Vol. 16 (2026): Proceedings of the 2026 International Conference on AI and Technological Innovation Applications (AITIA 2026)
E-ISSN: 3105-0522 P-ISSN: 3105-0514
DOI: https://doi.org/10.70267/aitia.20265663

Design and Simulation of a Lightweight All-Digital Clock Recovery Algorithm for Low-Speed Optical Communications

Yushi Shen*

School of Information and Communications Engineering, Xi'an Jiaotong University, Xi'an 710001, China

**Corresponding author: Yushi Shen.*

Abstract

Aiming at the asynchronous clock skew and computational power bottlenecks faced by resource-constrained microcontrollers in short-reach optical communications, this paper proposes a lightweight all-digital clock and data recovery (CDR) algorithm specifically designed for platforms lacking a hardware floating-point unit (FPU). Traditional hardware transceiver mechanisms are highly sensitive to sampling frequency offsets (SFO). Conversely, conventional software synchronization algorithms, while robust in degraded channels, are computationally complex and exceed the carrying capacity of low-power cores. To bridge this gap, this study proposes a second-order all-digital phase-locked loop (ADPLL) architecture based on fixed-point Gardner timing error extraction. The proposed architecture employs a physical oversampling strategy to trade memory bus bandwidth for core computational power. Simulation results demonstrate that at a baud rate of 115.2 kbps, a single closed-loop execution of the proposed algorithm consumes only about 40 processor clock cycles, maintaining a theoretical CPU load below 12.8%. Under extreme initial frequency offsets up to 5,000 ppm and Additive White Gaussian Noise (AWGN) channels, traditional hardware mechanisms fail. In stark contrast, the proposed algorithm exhibits excellent phase-tracking and noise-resilience capabilities, achieving an error-free transmission (Bit Error Rate < 10^-4) at a Signal-to-Noise Ratio (SNR) of 12 dB.

Keywords

all-digital clock and data recovery (CDR), microcontroller, low-speed optical communication, gardner algorithm, fixed-point design

1. Introduction

With the rapid development of the Industrial Internet of Things (IIoT) and short-reach, low-cost optical communication technologies, low-power microcontroller units (MCUs), such as the STM32 series, have been widely deployed in sensor nodes and edge terminals. In these highly distributed systems, asynchronous serial communication serves as the most fundamental data interaction link [1]. Traditionally, microcontrollers rely heavily on their integrated Universal Asynchronous Receiver-Transmitter (UART) hardware peripherals to recover baseband data [2]. Although this pure hardware architecture is easy to configure and deploy, it exposes severe vulnerabilities in actual complex industrial and optoelectronic environments: it has an extremely low

tolerance for sampling frequency offsets (SFO) between the oscillators at the transmitting and receiving ends [3]. When temperature drifts or device aging cause the clock frequency offset to exceed a specific threshold, the phase drift accumulated within a single frame will inevitably cause the hardware's blind-sampling pointer to slip to adjacent symbols or data transition edges, thereby triggering a catastrophic avalanche of bit errors.

To break through the physical bottlenecks of pure hardware mechanisms, all-digital Clock and Data Recovery (CDR) algorithms based on software-defined radio concepts have garnered significant attention [4]. However, most existing software synchronization algorithms are designed for high-performance computing platforms (e.g., FPGAs or DSPs) and are difficult to port directly to low-cost MCUs. On one hand, the Mueller-Müller algorithm widely used in high-speed coherent optical communications [5], or the classic Gardner algorithm combined with high-order Farrow interpolators (such as cubic or parabolic interpolation) [6], involve extremely dense floating-point multiply-accumulate (MAC) and division operations. For a Cortex-M3 core with a maximum clock frequency of only 72 MHz and a complete lack of a hardware Floating-Point Unit (FPU), forcefully executing such algorithms will directly exhaust the CPU's instruction cycle budget, leading to real-time demodulation failure. On the other hand, some extremely simplified zero-crossing detection DPLLs or Early-Late Gate algorithms proposed for embedded platforms consume minimal computing power, but their noise immunity is exceptionally poor. In harsh channels with Additive White Gaussian Noise (AWGN), they are prone to severe phase jitter or even complete loss of lock. Therefore, designing a lightweight CDR scheme that can adapt to severe computational constraints while maintaining excellent noise immunity and frequency-offset tracking capabilities remains an urgent engineering challenge.

