Novel Approaches to Resource-Efficient Hardware: Time-Domain Solutions for Clock Synchronization and Error Correction

by

Yang Ge

Submitted in partial fulfillment of the requirements
for the degree of Doctor of Philosophy

at

DALHOUSIE UNIVERSITY
Halifax, Nova Scotia
June 2024

Contents

List of Tables

List of Figures

Abstract

In time-domain processing, the time difference is directly proportional to the amplitude of the analog signal. Within the context of a time variable, it functions similarly to a pulse-width-modulated signal, where the pulse width is directly proportional to the signal's amplitude. Time variables exhibit a dual characteristic, acting both as analog and digital signals. The continuous amplitude of the analog signal is represented by the pulse duration, while the digital aspect is characterized by two distinct values (0 and 1). This duality allows time variables to engage in analog signal processing within a digital environment, a capability that is not shared by purely analog or purely digital variables.

Time-domain processing can be used in various operations, including addition, multiplication, amplification, integration, and quantization. The fundamental building blocks of time-domain circuits include digital-to-time converters (DTCs), time-to-digital converters (TDCs), time multipliers, and time quantizers, etc. In this thesis, we explore three specific applications of time-domain processing: **clock skew compensation system** in three-dimensional integrated circuits (3D-ICs), **minimum finder** (MF) for soft decoders, and **hybrid soft decoder**.

The three time-domain applications — clock skew compensation system in 3D-ICs, MF for soft decoders, and hybrid soft decoder — share an intrinsic connection in their reliance on the unique advantages of time-domain signal processing. Time-domain processing can be utilized for various operations, such as addition, multiplication, amplification, integration, and quantization. The fundamental building blocks of time-domain circuits include DTCs, TDCs, time multipliers, and time quantizers, among others.

In summary, these three applications demonstrate the versatility and efficiency of time-domain processing in modern integrated circuits and decoding technologies. By optimizing time-domain signal processing in various application scenarios, it is possible to significantly enhance system performance and reduce power consumption, thereby meeting the growing demand for high performance and low power.

List of Abbreviations Used

TDP	time domain processing
TSV	Through-silicon vias
VCDL	voltage-controlled delay line
DLL	delay lock loop
PG	pulse generator
DCDL	Digitally controlled delay line
3-D IC	three-dimensional integrated circuit
FEC	Forward error-correction
SDDs	soft decision decoders
MVA	minimum values and addresses finder
TD	time-domain
TS	tree structure
AETD	address-embedded time-domain
FPGA	field-programmable gate arrays
ILA	integrated logic analyzer
BRAM	block ram
SNR	signal to noise ratio
BER	bit error rate

Acknowledgements

I would like to take this opportunity to express my deep gratitude to everyone who supported me throughout my graduate studies.

First of all, I would like to thank my supervisor, Dmitry Trukhachev, and my co-supervisor, Dr. Kamal El-Sankary, for their invaluable guidance, constant encouragement, and incredible patience. Their extensive knowledge and experience greatly contributed to my research journey. They were always ready to provide new ideas for my project, as long as I demonstrated self-motivation and progress. They dedicated individual attention to each of their students, inspiring and assisting us in our research studies.

I am also deeply grateful to Dr. Masry and Dr. Hammad for serving as my committee members and to all the other professors in our faculty who provided advice and support. Special thanks to our department secretaries, Nicole, Tamara, and Vanessa, for their assistance, and to Mark for his technical support. Additionally, I would like to express my gratitude to Dr. Lihong Zhang from Memorial University for serving as the external examiner.

I want to express my thanks to my friends, Ming Yan, Abolfazl and Xiaoting, for their valuable research suggestions throughout my Ph.D. program. Moreover, I owe a great debt of gratitude to my family for their unwavering support and generosity, without their love and encouragement, none of this would have been possible. Finally, I would like to express my deepest appreciation to my girlfriend, Xiaohan Dai, for her constant support.

Chapter 1 Introduction and motivation

This thesis explores the innovative applications of time-domain processing (TDP) in the realm of three-dimensional integrated circuits (3D-ICs), soft decoders, and hybrid decoding systems, pushing the boundaries of existing technologies and introducing new methods for clock skew compensation, error correction, and signal decoding. The work is divided into three major applications, each addressing critical aspects of circuit design and signal processing.

1.1 A novel approach to clock skew compensation in 3D-ICs

Moore's Law, which predicts the doubling of transistors on an integrated circuit every two years, has shaped the landscape of semiconductor technology since the mid-20th century. As we approach the physical limits of transistor miniaturization, the semiconductor industry faces significant challenges that hinder further scaling down in two-dimensional (2D) planes. This limitation has catalyzed the shift towards three-dimensional integrated circuits (3D-ICs), where multiple dies are stacked vertically. This architectural innovation effectively multiplies the number of components per chip, ostensibly continuing the trend set by Moore's Law but within a three-dimensional space.

Despite its promising potential, the 3D integration approach introduces complex technical challenges, particularly in the realm of inter-die communication. One of the most critical issues is clock distribution—ensuring that clock signals are synchronized across multiple dies. Traditionally, methods like bonding wires and more recently, through-silicon vias (TSVs),

have been used to connect dies. TSVs, especially, have been favored for their ability to offer high density, low latency, and reduced power consumption. However, they also introduce variabilities such as delay fluctuations due to process, voltage, and temperature (PVT) variations, which can result in significant clock skew [1]-[9].

Current state-of-the-art solutions for die-to-die clock synchronization typically involve multiple TSVs to create robust forward and return paths, aiming to minimize skew and mismatches [1]-[8]. Nonetheless, these solutions are often limited to configurations involving only two dies and can suffer from substantial synchronization failures due to high defect rates and sensitivity to PVT variations [9]-[10]. Additionally, systems utilizing dual-locking digitally controlled delay lines (DCDLs) or delay-locked loops require intricate matching and high-resolution delay lines, which consume considerable chip space and often retain a residual skew post-lock-in [5].

Recently, research presented in [6] introduced a novel all-digital delay-locked loop topology that utilizes only one TSV for transmitting clock signals, effectively eliminating mismatches associated with multiple TSVs. However, this approach relies on assumptions such as identical buffer delays and perfectly matched delay lines, which are often impractical and can result in significant residual skew after lock-in. Another study [7] described a single DCDL-based architecture that reduces the need for multiple DCDLs and phase detectors (PDs), but still faces challenges with variations between two TSVs. Additionally, the analog dual-delay-locked loop (D-DLL) proposed in [8] for two-die clock synchronization achieves only a 2 ps skew but assumes perfect matching of delay lines and incurs high chip area and power consumption.

Other researches [11]-[13] have developed methods using a balanced tree structure for clock distribution across multiple dies, aiming to minimize the effects of within-die process and loading variations. Yet, no effective topology for die-to-die clock skew compensation for systems with more than two dies has been introduced. Current methodologies [3]-[8] are limited to supporting clock synchronization only up to two dies, underlining the urgent need for new approaches that can extend beyond these limitations.

The existing works [3]-[8] typically focus on a frequency of 1 GHz, common in digital architectures. In contrast, the analog circuit described in [8] can operate at a higher frequency of 2 GHz. The principle of 3-D-IC technology is versatile, potentially applicable to any frequency or protocol to enhance integration density. In this thesis, we demonstrate a proof of concept at a frequency of 1 GHz.

1.2 An efficient address-embedded time-domain implementation of minima finders for soft error-correction decoders

The fundamental challenge in digital communications is the corruption of transmitted data by noise in the channel. This issue is effectively addressed using forward error-correction (FEC) codes, which integrate redundant parity-check bits into the transmitted message, enabling error correction through decoding mechanisms [14]. This paper focuses on the applications of soft-decision decoding (SDD), which evaluates the reliability of received data bits, often expressed as log-likelihood ratios (LLRs), serving as a cornerstone for modern error-correction strategies [14]. Specifically, we explore high-throughput, low-power SDD applications in fiber-optical communications,

which are required to deliver throughputs exceeding 400 Gbps over a single wavelength while adhering to stringent reliability, latency, and power constraints [18].

Among the classes of SDDs, notable algorithms include Chase [19], ordered reliability bits guessing random additive noise decoding (ORBGRAND) [12], direct error-pattern testing (DEPT) [18], and ordered reliability DEPT (ORDEPT) [21]. These techniques necessitate that SDDs provide soft outputs, crucial for the soft iterative decoding of concatenated codes. Figure 1(a) depicts a generic block diagram of a test-pattern based soft-input soft-output (SISO) decoder, where LLRs obtained from the channel demodulator are fed into the decoder. The decoder then commences by identifying the positions and magnitudes of the least reliable LLRs, which is vital for decoders operating near pre-FEC thresholds of standard optical FEC codes [15], [16]. These decoders must locate at least 5 to 8 minima among a substantial number of quantized input LLR values, which often exceeds 256 [17], [18], [22]-[25]. The hardware of the minimum finder (MF) block significantly adds to the complexity of SDDs [18]-[26], and its latency frequently becomes a bottleneck, emphasizing the need for an efficient MF block to enhance throughput and reduce complexity and power consumption [18]-[26].

The positions of the smallest LLRs identified are crucial for generating test error patterns and comparing them based on likelihood, as utilized in algorithms like Chase and DEPT. The most probable error patterns are then used for error correction and generating the output LLRs [27].

Numerous studies have focused on resolving the minima finding challenges.

Generally, parallel [28]-[31] and serial [32], [33] architectures are the two prevalent approaches to determine minima values and their indices. The architecture dedicated to finding the first two minima and their indices is discussed in [28]. The parallel tree-structure (TS)-based architecture offers lower latency and is more hardware-efficient than sorting-based structures, which require tracking indices through all previous comparisons [28]. Modified two-minima finding TS architectures presented in [29], [30] offer reduced hardware complexity and latency. However, as the number of inputs increases, the index-generating unit becomes highly complex, resulting in a significant increase in the overall area and latency of the critical path. While finding just two minima may suffice for min-sum decoding of low-density parity-check codes, more complex error-pattern based decoding requires additional algorithms to identify more minima, as demonstrated in [31]. Iterative architectures [32] find minima sequentially but at the cost of increased hardware resources and latency.

Time-domain processing (TDP) provides advantages in latency, area, and energy efficiency compared to traditional digital processing. In TDP, a multi-bit quantized digital data value is represented by the pulse width of a single signal [33]-[35]. The conversion from digital to time domain involves a digital to time converter (DTC), consisting of a delay chain and a multiplexer (MUX). The delay chain generates multiple delayed versions of the input pulse, and the MUX selects the appropriate delayed version as the output, as efficiently demonstrated in a serial TD-based minima value finder [33].

This thesis introduces a novel architecture can be directly embedded into a fully TD Bose-Chaudhuri-Hocquenghem (BCH) decoder implementation,

enabling both the input and output of the minima value and address (MVA) finder to be in TD. The MVA finder that effectively identifies the six smallest values and their positions in a vector of 256 quantized input numbers. This architecture is specifically tailored for SDD implementation in high-speed optical communications using the open forward error correction (oFEC) code [26]. Our proposed architecture synergistically combines the benefits of both parallel TS and TDP techniques to achieve optimal performance. Compared to parallel architectures in [28]-[31], our TDP-based module significantly enhances throughput, particularly when inputs are highly quantized. The parallel TS-based architecture employed results in lower latency compared to the serial structure [32]. Furthermore, we introduce a novel format of address-embedded TD (AETD) signals that facilitate higher speed and lower complexity relative to conventional TD architectures [33]. The proposed algorithm implemented using TSMC 65-nm CMOS technology, this minima finder outstrips current solutions in both latency and complexity, making it a strong candidate for next-generation high-throughput forward error-correction decoders in fiber-optic communications.

1.3 A novel hybrid soft decoder

Contemporary fiber-optical communication systems operate at speeds of hundreds of gigabits per second per wavelength and require output bit error rates (BERs) below 10^{-15}. Achieving this necessitates advanced digital signal processing (DSP) and forward error-correction (FEC) techniques [36]. Convolutional product-like FEC schemes are widely used in fiber-optical communication standards for their substantial net coding gains with reasonable complexity and latency. Notable examples include the concatenated Hamming/staircase code in 400ZR [37] and the open FEC

(oFEC) [38] adopted by the Optical Inter-Networking Forum (OIF) for the open ROADM multi-source agreement schemes targeting 400 Gbps communications. The oFEC is also being considered as a potential FEC solution for 800 Gbps links [39].

Many optical FECs employ BCH component codes with double- or triple-error-correcting capabilities due to their suitability for Hard Decision Decoder (HDD) and Soft Decision Decoder (SDD) implementations [40], [41]. Both the oFEC and the staircase code included in 400ZR [42] are concatenated codes constructed from BCH component codes. Specifically, the oFEC is made from extended (256, 239, 2) double-error-correcting BCH codes and decoded using a soft-input soft-output (SISO) decoding algorithm that typically incorporates a Chase II SDD [43] with output softened by the Pyndiah algorithm [44]. Most soft BCH decoders are similarly designed and implemented using various adaptations of the Chase algorithm [43].

Achieving high throughput under constraints of power and latency requires an SDD implementation that employs numerous HDDs working in parallel on different test patterns. Conversely, a hardware-optimized BCH HDD should utilize an alternative to the Chien search technique, which occupies a significant portion of the circuit area and has high latency [46]. Although parallel implementations of the Berlekamp-Massey [47], Peterson [48], and inverse-Peterson [49] algorithms exist, there remains a need for a more efficient decoder to achieve higher speeds and throughputs beyond 800 Gbps under strict power constraints.

This thesis introduces a cutting-edge hybrid soft decoder architecture that

combines the precision of traditional digital decoding with the operational efficiency of time-domain processing (TDP). This novel approach aims to alleviate the high power and area demands of conventional soft decoders by integrating faster, more efficient TDP components. This integration not only enhances decoding performance in challenging noise conditions but also significantly reduces the power consumption and physical footprint of the decoding process. Together, these advancements highlight the potential of time-domain processing to enhance the efficiency and effectiveness of modern digital communication systems, offering significant improvements over traditional methods and paving the way for future innovations in integrated circuit design.

The proposed decoder, synthesized using the latest technology in the Synopsys Design Compiler and fabricated on a TSMC 65-nm CMOS process, represents a significant advancement in decoder design [50]-[51]. It has been thoroughly validated through FPGA-based simulations, confirming its capability to meet the demanding conditions of practical optical communication systems [52]-[53].

1.4 Research Objectives

This research aims to use the benefits of time-domain processing (TDP) to address challenges in digital signal processing across three pivotal applications: clock skew compensation in 3-dimensional integrated circuits (3D-ICs), the development of minima finders for soft decoders, and the creation of a novel hybrid soft decoder. Each application seeks to leverage the intrinsic advantages of TDP, such as reduced latency, decreased physical footprint, and

enhanced energy efficiency, to surpass the capabilities of traditional digital processing methods.

- Clock Skew Compensation in 3D-ICs

The primary goal in this application is to achieve precise clock synchronization across multiple dies in 3D-ICs operating at 1 GHz. The objective focuses on minimizing residual skew while utilizing the fewest possible resources like through-silicon vias (TSVs) and digitally controlled delay lines (DCDLs). The expected outcomes include a reduction in system complexity and power consumption, alongside shortened delay times when compared to existing methodologies. Verification of these improvements will be conducted through detailed synthesis and post-layout analysis using industry-standard electronic design automation (EDA) tools, such as Synopsys Design Compiler and Cadence Innovus.

- Minima Finders for Soft Decoders

The second application targets the enhancement of minima finder algorithms within soft decoders. The specific objective is to efficiently process and identify the addresses of the six smallest values from a dataset of 256 6-bit inputs, operating purely in the time domain. The new architecture is anticipated to offer significant improvements in terms of system complexity, critical delays, chip area, and power consumption. These advancements will be substantiated through rigorous synthesis and post-layout assessments, aiming to outperform existing solutions as documented in recent literature.

