [\cos(2\pi f_{10}t) + j \sin(2\pi f_{10}t)] \\ &\quad \cdot [\cos(2\pi f_1 t) + j \sin(2\pi f_1 t)] \\ &= \cos(2\pi f_{10}t) \cos(2\pi f_1 t) \\ &\quad - \sin(2\pi f_{10}t) \sin(2\pi f_1 t) \\ &\quad + j \Big[\sin(2\pi f_{10}t) \cos(2\pi f_1 t) \\ &\qquad + \cos(2\pi f_{10}t) \sin(2\pi f_1 t) \Big] \end{aligned} \tag{6.6}$$

Using trigonometric identities:

$$\cos A \cos B - \sin A \sin B = \cos(A + B) \tag{6.7}$$
$$\sin A \cos B + \cos A \sin B = \sin(A + B) \tag{6.8}$$

we obtain:

$$S_{11M}(t) = \cos(f_{11} \cdot t) + j \sin(f_{11} \cdot t) \tag{6.9}$$

In hardware, the above equations are implemented using multipliers and adders. The real and imaginary components are given by:

$$I_{11M} = I_{10M} I_{1M} - Q_{10M} Q_{1M} \tag{6.10}$$

$$Q_{11M} = I_{10M} Q_{1M} + Q_{10M} I_{1M} \tag{6.11}$$

where $I_{10M} = \cos(\omega_{10})$, $Q_{10M} = \sin(\omega_{10})$, $I_{1M} = \cos(\omega_1))$, and $Q_{1M} = \sin(\omega_1))$. This corresponds to complex mixing using $e^{j2\pi \cdot f_1 t}$, achieving frequency translation from 10 MHz to 11 MHz. In FPGA, the NCO generates the I/Q signals, while DSP blocks perform multiplications and adders form the final output.

Figure 6.6 illustrates the architecture used to generate the 11 MHz signal. The system consists of two NCO IP cores operating at 1 MHz and 10 MHz, IQ combination

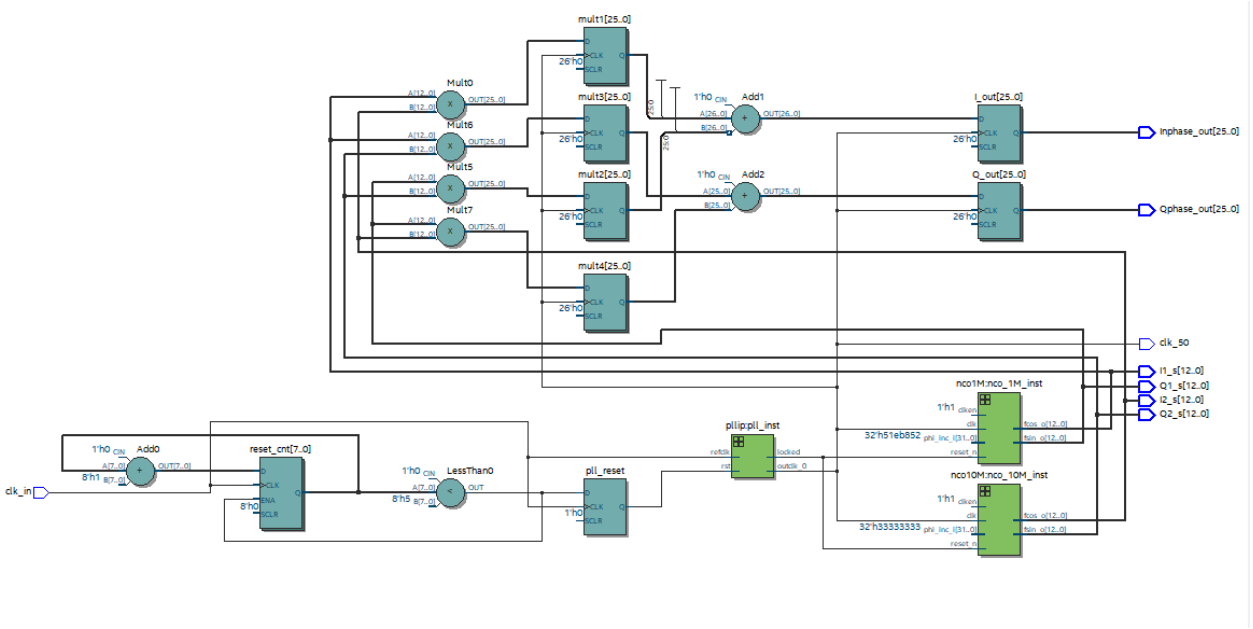


Figure 6.6: Block diagram of 11 MHz signal generation using NCO modules

logic, and fixed-point arithmetic blocks for multiplication and addition. The NCO IP cores are driven by a 50 MHz clock derived from the PLL and are configured with a 32-bit phase accumulator and a 13-bit output width. The frequency control word ϕ_{inc} is computed according to Equation 6.12, ensuring accurate frequency synthesis.

$$\phi_{\text{inc}} = \frac{f_{\text{out}}}{f_{\text{clk}}} \cdot 2^{32} \tag{6.12}$$

The outputs of the NCO blocks, namely `fsin_o` and `fcos_o`, provide sine and cosine waveforms respectively. These signals are treated as signed fixed-point values and are further processed within the digital domain to realize the final mixed-frequency output used in the system.

6.5 Matched Filter Implementation

The matched filter was implemented to maximize the SNR and improve pilot sequence detection performance. The complete matched filtering operation was implemented inside the FPGA using fixed-point arithmetic, with the filter coefficients derived from the known pilot/reference sequence .

Initially, a hardware-efficient peak detection approach based on adaptive moving-average thresholding was investigated. In this method, the received signal, including an added phase noise of 30°, was processed through a FIR matched filter, producing signed correlation outputs corresponding to the pilot sequence. To simplify subsequent comparison operations, the signed FIR output was converted into its absolute magnitude representation. The resulting magnitude samples were then processed using a moving-average block with a window length of 32 samples to generate an adaptive detection threshold.

However, the averaging-based thresholding approach was found to be insufficiently robust in noisy environments. When additive white Gaussian noise (AWGN) was introduced, the moving average increased along with the noise power, causing the adaptive threshold to rise dynamically. This resulted in unstable threshold behavior and multiple false peak detections, particularly at lower SNR values. Due to these limitations, the averaging-based method was not selected for the final implementation.

To improve detection robustness, a maximum-based adaptive thresholding technique was implemented. In this approach, the absolute value of the FIR output `abs_val_u` was continuously monitored and compared against a running maximum register `max_val`. Whenever a larger correlation value was observed, the maximum register was updated accordingly. At the completion of each PN sequence duration, the detected maximum value was latched and used as the threshold for the subsequent detection window. Peak detection was then performed by comparing the incoming signal magnitude against the threshold condition $\texttt{abs\_val\_u} \geq \texttt{threshold}$.

Since the threshold was derived from the strongest correlation peak rather than from an average noise-dependent estimate, the proposed method significantly reduced false detections and provided stable peak detection performance under noisy conditions. Simulation results verified reliable operation under no-noise conditions as well as in the presence of AWGN at 20 dB and 10 dB SNR. The matched filter output was verified in Quartus using internally generated IQ test signals, where the resulting correlation output demonstrated improved peak sharpness and enhanced pilot detectability.

6.6 Phase Estimation Using CORDIC IP

The received IQ samples exhibit phase rotation due to carrier frequency offset, channel impairments, and intentionally introduced phase offsets in the transmitted signal. In this work, a controlled phase offset of 30° was added to the transmitted IQ samples in order to validate the accuracy of the phase estimation stage. To efficiently estimate the instantaneous phase within the FPGA, the Intel FPGA CORDIC IP core was employed. The CORDIC algorithm is well-suited for real-time digital signal processing because it is implemented using only shift-and-add operations, avoids multipliers and dividers, and can be fully pipelined to achieve high-throughput streaming operation.

The CORDIC IP core performs coordinate rotation digital computation and enables the conversion of Cartesian IQ samples into polar form. Specifically, the phase angle is computed using the $\texttt{atan2}(Q, I)$ function, expressed as $\hat{\theta} = \text{atan2}(Q, I)$, which maps the in-phase and quadrature components into an instantaneous phase estimate. In addition to phase extraction, the CORDIC block also provides the magnitude of the input signal, making it suitable for both amplitude and phase analysis in FPGA-based receivers.