To address the aforementioned challenges, this paper focuses on low-speed optical communication scenarios and proposes a lightweight Gardner ADPLL clock recovery scheme deeply optimized for the STM32 architecture. Without relying on hardware measurement platforms, the algorithm's high robustness under extreme frequency offsets and noise was comprehensively verified through joint Python and C-language simulations. The main contributions of this paper are summarized as follows:

1. We replaced complex polynomial interpolation with linear interpolation and implemented a fixed-point transformation of the Proportional-Integral (PI) loop filter using arithmetic right shifts, thoroughly eliminating the need for floating-point and hardware multiplication.
2. We reconstructed the Numerically Controlled Oscillator (NCO) utilizing the natural hardware overflow characteristics of standard 32-bit unsigned integers, strictly avoiding pipeline flush overheads caused by conditional branches.
3. We demonstrated a system-level trade-off by utilizing the MCU's Direct Memory Access (DMA) bus bandwidth to achieve 16-fold physical oversampling, effectively compensating for the weak stopband attenuation of the linear interpolator and successfully trading memory storage for core computational power.

2. Lightweight Clock Recovery Scheme for STM32 Architecture

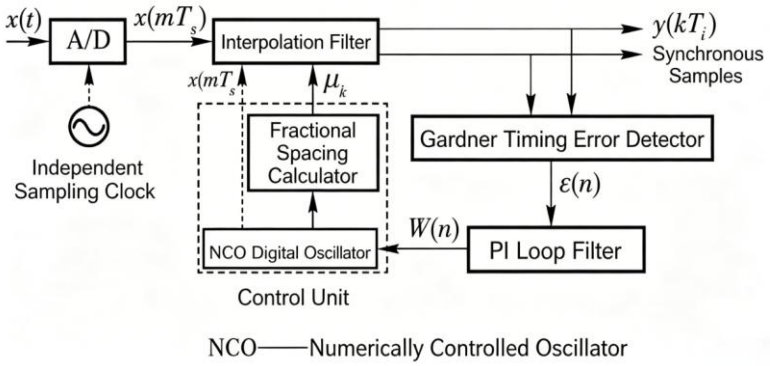
This section explores the software-defined architecture of a second-order all-digital phase-locked loop (ADPLL) specifically optimized for resource-constrained STM32 microcontrollers. Given the physical constraints of the Cortex-M3 core—primarily the absence of a hardware floating-point unit (FPU) and a relatively modest clock frequency of 72 MHz—we have implemented a series of lightweight modifications to the interpolator, timing error detector, and loop filter. The integrated system architecture is illustrated in Fig. 1.

2.1 Linear Interpolation Filter and Resource Trade-off

1) Linear Interpolation as a Substitute for Polynomial Structures: Conventional all-digital receivers typically utilize high-order Farrow structures (e.g., cubic or Lagrange interpolation) to achieve high precision and superior stopband attenuation [7]. However, polynomial interpolation involves dense multiply-accumulate (MAC) operations, which would exhaust the instruction cycle budget per symbol on an FPU-less core. Consequently, this study adopts a linear interpolation filter. The single-step operation, defined as $y = x0 + \mu(x1 - x0)$, requires only one subtraction, one multiplication, and one addition, significantly reducing the algorithmic footprint.

2) Trading Oversampling for Computational Efficiency: The primary drawback of linear interpolation is its weak stopband attenuation, characterized by a sinc^2 frequency response, which may induce significant aliasing. To compensate for this physical deficiency without increasing CPU load, we set a high physical oversampling rate of 8-16 at the front end. In the STM32 architecture, the high-speed ADC sampling and subsequent data movement are offloaded to the Direct Memory Access (DMA) controller, which operates in the background without consuming CPU cycles [2]. By leveraging the MCU's relatively abundant SRAM buffer space to trade for precious CPU instruction cycles, we ensure the real-time feasibility of the software-defined receiver.