- Development of a Hybrid Soft Decoder

The final application focuses on developing a hybrid soft decoder that incorporates time-domain components into a traditionally digital decoder framework. This involves the integration of time-domain implementations for the minimum finder (MF), LLR summation, and address finder—utilizing the MF module designed in the second application—while maintaining digital components for more complex processing tasks. The hybrid architecture will be evaluated using multiple EDA tools to ensure its functionality and performance exceed those of existing digital decoders. Final validation will occur through extensive simulation on FPGA boards, demonstrating the decoder's enhanced performance and operational efficiency.

Overall, these objectives are designed to push the boundaries of current technology using time-domain processing, aiming to set new standards for performance and efficiency in integrated circuit design and digital processing.

1.5 Thesis Contributions

The thesis combines two published journal papers ([49]-[50]) and one submitted journal paper. The three papers—3D-ICs, the MF for soft decoders, and the hybrid soft decoder—are inherently connected through their utilization of TDP techniques to enhance performance, reduce power consumption, and improve overall system efficiency.

[49] presents a single-TSV and single-DCDL (STSD) approach for multi-die clock synchronization.

The proposed MVA [50] introduces a novel algorithm that accurately determines the six smallest values and their corresponding indices within a vector of 256 quantized positive real numbers.

The last application proposed a hybrid soft decoder, which combines elements of both time-domain and digital-domain processing to enhance performance and efficiency.

In conclusion, these papers illustrate the versatile and complementary nature of time-domain processing in enhancing various aspects of modern electronic systems. By utilizing the inherent strengths of time-domain techniques—such as low latency, reduced power consumption, and simplicity of implementation—these applications showcase innovative solutions that address the growing demands for high performance and energy efficiency in advanced integrated circuits and communication systems.

1.6 Thesis Outline

The rest of this thesis is organized as follows, each chapter of chapter 3-5 presents one published journal paper or submitter paper: The thesis is from these three papers ([54]-[55] and one submitted paper).

- Chapter 2: This chapter introduces the necessary background of the research.
- Chapter 3: This chapter details the architecture, circuit implementation, and simulation results of the STSD approach for multi-die clock synchronization.

- Chapter 4: This chapter presents the MVA approach along with its post layout experimental results.
- Chapter 5: This chapter introduces the proposed hybrid decoder, including its block diagram, circuit design, and FPGA implementation results.
- Chapter 6: This chapter concludes the thesis and outlines future work.

Chapter 2 Background

2.1 Clock skew problems in 3D-ICs

Clock distribution is a fundamental aspect of integrated circuit design that ensures synchronous operation across all parts of a chip. In a typical 2D integrated circuit, clock signals are distributed using a network of buffers and interconnects that propagate the clock signal from a central point to various components within the IC. However, as the transition to three-dimensional integrated circuits (3D-ICs) continues, the complexity of clock distribution increases significantly. This complexity arises from the need to synchronize multiple stacked dies, each potentially containing millions of transistors, within acceptable skew and jitter limits.

2.1.1 Clock Skew and Its Impact

Clock skew refers to the variation in timing with which the clock signal arrives at different parts of the circuit. In the context of 3D-ICs, skew can be particularly problematic due to the vertical stacking of dies, which introduces additional variables such as inter-die capacitance and resistance. These variables can affect the timing of the clock signal as it passes through different layers. Excessive skew can lead to setup and hold time violations, ultimately causing functional errors or leading to a decrease in the overall performance of the IC.

1. Current Clock Distribution Techniques and Their Limitations

- Bonding Wires and Traditional TSVs: Initially, bonding wires were used to connect different layers in multi-die configurations. However, this method was soon replaced by through-silicon vias (TSVs) because TSVs can reduce latency and power consumption while increasing the density of interconnects.

Despite these advantages, traditional TSVs introduce their own challenges, including:

- PVT Variations: Process, voltage, and temperature variations can affect the delay introduced by TSVs, leading to unpredictable clock skew.
- Complexity in Multi-Die Systems: Most existing TSV-based clock distribution strategies are limited to configurations of two dies due to increased complexity and higher likelihood of synchronization errors as more dies are added.

2. Dual-Locking DCDLs and Delay-Locked Loops: These systems use feedback mechanisms to adjust the delay of the clock signal to match a reference timing. While effective in minimizing skew in two-die systems, they are:

- Resource-Intensive: These systems require precise matching of delay lines and control circuits, which can consume significant chip area and power.
- Scalability Issues: Scaling these systems to more than two dies increase the complexity exponentially, often making them impractical for larger 3D-IC stacks.

2.2 The Open Forward Error Correction (oFEC) code and system model

This section introduces the necessary background and model of the 2nd and 3rd papers. Help readers better understand the proposed architecture.

Basic Error Correction Mechanism

Error correction mechanisms are vital for ensuring the integrity of data in digital communication and storage. These mechanisms work by adding extra data, known as redundancy, to the original information before transmission or storage. This redundancy helps in detecting and correcting errors that might occur due to disturbances like noise and interference.

One of the simplest error correction strategies is the **repetition code**. In this method, each bit of the original data is replicated multiple times to create redundancy. For example, using a repetition factor of three, a bit sequence "101" would be transmitted as "111 000 111". The advantage of this approach is its simplicity in implementation; however, it is not bandwidth efficient as it triples the amount of data transmitted. Error correction is achieved by majority voting; for instance, if "110" is received, it is corrected to "1", based on the majority of the bits.

A more sophisticated and efficient method is the **Hamming code**, which not only corrects single-bit errors but also detects double-bit errors. This code inserts multiple parity bits at specific positions within the data. These positions correspond to powers of two (e.g., 1, 2, 4, 8). Each parity bit is calculated to ensure a specific parity (even or odd) across particular groups of bits in the data. For example, in a (7,4) Hamming code—indicating seven bits in the codeword, of which four are data bits and three are parity bits—the parity bits

are arranged to check overlapping groups of bits. If any of the parity checks fail upon reception, the pattern of the failures will indicate the exact location of the error, allowing the system to correct the specific erroneous bit.

Thus, while repetition codes provide a basic level of error protection suitable for scenarios where bandwidth usage is not a primary concern, Hamming codes offer a more refined solution that balances redundancy with error detection and correction capabilities, making them ideal for applications where data integrity is critical without excessively increasing the data load.

Bose-Chaudhuri-Hocquenghem (BCH) codes represent a significant advancement in the field of error-correcting codes, designed to correct multiple errors in data transmission and storage systems. These codes belong to a class of cyclic error-correcting codes that can be specifically tailored to correct different numbers of errors, making them extremely versatile and widely applicable in various digital communication and storage technologies.

BCH codes are constructed using algebraic methods, primarily through the application of finite field arithmetic. The strength of BCH codes lies in their capability to define the error correction radius explicitly—that is, the maximum number of errors that can be corrected by the code. This is achieved by setting parameters during the code's creation, which dictate the number of check symbols added to the data. These parameters directly influence the error-correcting capability and complexity of the code.

The error correction radius of a BCH code is a crucial concept. It defines the half the minimum distance of the code, which mathematically represents the

number of errors that can be independently corrected within a codeword. For instance, a BCH code designed with an error correction radius of three can detect and correct up to three errors in each codeword, regardless of the codeword's length. This ability to maintain integrity over a specified number of errors makes BCH codes particularly useful in environments where data transmissions are susceptible to noise and other errors that can corrupt multiple bits simultaneously.

Overall, the development of BCH codes has allowed for more reliable data transmission by providing robust mechanisms to detect and correct multiple simultaneous errors. Their application ranges from satellite communication to consumer electronics, highlighting their importance in modern digital communications infrastructure.

In the study of error-correcting codes, the **(236, 239) extended BCH code** represents a specific configuration within the broader family of BCH codes. This code is defined by its length of 239 bits and is capable of correcting errors across 236 of these bits. Its design makes it a powerful tool in scenarios where robust error correction is paramount, such as in digital communications and data storage systems prone to interference and data degradation.

The primary attribute of the (236, 239) code is its high error correction capability relative to its length, providing a significant level of redundancy while maintaining data efficiency. The code's structure allows for the correction of multiple errors, enhancing the reliability of data transmissions over noisy channels. This makes the (236, 239) code particularly valuable in applications where the integrity of each bit is critical to the overall data quality

and where errors are likely to occur due to environmental factors or system noise.

Moreover, the (236, 239) code employs a systematic encoding method that facilitates easy implementation in hardware and software. The encoding process embeds redundancy in a way that error detection and correction procedures can be executed efficiently, thus minimizing the processing overhead. This efficiency is crucial in high-speed communication environments where processing delays can impede performance.

Overall, the (236, 239) code exemplifies a balanced approach to error correction, offering strong protection against data corruption with minimal compromise on data throughput and system performance. Its integration into communication systems significantly enhances data reliability, ensuring that even in the presence of substantial noise, the integrity of the transmitted information is preserved. This characteristic is essential for maintaining the efficacy of modern digital communication infrastructures.

Soft Decoding and Chase Algorithm

Soft decoding represents a sophisticated method in error-correction coding that leverages probabilistic information about the received signals to enhance decoding accuracy, particularly under noisy conditions. A prominent example of this approach is the **Chase algorithm**, which is widely recognized for its ability to improve error correction performance beyond conventional hard decision decoding methods.

The Chase algorithm is a form of soft decoding that initially performs a hard

decision on the received data to form a tentative decoded sequence. It then identifies a subset of bits with the lowest reliability, typically based on the smallest absolute values of the LLRs. These bits are considered the most likely to have been corrupted. The algorithm generates multiple test patterns by flipping these uncertain bits in all possible combinations, creating a set of candidate codewords from the original hard-decision output.

For each candidate codeword generated, the Chase algorithm recalculates the syndrome values, a step which is essential for identifying and verifying the presence of errors. By comparing these syndrome values with the original, the algorithm can determine which of the candidate codewords most likely represents the transmitted data. This comparison heavily utilizes the soft information provided by the LLRs, enhancing the decision-making process by considering the probabilistic confidence levels associated with each bit.

Ordered Reliability Direct Error Pattern Testing (ORDEPT) algorithm

The **ORDEPT algorithm**, an advanced decoding technique, is designed to enhance the efficiency and accuracy of error correction in digital communication systems. This algorithm stands out due to its systematic approach to decoding complex codewords, making it highly effective in environments where data integrity is paramount.

At its core, the ORDEPT algorithm employs an ordered statistic decoding strategy, which focuses on leveraging the reliability of received signal information to prioritize the processing of bits within a codeword. This method involves arranging the received bits according to their reliability measures, typically derived from their likelihood ratios or soft metrics. Such

organization allows the algorithm to first focus on the most reliable bits, thereby increasing the probability of accurate error detection and correction.

The algorithm then iteratively decodes the codeword by examining a subset of the least reliable bits, which are most susceptible to errors. By generating hypotheses about these bits and testing these hypotheses against the overall codeword structure, the ORDEPT algorithm can effectively isolate and correct errors. This iterative approach ensures that each decision is informed by the maximum available context, significantly enhancing the decoding process's accuracy.

The Open Forward Error Correction (oFEC)

The oFEC is a new generation error correction coding scheme for fiber optical communication, characterized by high efficiency and code rate, and is standardized by Open ROADM [56]. According to the standard documentation, oFEC achieves a Net Coding Gain (NCG) of 11.1dB for BPSK/QPSK with a pre-FEC BER of $2e^{-2}$ and 11.6dB for 16QAM following three soft-decision iterations. In a recent study [57], the Chase-Pyndiah algorithm was suggested for using oFEC to explore its effectiveness across various numbers of error patterns, particularly focusing on power dissipation.

The structural inspiration for the oFEC code comes from braided block codes (BBCs) [58], a type of sliding code first introduced in 2002 (Lentemaier, Truhachev, and Zigangirov, 2002, pp. 190-193), (Truhachev, Lentemaier, and Zigangirov, 2003). BBCs incorporate elements like convolutional codes and diamond codes. The braided coding approach, akin to that of the oFEC, is designed for iterative decoding within a continuous data stream and features

interconnected horizontal and vertical elements of two-component block codes. The Ordered Reliability Bits Guessing Random Additive Noise Decoding (ORBGRAND) algorithm, detailed in [59], and the recently introduced ORDEPT decoding algorithm will both be implemented in the oFEC code framework.

Generally, the Chase algorithm produces error sequences by considering the bit's reliability, which is derived from the absolute LLRs output from the AWGN channel. The number of least reliable bit (LRB) positions is denoted as p, and subsequently, 2^p error sequences are synthesized. This implies that the number of error sequences increases exponentially as more LRBs are utilized for decoding. Consequently, the complexity of the decoding process also increases significantly. As a result, both the efficiency and accuracy of decoding are comparatively low.

2.2.1 System model

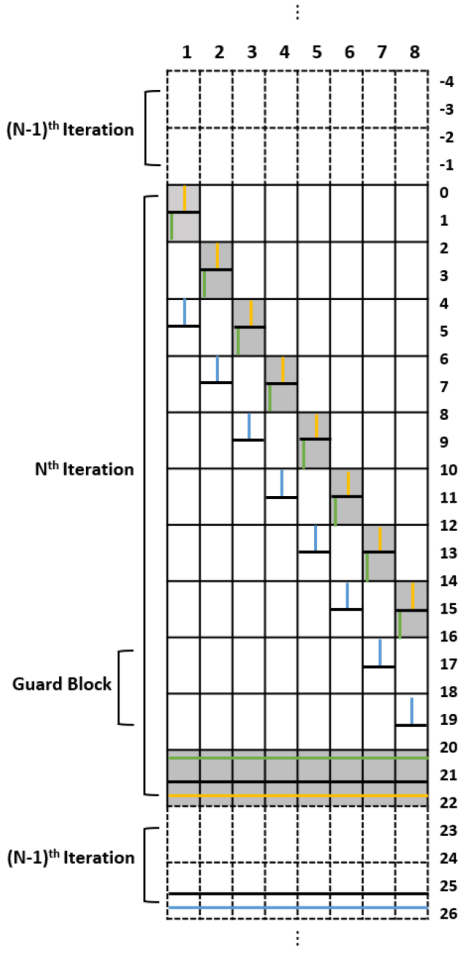


Fig. 2.1. Single iteration oFEC block window.

Each iteration block of the encoder or decoder is structured as a unit oFEC configuration, as illustrated in Fig. 2.1. The matrix is divided into a series of B * B square blocks (where B=16), representing the dimension of each unit block. The (239, 256) extended BCH codeword is utilized as the component codewords. Consequently, the memory window is segmented into N/B block columns, with N defined as half the length of the component codeword, N=128. The codewords within the oFEC are typically split into front and back sections, with the first N\) bits referred to as “front” bits, and the remaining

N\bits termed “back” bits. Each memory bit's position is identifiable by a quadruple (R, C, r, c), where R and C denote the block row and column numbers as outlined by the square blocks, and r and c indicate the specific row and column numbers within each B * B block.

The matrix block is categorized into three sections as depicted in Fig. 2.1. The block row numbers from 0 to 15 designate the location for the "front" bits, corresponding to bits 0 to 127 of the component codeword. Conversely, the "back" bits, encompassing bits 128 to 255 of the component codeword, are allocated to block row numbers beginning after 20. The intervening space functions as a guard-block area, necessitating that the row number be twice the value of G and be even. The magnitude of G directly influences both the execution time of the system and the dimensions of the matrix block. As G increases, so too does the pipeline latency, allowing for more codewords to be encoded or decoded in parallel, albeit at the cost of expanding the memory block's size.