In the FPGA implementation, the Intel CORDIC IP core was configured in vectoring mode, which is optimized for converting Cartesian inputs into magnitude and

phase outputs. The I and Q samples were provided as inputs to the CORDIC block, while the corresponding outputs were the estimated phase angle and signal magnitude. This configuration allows efficient real-time computation of phase information without requiring complex arithmetic resources.

6.7 Phase Correction (Derotation)

The derotation stage was implemented to compensate for the residual carrier phase offset present in the received baseband signal. Although the phase estimation stage using the CORDIC IP provides an accurate estimate of the instantaneous phase, the received signal may still exhibit a fixed phase rotation due to transmitter–receiver oscillator mismatch. To remove this effect, a phase correction stage is applied after correlation with the PN sequence in order to align the signal with the in-phase axis and enable reliable symbol detection.

Mathematically, the derotation operation is performed by multiplying the received complex signal with the conjugate of the estimated phase. This can be expressed in terms of in-phase and quadrature components as:

$$I_{\text{corr}} = I\cos(\hat{\theta}) + Q\sin(\hat{\theta}) \tag{6.13}$$

$$Q_{\text{corr}} = -I\sin(\hat{\theta}) + Q\cos(\hat{\theta}) \tag{6.14}$$

where $\hat{\theta}$ represents the estimated phase obtained from the CORDIC-based phase estimator. This transformation effectively rotates the received constellation back to its original orientation, ensuring that the signal energy is concentrated along the in-phase component.

In the FPGA implementation, the derotation block was realized using a hardware-efficient complex multiplication structure. The estimated phase is used to generate sine and cosine terms, which are then combined with the incoming I and Q samples through multipliers and adders/subtractors. This architecture reduces the number of required DSP blocks compared to a direct trigonometric implementation and avoids the use of resource-intensive sine/cosine function evaluation. As a result, it achieves efficient utilization of FPGA logic resources while supporting a fully pipelined structure for real-time operation.

6.8 FPGA-based DPLL for CFO Estimation

The DPLL is a feedback control system used for carrier frequency and phase synchronization in digital communication systems. It operates by continuously estimating the phase difference between a received signal and a locally generated reference and correcting this error through a closed-loop mechanism.

In the synchronization architecture, the phase detector computes the instantaneous phase error between the incoming signal and the NCO output. This error is passed through a PI loop filter, which suppresses noise and generates a stable control signal

for frequency correction. The proportional path provides fast response to phase variations, while the integral path eliminates steady-state frequency offset.

The filtered output adjusts the frequency of the NCO, which generates the local carrier used for mixing and phase alignment. As the loop operates, the system converges when the phase error approaches zero and the NCO frequency matches the incoming carrier frequency, thereby achieving CFO compensation. This closed-loop mechanism enables continuous tracking of phase and frequency variations, making the DPLL suitable for real-time carrier recovery in FPGA-based digital receivers.

Current results demonstrate stable DPLL operation, with the phase error remaining bounded and the frequency control signal converging to a steady value. The implemented FPGA-based DPLL successfully achieves frequency lock at approximately 12 kHz, indicating correct CFO estimation and tracking behavior.

The observed FPGA results closely match those obtained from the fixed-point MATLAB simulation model, providing strong validation of the hardware implementation. This agreement confirms the correctness of the fixed-point scaling strategy, PI loop filter design, phase accumulator implementation, and NCO control architecture. The objective of this stage is to achieve accurate and stable CFO estimation with minimal steady-state error while maintaining hardware efficiency for real-time FPGA implementation.

7

Results

This chapter presents the results and performance evaluation of the proposed digital carrier recovery system. The results are organized into two main parts to reflect the development flow of the system. First, MATLAB simulation results are presented to verify the functional correctness and analyze the performance of the proposed algorithms under ideal and impaired channel conditions. Second, FPGA implementation results are provided to validate the hardware realization of the system and to confirm correct operation of the signal processing chain in real-time.

7.1 MATLAB Simulation Results and Analysis

This section presents the MATLAB simulation results and performance analysis of the proposed digital carrier recovery system. The system is evaluated in terms of time-domain and spectral behavior, pilot signal detection, carrier phase and frequency synchronization, and constellation performance under CFO and noise conditions. In addition, fixed-point simulations are carried out to verify the feasibility of FPGA implementation and to assess the impact of finite word-length effects on synchronization accuracy.

7.1.1 Simulation Setup and Design Parameters

The proposed digital carrier recovery architecture was first validated using MATLAB simulations. The system consists of a spread-spectrum data signal combined with a low-power pilot tone, followed by a digital receiver implementing matched filtering and a DPLL. The key simulation parameters include a carrier frequency of 1 GHz, a sampling rate of 40 MSPS, a PN sequence length of 511 based on an m-sequence, a pilot frequency of 1 MHz, and a DPLL loop bandwidth of 1 kHz.

A long ZC sequence of length 511 was selected to achieve high processing gain and improved noise immunity. The pilot signal is intentionally designed as a narrowband low-frequency tone to enable robust carrier tracking even under severe interference conditions. The selected loop bandwidth of 10 kHz represents a trade-off between noise suppression and tracking capability, where a narrower bandwidth improves noise filtering but limits the range of frequency offsets that can be reliably tracked.

To overcome this limitation, a coarse frequency acquisition technique can be employed using multiple parallel frequency hypotheses. In this approach, several NCOs are configured with different frequency offsets (e.g., 1 kHz, 2 kHz, 3 kHz, and so

on), and the received signal is mixed with each of these candidate frequencies. The output of each branch is then passed through the matched filter, and the branch producing the highest correlation peak indicates the closest estimate of the actual CFO.

This approach effectively shifts the received signal toward baseband prior to fine synchronization, thereby extending the detectable CFO range well beyond the limitation imposed by the PN sequence length and loop bandwidth. As a result, even with a relatively narrow loop bandwidth, the system can achieve a wide acquisition range (e.g., tens or hundreds of kHz), provided sufficient hardware resources are available. Once the coarse CFO is identified, a fine tracking loop can be applied for accurate phase and frequency synchronization. This hierarchical approach enables both wide acquisition range and high-precision tracking within the same system.

7.1.2 Time-Domain and Spectral Analysis

Figure 7.1(a) shows the transmitted time-domain signal, which consists of high-rate PN-modulated data combined with a low-power pilot component. The waveform exhibits rapid transitions due to spreading, confirming wideband signal characteristics.

The TX spectrum in Figure 7.1(b) presents the transmitted spectrum. The signal energy is spread over a wide bandwidth due to PN modulation, while a narrowband pilot component is embedded within the spectrum.

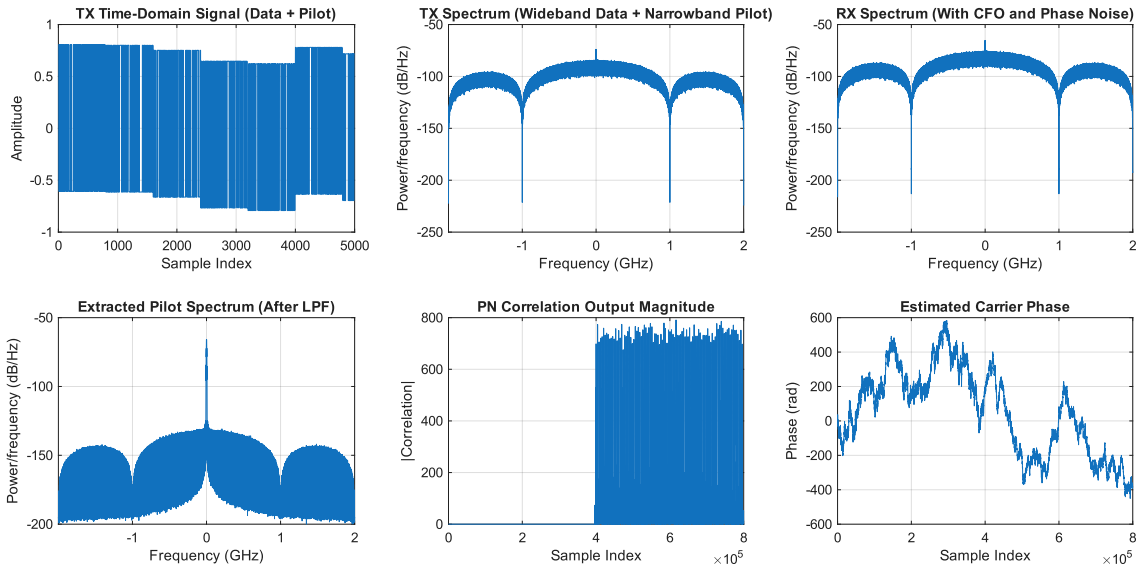


Figure 7.1: MATLAB simulation results of the proposed system demonstrating robust carrier recovery and pilot detection: (a) TX Time-domain Signal, (b) TX spectrum, (c) Rx spectrum, (d) extracted pilot Spectrum, (e) PN correlation output Magnitude providing processing gain, and (f) estimated carrier phase. The results confirm reliable detection of a pilot signal buried up to 30 dB below the data signal.