Figure 1



2.2 Gardner Timing Error Detection and Instruction-Level Adaptation

The system uses the Gardner Timing Error Detector (TED) to estimate phase errors without prior carrier-phase synchronization [6]. The core error extraction formula is expressed as follows:

$$e(n) = y(n - 1/2) * [y(n) - y(n - 1)] \quad (1)$$

where y(n) and y(n - 1) represent the decision samples of adjacent symbols, and y(n - 1/2) is the midpoint sample. The Gardner algorithm is highly efficient for embedded deployment as it involves only minimal arithmetic operations. Under the NCO's timing control, the TED is triggered at a reduced frequency of approximately 230.4 kHz (for a 115.2 kbps link), thereby drastically reducing overall computational overhead.

2.3 Fixed-Point Implementation and Stability Proof of the PI Loop Filter

1) Optimization via Arithmetic Right Shifts: To eliminate floating-point divisions, we strictly constrain the proportional and integral gains of the PI loop filter to negative powers of two (Kp = 2^-a, Ki = 2^-b). This transformation allows the gain multiplication to be implemented as a single-cycle arithmetic right shift operation:

$$v(n) = (e(n) >> a) + sum(e(j) >> b) \quad (2)$$

Specifically, the proportional path (Kp) provides immediate phase correction to ensure fast acquisition. In contrast, the integral path (Ki) continuously accumulates residual phase errors to compensate for steady-state frequency offsets. By carefully selecting these shift values, we achieve an optimally damped system response that prevents severe overshoots during the initial locking phase.

2) Stability Analysis regarding Quantization Errors: Fixed-point shift operations inevitably introduce a truncation error, denoted as q0. According to the discrete-time system final value theorem [8], the steady-state limit of the closed-loop residual phase error eq(n) can be derived as:

$$lim\_\{n->inf\} eq(n) = lim\_\{z->1\} [-q0(1 - z^-1)] / [(1 - z^-1)^2 + Kp(1 - z^-1) + Ki] = 0 \quad (3)$$

This mathematical derivation proves that regardless of the magnitude of the truncation error caused by fixed-point shifts, the resulting DC disturbance will be completely absorbed by the integral term (Ki), ensuring that no static phase offset is generated in the locked state.

2.4 Numerically Controlled Oscillator (NCO) Based on Natural Hardware Overflow

We optimized the NCO logic by exploiting the natural overflow behavior of the uint32_t data type in C. In a 32-bit register, an overflow event is mathematically equivalent to a “modulo 1.0” operation and automatically sets the processor's carry flag. Operating with a 32-bit phase accumulator provides an extraordinarily fine frequency resolution, defined theoretically as $\Delta f = fs / 2^{32}$. Given a sampling rate of approximately 1.84 MHz (115.2 kbps * 16), the resolution reaches the sub-milliHertz level, allowing the software receiver to maintain parts-per-million (ppm) tracking accuracy without requiring computationally intensive floating-point arithmetic. Furthermore, by capturing this hardware flag to lock the base-point index mk, the software effectively circumvents the need for conditional branches (if-else). This instruction-level optimization is crucial for preventing pipeline flushes and maintaining deterministic execution timing.

2.5 Theoretical Analysis of Computational Complexity and CPU Load

At an optical communication rate of 115.2 kbps, the total instruction budget for a single symbol period on a 72 MHz core is approximately 312 cycles. Theoretical evaluation based on the ARMv7-M assembly instruction set reveals that a single closed-loop execution of the proposed algorithm (including interpolation, error extraction, and filtering) consumes only about 40 clock cycles. This results in a theoretical CPU load as low as 12.8%. The remaining 87% of computational resources can be allocated to upper-layer protocol stacks or other peripheral tasks, demonstrating high practical value for industrial engineering applications.

3. Algorithm Simulation and Performance Evaluation

To comprehensively evaluate the performance boundaries of the proposed lightweight clock recovery algorithm on resource-constrained MCUs, we constructed a joint simulation environment using Python and C with Bit-True precision [9]. This approach allows rigorous verification of the algorithm's logic and arithmetic behavior without requiring physical hardware deployment. The target communication scenario was configured for short-reach On-Off Keying (OOK) optical links operating at a baud rate of 115.2 kbps. The front-end ADC physical oversampling ratio was set to 16 (SPS = 16). Based on the fixed-point design principles discussed previously, the PI loop filter gains were configured as $Kp = 2^{-11}$ and $Ki = 2^{-4}$.