2.2.2 Galois Filed

This work is substituted with a pure combinational logic gate design that developed by simplifying the error-location polynomial over **Galois field**. In the context of error-correcting codes, particularly those like BCH and Reed-Solomon codes, the use of polynomials defined over a Galois field plays a central role. A Galois field, often denoted as GF(q), where q is a prime power, provides a finite field of elements with which to construct polynomials that are crucial for coding and decoding processes.

A polynomial over a Galois field, such as $GF(2^m)$ for binary BCH codes, is

constructed using elements from that field as coefficients. This arrangement ensures that all arithmetic operations within the polynomial, such as addition and multiplication, are performed modulo a prime number or a prime power, depending on the characteristics of the field. For coding applications, this finite nature of the field facilitates the design of algorithms that can efficiently manage and correct errors in data transmissions.

In practical terms, these polynomials serve as the basis for generating codewords in error-correcting codes. The roots of these polynomials, which are also elements of the Galois field, are crucial in the construction and definition of the code. For example, in BCH codes, the chosen polynomial's roots are specifically selected to achieve a desired minimum distance between codewords, which directly influences the code's error-correcting capability.

2.2.3 Encoder and decoder

The oFEC system utilizes two dedicated memory buffers, one each for the encoder and decoder, to store the encoded and decoded bits during specific intervals. The architecture of the oFEC code differs from typical block codes such as turbo product codes, which are determined by the lengths of the codewords C1 and C2 in the block’s rows and columns. In contrast, the oFEC's block dimensions are semi-infinite, dependent on the number of iterations and the specific component codeword used. In this research, the dimension of the block matrix for each iteration measures 352 by 128. Given the codeword is segmented into front and back bits, 128 columns are reserved for horizontally positioning the codeword within these matrices. This matrix comprises 32 rows for input codewords, 64 rows as a guard block, and 256

rows dedicated to front bit storage. Both the encoder and decoder utilize the same block matrix configuration to ensure synchronization of messages.

To encode a (256,239) extended BCH codeword, a generator matrix G and a parity check matrix H are essential. The codeword polynomial is expressed as:

$$C(x) = c_0 + c_1 + c_2x^2 + \cdots + c_{n-1}x^{n-1} \quad (1)$$

where a, a^2, a^2, a^{2t} are the roots of the generator polynomial. The matrix H must adhere to the following structure:

$$H = \begin{bmatrix} a^{a-1} & a^{n-2} & & a & 1 \\ a^{3(n-1)} & a^{3(n-1)} & \dots & a^3 & 1 \\ a^{5(n-1)} & a^{5(n-1)} & \dots & a^5 & 1 \\ \cdots & \cdots & \dots & \cdots & 1 \\ a^{(2t-1)(n-1)} & a^{(2t-1)(n-1)} & \cdots & a^{2t-1} & 1 \end{bmatrix} \quad (2)$$

The encoded message is derived by convolving the input message with the generator matrix G, structured as follows:

$$G = |I_K|P \quad (3)$$

$$GH^T = [I_K \quad P]\begin{bmatrix} P \\ I_{n-K} \end{bmatrix} = 0 \quad (4)$$

$$c = mG \quad (5)$$

Decoding is performed using the Chase-II Pyndiah algorithms, recognized for their effective soft-in soft-out (SISO) iterative decoding capabilities. This method handles an exponentially increasing number of least reliable bits

(LRB), adding complexity to the system. Additionally, alternative error pattern selection methods such as Direct Error-Pattern Testing (DEPT) and DEPT-ORBGRAND complement the oFEC system by offering varied approaches to handling error patterns compared to Chase-II. A comprehensive overview of these algorithms will be provided in the subsequent section.

2.2.4 oFEC Adapted Decoding Algorithm

2.2.4.1 Chase algorithm

The Chase algorithm, first introduced by David Chase [60], represents a class of decoding algorithms that leverage channel measurement information. This approach to utilizing channel measurement data for block code decoding was initially introduced by Wagner decoding [61]. Given that this method is effective across all block codes, it can significantly enhance the error-correcting capabilities of any code. Suppose the received sequence is denoted by Y and that the codewords C_i (e.g., C1, C2, C3) are encapsulated by Y within a sphere of radius (d-1)/2, as depicted in Fig. 2.2. It is noted that only the codeword C2 falls within the sphere of the received codeword Y, thus acting as one of the potential candidates codewords for Y. This concept underpins three algorithms for generating sets of codeword C_i, and among these, the Chase-II algorithm is employed in this research. The absolute log-likelihood ratios (LLRs) determine the reliability of each bit in \(Y\); higher absolute LLRs indicate greater confidence in the bit's accuracy. Based on this reliability, a number of least reliable bits (LRB), p, are capable of generating 2^p test patterns by varying the combinations within these LRB positions.

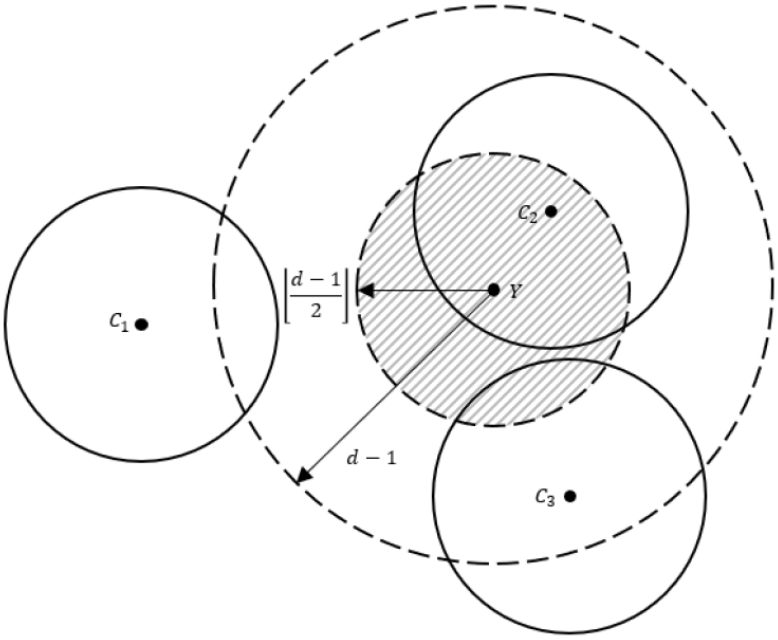


Fig. 2.2. Decoding with channel measurement information.

Moreover, the paper introduces the Chase-I and Chase-III algorithms alongside Chase-II. Among them, the Chase-II algorithm is noted for its high efficiency and low complexity. The Chase-I algorithm considers all error patterns within a sphere of radius (d-1) around the received sequence Y, generating all possible error sequences with a binary weight less than the minimum distance d. Due to the vast number of error sequences, Chase-I is only feasible for codes with a small minimum distance. Conversely, the Chase-III algorithm, while similar to Chase-II, incorporates an additional strategy to reduce the number of error sequences by categorizing them based on whether the count of d in the received sequence Y is even or odd. Although Chase-III may be slightly less effective than Chase-I and Chase-II, its application is particularly valuable for codes with a large minimum distance, making it a viable option in certain scenarios.

2.2.4.2 Pyndiah algorithm

[62] details a decoding algorithm used to estimate the log-likelihood ratio (LLR) of binary decisions, specifically when utilizing the Chase decoder alongside Block Turbo Codes (BTC). The Pyndiah algorithm, also relevant to this research, is suitable for any product code derived from linear block codes.

To calculate the output LLRs, the process begins by identifying the closest codeword D and the second closest codeword C as competing codewords based on the Euclidean distance between the received LLRs, ensuring c_j, d_j. Using these codewords, the normalized output LLR r'_j is computed according to Equation 8. If codeword C is unidentifiable—likely due to a significant Euclidean distance from the received codeword R —an approximation of r'_j is made using Equation 9. Once all output LLRs are estimated, the correction term \(W\) is calculated by subtracting it from the input LLRs as per Equation 10. The final step involves updating the received codewords for further iterations using Equ. 9, with the process depicted in Fig. 2.3.

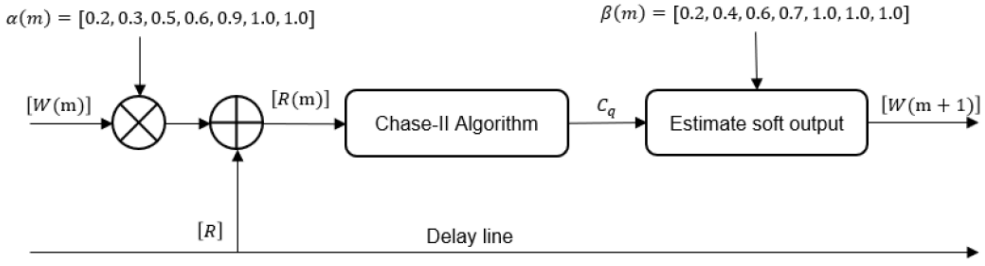


Fig. 2.3. Pyndiah algorithm elementary block diagram.

The equations are detailed as follows:

$$r'_j = \frac{|R-C|^2 - |R-D|^2}{4} d_j \quad (6)$$

$$r'_j = \beta * d_j \quad (7)$$

$$r'_j = \mathrm{r} + w_j \quad (8)$$

$$r'_j = [R] + \alpha(2)[\mathrm{W}(2)] \quad (9)$$

The weighting factor α and the reliability factor β are pivotal in the decoding procedure. The scaling factor α is particularly crucial in the initial iteration to mitigate the impact of the extrinsic information W, which differs in standard deviation from the received codeword R. This scaling factor will gradually increase to one as the number of iterations progresses. The exact values of α and β are not predetermined as they depend on specific conditions and are optimized through experimental statistics. Given the distinct needs of the front and back bits in the decoding process, the values of α and β are necessarily doubled to accommodate their unique requirements during decoding.

2.2.4.3 ORDEPT Decoding Algorithm

The Direct Error-Pattern Testing (DEPT) algorithm, as detailed in [63], is implemented by predefining Partial Error Patterns (PEPs) for a specific Bit Error Rate (BER) target to optimize performance for a given Signal-to-Noise Ratio (SNR). Similarly, the ORDEPT decoding algorithm operates by generating a set of PEPs ordered by their approximate likelihood. These PEPs are defined in a binary sequence, denoted as e_1, e_2 … e_q, with q representing the index of the error pattern queries, and iii being the index of the error pattern element (i = 1, 2, ..., n). The sum of the error positions for each pattern is estimated and ranked by its logistic weight, defined as:

$$W_L(e^q) = \sum_{i=1}^{n} i\, e_i^q \quad (10)$$

The sequence starts with no bits consists of zeros. The subsequent queries increase the logistic weight sequentially, flipping the least reliable bits first. The last error position, determined by the lowest likelihood, uses a partial syndrome s′ computation, defined by:

$$s' = (c \oplus \mathrm{e})^{H}T = s \oplus h_{j1} \oplus h_{j2} \oplus \ldots \oplus h_{jN-1} \quad (11)$$

where c represents the hard decision of the received sequence, s is the computation of its syndrome, consisting of a (n−k)(n-k)(n−k) binary vector, and h_{ji} represents the i_{th} column binary vector of the parity check matrix H.

This methodology generates PEP sets with increasing logistic weight, using a 'build-mountain' routine and 'landslide' algorithm. All error positions except the last one are searchable in the pre-generated PEP set. The PEPs are divided into odd and even sets based on the parity of the total number of errors, which reduces the algorithm's complexity by decreasing the number of PEPs examined.

Chapter 3 A Single-TSV and Single-DCDL Approach for Skew Compensation of Multi-Dies Clock Synchronization in 3-D-ICs

Yang Ge, Tejinder Singh Sandhu, Dmitri Truhachev,
and Kamal El-Sankary

3.1. Abstract

Existing methods used for the clock distribution of multiple dies employ a balanced tree structure to minimize the impact of the within-die process and loading variations. No topology for die-to-die clock skew compensation of more than two dies has been presented yet. This paper presents a novel die-to-die clock skew compensation topology to address these limitations. Unlike existing designs, the proposed topology does not need a phase detector (hence, no dead zone); it only requires one through-silicon via (TSV) to connect a pair of dies and one digitally controlled delay line (DCDL) in each die; thus, there is no skew from extra TSVs and DCDLs. Accordingly, the system has a small chip area and low lock time. The postsynthesis of this work was accomplished in a 65-nm CMOS process. The performance of our design was evaluated theoretically and practically in terms of mismatch/finite resolution of delay lines, buffer mismatch, and TSV delay. Under identical conditions, the residual skew of the proposed design was as low as 13 ps at 1 GHz. This study is the first to obtain the solution for die-to-die clock synchronization of multiple dies (more than two dies) in a three-dimensional integrated circuit, while other systems can only support two dies.

Index Terms—3D-IC, multi-die clock synchronization, through-silicon via (TSV), digitally controlled delay line (DCDL), skew compensation.

3.2 Introduction

MOORE'S law states that the number of transistors on an integrated circuit (IC) doubles every two years; this has mostly been true since the first IC was introduced. However, it has become increasingly difficult to reduce the size of

transistors. Eventually, transistors will become too small to be further reduced, and their physical constraints will essentially limit the growth of transistor density on chips. To deal with this limitation, researchers have focused on stacking multiple dies on top of each other, known as three-dimensional (3D)-IC; this topology can significantly increase the number of transistors contained in a package, since the number of all components is effectively multiplied in each chip. Bonding wires have also been utilized to connect these two-dimensional (2D) fabricated dies. However, a recent die-to-die interconnection known as through-silicon via (TSV) has attracted considerable attention because of its high density, low latency, and power reduction capability [1]. In addition, TSVs can be placed anywhere on dies, increasing the number of I/Os while significantly reducing the wire length between dies. However, distributing clocks in 3D-ICs with TSVs is a significant challenge because of the following reasons: (i) The delay across TSVs leads to critical clock skew among dies, which can accumulate every time a clock signal transmits through TSVs; (ii) TSV delay is susceptible to process, voltage and temperature (PVT) variations [1]-[9], thus the assumption of fixed TSV delay is not practical in the design. Consequently, the propagation delay across the TSV exhibits very large delay variations, and an approach that tolerates any unanticipated TSV delay variations for inter-die clock distribution is needed.

Numerous studies have focused on solving the TSV variation issue. Conventionally, two or three TSV channels [3]-[8] are employed in 3D-IC die-to-die clock synchronization with the purpose of creating forward and return paths. However, the mismatch between two TSVs can be critical [9] due to their high defect rate and susceptible to electro-migration and PVT variations [10]. These TSV mismatches lead to longer lock-in time or even cause system

synchronization failure [5]; thus, using a single TSV can prevent the aforementioned issues.

A dual-locking digitally controlled delay line (DCDL) was proposed in [3] for die-to-die clock synchronization, which needs perfect matching of two non-ideal delay lines and equal buffer delays. Even if there is a maintenance mode after locking, the dual-locking DCDL must be restarted to cancel the clock skew. An all-digital delay-locked loop and a mismatch-insensitive skew compensation architecture were presented in [4] and [5]; these techniques rely on two high resolution DCDLs to calibrate the clock skew, thereby requiring a substantial amount of chip space to accommodate the delay lines. Additionally, the mismatch between two identical delay lines can be an order of magnitude higher than hundreds of picoseconds [5]; this can result in considerable clock skew after the DCDL is locked. The study presented in [5] proposed an iteration process to eliminate the residual skew; however, it comes at the expense of longer lock-in time.