The RX spectrum in Figure 7.1(c) illustrates the received spectrum in the presence of CFO and phase noise. The spectral distortion and spreading confirm the impact

of channel impairments and motivate the need for carrier recovery.

Pilot Extraction and Matched Filtering

Figure 7.1(d) presents the spectrum of the extracted pilot signal after low-pass filtering. The low-pass filter suppresses the wideband data components and retains the narrowband pilot signal located around the pilot frequency. This filtering stage improves the signal-to-noise ratio of the pilot and prepares the signal for subsequent matched filtering. The result confirms that the pilot can be reliably isolated even when transmitted at a substantially lower power level than the data signal.

Figure 7.1(e) shows the output of the matched filter applied to the received signal containing the embedded PN-coded pilot. A distinct correlation peak is observed when the received signal aligns with the reference PN sequence.

This peak indicates successful detection of the pilot signal and confirms the effectiveness of the matched filtering approach. The presence of lower-amplitude sidelobes is consistent with the autocorrelation properties of the PN sequence.

The matched filter output demonstrates that the PN-coded pilot can be reliably detected despite being embedded at a lower power level than the data signal. This processing gain is a key advantage of the proposed synchronization method.

Carrier Phase Estimation and Tracking

Figure 7.1(f) shows the estimated carrier phase over time. The phase exhibits smooth variations due to CFO and phase noise, demonstrating that the phase detector and loop filter are effectively tracking the carrier.

The estimated phase is used to drive the numerically controlled oscillator (NCO), forming a closed-loop digital PLL for carrier correction.

7.1.3 Constellation Analysis

Figure 7.2 shows the received constellation before and after CFO correction. Prior to correction, the constellation points form a circular trajectory due to continuous phase rotation. After applying the DPLL, the constellation stabilizes, indicating successful PLL-based carrier recovery.

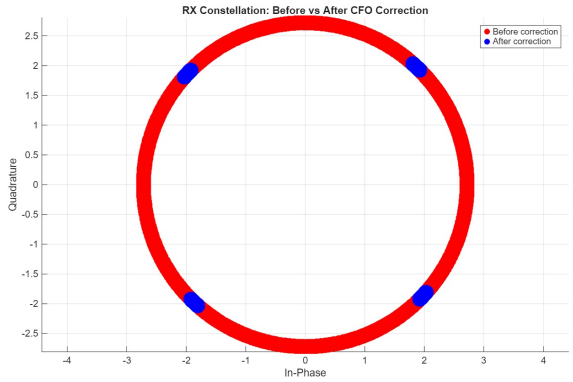


Figure 7.2: Received constellation before and after CFO correction

7.1.4 Frequency Locking Performance

The frequency estimation results in Figure 7.3 demonstrate convergence to a stable value after initial transients, indicating successful locking of the digital phase-locked loop. Minor fluctuations are attributed to noise and finite loop bandwidth. This confirms the effectiveness of the DPLL in compensating frequency offsets.

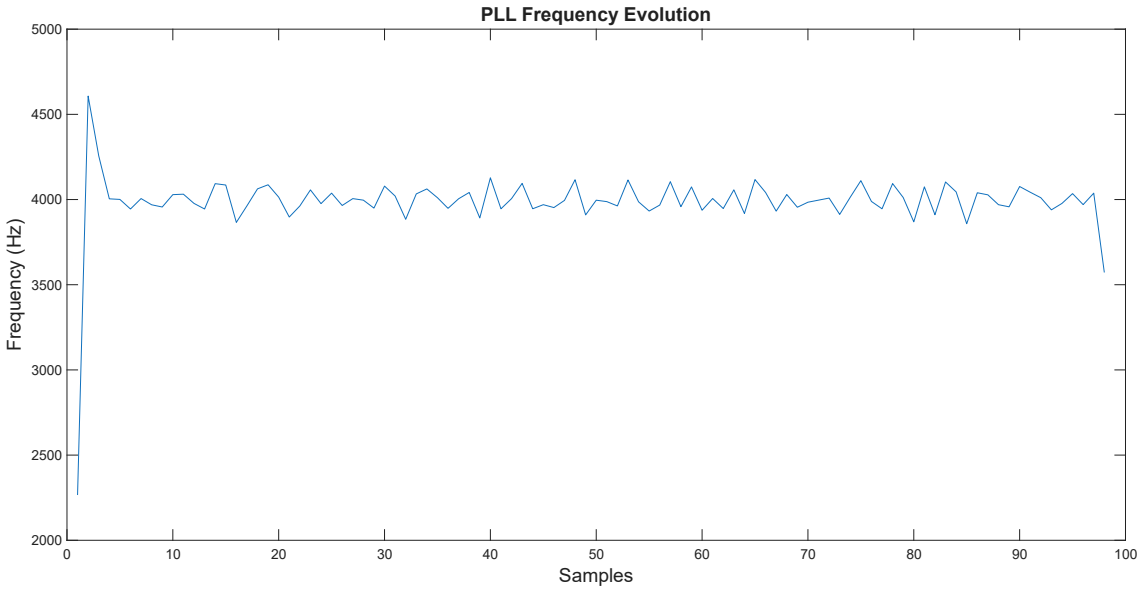


Figure 7.3: Estimated frequency/phase evolution showing PLL lock around 4 kHz under injected CFO

One of the key outcomes of this work is the ability to detect a pilot signal that is significantly weaker than the data signal. Owing to the processing gain provided by the PN-based matched filtering approach, the system is capable of reliably detecting a pilot signal that is buried down to 30 dB below the data signal. This demonstrates the robustness of the proposed synchronization architecture and highlights its suitability for high-speed communication systems in which pilot tones must operate at very low power levels without degrading detection performance.

The MATLAB simulation results further validate the effectiveness of the proposed digital carrier recovery system. The system successfully achieves accurate carrier phase tracking using a DPLL, while also enabling reliable detection of the PN sequence through matched filtering. In addition, the architecture maintains stable operation under carrier frequency offset and additive noise conditions. Most importantly, the simulations confirm that pilot signals can still be reliably detected even when their power is 30 dB lower than the data signal, indicating strong interference resilience.

Overall, these results demonstrate that the receiver architecture is capable of isolating a narrowband pilot signal from a widerband composite signal. This capability is essential for the proposed PN-based synchronization approach, where reliable extraction of a low-power pilot embedded within a high-power data stream is required for accurate carrier recovery and timing alignment.

7.1.5 Fixed-Point Validation of Synchronization Algorithms

To evaluate the feasibility of FPGA implementation, the synchronization algorithms were converted from floating-point to fixed-point arithmetic in MATLAB. This step was performed to emulate the finite word-length effects that occur in digital hardware and to verify that the proposed algorithms maintain acceptable performance under quantization constraints.

The fixed-point simulations were implemented using MATLAB fixed-point data types, with wordlengths selected according to the target FPGA architecture. The matched filter, phase detector, loop filter, and NCO blocks were modeled using fixed-point arithmetic to investigate the impact of quantization on synchronization accuracy.