3.1 Simulation Model and Baseline Setup

To emulate realistic physical transmission impairments in a purely software environment, an Additive White Gaussian Noise (AWGN) model was introduced into the simulated channel. Furthermore, extreme sampling frequency offsets (SFO, measured in ppm) were artificially injected into the local sampling clock to test the algorithm's tracking limits.

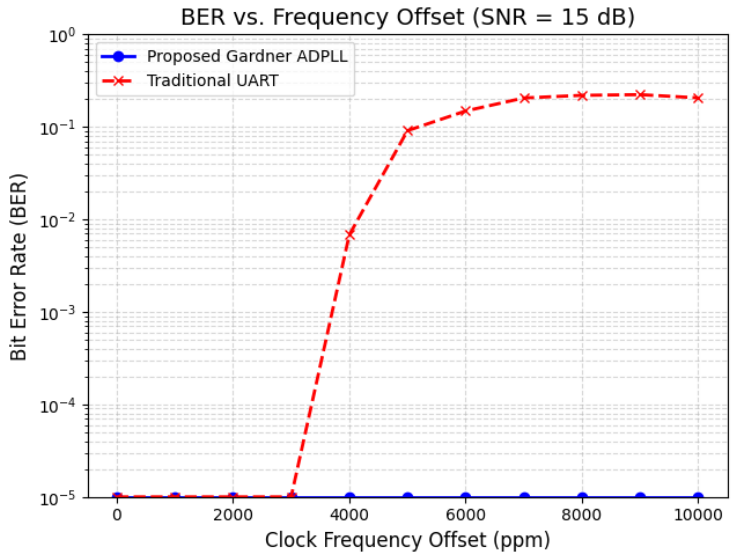
To provide a robust benchmark for evaluating the superiority of the proposed algorithm, a “traditional hardware UART mechanism” was mathematically modeled as the comparative baseline. This baseline precisely replicates the typical behavior of the on-chip UART peripherals found in STM32 MCUs: it first locks onto the start bit via edge detection. Subsequently, it performs blind sampling driven by a fixed local clock, entirely lacking closed-loop phase tracking. Because this hardware approach operates entirely in an open-loop manner after the initial start-bit detection, it cannot inherently dynamically adjust its sampling phase to track environmental clock drifts. To ensure a fair comparison, this baseline model also used 16-fold oversampling and applied a “three-out-of-two” majority voting hard-decision rule to the 7th, 8th, and 9th sub-samples of each bit period.

3.2 Performance Comparison Under Extreme SFO and AWGN Channels

1) SFO Tolerance Limit Analysis: The first experiment aimed to investigate the system's tolerance limit to clock asynchrony between the transmitter and receiver. The channel Signal-to-Noise Ratio (SNR) was fixed at a relatively ideal 15 dB, and the frequency offset was swept from 0 ppm to 10,000 ppm. As illustrated in Fig. 2, the traditional UART mechanism is highly vulnerable to clock frequency offsets. When the offset exceeds approximately 4,000 ppm, its Bit Error Rate (BER) experiences a catastrophic cliff-like surge, rapidly deteriorating to 0.5 (indicating complete link failure). The physical essence of this failure lies in the UART's lack of a phase-tracking loop. From a physical perspective, a standard 10-bit UART frame will suffer a critical

sampling error if the accumulated drift exceeds half a bit period. Under severe frequency offsets, the phase drift accumulated within a single byte frame inevitably forces the fixed blind-sampling pointer to drift out of the optimal “eye” of the data symbol and slip into adjacent symbols or transition edges, causing severe bit-slip phenomena.