Recently, the study in [6] introduced an all-digital delay-locked loop topology, which has only one TSV to transmit clock signals, meaning that there is no mismatch between multiple TSVs. However, the assumptions of identical buffer delays and perfect matching delay lines are not practical, which can lead to significant residual skew after lock-in. In [7], a single DCDL-based architecture was explained. The architecture avoids the requirements for several DCDLs and phase detectors, but the variations between two TSVs cannot be ignored. The analog dual-delay-locked loop (D-DLL) [8] is proposed for two die clock synchronization, reports a skew of only 2 ps. However, the assumption of perfect matching delay lines is not practical. Also,

it suffers from massive chip area and power consumption.

The existing work [3]-[8] focusses on the frequency of 1GHz which is common in digital architectures. The analog circuit [8] can achieve a higher operating frequency of 2GHz. The principle of 3D-IC can be applied to any frequency or protocol in order to increase the integration density. In our paper, we use the frequency of 1GHz to demonstrate a proof of concept.

This paper presents a single-TSV and single-DCDL (STSD) approach for multi-die clock synchronization. The proposed STSD employs a single TSV channel and a time-to-digital converter (TDC)-embedded DCDL in each die. Thus, there is no extra clock skew between averaging delay lines as in previous works [3]-[8]. In addition, numerous dies can be synchronized simultaneously. A three-die example is presented in this paper; this approach can be extended to more dies. The proposed STSD is implemented in a 65-nm CMOS process utilizing standard cells from the TSMC library.

The rest of this paper is organized as follows. Section 3.3 describes the design and limitations of D-DLL topology [8]. The proposed STSD architecture is presented in Section 3.4. The circuit implementation is demonstrated in Section 3.5. The testbench setup is described in Section 3.6. The results and discussion are presented in Section 3.7, followed by the conclusions in Section 3.8.

3.3 D-DLL Architecture and Limitation

The architecture of the D-DLL [8] is shown in Fig. 3.1. The aim of this design is to ensure that the reference clock in Die1 (*Die1_CLK*) synchronizes with the

distributed clock in Die2 (*Die2_CLK*) eventually. TSV1 is used to send the clock signal from Die1 to Die2, while TSV2 is used for feeding back the transmitted signal from Die2 back to Die1. The D-DLL contains two DLLs, and each DLL includes a phase detector (PD), a charge pump (CP), a loop filter (not shown in Fig. 3.1), and voltage-controlled delay line (VCDL). Notice that, Die1 needs two VCDLs, while Die2 only requires one.

For the D-DLL, it consists of two loops (Loop1 and Loop2 in Fig. 3.1). The loop delay is measured as the time it takes for the reference clock to propagate through buffers, TSV, VCDL and arrived at the PD. The delay of two loops expressed as follows

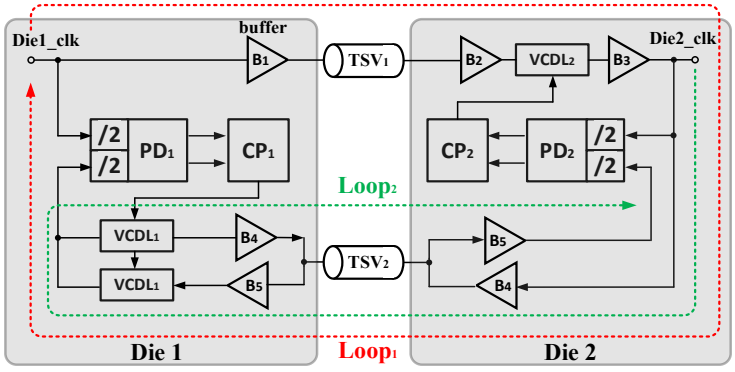


Fig. 3.1. D-DLL synchronization topology [8].

$$T_{loop1} = t_{B1} + t_{TSV1} + t_{B2} + t_{VCDL2} + t_{B3} + t_{B4} + t_{TSV2} + t_{B5} + t_{VCDL1} \quad (1)$$

and

$$T_{loop2} = 2(t_{TSV2} + t_{B4} + t_{B5} + t_{VCDL1}) \quad (2)$$

where t_{TSV1} and t_{TSV2} are the delay of the TSV_1 and TSV_2 respectively. t_{B1-B5} are the delays caused by buffers. A delay of t_{VCDL1} or t_{VCDL2} is added once the signal propagates through the $VCDL_1$ or $VCDL_2$. In the steady state, the *Die2_CLK* is synchronized with *Die1_CLK*. As a result, the *Die1_CLK* is

one clock period ahead of *Die2_CLK*, meaning that *Die2_CLK* is synchronized with *Die1_CLK*, i.e.,

$$T_{loop1} = M \cdot T_{Ref}$$

and

$$T_{loop2} = N \cdot T_{Ref} \qquad (3)$$

where T_{Ref} is the period of the reference clock, M and N are integers. Substituting (3) into (1) and (2) obtained

$$t_{B1} + t_{TSV1} + t_{B2} + t_{VCDL2} + t_{B3} + t_{B4} + t_{TSV2} + t_{B5} + t_{VCDL1} = M \cdot T_{Ref} \quad (4)$$

$$t_{TSV2} + t_{B4} + t_{B5} + t_{VCDL1} = \frac{N \cdot T_{Ref}}{2} \qquad (5)$$

Subtracting (5) from (4) becomes

$$t_{B1} + t_{TSV1} + t_{B2} + t_{VCDL2} + t_{B3} = (M - \frac{N}{2})T_{Ref} \qquad (6)$$

From (6), N must be an even integer to ensure clock synchronization, so a divide-by-2 circuit is added before the phase detector (PD).

However, the D-DLL topology has the following limitations:

- Perfect matching delay of two TSVs and buffers is not practical, leading to significant residual skew. A single TSV can solve this issue, as proven in our design in Section 2.3.
- Due to the mismatches in delay cells, the delay generated by the DLL is not same as the delay measured by PD. Therefore, a TDC-embedded

DCDL is utilized in the proposed design, and the same delay line is used to measure and create delay.

- An extremely large chip space is required to contain three delay lines of three DLL, whereas the proposed work only needs one delay line in each die.
- It is challenging to design an analog DLL with wide frequency range, the problems are the high leakage current and high voltage gain is required under low supply voltage. Thus, a digital DLL is employed in the proposed work to resolve these problems.
- The D-DLL can only support clock synchronization between two dies and cannot be extended to more dies, such as three.

The proposed STSD addresses all these limitations and eliminates any residual skew from TSVs and delay lines, enabling multi-die clock synchronization.

3.4 Proposed STSD Architecture and Operation

The simplified full schematic of the proposed architecture is shown in Fig. 3.2. The goal of the STSD is to align *Die3_clk* and *Die2_clk* with *Die1_clk* (as shown in Fig. 3.6) regardless of imperfect dividers and mismatch between unequaled buffers. Die1 contains one pulse generator (PG), one TDC-embedded DCDL, one inverter, one divide-by-2 divider, one mult_by_2 multiplier, and two tri-state buffers. Similarly, Die2 consists of one TDC-embedded DCDL, one divide-by-2 divider, and six tri-state buffers. Die3 has the least components, and only includes one tri-state buffer. TDC-embedded DCDL is the main component that has two functionalities: compare input signals and produce a specific delay. The TDC-embedded DCDL has two inputs in both Die1 and Die2, which are *Forward1*, *Reverse1* and *Forward2*, *Reverse2*, respectively. The phase difference between two input signals is

converted to digital words *Code1* and *Code2* (Fig. 3.2), respectively; after the computation, the control code *Tune1* and *Tune2* are applied to the DCDLs to adjust the delay.

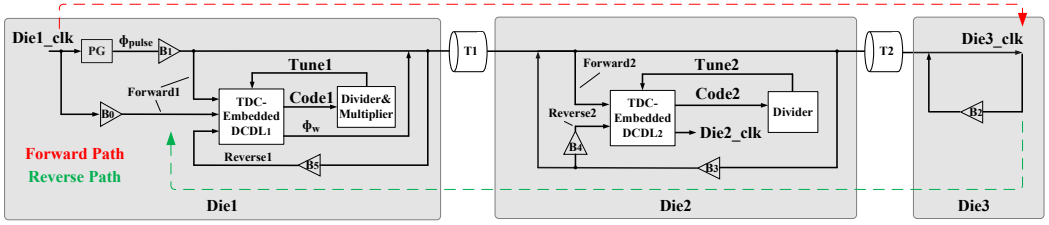


Fig. 3.2. Proposed STSD architecture.

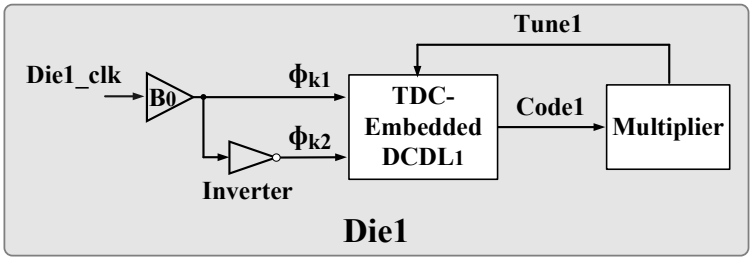


Fig. 3.3. State 1 of proposed STSD architecture.

The basic clock-synchronization process of three dies is described as follows. a) In Die1, firstly, we adjust the delay in the *DCDL1* equals to one clock period i.e., CLK. Secondly, measure the phase difference between *Forward1*, *Reverse1*. Thirdly, divide the phase differences by two and subtract from CLK to get *Tune1* which equals to $\boldsymbol{CLK - T1 - T2}$. Here, $T1$ and $T2$ are the delays of a single TSV. b) In Die2, similarly, the phase difference between *Forward2*, *Reverse2* are measured, then divide by two to make *Tune2* equals to $\boldsymbol{T2}$. c) In Die3, there is no delay line adjustment required, the distributed clock will be automatically synchronized with *Die1_clk* and *Die2_clk*.

After the system finishes the calibration, the reference clock will transmit through Die2 and Die3 via DCDLs and TSVs. As shown in Fig. 3.6, in Die2, the distributed clock equals to $(\boldsymbol{CLK - T1 - T2}) + \boldsymbol{T1} + (\boldsymbol{T2}) = \boldsymbol{CLK}$. When the reference clock arrives at Die3, becomes $(\boldsymbol{CLK - T1 - T2}) + \boldsymbol{T1 + T2} = \boldsymbol{CLK}$, meaning that the clocks in all dies are in phase. The STSD architecture can be divided into three states, as shown in Figs. 3-5. State 1-2 aim to tune the delay lines of DCDL, state 3 compensate the skew generated by previous stages. The three states are described as follows: state 1 measures the period of source clock CLK and tunes the delay in the DCDL of Die1 to be equal to CLK; based on the results of TDCs, state 2 adjusts the delays of the DCDLs of Die1 and Die2; if there is any mismatch, the error is extracted and minimized in state 3; finally, all mismatches are cancelled out, and the clocks in all dies are synchronized.

The detailed process of the STSD architecture is explained as follows

A. State 1

Fig. 3.3 presents the first state of the STSD architecture, the timing diagram of TDC is displayed in Fig. 3.3(a) of Section IV. The inverted version of $Die1\_clk$ acts as the sampling signal to produce the output code $\frac{CLK}{2} \pm lsb$ (*Code1* in Fig. 3.3), where lsb is the minimum resolution of TDC; then, the mult_by_2 module multiplies the digital code by two to obtain the source period $CLK \pm 2lsb$. The digital code of $CLK \pm 2lsb$ (*Tune1* in Fig. 3.1) is directly given to the DCDL to adjust the delay time within it.

B. State 2

State 2 of STSD is denoted in Fig. 3.4. The PG sends a single pulse from Die1 that propagates through Die2 and circulates in Die3; then, the differences of $Forward$ and $Reverse$ are extracted by TDCs of Die1 and Die2. After that, the delays in DCDLs of two dies are adjusted to $Delay1$ and $Delay2$, respectively. The specific propagation routes of the $Forward$ and $Reverse$ paths of state 2 are described below:

1. $Forward\ Path$ (B1 is on, while B2-B5 are off):

from ϕ_{pulse} to $Die3\_clk$ across PG → B1 → T1 → T2. ϕ_{p1} and ϕ_{s1} are captured by the TDCs of Die1 and Die2, respectively, and are expressed as

$$\phi_{p1} = \phi_{pulse} + B$$

and

$$\phi_{s1} = \phi_{pulse} + T1 + B \qquad (7)$$

$Reverse\ Path$ (B1 is off, while B2-B5 are on): from Die3_clk to ϕ_{p2} across B2 → T2 → B3 → T1 → B5. ϕ_{s2} and ϕ_{p2} are captured by the TDCs of Die1 and Die2, respectively, and are expressed as

$$\phi_{s2} = \phi_{pulse} + T1 + 2 \cdot T2 + 4B \qquad (8)$$

$$\phi_{p2} = \phi_{pulse} + 2(T1 + T2) + 4B \qquad (9)$$

Here, T1 and T2 are the delays of the two TSVs; assuming that the buffers have identical delay, B is the delay of each buffer.

The TDCs in Die1 and Die2 calculate the phase differences between input signals; then, the divider modules divide the digital code by two to get *Code1*

and *Code2*. Thereafter, *Delay1* is generated by subtracting *Code1* from one source period (measured in state 1) and *Delay2* equals to *Code2*:

$$\begin{aligned} Delay1 &= CLK \pm 2lsb - Code1 \\ &= CLK \pm 2lsb - \frac{\phi_{p2} - \phi_{p1} \pm lsb}{2} \\ &= CLK - T1 - T2 - \tfrac{3}{2}B \pm \tfrac{5}{2}lsb \end{aligned} \qquad (10)$$

$$Delay2 = Code2 = \frac{\phi_{s2} - \phi_{s1} \pm lsb}{2} = T2 + \tfrac{3}{2}B \pm \tfrac{1}{2}lsb \qquad (11)$$

The digital codes of *Delay1* and *Delay2* are applied to delay lines before state 3.

C. State 3

The residual skew after state 2 is caused by non-linear delay lines and unequal buffers. It is detected, and minimized in state 3 (Fig. 3.5), these skews are denoted ϕ_{e1} and ϕ_{e2} in the DCDLs of Die1 and Die2, respectively. Consequently, (10) and (11) are modified as follows

$$Delay1' = CLK - T1 - T2 - \tfrac{3}{2}B \pm \tfrac{5}{2}lsb + \phi_{e1} \qquad (12)$$

$$Delay2' = T2 + \tfrac{3}{2}B \pm \tfrac{1}{2}lsb + \phi_{e2} \qquad (13)$$

Similar to state 2, the PG sends another pulse from Die1 that propagates through Die2 and circulates in Die3; the TDCs extract the difference of the *Forward* and *Reverse* pulses. The difference between states 2 and 3 is that, in state 3, the sampling pulses of TDCs in Die1 and Die2 (ϕ_{t2} and ϕ_{d1} in Fig.