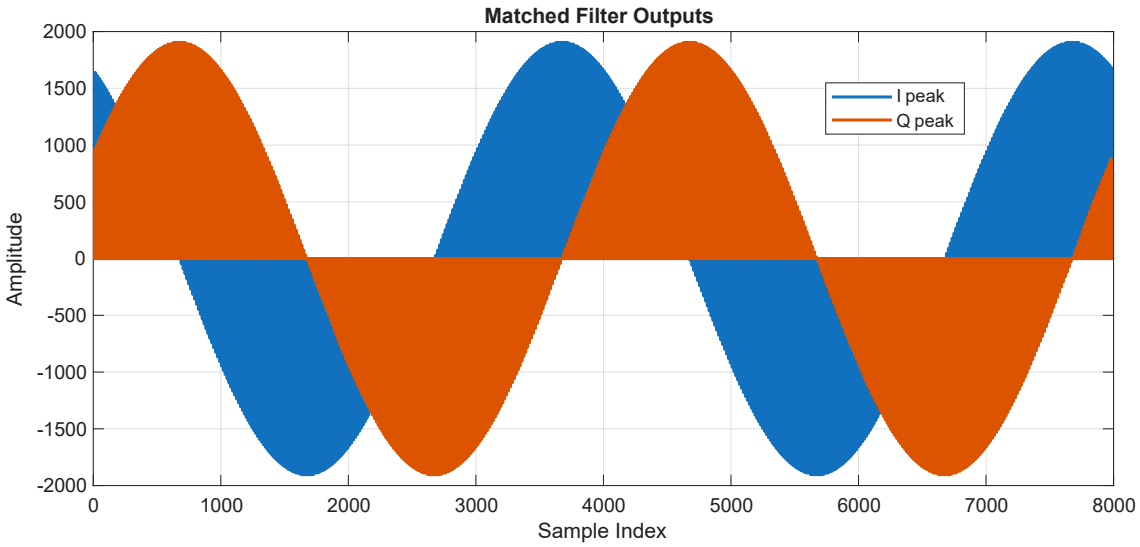


Figure 7.4: Fixed-point matched filter output obtained from MATLAB simulation. The correlation peaks remain clearly detectable despite finite word-length quantization effects

Figure 7.4 shows the matched filter output obtained using fixed-point arithmetic. The correlation peak remains clearly distinguishable despite the quantization effects, indicating that the pilot sequence can still be reliably detected. The observed peak location matches the floating-point reference implementation, confirming that the synchronization algorithm is robust to finite precision effects.

The results demonstrate that fixed-point implementation introduces only minor amplitude quantization while preserving the overall correlation characteristics. Consequently, the proposed synchronization architecture is suitable for FPGA realization using fixed-point arithmetic.

The fixed-point simulation results provide confidence that the proposed synchronization algorithms can be implemented on FPGA hardware with limited numerical precision while maintaining reliable pilot detection and synchronization performance.

7.1.6 Fixed-Point DPLL Verification

In this work, the DPLL employs a PI loop filter to track residual CFO and phase variations. The proportional component provides an immediate response to phase errors, while the integral component accumulates phase errors over time and eliminates steady-state frequency offsets.

The proportional gain K_p determines the speed of the loop response to phase errors. A larger K_p results in faster acquisition but may increase loop oscillations. The integral gain K_i accumulates phase errors and removes residual frequency offsets that cannot be corrected by the proportional path alone. Proper selection of K_p and K_i is essential to achieve a stable and responsive synchronization loop.

To verify the suitability of the proposed carrier synchronization architecture for FPGA implementation, the DPLL was modeled using fixed-point arithmetic in MATLAB. The simulation was performed using the same loop structure intended for hardware realization, including the phase detector, PI loop filter, and NCO.

Figure 7.5 illustrates the evolution of the main internal signals of the DPLL during the carrier synchronization process. The `Phase Error` signal represents the instantaneous phase difference between the received pilot signal and the locally generated reference signal. During the acquisition phase, the phase error is initially large and gradually decreases as the loop approaches lock. The `i_term` signal corresponds to the accumulated phase error generated by the integral branch of the PI controller. This accumulated term provides long-term frequency correction and eliminates residual carrier frequency offsets that cannot be removed by the proportional path alone. The `Frequency Control Word` signal is the output of the PI loop filter, combining both proportional and integral contributions, and serves as the control input to the NCO. The `DDS LUT Address` signal represents the accumulated phase generated by the NCO phase accumulator, which continuously updates the local carrier phase and frequency to compensate for the estimated synchronization error.The `Estimated CFO (Hz)` signal shows the carrier frequency offset estimate generated by the loop. As the DPLL converges, this estimate settles to approximately 12 kHz and remains stable, indicating successful frequency lock. Simultaneously, the phase

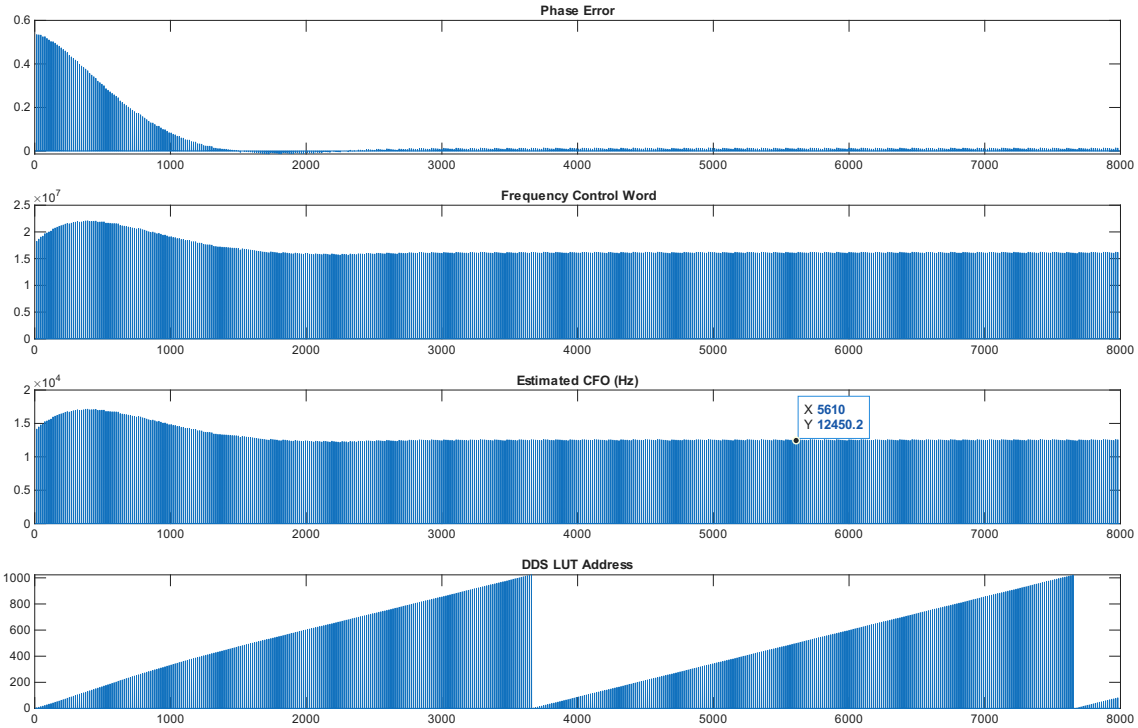


Figure 7.5: Fixed-point DPLL simulation showing the evolution of phase error, integral term, frequency control signal, and NCO phase during carrier synchronization. The convergence of the phase error and stabilization of the control signal indicate successful loop operation.

error approaches zero while the frequency control signal stabilizes, confirming successful carrier tracking and synchronization.

The results demonstrate that the phase error converges toward zero while the frequency control signal stabilizes at a constant value. Simultaneously, the NCO phase evolves smoothly, indicating successful carrier tracking. The behavior of the fixed-point implementation closely matches the floating-point reference model, confirming that quantization effects do not significantly degrade synchronization performance.

The observed behavior confirms the correct operation of the DPLL and demonstrates that the fixed-point implementation is capable of achieving stable carrier synchronization while maintaining compatibility with FPGA-based hardware realization. This validates the feasibility of implementing the proposed DPLL architecture using fixed-point arithmetic on FPGA hardware.