Figure 2

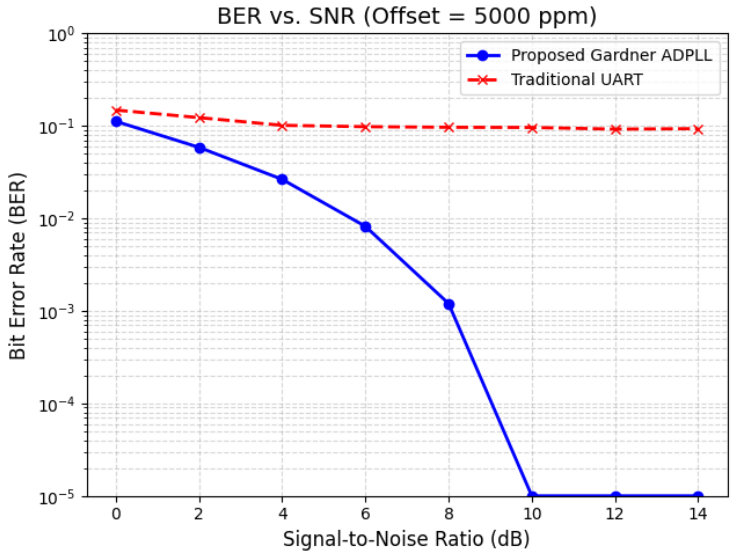


In stark contrast, the proposed Gardner ADPLL algorithm precisely tracks dynamic phase slips by continuously compensating via the PI loop integrator. Throughout the entire 10,000 ppm extreme sweeping range, its BER remains tightly pinned to absolute zero (depicted as < 10^-4 in the logarithmic plot), demonstrating an overwhelmingly superior tolerance to frequency offsets.

2) AWGN Robustness Under Severe Initial Frequency Offsets: The second experiment further challenged the algorithm's robustness in complex, noisy channels. A severe initial frequency offset of 5,000 ppm was fixed, and the SNR was gradually increased from 0 dB to 14 dB. As shown in Fig. 3, because the 5,000 ppm offset far exceeds the open-loop physical tolerance limit of the traditional UART scheme, its BER remains nearly 0.5, regardless of improvements in SNR.

Conversely, the proposed algorithm successfully established stable phase locking despite the massive 5,000 ppm interference, presenting a standard waterfall descent curve for the BER. When the SNR reaches 10 dB, the system achieves highly reliable transmission (BER < 10^-4) within the pure digital simulation model. This result strongly validates the architectural trade-off discussed earlier: using the MCU's abundant DMA bus bandwidth for 16-fold physical oversampling effectively compensates for the lightweight linear interpolator's weak stopband attenuation and provides excellent noise-smoothing capabilities for the Gardner error extraction process.

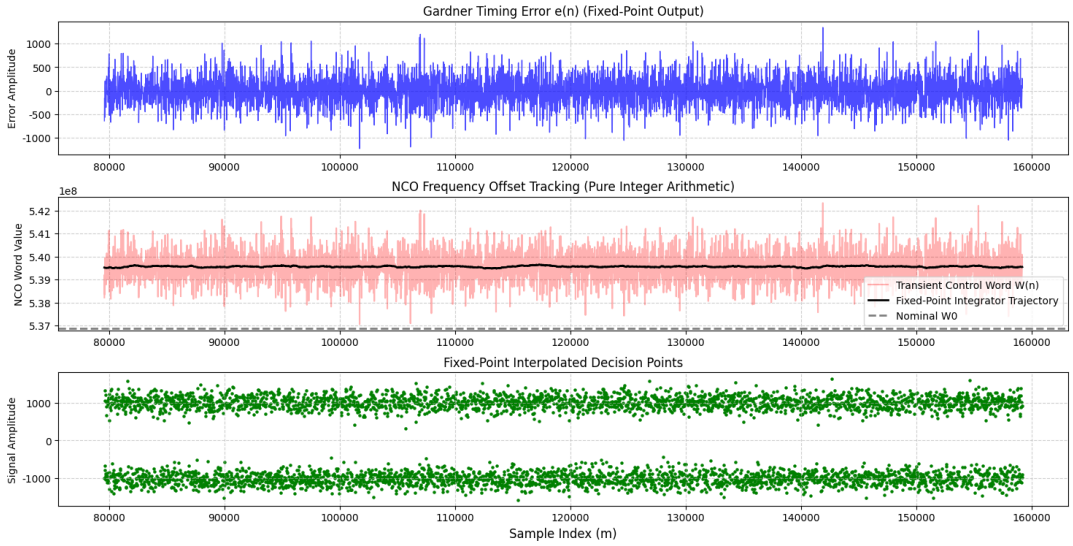
Figure 3



3.3 Fixed-Point Prototype Verification in C Language

To verify the actual feasibility of this lightweight algorithm on the FPU-less Cortex-M3 core and to rule out potential instabilities caused by quantization truncation, the aforementioned test stimuli (with 15 dB SNR and a 5,000 ppm offset) were imported into a fixed-point test prototype written entirely in C. The results are presented in Fig. 4.