3.5) circulate in the DCDL before giving to the TDC for comparison, whereas the sampling signals (ϕ_{p2} and ϕ_{s2} in Fig. 3.4) in state 2 do not go through the DCDL. After the residual skew is

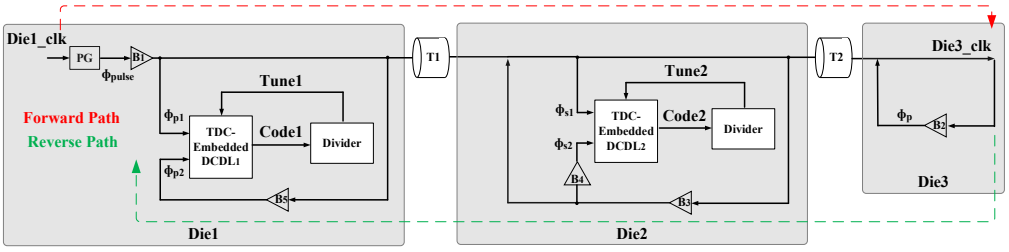


Fig. 3.4. State 2 of proposed STSD architecture

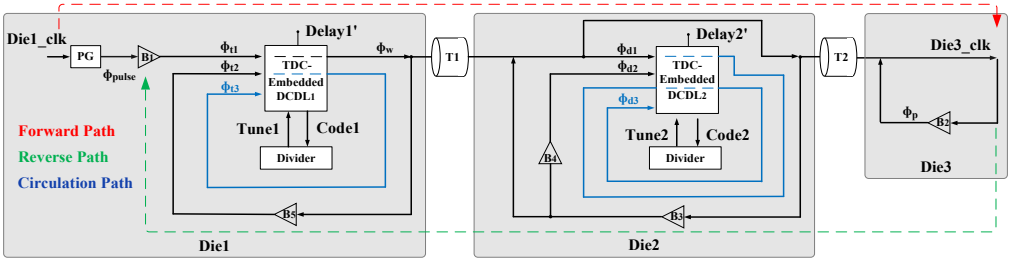


Fig. 3.5. State 3 of proposed STSD architecture.

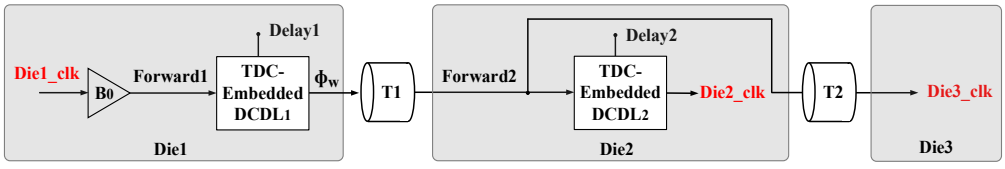


Fig. 3.6. Proposed STSD architecture after skew compensation completed.

extracted, the controller tunes the DCDL to cancel the error. The specific propagation route of $Forward$ and $Reverse$ of state 3 (Fig. 3.5) are described as follows

1. $Forward\ Path$ (B1 is on, other buffers are off):
    a) Die1: From ϕ_{pulse} to ϕ_{t1} through PG → B1 → ϕ_{t1}.

b) Die2: From ϕ_{pulse} to ϕ_{d3} through PG → B1 → DCDL1 → T1 → ϕ_{d1} → DCDL2 → DCDL2 → ϕ_{d3}.

c) Die3: From ϕ_{pulse} to $Die3\_clk$ through PG → B1 → DCDL1 → T1 → T2 → $Die3\_clk$.

$$\phi_{d1} = \phi_{pulse} + T1 + B + Delay1' \qquad (14)$$

After the returning pulse ϕ_{t2} arrives at Die1, the PG sends a second pulse at the next rising edge of $Die1\_clk$. The second pulse of ϕ_{pulse} is $2 \cdot CLK$ behind the first one; then the following is obtained:

$$\phi_{t1} = \phi_{pulse} + 2 \cdot CLK + B \qquad (15)$$

$$\begin{aligned} \phi_{d3} &= \phi_{d1} + 2 \cdot Delay2' \\ &= \phi_{pulse} + T1 + B + Delay1' + 2 \cdot Delay2' \\ &= \phi_{pulse} + CLK + T2 + \frac{5}{2}B + \phi_{e1} + 2 \cdot \phi_{e2} \pm \frac{7}{2}lsb \end{aligned} \qquad (16)$$

2. $Reverse\ Path$ (B2-B5 are on, other buffers are off):

a) Die3: From $Die3\_clk$ to ϕ_p through $Die3\_clk$ → B2 → ϕ_p.

b) Die2: From $Die3\_clk$ to ϕ_{d2} through $Die3\_clk$ → B2 → T2 → B3 → B4 → ϕ_{d2}.

c) Die1: From $Die3\_clk$ to ϕ_{t3} through $Die3\_clk$ → B2 → T2 → B3 → T1 → B5 → ϕ_{t2} → DCDL1 → ϕ_{t3}.

$$\begin{aligned} Die3\_clk &= \phi_{pulse} + T1 + T2 + B + Delay1' \\ &= \phi_{pulse} + CLK + \phi_{e1} - \frac{1}{2}B \pm \frac{5}{2}lsb \end{aligned} \qquad (17)$$

$$\phi_{d2} = Die3\_clk + T2 + 3B$$
$$= \phi_{pulse} + CLK + T2 + \frac{5}{2}B + \phi_{e1} \pm \frac{5}{2}lsb \qquad (18)$$

$$\phi_{t2} = Die3\_clk + T2 + T1 + 3B$$
$$= \phi_{pulse} + CLK + T1 + T2 + \frac{5}{2}B + \phi_{e1} \pm \frac{5}{2}lsb \quad (19)$$

$$\phi_{t3} = \phi_{t2} + Delay1'$$
$$= \phi_{pulse} + 2 \cdot CLK + B + 2\phi_{e1} \pm 5lsb \qquad (20)$$

The divider module in Die1 converts the output code of the TDC-embedded DCDL1 to half, using (15) and (20), and the tuning code $Tune1$ becomes

$$Tune1 = \frac{\phi_{t3} - \phi_{t1} \pm lsb}{2} = \phi_{e1} \pm 3lsb \qquad (21)$$

Similarly, the divider module in Die2 divides the output code of TDC-embedded DCDL2 by 2, using (16) and (18). Then, the tuning code Tune2 becomes

$$Tune2 = \frac{\phi_{d3} - \phi_{d2} \pm lsb}{2} = \phi_{e2} \pm lsb \qquad (22)$$

The control codes $Tune1$ and $Tune2$ are then subtracted from the DCDLs to adjust the skew in the delay lines using (12), (21), (13) and (22); (10) and (11) can be modified as follows

$$Delay1 = Delay1' - Tune1$$

$$= \left(CLK - T1 - T2 - \frac{3}{2}B \pm \frac{5}{2}lsb + \phi_{e1}\right) - (\phi_{e1} \pm 3lsb)$$

$$= CLK - T1 - T2 - \tfrac{3}{2}B \pm \tfrac{1}{2}lsb \qquad (23)$$

$$Delay2 = Delay2' - Tune2$$

$$= \left(T2 + \tfrac{3}{2}B \pm \tfrac{1}{2}lsb + \phi_{e2}\right) - (\phi_{e2} \pm lsb)$$

$$= T2 + \tfrac{3}{2}B \pm \tfrac{1}{2}lsb \qquad (24)$$

Here, (23) and (24) are the designed delay set up in the DCDLs of Die1 and Die2, respectively.

D. Clock synchronized after the skew compensation

As presented in Fig. 3.6, when only B0 is on, other buffers are off. The synchronized clocks in Die2 and Die3 are expressed as

$$Die2\_clk = Die1\_clk + B + Delay1 + T1 + Delay2$$

$$= Die1\_clk + B \pm lsb \qquad (25)$$

$$Die3\_clk = Die1\_clk + B + Delay1 + T1 + T2$$

$$= Die1_{clk} - \tfrac{1}{2}B \pm \tfrac{1}{2}lsb \qquad (26)$$

As depicted in (25) and (26), the *Die2_clk* and *Die3_clk* are aligned with *Die1_clk*, and clocks are synchronized within ($B \pm lsb$) in Die2 and ($-\frac{1}{2}B \pm \frac{1}{2}lsb$) in Die3 after completing the skew compensation.

E. Back-and-forth calibration

In the presence of non-linear delay line, the tuning code created by divider circuit in (21) and (22), may not tunes the delay line precisely. Therefore, the back-and-forth process is employed to fully eliminate skew caused by the divider circuit and non-linear delay line. Comparing with [5] that also implemented a back-and-forth process, ours has much less lock-in time since there is only one delay line and one TSV utilized in one die, while [5] includes two delay lines and three TSVs, meaning that [5] has larger residual skew and longer lock-in time.

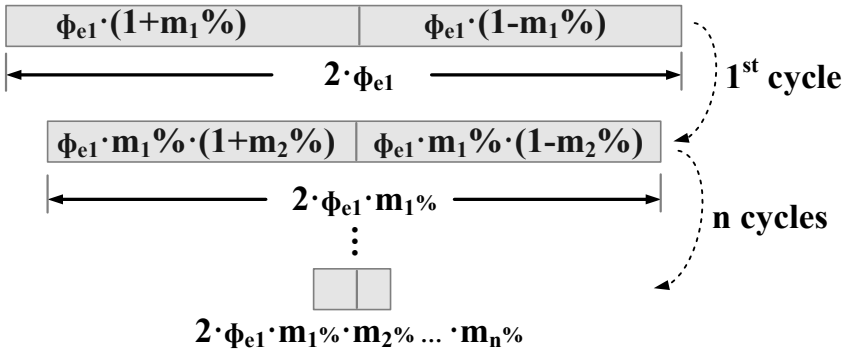


Fig. 3.7. Back-and-forth calibration process.

It should be noted that after state 2, the residual skew of the DCDL in Die1 is ϕ_{e1}; after circulating in the DCDL twice, the extracted residual skew given to the divider is $2 \cdot \phi_{e1}$. Assuming that the divider circuit and TDC has m_1 % mismatch in the beginning after state 3, as depicted in Fig. 3.7, the input code of the divider $2 \cdot \phi_{e1}$ can be divided into two parts: $\phi_{e1} \cdot (1 + m_1\%)$ and $\phi_{e1} \cdot (1 - m_1\%)$. After taking the first part to be $Tune1$ (taking the second part to be $Tune1$ will have the same result), (23) becomes

$$Delay1 = Delay1' - Tune1$$
$$= Delay1' - (\phi_{e1} \pm 3lsb) \cdot (1 + m_1\%)$$

$$= \left(CLK - T1 - T2 - \frac{3}{2}B \pm \frac{1}{2}lsb\right) - (\phi_{e1} \pm 3lsb)m_1\% \quad (27)$$

Next, we send another pulse, and this time suppose that the divider circuit has m_2 % mismatch; thus, (27) becomes:

$$\begin{aligned} Delay1 = Delay1' - Tune1 \\ = \left(CLK - T1 - T2 - \frac{3}{2}B \pm \frac{1}{2}lsb\right) \\ +(\phi_{e1} \pm 3lsb) \cdot m_1\% \cdot m_2\% \quad (28) \end{aligned}$$

After n cycles, (28) becomes

$$\begin{aligned} Delay1 = \left(CLK - T1 - T2 - \frac{3}{2}B \pm \frac{1}{2}lsb\right) \\ +(\phi_{e1} \pm 3lsb) \cdot \prod_{n=1}^{k} m_n\% \\ \approx CLK - T1 - T2 - \tfrac{3}{2}B \pm \tfrac{1}{2}lsb \quad (29) \end{aligned}$$

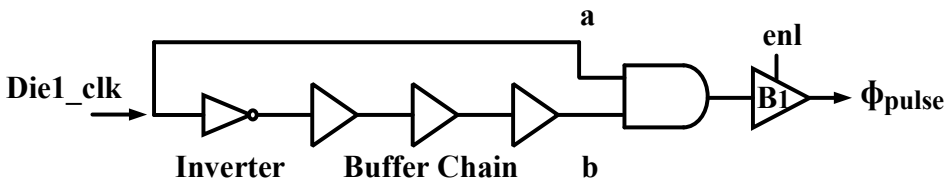
Similarly, in the DCDL of Die2, (24) becomes

$$\begin{aligned} Delay2 = \left(T2 + \frac{3}{2}B \pm \frac{1}{2}lsb\right) - (\phi_{e2} \pm lsb) \cdot \prod_{n=1}^{k} m_n\% \\ \approx T2 + \tfrac{3}{2}B \pm \tfrac{1}{2}lsb \quad (30) \end{aligned}$$

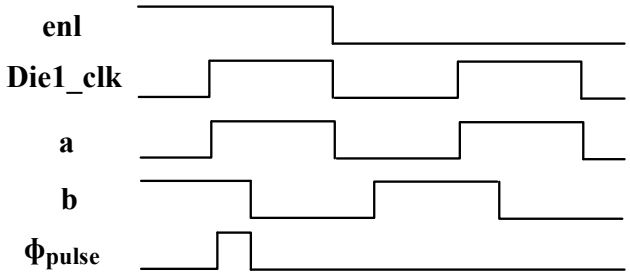
it is noteworthy that the residual skew in (29) and (30) is approximately equal to zero after a few cycles, the results of (29) and (30) are exactly same with (23) and (24). In other words, the STSD completely eliminates the skew caused by the division process.

3.5 Circuit Implementation

This section describes the circuit details of the main blocks: pulse generator (PG) and TDC-embedded DCDL.



(a)



(b)

Fig. 3.8. Block and timing diagram of the PG. (a) Block diagram. (b)Timing diagram.

A. Pulse Generator

Fig. 3.8 displays the block and timing diagram of the PG. The PG generates narrow pulses whose pulse width equals to the delay of buffer chain. A tri-state buffer is employed to enable/disable the PG. As shown in Fig. 3.8(b), once *enl* is set to high, signal "*a*" propagates through B1 and signal "*b*" transmits through buffer chain and B1. The AND gate generates the pulse signal ϕ_{pulse}, the pulse width can be adjusted by modifying the delay time of the buffer chain.

B. TDC-embedded DCDL

Fig. 3.9 illustrates an example of a three-delay-setting TDC-embedded DCDL; it comprises of three coarse delay cells, three fine delay cells, and two TDC-embedded delay lines (i.e., one for coarse tuning and another one for fine tuning). Each coarse and fine delay cell can generate a delay of ϕ_{coarse} and ϕ_{fine}, respectively. The Forward signal propagates through the delay line during the measurement, and the Reverse signal appears later. Thus, it appears that the Forward signal follows the Reverse signal. During each state, the Forward catches up by the lsb_coarse and lsb_fine in the coarse-TDC and fine-TDC, respectively. Moreover, a-c are the delay versions of Forward signal, and d-f are the delay versions of b (also shown in Fig. 3.10). T[0-2] and Q[0-2] (shown in Fig. 3.9) are the output data generated by registers. The DCDL is realized by selecting a signal from the delayed signals a-c and d-f according to the control code C[0-2] and F[0-2], respectively.

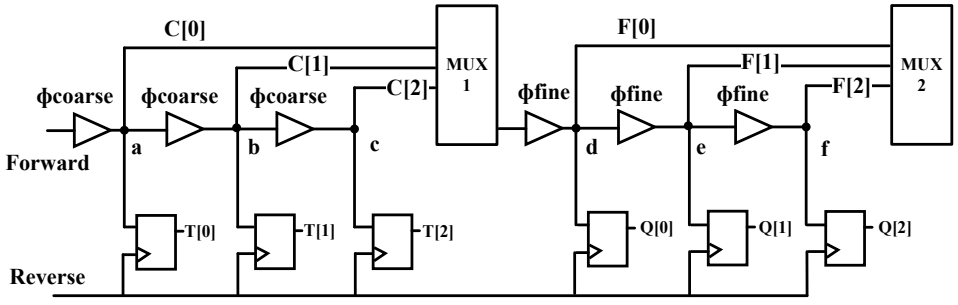


Fig. 3.9. TDC-embedded DCDL.

The timing diagram of the TDC in state 1 is shown in Fig. 3.10(a), the inverted version of $Die1\_clk$ acts as the sampling signal to produce the code equals to half of the clock period. Fig. 3.10(b)-(c) show the timing diagrams of the coarse-TDC and fine-TDC, where the phase difference between *Forward* and

Reverse is first measured by coarse-TDC; then, the residual phase difference is determined by the fine-TDC.

The delay lines of DCDLs in Die1 and Die2 consist of 60 coarse and 25 fine delay elements. From Fig. 3.10(b)-(c), we know that the fine delay range must be at least the same with the coarse delay resolution. Therefore, the coarse and fine resolutions are assigned at 50 ps and 2 ps, respectively, giving a maximum delay of 3 ns.