7.2 FPGA: Signal Chain Verification Results

This section presents the FPGA-based signal chain verification results, focusing on the functionality of key digital processing blocks implemented in the system. The primary objective is to validate both frequency-selective filtering and I/Q signal processing within the receiver chain. In particular, the results demonstrate the ability to isolate a low-frequency pilot signal from a composite multi-tone input using

digital filtering, as well as to verify correct I/Q signal generation and processing for subsequent carrier synchronization tasks. These steps are essential to ensure that the digital front-end can reliably suppress unwanted spectral components before further synchronization and estimation stages.

7.2.1 Frequency-Selective Filtering Results

To validate the ability to extract a narrowband pilot signal from a composite signal, frequency-selective filtering was performed on signals containing multiple frequency components. A composite signal consisting of 1 MHz and 10 MHz frequency components was generated and transmitted through the FPGA signal chain. The FIR filter was designed to pass the low-frequency pilot component while attenuating higher-frequency spectral components.

Table 7.1 summarizes the measured signal amplitudes at different stages of the processing chain.

Table 7.1: Verification of ADC-DAC Loopback and FIR Filtering

Processing Stage	**1 MHz Component**	**10 MHz Component**
NCO Signal Generation	0 dB	0 dB
ADC-DAC Loopback	-0.2 dB	-0.3 dB
FIR Filter Output	0 dB	-45 dB

The results demonstrate that both frequency components are preserved through the ADC-DAC loopback path with negligible attenuation. After digital filtering, the desired 1 MHz component remains essentially unchanged, while the 10 MHz component is significantly suppressed. This confirms the correct operation of the FIR filter and validates the FPGA signal-processing chain prior to the implementation of carrier synchronization algorithms.

Clocking Considerations

Special attention was given to clock domain synchronization to ensure reliable system operation across all processing blocks. A PLL was employed to generate phase-aligned clock signals for all modules, thereby minimizing timing skew and reducing the possibility of sampling errors. In addition, proper differential clock routing was implemented for the ADC and DAC interfaces to preserve signal integrity at high sampling rates.

The loopback experiment confirms correct operation of the complete signal chain from digital signal generation to analog conversion and back to digital processing. The system employs multiple clock domains, including the ADC interface clocks, DAC interface clocks, and the FPGA core processing clock. These clocks are derived from a common reference clock using a PLL to ensure phase alignment and stable frequency generation. Proper synchronization between ADC/DAC data clocks and the FPGA processing clock is essential to avoid sampling errors and timing mismatches in the data path. The FIR filter and NCO-based signal generation operate

synchronously within this clocking framework, ensuring reliable real-time processing of multi-frequency signals.

7.2.2 IQ Signal Generation and Verification

The generation of accurate I and Q signals is a fundamental requirement for carrier synchronization, phase estimation, and complex baseband processing. To verify the FPGA implementation, two NCO outputs operating at 1 MHz and 10 MHz were generated and combined to form a complex signal corresponding to an effective carrier frequency of 11 MHz.

Table 7.2 summarizes the verification results obtained from the FPGA implementation.

Table 7.2: Verification of FPGA-Based IQ Signal Generation

Parameter	**Measured Result**
I/Q Frequency	11 MHz
Phase Difference (I-Q)	90°
Signal Stability	Stable over multiple cycles
Amplitude Consistency	No observable drift
Overflow / Saturation	Not observed
Fixed-Point Operation	Verified

The measurements confirm that the FPGA correctly generates orthogonal I and Q signals with the expected quadrature phase relationship. The generated signals remain stable over multiple observation intervals, indicating correct operation of the NCO phase accumulator, lookup-table generation, and fixed-point arithmetic implementation.

Furthermore, no clipping, overflow, or numerical instability was observed within the selected fixed-point representation. These results validate the suitability of the implemented IQ signal generation architecture for subsequent synchronization functions, including matched filtering, phase estimation, carrier phase correction, and DPLL-based carrier tracking.

7.2.3 Matched filter Verification

Figure 7.6 demonstrates how the FPGA detects correlation peaks from the received PN sequence using the maximum-based thresholding method implemented in Quartus. The received PN sequence with added phase offset is processed through the FIR matched filter, producing correlation outputs represented by `fir_out_matched`. Since the FIR output contains signed values, the signal is converted into its absolute magnitude form `abs_val` before threshold comparison. A running maximum value `max_val` is continuously updated during each PN sequence duration by comparing the current magnitude with the previously stored maximum value. At the end of the PN observation window, this maximum value is latched and used as the adaptive threshold `threshold` for the subsequent detection interval. Peak detection is

performed whenever the condition `abs_val` $\geq$ `threshold` is satisfied, resulting in assertion of the `peak_flag` signal. The waveform demonstrates stable and accurate detection of the correlation peaks with only one valid peak detected per PN sequence window. The results confirm the correct operation of the maximum-based adaptive thresholding technique and validate the effectiveness of the implemented FPGA-based matched filtering and synchronization architecture.

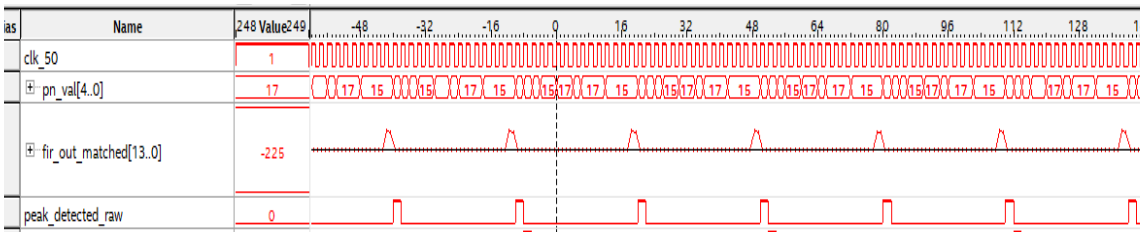


Figure 7.6: Peak Detection Without Noise

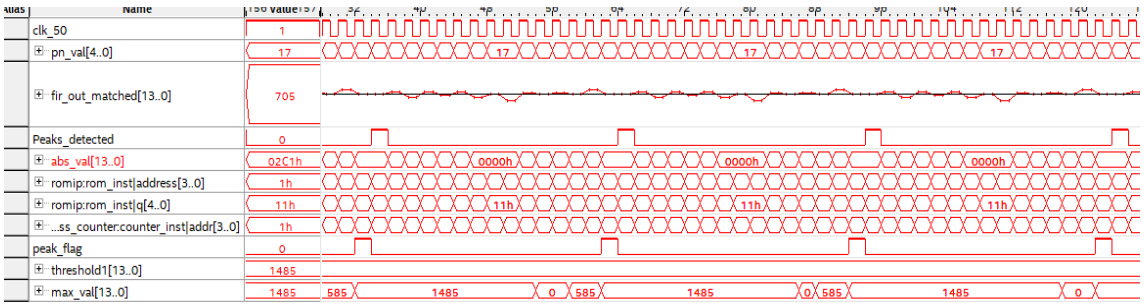


Figure 7.7: Peak Detection with AWGN at 20 dB SNR

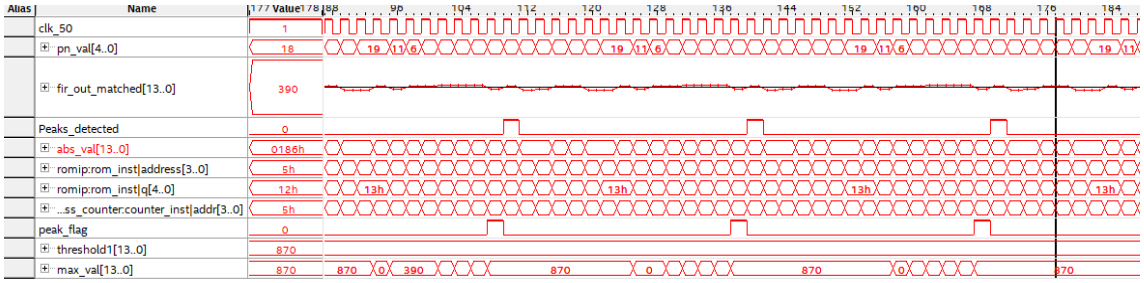


Figure 7.8: Peak Detection with AWGN at 10 dB SNR

Figure 7.7 and Figure 7.8 show the matched filter output with additive white Gaussian noise at 20 dB SNR and at 10 dB SNR respectively. The observed correlation peaks in all above three cases align correctly with the adaptive threshold, demonstrating reliable pilot detection. Compared to the moving-average thresholding approach, the maximum-based thresholding method provides significantly improved stability and reduced false detections, especially in noisy environments.