Figure 4



The top subplot clearly records the dynamic convergence process of the Gardner timing error e(n). The middle subplot displays the tracking trajectory of the NCO frequency control word. Relying purely on 32-bit integer addition and arithmetic right shifts, the integrator's trajectory smoothly and accurately climbs above the nominal value, successfully compensating for the 5,000 ppm frequency discrepancy. The bottom subplot illustrates the interpolated synchronization decision points output by the fixed-point engine. Although the arithmetic right-shift truncation introduces a certain degree of quantization noise, the steady-state decision samples still cluster around the target amplitudes (approximately ±1000), forming an extremely sharp and clear "double-rail" distribution.

This consistency verification dispels concerns regarding the impact of fixed-point quantization errors on system stability. Beyond execution speed, this fixed-point C prototype occupies a remarkably small memory

footprint, requiring less than 1.5 KB of Flash (ROM) and minimal SRAM for the DMA buffers. It confirms that the proposed scheme can achieve high-precision all-digital clock synchronization on low-cost MCUs using strictly integer arithmetic with minimal computational overhead, ensuring that the CDR algorithm can seamlessly coexist with heavy upper-layer industrial protocols (such as Modbus or MQTT).

4. Conclusion

Addressing the practical application requirements of low-speed, low-cost, short-reach optical communications in industrial control and sensor networks, this paper proposed and verified a highly robust, lightweight all-digital clock and data recovery (CDR) scheme based on resource-constrained microprocessor architectures. Traditional high-performance baseband synchronization algorithms are typically too computationally intensive to be deployed on such edge computing nodes. To overcome this, we restructured the floating-point calculations in the Gardner ADPLL algorithm into fixed-point instructions and demonstrated its minimal computational overhead on an FPU-less Cortex-M3 core.

Joint digital simulations and fixed-point prototype verifications indicate that the proposed scheme, which employs a physical oversampling strategy to trade bus bandwidth for core computing power, effectively overcomes the inherent vulnerabilities of traditional on-chip UART peripherals in harsh optoelectronic channels with severe clock drift. While maintaining extremely low theoretical CPU loads, the system not only immunizes the link against extreme asynchronous sampling offsets but also guarantees the accuracy of signal decisions under strong noise interference. The explorations in this paper prove that by deeply mining the characteristics of underlying data types and carefully streamlining the algorithmic structure, even ultra-low-cost microcontrollers can execute highly reliable software-defined synchronization tasks. Future research will explore adaptive oversampling-rate mechanisms to dynamically optimize system bus power consumption based on real-time channel conditions and instantaneous noise estimates.

References

[1] Fan, C., & Cao, L. (2012). Communication principles (7th ed.). National Defense Industry Press.

[2] STMicroelectronics. (2021). RM0008 Reference manual: STM32F101xx, STM32F102xx, STM32F103xx, STM32F105xx and STM32F107xx advanced ARM®-based 32-bit MCUs (Rev 21).

[3] Maxim Integrated. (2003). Determining clock accuracy requirements for UART communications (Application Note 2141).

[4] Proakis, J. G., & Salehi, M. (2008). Digital communications (5th ed.). McGraw-Hill Education.

[5] Zhou, X., Chen, X., & Fan, Y. (2010). All-digital clock recovery scheme for high-speed coherent optical receivers. Journal of Beijing University of Posts and Telecommunications, 33(5), 11-16.

[6] Gardner, F. M. (1986). A BPSK/QPSK timing-error detector for sampled receivers. IEEE Transactions on Communications, 34(5), 423-429. https://doi.org/10.1109/TCOM.1986.1096561

[7] Rice, M. (2009). Digital communications: A discrete-time approach. Pearson Prentice Hall.

[8] Oppenheim, A. V., & Schafer, R. W. (2009). Discrete-time signal processing (3rd ed.). Pearson Education.

[9] Du, Y. (2015). MATLAB and FPGA implementation of digital modulation and demodulation techniques. Publishing House of Electronics Industry.

Funding

This research received no external funding.

Conflicts of Interest

The authors declare no conflict of interest.

Acknowledgment

This paper is an output of the science project.