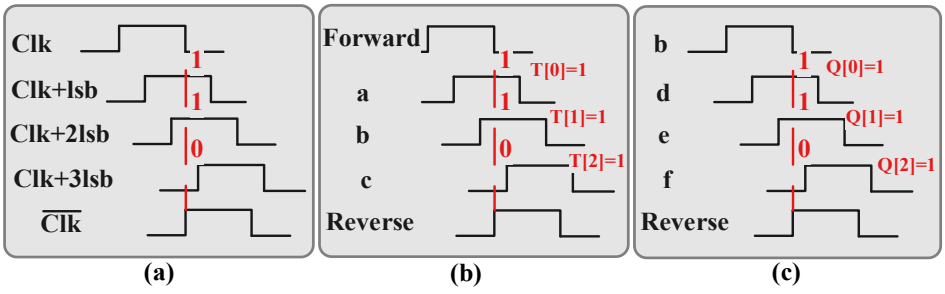


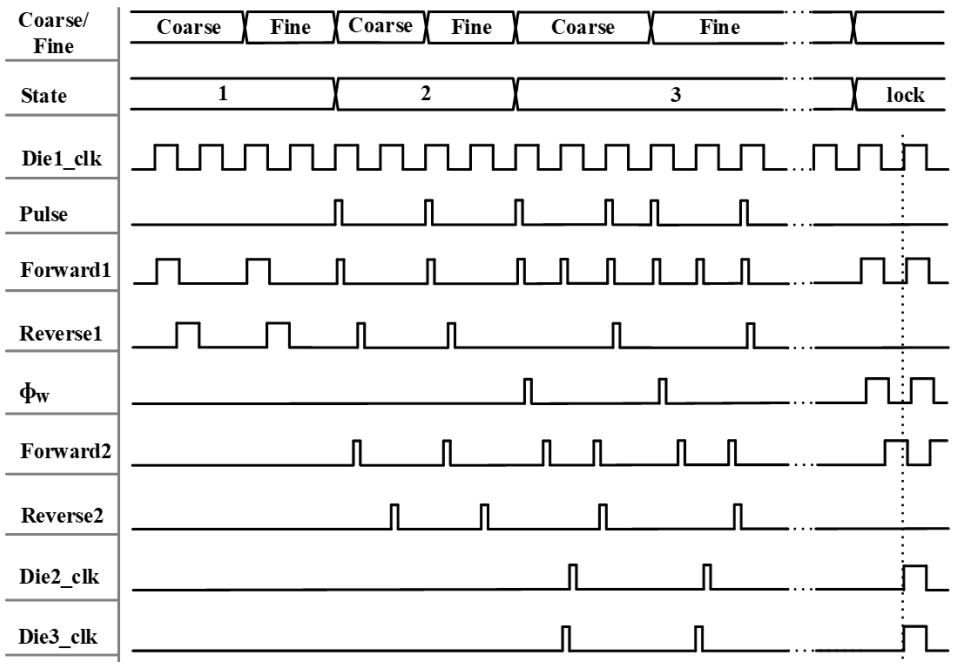
Fig. 3.10. Timing diagrams (a) TDC in state 1. (b) Coarse vernier TDC. (c) Fine vernier TDC.

3.6 Testbench Setup

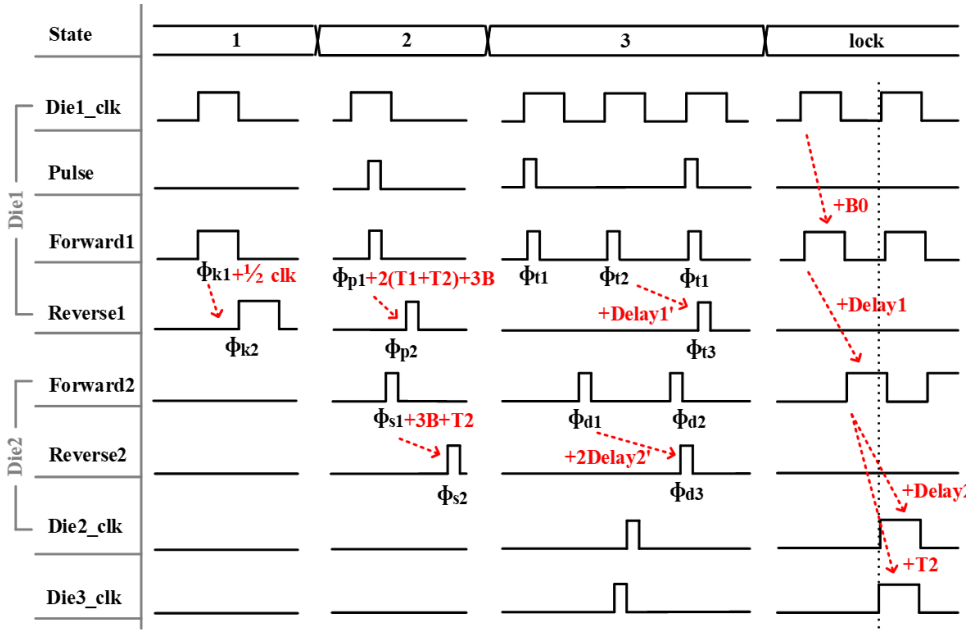
In the testbench, we have completed 10,000 emulation runs to evaluate the performance with various skew sources. Prior to fabricating the chips, our strategy is to use a large number of testbench emulations runs to perform an accurate verification of the design. According to [5] equation

$$Clk = nT \pm \frac{3}{2} lsb \pm \frac{\phi_m}{2} \pm \frac{\phi_{buf}}{2} \pm \frac{\phi_{pd}}{2} \pm \phi_j \qquad (31)$$

is the expression of the synchronized clock [3]. While ideally, $Clk = nT$, in practice, a number of residual skews - the other terms that appear on the right-



(a)



(b)

Fig. 3.11. Postsynthesis simulation of STSD architecture at 1 GHz in 65-nm CMOS. (a) Full synchronization process of STSD. (b) Zoomed views of different states.

hand side of (31) - are also present. The composite value of the residual skews reported in [3] operating at 600 MHz equals 9.6 ps. We note, however, that for the 90 nm CMOS the mismatch of a bidirectional buffer ϕ_{buf} can be as high as 22 ps, and the peak mismatch of the phase detector ϕ_{pd} equals 38 ps [5]. The sum of the two above-mentioned skews is much higher than the reported total skew due to the fact that these random skews are added out of phase.

To explain the effect of random skews, consider two random vectors ‘a’ and ‘b’ of length x such that their components are independent and distributed uniformly in the interval [-2, 2]. The components of the sum of ‘a’ and ‘b’ follow the triangular distribution. The resulting pdf is maximized at the central value ‘0’ but yields the peak sums a + b = ±4 with a probability diminishing to 0. Similarly, skew sources such as non-linear delay lines, unequal buffers, defected TSV etc. can be treated as independent random variables that appear in (31). Depending on their values, signs, and distributions, they may add constructively or destructively. Most importantly *they are not known a priori before the fabrication.* It is time consuming and expensive to perform an extensive number of fabrications (at least 1000) with random total mismatch to find the distribution of the residual skew. Instead, it is more effective to follow our proposed strategy and evaluate the design by using the testbench and sweeping each skew variable between its peak values emulating a large number of fabrication runs.

The test flow of the proposed design is described below. At the first step, the design is described by Verilog and given to the Synopsys Design Compiler (DC). Following the synthesis in Synopsys DC, the gate level netlist and timing information are derived from the layout in Cadence Innovus. This

includes all practical delays caused by the physical design, such as wire delay, set/hold time delay, fanout delay, gate delay etc. The generated physical netlist and timing information (SDF) files are then imported into ModelSim that is used to run the testbench emulating 10,000 distinct fabrication runs. In the following, we focus on several major skew sources and their handling in the testbench.

1) **Skew in TSVs.** TSVs contain some of the same faults present in conventional 2D interconnecting wires, such as open faults and short. Meanwhile, TSVs also suffer from some unique defects such as air holes, that are hard to model since they generally affect the performance of TSV rather than its logical function [65], [67]. Thus, an accurate test model of a TSV is always challenging to create. For the purpose of basic analysis, TSV models based on some simplifying assumptions are commonly used. These include transmission line-based models [65], [67], a programming-based model [68], and a lumped TSV equivalent circuit [69]. In our architecture, the TSVs are modeled as delay lines, which is a general approach used in recent 3D-IC synchronization topologies [3]-[8]. The two TSVs that are implemented in our design are replaced with behavioral models to emulate the PVT-induced delay mismatch across them. A typical delay time of a single TSV equals 75 ps [14] with the delay variance of about 10% as found in [65] and [66] using TSMC 65 nm CMOS technology. Thus, for our purpose, the delay times of the two TSVs are selected independently uniformly at random from the interval [65, 85] ps with the resolution of 2ps.

2) **Other Skew sources.** The other skew sources such as wire propagation delay, unequal buffers, delay line mismatches, gate delay, set/hold time delay,

fanout delay etc., are all implemented physically using standard cells in 65 nm CMOS under different corner analysis. More specifically, the peak buffers mismatch is ±22 ps in 65nm CMOS [5], the delay of each buffer is generated uniformly in the interval [-22 ps, 22 ps]. Similarly, we introduced mismatch in delay line by constructing them with varied delay cells. Wire delay, gate delay, set/hold time delay, fanout delay etc., are defined by Synopsis DC and Cadence Innovus. All the above-mentioned skews are recorded in the generated physical netlist and timing information (SDF) files and then imported to ModelSim for emulation runs.

3.7 Results and Discussion

The proposed STSD is created in Verilog by using standard cells from the 65-nm TSMC library. Then, it is converted to the gate-level netlist with the synthesis tool Synopsys Design Compiler (DC); the output netlist is constrained in terms of area, setup/hold time and propagation delay. The timing information that includes delays is extracted from the DC and imported into ModelSim for further simulation. In addition, the TSV model contains a tunable delay line that mimics the TSV delay between dies. This STSD design consumes 1.6 mW, while the core area is 0.0056 mm^2. The detailed timing diagram of a synchronization cycle of the STSD at 1 GHz is shown in Fig. 3.11(a), followed by the zoomed views of different states presented in Fig. 3.11(b). In Fig. 3.11, the STSD includes three states and a lock state, and state 3 is repeated twice to compensate for the residual skew. The *Forward1*, *Reverse1* and *Forward2*, *Reverse2* in Fig. 3.11 are the data and sampling signals (that can be seen in Fig. 3.10(b)-(c)) of Die1 and Die2, respectively. The aim of the STSD is to synchronize *Die1_clk*, *Die2_clk* and *Die3_clk*. The explanations of Fig. 3.11 in the order of state1-lock are given as follows:

a) In state 1 of Die1, the phase difference between ϕ_{k1} and ϕ_{k2} is $\frac{1}{2}clk$; this difference is measured by the TDC and given to the multiplier circuit.

b) During state 2 of Die1, the pulse signal that propagates through buffer B1 arrives at the TDC, which becomes ϕ_{p1}, and acts as the data signal. Meanwhile, another path of the pulse across T1 and T2 circulates in Die3 and transmits back to Die1; this reverse pulse ϕ_{p2} acts as the sampling signal of the TDC. The phase difference between ϕ_{p1} and ϕ_{p2} is computed and applied to the DCDL as a digital word $Tune1$.

c) During state 2 of Die2, the pulse signal transmits from Die1 through T1 and B4, turning to ϕ_{s1}, which is the data signal of the TDC. Similar with (b), the other path of the pulse transmits through T2 and circulates in Die3, and then, back to the TDC in Die2; this reverse signal ϕ_{s2} plays the role of the sampling signal. The phase difference between ϕ_{s1} and ϕ_{s2} is converted to a digital word $Tune2$ and applied to the DCDL.

d) During state 3 in Die1, before the pulse signal is generated in state 3, the digital codes of $Tune1$ and $Tune2$ are applied to the DCDLs in Die1 and Die2, respectively. In state 3 of Die1, due to the skew created in the previous states, the actual delays in the DCDLs become $Delay1'$ and $Delay2'$. The pulse signal ϕ_{pulse} propagates across B1, DCDL1, thereafter, across T1, T2 and circulates in Die3. The reverse signal transmits through B2, T2, B3, T1, B5, and then, the derived signal ϕ_{t2} transmits through the DCDL1 ending up with ϕ_{t3}, which is the sampling signal of the TDC. After ϕ_{t2} arriving at the TDC, the PG generates another pulse at the next rising edge of Die1_clk to act as the data signal of the TDC (ϕ_{t1}). The phase difference between ϕ_{t1} and

ϕ_{t3} is calculated to a digital code *Tune1*, which is applied to the DCDL once it is created.

e) During state 3 in Die2, the transmitted signal (ϕ_w) from Die1 that transmits through T1 yields ϕ_{d1}, and then, circulates within the DCDL twice to obtain ϕ_{d3}. The other path from ϕ_w via T1, T2, B2, B3 and B4, receives the signal ϕ_{d2}. The phase difference between $\phi_{d2}$and ϕ_{d3} is extracted and transferred to a compensation code *Tune2*.

f) In the lock-in state, Die1_clk transmits through B0, DCDL1, T1 and DCDL2, and obtains Die2_clk. Another path of Forward2 propagates through T2 to receive *Die3_clk*. Finally, *Die2_clk* and *Die3_clk* are in phase with *Die1_clk*.

Fig. 3.12 shows the STSD layout implemented in Cadence Innovus using standard cells in the 65-nm CMOS. Notice that, a three-die system is implemented in Fig. 3.12, if we implement a two-die structure like other works [3]-[8], the speed, area and power could be even better. Table I compares the proposed STSD with existing topologies. It can be seen that the peak skews in Die2 and Die3 are 26 and 13 ps, respectively. Also, the proposed work has the least number of TSV and delay line, this is the main reason that the proposed work occupies the least chip area, moreover, takes the least time to compensate the mismatch and complete synchronization. Compared to the existing topologies, the all-digital system from Chung [4] has less residual skew than the proposed design, however, with the expense of longer lock-in time. [5] reported a relatively low power and residual skew, while it requires longer time to compensate the mismatch caused by different TSVs and DCDLs. The analog topology shown in [8] has the least residual skew, however, the proposed architecture requires much less power and chip

area. With the help of single-TSV and single- DCDL, the STSD accomplishes the relatively small residual skew and fastest lock time with the smallest chip area and lowest power consumption. Also note that, from Table I, the STSD is the only architecture that can synchronize clocks for more than two dies.

The distribution is obtained based on 10,000 testbench simulation runs at 1 GHz and considering all skew sources as mentioned in Section V. Note that the ss corner has the largest residual skew, while the smallest residual skew is achieved at the ff corner. Additionally, we note that the residual skews are concentrated around the median of the distribution which is close to zero. The peak residual skews in Die2 and Die3 are around 26 ps and 13 ps, respectively. Notably, these values are matched to the residual skew shown in Table I.