7.2.4 Verification of CORDIC-Based Phase Estimation

The simulation waveform in Figure 7.9 illustrates the internal operation of the phase estimation process. The signals `peak_flag_I` and `peak_flag_Q` indicate the detection of valid peak samples in the received I and Q channels. At these instants, the corresponding `I_peak` and `Q_peak` values are captured and applied to the CORDIC block for phase computation. The CORDIC output, `theta_cordic`, rises to approximately 30° whenever a valid peak is detected, corresponding to the imposed phase offset. Following this computation, the final processed phase signal, `theta_deg_final`, remains stable and closely aligned with the expected theoretical value. This behavior confirms correct synchronization between peak detection, sample capture, and phase computation, and validates that the CORDIC IP correctly converts IQ samples into accurate and stable phase estimates.

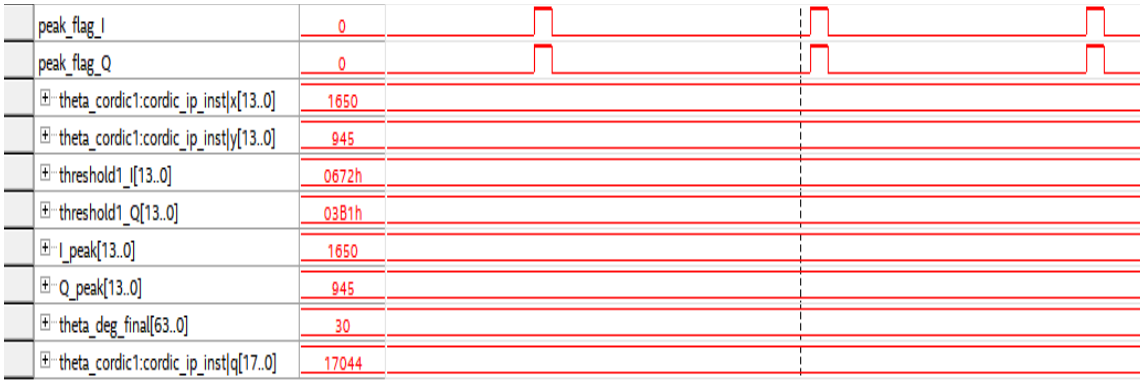


Figure 7.9: Simulation waveform of CORDIC-based phase estimation using IQ samples with 30° phase offset.

7.2.5 Verification of FPGA-Based Derotation

The simulation results in Figure 7.10 confirm the correct operation of the derotation stage. After phase correction, the in-phase component $I_{\texttt{corr}}$ aligns cleanly with the expected PN chip sequence, accurately reproducing the transmitted pattern $[1, 1, 1, 1, -1, 1, -1, 1, 1, -1, -1, 1, -1, -1, -1]$ with the correct sign and amplitude. This demonstrates that the applied transformation successfully compensates for the phase offset introduced in the received signal and restores proper alignment with the in-phase axis.

In contrast, the quadrature component $Q_{\texttt{corr}}$ collapses to values close to zero, exhibiting only minor residual fluctuations due to noise. This behavior is expected in a correctly phase-aligned system, as ideally all signal energy is transferred to the in-phase component while the quadrature component contains only negligible residual error. The near-zero quadrature output therefore further validates the correctness of the derotation process.

The simulation waveform clearly illustrates this behavior, showing distinct alignment of $I_{\texttt{corr}}$ with the PN sequence and suppression of $Q_{\texttt{corr}}$ around zero. Overall, the results confirm that the derotation and correlation chain operates correctly and

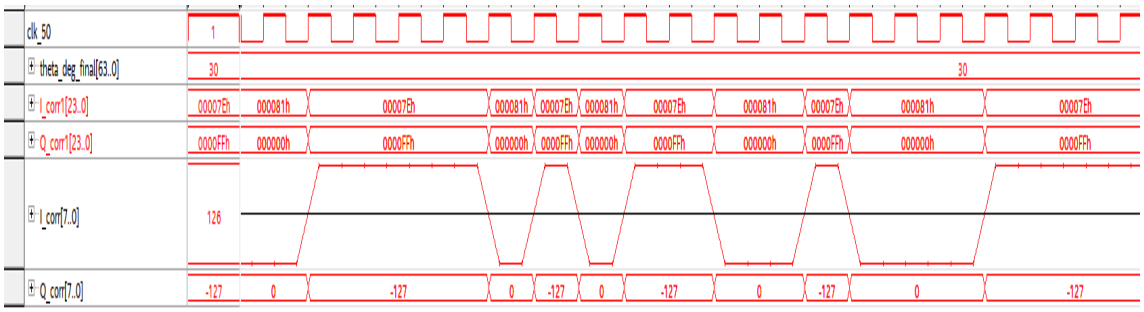


Figure 7.10: Simulation waveform of the derotation (phase correction) stage showing the corrected in-phase and quadrature components

effectively removes residual phase rotation, ensuring accurate signal recovery in the FPGA-based receiver.

7.2.6 Verification of FPGA-based DPLL

The simulation waveform in Figure 7.11 illustrates the internal operation of the implemented fixed-point DPLL on the FPGA. The waveform shows the evolution of the main synchronization signals, including the phase error, integral term, frequency control signal, NCO phase accumulator, and estimated CFO.

The `phase_error` signal remains bounded and stable throughout the observation interval, indicating correct operation of the phase detector and successful tracking of the residual carrier phase error. The `i_term` signal accumulates the phase error over time and converges to a constant value, providing the long-term frequency correction required for carrier synchronization. The `freq_control` signal also stabilizes, demonstrating proper operation of the PI loop filter.

The `nco_phase` signal increases continuously as expected for the NCO, confirming correct phase accumulation and frequency correction behavior. Most importantly, the estimated CFO, represented by `cfo_hz_out`, converges to approximately 12 kHz and remains stable over time.

The observed FPGA results closely match the behavior obtained from the fixed-point MATLAB DPLL model, providing strong validation of the hardware implementation. The successful convergence of the CFO estimate confirms that the fixed-point scaling, loop filter design, phase accumulator implementation, and NCO control architecture operate correctly in hardware. These components will complete the receiver synchronization chain and enable real-time carrier recovery.

The results demonstrate successful carrier frequency offset estimation and stable DPLL operation on the FPGA. Furthermore, the close agreement between the FPGA measurements and the fixed-point MATLAB simulations provides confidence in the correctness of the implemented synchronization architecture and validates the feasibility of real-time carrier recovery using a hardware-efficient fixed-point DPLL.

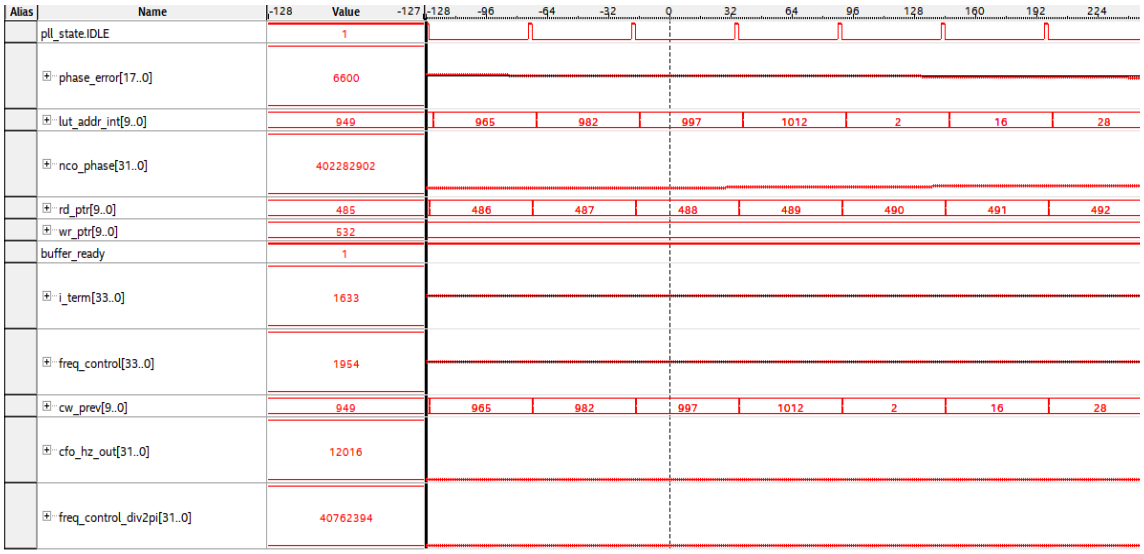


Figure 7.11: SignalTap waveform showing successful FPGA-based fixed-point DPLL operation and CFO lock at approximately 12 kHz.

7.2.7 Resource Utilization and Timing Analysis

The FPGA resource utilization and timing performance of the implemented synchronization architecture were evaluated using the Quartus Prime compilation and timing analysis reports. The target device used in this work was the Intel Arria V FPGA (5AGXFB3H4F35C4) [37].

To illustrate the scalability of the proposed architecture, resource utilization was analyzed at three different implementation stages: ADC-DAC loopback verification, IQ signal processing, and the complete synchronization chain including the fixed-point DPLL.

Table 7.3: FPGA Resource Utilization for Different Design Stages

Design Stage	ALMs	Registers	DSPs	Memory Bits	I/O Pins
ADC-DAC	2201 (2%)	5210	19 (2%)	157,696 (<1%)	96 (15%)
IQ Processing	1947 (1%)	4531	2 (<1%)	105,472 (<1%)	106 (16%)
DPLL System	2565 (2%)	4195	24 (2%)	410,402 (2%)	220 (34%)

Table 7.3 shows that the resource utilization increases as additional synchronization blocks are integrated into the design. The ADC-DAC loopback configuration represents the minimum hardware implementation used for initial verification, while the IQ processing stage introduces complex baseband signal generation and processing. The final implementation includes matched filtering, peak detection, phase estimation, carrier correction, and fixed-point DPLL-based carrier tracking.

Despite the additional functionality, the overall resource utilization remains low, with logic utilization below 2% and DSP utilization below 2% of the available Arria V FPGA resources. The results demonstrate that the proposed synchronization

architecture is highly hardware-efficient and leaves substantial resources available for future extensions and integration with additional receiver processing blocks.

Timing analysis was performed to verify that all implemented designs satisfy the timing constraints required for real-time FPGA operation. The designs were constrained to operate at a target clock frequency of 50 MHz, corresponding to a clock period of 20 ns. The worst-case setup and hold slack values reported by the Quartus TimeQuest Timing Analyzer are summarized in Table 7.4.

Table 7.4: Timing Analysis Summary at 50 MHz (20 ns Clock Period)

Design Stage	**Setup Slack (ns)**	**Hold Slack (ns)**
ADC-DAC Loopback	3.921	0.187
IQ Processing	3.478	0.263
Final DPLL System	2.558	0.157

As shown in Table 7.4, all implemented designs achieve positive setup and hold slack values, indicating that the timing requirements are successfully met without violations at the target operating frequency of 50 MHz. The ADC-DAC loopback and IQ processing designs exhibit comfortable timing margins, while the integration of additional synchronization blocks in the final DPLL system increases the critical path length and reduces the available setup slack.

Despite the increased complexity associated with matched filtering, phase estimation, NCO generation, and DPLL-based carrier tracking, the final implementation maintains a positive setup slack of 2.558 ns and a positive hold slack of 0.157 ns. These results confirm that all timing constraints are satisfied and demonstrate that the complete synchronization architecture is capable of reliable real-time operation on the Arria V FPGA.

Overall, the resource utilization and timing analysis results validate that the proposed fixed-point synchronization architecture can be efficiently implemented on FPGA hardware while maintaining low resource consumption and reliable timing performance.

8

Conclusion

This thesis presented the design, verification, and FPGA implementation of a PN-coded pilot-based synchronization framework for wideband sub-THz communication systems. The synchronization architecture developed in this work addresses key receiver challenges including carrier frequency offset (CFO), phase rotation, reliable pilot detection, and fixed-point hardware implementation constraints.

A comprehensive MATLAB-based system model was developed to evaluate synchronization performance under realistic impairments such as carrier frequency offset, phase offset, and additive noise. The results demonstrated that PN-coded pilot sequences provide significant processing gain, enabling reliable pilot detection through matched filtering even at low pilot power levels.

The work included the development of floating-point and fixed-point MATLAB models for matched filtering, peak detection, phase estimation, phase correction, CFO estimation, and Digital Phase-Locked Loop (DPLL) carrier recovery. The fixed-point results closely matched the floating-point reference models, providing confidence in the subsequent FPGA implementation.

The synchronization framework was successfully implemented on an Intel Arria V FPGA using VHDL. The complete receiver chain, including IQ signal generation, matched filtering, adaptive peak detection, CORDIC-based phase estimation, phase correction through IQ derotation, CFO estimation, and fixed-point DPLL carrier recovery, was verified through simulation and real-time FPGA measurements.

A significant contribution of this work is the successful implementation and validation of a fixed-point DPLL for carrier synchronization. FPGA measurements demonstrated stable phase tracking, convergence of the loop filter, correct NCO operation, and successful carrier frequency offset estimation. The estimated CFO converged to approximately 12 kHz, closely matching the behavior observed in the fixed-point MATLAB model. The strong agreement between MATLAB and FPGA results validates the correctness of the developed synchronization architecture and confirms the feasibility of real-time carrier recovery using fixed-point digital signal processing techniques.

Overall, the results demonstrate that the synchronization architecture developed in this thesis provides a hardware-efficient and scalable solution for wideband wireless communication systems. The successful FPGA implementation, together with the close correlation between MATLAB and hardware results, confirms the practicality of PN-coded pilot synchronization for future high-data-rate and sub-THz communication systems.

9

Future Work

Although the synchronization architecture has been successfully validated in both MATLAB and FPGA implementations, several extensions can further improve the system performance and practical applicability.

Possible avenues for future work include the integration of the complete synchronization chain with an RF front-end platform. Such an implementation would enable the generation and transmission of communication signals over a wireless link, allowing end-to-end validation of the synchronization architecture under realistic propagation conditions.

Further investigations could evaluate the robustness of the synchronization architecture in the presence of practical impairments such as large carrier frequency offsets, phase noise, additive white Gaussian noise (AWGN), and channel fading. These studies may provide additional insight into the operational limits and performance trade-offs of the synchronization scheme.

The current implementation could also be extended to support higher-order modulation formats and wider-bandwidth communication signals. In addition, optimization of FPGA resource utilization, timing performance, and power consumption may be explored to facilitate deployment in future high-data-rate communication systems.

Another potential research direction is the application of the synchronization architecture to multi-user and multi-carrier communication scenarios. Such extensions could broaden its applicability to future sub-THz and beyond-5G/6G wireless communication systems.

Bibliography

[1] S. An *et al.*, "A Synchronous Baseband Receiver for High-Data-Rate Millimeter-Wave Communication Systems," *IEEE Microwave and Wireless Components Letters*, vol. 29, no. 6, pp. 412–415, 2019.

[2] ——, "Coded Pilot Assisted Baseband Receiver for High Data Rate Millimeter-Wave Communications," *IEEE Transactions on Microwave Theory and Techniques*, vol. 68, no. 11, pp. 4719–4727, 2020.

[3] T. S. Rappaport *et al.*, "Wireless Communications Above 100 GHz: Opportunities and Challenges for 6G and Beyond."

[4] R. Chen, B. Yan, and M.-C. F. Chang, "A Review of Circuits and Systems for Advanced Sub-THz Transceivers in Wireless Communication," *Electronics*, vol. 14, no. 5, p. 861, 2025.

[5] A. Demir, A. Mehrotra, and J. Roychowdhury, "Phase Noise in Oscillators: A Unifying Theory and Numerical Methods for Characterization," *IEEE Transactions on Circuits and Systems I*, vol. 47, no. 5, pp. 655–674, 2000.

[6] U. Mengali and A. N. D'Andrea, *Synchronization Techniques for Digital Receivers*. Springer, 1997.

[7] J. G. Proakis, *Digital Communications*, 5th ed. McGraw-Hill, 2007.