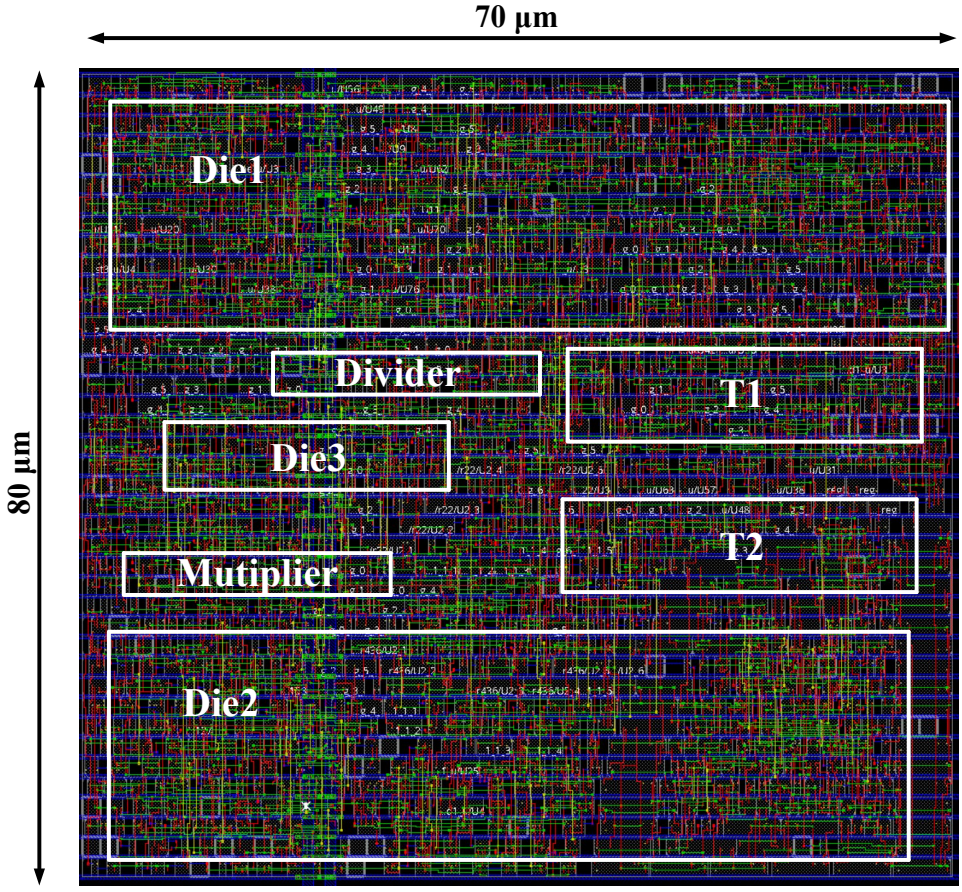


Fig. 3.12. Complete STSD layout implemented on a single die, measuring 70 μm × 80 μm.

TABLE I

PERFORMANCE SUMMARY AND COMPARISONS WITH THE STATE-OF-THE-ART

Ref.	Ke'12 [3]	Chung'14 [4]	Sandhu'16 [5]	Chung'17 [6]	Sadi'17 [7]	Yang'21 [8]	This work
Process	90	90	65	90	180	180	65

(nm)							
Type	Digital	Digital	Digital	Digital	Digital	Analog	Digital
Supply Voltage (V)	1	1	1	1	1.8	1.8	1
Maximum Number of Delay Lines in one Die	2	2	2	2	4	2	1
Number of TSVs	2	2	3	1	2	2	1
Frequency Range (MHz)	50-600	300-1000	250-1000	200-1000	556-1000	20-2000	200-1000
Active Area (mm^2)	0.0088	0.045	0.0081	0.0289	N/A	0.82	0.0056
Lock Time (Cycles)	54 @600MHz	79 @1GHz	180 @1GHz	50 @1GHz	N/A	N/A	40 @1GHz

Power (mW)	1.8 @600 MHz	3.27 @1GHz	2.1 @1GHz	3.27 @1GHz	56 @1.5 GHz	20 @2 GHz	1.6 @1GHz
Residual Skew (ps)	119 @600 MHz	21.9 @1GHz	32 @1GHz	40 @1GHz	116 @1.5 GHz	2 @2 GHz 60 @20 MHz	23 (Die2) 11 (Die3) @1GHz

3.8 Conclusion

In this paper, a STSD architecture with single TSV and single DCDL was proposed to synchronize the clocks between multiple dies. The performance of our architecture was verified theoretically and experimentally regarding mismatch/finite resolution of delay lines, buffer mismatch, and TSV delay. By using the single-TSV and single-delay-line structure, compared to the state-of-arts, the proposed STSD has the smallest lock-in time, lowest power and chip area. STSD can operate at a range from 250 MHz to 1 GHz, with a maximum residual skew of 26 and 13 ps in Die2 and Die3, respectively. In addition, STSD is designed with standard cells in a TSMC 65-nm CMOS process; thus, the proposed approach can be easily adapted to different processes.

Chapter 4. Efficient Address-Embedded Time-Domain Implementation of Minima Finders for Soft Error-Correction Decoders

Yang Ge, Dmitri Truhachev, Abolfazl Zokaei, Xxiaotong Lu, Xiaoting Huang and Kamal El-Sankary,

4.1 Abstract

This brief presents a novel algorithm that identifies the first six smallest values and their positions in a vector of 256 quantized positive real numbers. The proposed technique is designed for efficient time domain implementation. It operates with signals that are specially structured to contain a pulse width representation of a real number and a pulse-code of its address. The signals are passed through a tree structure where the number of stages and the comparison blocks are devised to yield the best overall performance. To demonstrate the hardware efficiency of this technique, it was successfully implemented using TSMC 65-nm CMOS technology. Experimental results unequivocally showcase that the proposed "minima finder" (MF) algorithm outperforms state-of-the-art solutions in terms of latency and complexity. This makes the proposed architecture a compelling candidate for the hardware implementation of the next-generation high-throughput forward error-correction (FEC) decoders in the realm of fiber-optical communications.

4.2 Introduction

The fundamental problem of digital communication is the corruption of the transmitted data by the noisy channel. This problem is addressed by the utilization of forward error correction (FEC) codes that add redundant parity-check bits to the transmitted message. The resulting parity-check equations allow the receiver to correct the errors by means of decoding [14]. Soft-decision decoding (SDD), which takes into account reliability of received data bits, typically expressed in form of log-likelihood ratios (LLRs), is the workhorse of modern error-correction [14]. In this paper we focus on high-throughput low-power applications of SDD such as fiber optical communications [15]–[17] that needs to deliver throughput over 400 Gbps

over a single wavelength under strict reliability, latency, and power constraints [18].

The class of SDDs includes, among others, Chase [19], ordered reliability bits guessing random additive noise decoding (ORBGRAND) [20], a direct error-pattern testing (DEPT) [18], and ordered reliability DEPT (ORDEPT) algorithm [21]. Soft iterative decoding of concatenated codes requires SDDs to provide soft output. Fig. 4.1(a) illustrates a generic block diagram of a test-pattern based soft-input soft-output (SISO) decoder. The LLRs of the data bits obtained from the channel demodulator are input to the decoder. The decoder starts identifying the positions (indices) and magnitudes of a few least reliable LLRs (the first block in Fig. 4.1(a)). Often, soft decoders that operate near pre-FEC thresholds of standard optical FEC codes [15], [16] need to find at least 5 − 8 minima among a large number (256 and above) of quantized input LLR values [17], [18], [22]–[25]. The hardware of the MF block contributes significantly to the complexity of the SDDs [18]-[26]. Furthermore, the latency of the MF often becomes a bottleneck [18]-[26]. Consequently, an efficient MF block is of paramount importance to improve the throughput and decrease the complexity and power consumption of SDDs. The identified positions of the smallest LLRs are used to generate test error patterns and to compare them in terms of likelihood (in Chase and DEPT-like algorithms). The most likely error patterns are further used for error correction and for generation of the outputs LLRs [27]. Numerous studies have focused on solving minima finding problems. In general, parallel [28]-[31] and serial [32], [30] are the two common architectures to determine minima values and the corresponding indices. The architecture, dedicated to finding the first two minima and their indices have been discussed in [28]. The parallel tree-

structure (TS)-based architecture provides lower latency and is more hardware-efficient than the other, sorting-based structure, which needs recording of the indices during all previous comparisons [28]. Modified two-minima finding TS were presented in [29], [30] and provide reduced hardware complexity and latency. However, as the number of inputs increases, the index-generating unit becomes extremely complex, leading to a significant increase in the overall area and the latency of the critical path. We also note that finding of just two minima, that is adequate for min-sum decoding low-density parity-check codes, is typically insufficient for error-pattern based decoding. In [31], an efficient four minima value finder was presented. However, it requires additional algorithms for finding more minima. Iterative architecture [32] finds the minima sequentially, but at the expense of larger hardware resources and increased latency.

Time-domain processing (TDP) offers advantages in latency, area and energy efficiency in comparison to the digital processing. In TD, a multi-bit quantized digital data value is represented by a pulse width of a single signal [30]–[32]. The data may be transferred from digital to time domain with use of a digital to time converter (DTC) consisting of a delay chain and a multiplexer (MUX). The delay chain generates multiple delay versions of the input pulse, and the MUX selects the appropriate delay version as the output. In [70], an efficient serial TD-based minima value finder is proposed.

In this paper, we introduce a novel architecture for a minima value and address (MVA) finder that efficiently identifies six smallest values and their positions in a vector of 256 quantized input numbers. This scenario is directly applicable to SDD implementation of the high-speed optical communications open

forward error correction (oFEC) code [16]. Our proposed architecture combines the benefits of both parallel TS and TDP techniques to achieve the optimal performance. Specifically, compared to parallel architectures in [28]-[31], the TDP-based module can further improve the throughput, especially for the case when the inputs are quantized with a large number of bits. The parallel TS-based architecture employed in our work results in lower latency, compared to the serial structure [32]. In addition, we propose a novel format of address-embedded TD (AETD) signals that enable higher speed and lower complexity compared to the conventional TD architecture [30].

The proposed MVA can be directly embedded into a fully TD BCH decoder implementation, enabling both the input and output of the MVA finder to be in TD. Fig. 4.1(a) and (b) jointly demonstrate a schematic of an SDD that operates with TD SISO and works as a component decoder in a complex concatenated code structure. The channel demodulator inputs digital LLRs that pass through a converter a DTC converts the LLR values to the TD. Synchronously, a pulse generator reads and converts the addresses of the corresponding LLRs into single pulses. These address pulses are subsequently combined with the TD LLR values into LLR signals. The outputs produced in the decoding process are also AETD signals (see Fig. 4.1(b)), and they can be used iteratively within the concatenated decoder without the need to use conversion at every iteration.

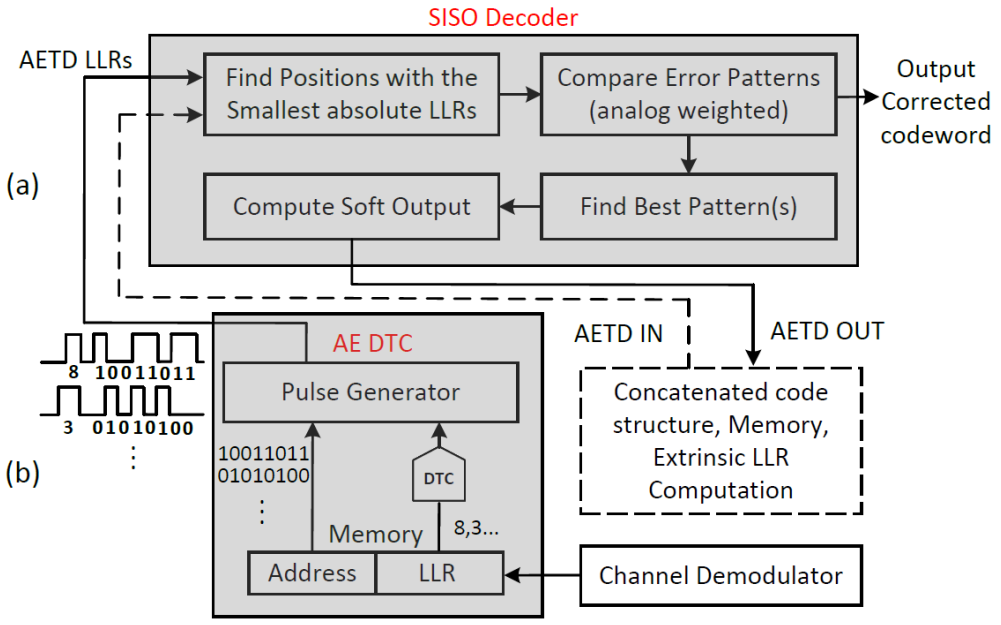


Fig. 4.1. A block diagram of an error-pattern based time-domain SDD with soft output. (a) Generic SDD. (b) Proposed AE DTC.

The paper is organized as follows: Section 4.3 explains our proposed AETD format. Section 4.4 presents our novel TS-based MVA finder algorithm. Section 4.5 describes the architecture for finding the first six minima from 256 inputs. Section 4.6 provides the implementation results and discussion, followed by the conclusion in Section 4.7.

4.3 Address-Embedded TDP

In order to find the minima and their positions in a single shot TDP architecture, we propose an AETD structure that contains both the data value and the address. The structure of the input TD signals and the operation of the proposed MVA finder for the case of two inputs are illustrated in Fig. 4.2. Within TDP, the magnitude of a value is represented by the width of the signal pulse. This is illustrated in Fig. 4.2(a). The Data pulse of signal A is narrower

than the Data pulse of signal B indicating that the Data value of A is larger than the Data of B.

As depicted in Fig. 4.1(b), the LLR values and their corresponding addresses are initially stored in a digital format and then transformed into TD using an AE DTC. Fig. 4.2 illustrates the conversion process of two LLRs, λa with address (index) 217 and λb with address (index) 42 into two AETD signals A and B. At the beginning, the internal DTC (see Fig. 4.1(b)) receives the reference pulse Valid and subsequently generates a set of delayed versions of Valid with a variety of different delays. Following this, the MUX (see Fig. 17(b)) selects the appropriate delayed version for the Data intervals (Fig. 4.2(a)) of the signals A and B, respectively, based on λa and λb. At the falling edge of the Valid pulse, the addresses corresponding to each LLR are read from memory (Fig. 4.1(b)) and converted with the use of pulse generator as follows. The bits of an address are shifted to the right by one bit width on each rising edge of the clock (Fig. 4.2(a)). Subsequently, the Address part of the signals A and B are generated by sequentially outputting the bits of the address from the least significant bit (LSB) to the most significant bit (MSB). Fig. 4.2(a) presents the proposed representation where a bit value of 1 is represented by a pulse width of one clock cycle, and 0 by one empty signal with one clock duration. The pulse generator, used to convert the address part, simplifies the hardware and also combines the address pulsed with the data pulses.

In the selecting process searching for the minimum between λa and λb represented by the Data parts of signals A and B, as shown in Fig. 4.2, the

pulse Valid indicates the start of the data comparison. Once Valid is detected, the data pulses are extracted and compared. The selector that consists a 2:2 MUX constructed from two 2:1 MUXs, extracts the data pulses of A and B by monitoring the rising and falling edges of signal Valid. At the start of the process Valid = 0 indicating the absence of data. Once Valid = 1, the data starts to propagate producing the outputs C and D that are equal to A and B, respectively. Note that at the start C and D are reset to 0 at the rising edge of Rst and are also reset to 0 before the next iteration of data signal arrival. As the data parts of the signals A and B propagate, the control signal Contrl is generated by the TD Comparator, which is implemented as an SR-latch. The logic behind the TD Comparator is as follows. In case {c,d} = {1,0} $\lambda b > \lambda a$ and Contrl = 0. Otherwise, if {c,d} = {0,1}, $\lambda b \leqslant \lambda a$ and Contrl = 1. In case {c,d} = {1,1}. Contrl remains unchanged. To prevent timing violations, we generate the laged versions A_lag and B_lag of A and B delayed by one clock cycle, generated by a controllable delay chain. Finally, the resulting decision code Contrl (Contrl = 1 in the example of Fig. 4.2) is given to a 2:1 MUX. The minimum output Min(A,B) is copied from either A_lag or B_lag, depending on which signal gives the minimum. Note that the address information can be easily converted back to digital domain at the final SDD output by sampling the address pulses using the same clock.