[8] R. C. Dixon, *Spread Spectrum Systems with Commercial Applications*. John Wiley & Sons, 1994.

[9] M. K. Simon, J. K. Omura, R. A. Scholtz, and B. K. Levitt, *Spread Spectrum Communications Handbook*. McGraw-Hill, 1994.

[10] "Arria V GT FPGA Development Kit User Guide," Intel (Altera), 2018.

[11] "Terasic Arria V GX starter kit user manual," Terasic Technologies, 2016.

[12] "Quartus Prime Standard Edition User Guide," Intel FPGA, 2018.

[13] E. Börjeson and P. Larsson-Edefors, "Energy-Efficient Implementation of Carrier Phase Recovery for Higher-Order Modulation Formats," *IEEE Journal of Lightwave Technology*, vol. 39, no. 2, pp. 505–510, 2021.

[14] V. V. Biryukov, V. L. Vaks, S. A. Kapustin, V. A. Malakhov, A. N. Panin, S. I. Pripolzin, A. S. Raevskiy, Y. V. Raevskaya, and V. V. Shcherbakov, "The Wireless Communications Systems in Subterahertz Frequency Range," *Physics of Wave Processes and Radio Systems*, vol. 26, no. 4, pp. 48–59, 2023.

[15] T. S. Rappaport, O. Kanhere, Y. Xing, S. Ju, A. Madanayake, S. Mandal, A. Alkhateeb, and G. C. Trichopoulos, "Wireless Communications and Applications Above 100 GHz: Opportunities and Challenges for 6G and Beyond," *IEEE Access*, vol. 7, pp. 78 729–78 757, 2019.

[16] J. G. Proakis and M. Salehi, *Digital Communications*, 5th ed. New York, NY, USA: McGraw-Hill, 2008.

[17] R. E. Best, *Phase-Locked Loops: Design, Simulation, and Applications*, 6th ed. McGraw-Hill, 2007.

[18] F.-c. Xing, S.-z. Wang, W.-h. Zong, and S.-j. Zhang, "Design and Implementation of an Improved Costas Loop Circuits by Using FPGA," in *International Conference on Computer Science and Electronic Technology (CSET)*. Qingdao, China: Qingdao University, 2016.

[19] L. M. Patel and J. N. Patel, "Digital Implementation of Costas Loop with Carrier Recovery," *International Journal of Engineering Research and Development (IJERD)*, vol. 11, no. 2, pp. 18–23, February 2015, e-ISSN: 2278-067X, p-ISSN: 2278-800X, www.ijerd.com.

[20] Q. Chaudhari, "Direct Digital Synthesizer (DDS)," https://wirelesspi.com/direct-digital-synthesizer-dds/, 2023, wireless Pi, Accessed: Jun. 4, 2026.

[21] "NCO Intel FPGA IP User Guide (UG-683406)," Intel Corporation, accessed: 2026-04-19. [Online]. Available: https://docs.altera.com/r/docs/683406/24.3/nco-ip-user-guide/small-rom-architecture

[22] Y. Chen and J. Yang, "Design and Implementation of Improved NCO Based on FPGA," in *Proceedings of the 3rd International Conference on Computer Engineering, Information Science & Application Technology (ICCIA)*, ser. Advances in Computer Science Research, vol. 90. Atlantis Press, 2019.

[23] FPGA Implementation of Digitally Controlled Oscillator (NCO). Accessed: 2026-04-19. [Online]. Available: https://en.eeworld.com.cn/news/qrs/eic92348.html

[24] FPGAPS, "CORDIC-based NCO Implementation in FPGA," 2025, accessed: 2026-04-22. [Online]. Available: https://www.hackster.io/fpgaps/cordic-ip-tutorial-create-a-nco-for-sine-cosine-generation-d72be5

[25] NPTEL, IIT Madras, "CORDIC Algorithm Lecture," https://www.youtube.com/watch?v=PalPWv0fU-s, 2020, accessed: 2026-04-22.

[26] NCOs are everywhere - here's how to make one using an FPGA. YouTube, Accessed: 2026-04-19. [Online]. Available: https://www.youtube.com/watch?v=wWadNvkdLqY

[27] FPGA 16 - Intel Altera Verilog CORDIC Sine/Cosine Generator. YouTube, Accessed: 2026-04-19. [Online]. Available: https://www.youtube.com/watch?v=l7VO5EnShhw

[28] CORDIC Algorithm for Angle Calculations. YouTube, Accessed: 2026-04-19. [Online]. Available: https://www.youtube.com/watch?v=4HxdV5xEe9k

[29] FPGA 17 - Intel Altera VHDL CORDIC Sine/Cosine Generator. YouTube, Accessed: 2026-04-19. [Online]. Available: https://www.youtube.com/watch?v=8HQyv6EBh-U

[30] G. Giannakopoulos, P. Adegbenro, and M. A. Perez, "Design and Implementation of FIR Filters in VHDL," *Preprints*, 2025, https://www.preprints.org/manuscript/202502.0761/v2.

[31] S. C. D. Roy, "Fir and iir filters lecture (digital signal processing lecture series)," https://www.youtube.com/watch?v=GpqMAzGEXXk, 2019, lecture Series on Digital Signal Processing, IIT Delhi. Accessed: 2026-04-22.

[32] R. C. Dixon, *Spread Spectrum Systems with Commercial Applications*. Wiley, 1994.

[33] D. C. Chu, "Polyphase Codes with Good Periodic Correlation Properties," *IEEE Transactions on Information Theory*, 1972.

[34] J. A. Gansman, M. P. Fitz, and J. V. Krogmeier, "Frame synchronization for pilot symbol assisted modulation," Purdue University, School of Electrical Engineering, West Lafayette, IN, USA, Tech. Rep., 1996, technical Report.

[35] S. Kapoor, D. J. Marchok, and E.-F. Huang, "Pilot Assisted Synchronization for Wireless OFDM Systems over Fast Time Varying Fading Channels," in *IEEE Vehicular Technology Conference (VTC)*, 1998.

[36] A. M. A., "FPGA Based Implementation of Replica Correlation Using Xilinx System Generator for High Performance Signal Processing Applications," *International Journal of Advanced Research in Electrical, Electronics and Instrumentation Engineering (IJAREEIE)*, vol. 14, no. 2, February 2025.

[37] "Arria V Device Handbook, Volume 1: Device Overview and Datasheet," Intel Corporation, Santa Clara, CA, USA, 2014, accessed: 19-Apr-2026. [Online]. Available: https://docs.rs-online.com/7b15/0900766b8133de6b.pdf

[38] "Arria V GX Starter Kit Board Schematic," Analog Devices, accessed: 2026-04-19. [Online]. Available: https://www.analog.com/media/en/technical-documentation/eval-board-schematic/a5gx_starter_c.pdf

[39] "Data Conversion HSMC Reference Manual," Terasic Inc., accessed: 2026-04-19. [Online]. Available: https://www.terasic.com.tw/cgi-bin/page/archive.pl?Language=English&CategoryNo=67&No=360&PartNo=3

[40] "DCC Data Conversion Demo Documentation (Loopback + NCO Adder System), organization = Altera / Terasic, note = Local file: DCC_DataConvoDemo_1.0.2.pdf."

[41] "Terasic FPGA Development Board Documentation," Terasic Inc., accessed: 2026-04-19. [Online]. Available: https://www.terasic.com.tw/cgi-bin/page/archive.pl?Language=English&CategoryNo=67&No=360&PartNo=3

[42] "AD9254: 14-Bit, 150 MSPS Analog-to-Digital Converter Datasheet," Analog Devices, accessed: 2026-04-19. [Online]. Available: https://www.analog.com/media/en/technical-documentation/data-sheets/ad9254.pdf

[43] "DAC5672: 14-Bit, 275 MSPS Digital-to-Analog Converter Datasheet," Texas Instruments, accessed: 2026-04-19. [Online]. Available: https://www.ti.com/lit/ds/symlink/dac5672.pdf

[44] "Data Conversion HSMC Interface Pin Configuration Manual," Terasic Inc. / Altera, local CD-ROM documentation: data_conversion_hsmc_0a.pdf.

[45] "Quartus Prime Introduction: Using VHDL Designs Tutorial," Intel Corporation. [Online]. Available: https://intelfpga.com