This proposed conversion and address embedding methodology greatly simplifies the process of memory access and data retrieval, particularly in the subsequent steps of patterns selection, LLR summation, and comparison inside a TD SDD (see Fig. 4.1). Additionally, this approach avoids the utilization of complex time-to-digital converter (TDC) that would need be employed to convert TD addresses into a digital format inside the SDD. This could greatly

benefit a high-throughput decoder reducing the power consumption and latency.

4.4 TS-Based MVA Finder Algorithm

Consider two sorted sequences A and B, that contain six elements sorted in the ascending order, A = (A1, A2, A3, A4, A5, A6) and B = (B1, B2, B3, B4, B5, B6). Fig. 4.3 illustrates the process of determining the candidates for the first six minima in the set A ∪ B.

Since the elements within the same input sequence are already sorted, comparisons between them are unnecessary. The candidates of the first minimum denoted by min1 are A1 and B1. If A1 is selected as min1, it is excluded from the subsequent minimum search. The candidates for the second minimum are then A2 and B1. Similarly, if B1 is selected as the first minimum, the candidates for min2 become A1 and B2. To determine the candidates for the remaining four minima, we can apply the same principle of exclusion we use to decide for min1 and min2. This way we perform the merging of the sorted A and B, but only to the point of the sixth minimum. Some of the intermediate comparisons are repeated several times (shown by red rectangles in Fig. 4.3). The repeated comparisons are consolidated into a single comparison unit in the hardware implementation.

Fig. 4.4 depicts the resulting parallel TSs used to find six minima out of twelve elements. The intermediate candidates for the first six minima used in these TSs are shown in Fig. 4.3. The respective MVA finder which encapsulates the TSs from Fig. 4.4 is denoted by MVA_6of12 and includes

only TD comparators and MUXs.

The TS design, analysis, and the corresponding figure (Fig. 4.3) demonstrate a systematic architecture for finding the first six minima and their addresses. This methodology provides valuable insights into design of a generic MVA finder.

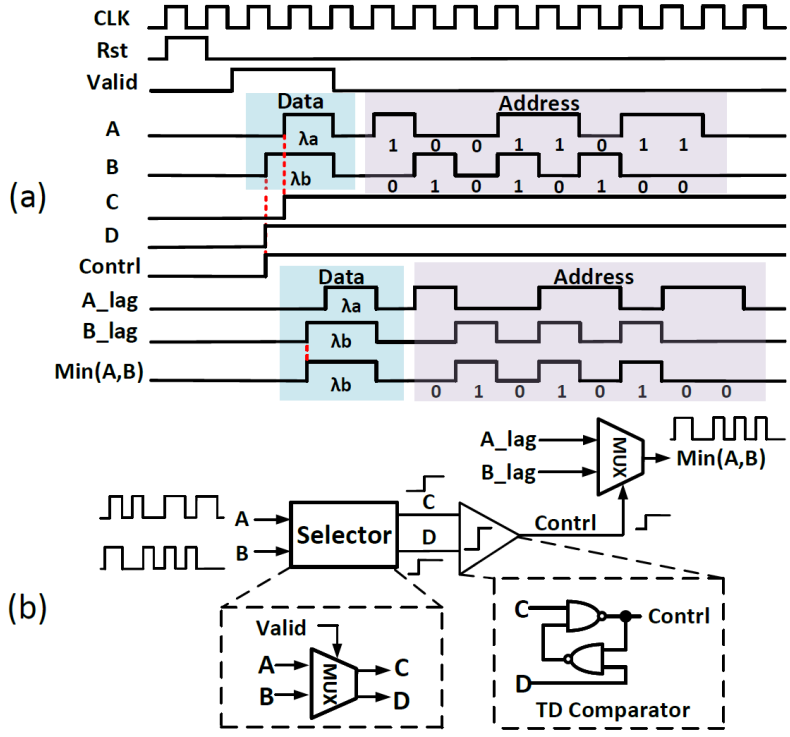


Fig. 4.2. Process of finding a minimum of two signals in the proposed AETD format. (a) Timing diagram. (b) Block diagram (MF module).

4.5 Proposed Architecture to Find Indices of the First 6 minima over 256 Inputs

By utilizing the MVA_6of12 blocks (structurally shown in Fig. 4.4), we can construct a TS architecture to find 6 minima out of 256 inputs as presented in Fig. 4.2. The inputs are split into 16 groups of 16 elements. Each group is first processed by the Sort_6of16 (denoted by 16/6 blocks in Fig. 4.2) pruned

bitonic sorting (PBS), resulting a sequence of 6 smallest elements (out of 16) sorted in an ascending order. The modified PBS (shown in Fig. 4.2) was introduced based on the full bitonic sorting [34] where instead of sorting all 16 inputs, we only find the first 6 minima and prune the rest of the graph. Each of the consecutive levels in Fig. 4.2 consists of finding 6 minima using MVA_6of12 allowing 6 minima to propagate to the root of the TS.

Instead of the 16/6 + 12/6 architecture shown in Fig. 4.2, other options to find 6 minima from 256 inputs, such as combinations of 8/6 + 12/6, 32/6 + 12/6 and 64/6 + 12/6 can also be considered. To find the optimal architecture analytically, let us consider a general problem of identifying m first minima (sorted in an ascending order) from a total of k inputs. Each module in the first stage is designed to perform PBS of n inputs. For example, the parameter values for the architecture presented in Fig. 4.2 equal m = 6, k = 256, and n = 16. Note that the principle of our proposed algorithm can be applied to find larger number of minima that may be required in GRAND-type algorithms, for example. The number of comparators and 2:2 MUXs required for the PBS architecture are given by

$$f_{pbs} = \frac{\mathrm{k}(\log_2^2 n)+\log_2 n)}{2a} - \frac{k\,L}{n} \quad (32)$$

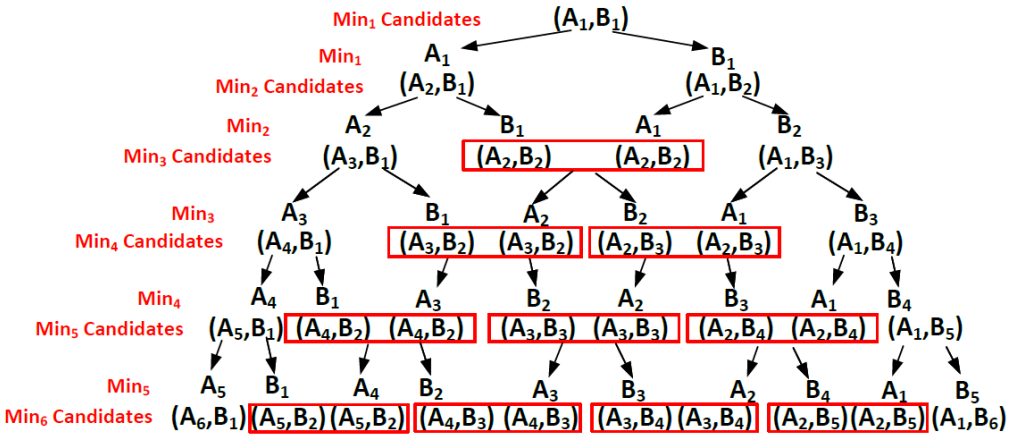


Fig. 4.3. Candidates of the first six minima identified by the proposed TS. The repeated comparisons are highlighted by red rectangles and consolidated into a single comparison unit in Fig. 4.4.

where L denotes the number of comparisons that need to be removed (pruned) from a full n × n bitonic sorting module. For instance, PBS in Fig. 4.2 corresponds to L = 13.

The expression for the number of comparators and 2:2 MUXs required for the proposed MVA finders in a k/m architecture can be computed via

$$f_{comp} = \frac{m(m+1)}{2}\sum_{j=1}^{\log_2\frac{k}{n}}\frac{k}{2^j n} \quad (33)$$

Finally, the number of 2:1 MUXs required for the proposed MVA finders in a k/m architecture is given by

$$f_{mux} = \frac{m(m+1)(m+2)}{6}\sum_{j=1}^{\log_2\frac{k}{n}}\frac{k}{2^j n} \quad (34)$$

The total critical delay is determined by multiplying the number of path delays of each module by their corresponding delay time and summing the results.

After utilizing (1) – (3), we find that the combination of 16/6 (PBS) + 12/6 (MVA) achieves the lowest complexity, while the combination of 32/6 + 12/6 achieves the lowest latency. These observations are concluded in Section 4.6.

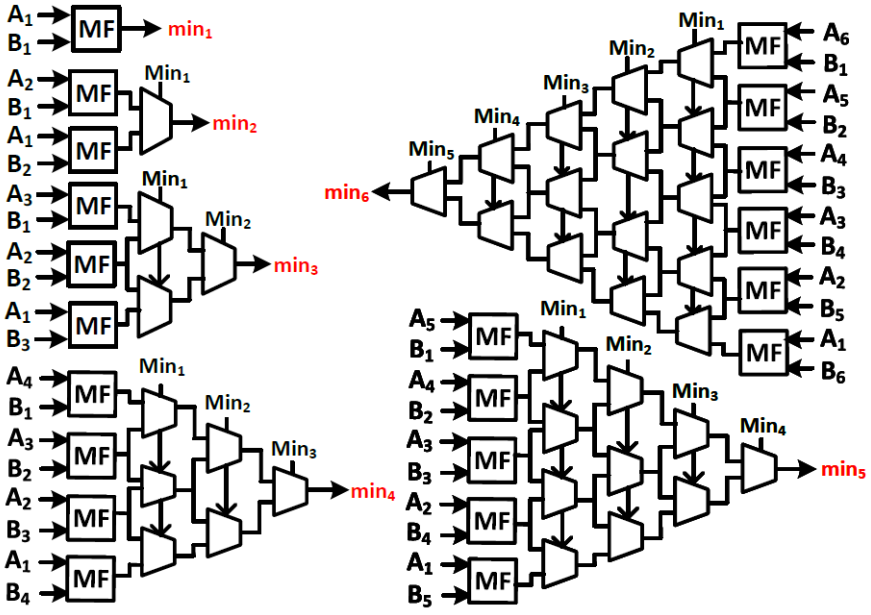


Fig. 4.4. MVA_6of12 tree structure. The inputs of each MF module are obtained from the candidates of min1-min6 shown in Fig. 4.3.

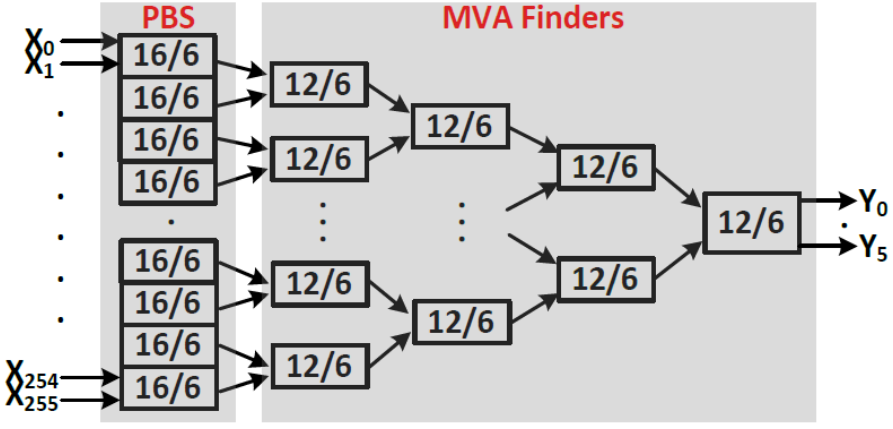


Fig. 4.5. Architecture to identify the first 6 minima and their addresses from 256 inputs. The first stage of the TS consists of 16 Sort_6of16 PBS modules. The subsequent stages involve 8, 4, 2, and 1 MVA_6of12 modules, respectively.

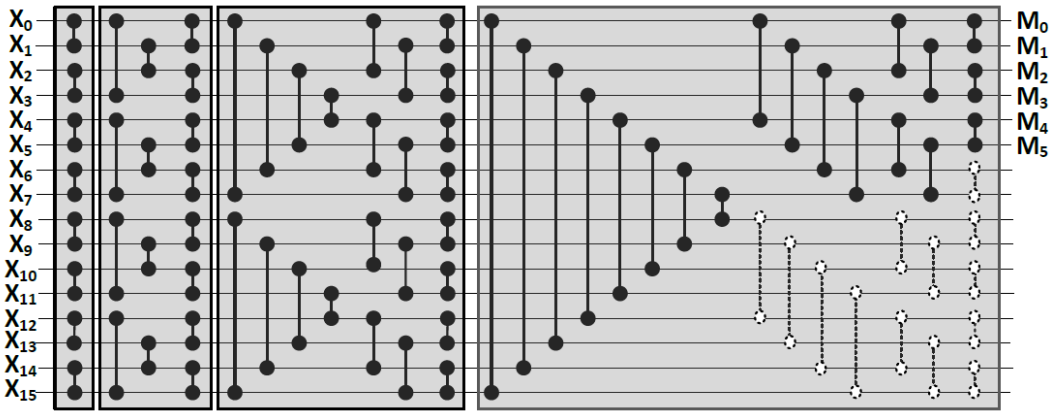


Fig. 4.6. Sort_6of16 PBS architecture based on bitonic sorting to identify the first 6 minima out of 16 inputs. Each rectangular box represents a comparison stage. The dotted lines denote pruned comparisons.

4.6 Results and Discussion

We utilized Verilog-HDL and Synopsys design tools with TSMC 65 nm CMOS technology to implement our proposed architectures and compare them with the current state-of-the-art designs [28], [70]. The layout was realized using Cadence Innovus, and post-layout simulations were conducted using Cadence Virtuoso. To ensure a fair comparison, we consider two versions of the proposed architecture. The first finds two minima (as shown in the first sub-column of the "Proposed" column in Table II), while the other focuses on finding six minima (as depicted in the second and third sub-columns of the "Proposed" column in Table II).

Table II presents a comparison between the proposed work and the existing architectures, applied to 256 inputs. All results have been scaled to 65 nm technology following the technique of [73]. The results of the proposed design

include the DTC and the pulse generator. Our proposed MVA finder with efficient TS achieves significant improvements in terms of the area and latency. The all-digital algorithms proposed in [28]-[30] require more than twice higher numbers of NAND gates compared to the proposed architecture for finding two minima. Moreover, the smallest critical delay in [28]-[30] is approximately twice as large as the critical delay of the proposed design. The serial architecture [32] reports a much larger circuit area compared to the proposed parallel architectures. The TDP architecture [70] also demonstrates a critical delay which is higher than the delay of the proposed architecture. In addition, [70] is a more complex design. Besides, the architectures [31], [70] consider fewer quantization bits. More importantly, most architectures [28]-[30], [32], [70] are designed to determine only minima values and do not provide their address information.

By using the advantages of the proposed AETD signal and an efficient TS algorithm, our proposed parallel MVA architecture achieves lower critical delay and lower complexity. It is worth noting that the proposed architecture is the only one in Table II that finds six minima with six-bit input quantization. A comparison to full six-bit and six-minimum and address finders implemented based on the state-of-the-art work could reveal even larger performance advantages of the proposed architecture.

A detailed comparison at the quantization level set to 6 bits, 65 nm technology and a frequency of 1 GHz between the proposed work, and an implemented conventional bitonic sorter, as well as the most recent digital work [31] (from Table II) has also been conducted. The synthesis results are presented in Table III. We can see that the proposed architecture demonstrates superior

TABLE II: Implementation results for the proposed design and the previous works for 256 inputs

	[33]	[34]	[35]	[36]	[37]	[38]	Proposed		
							2 mins	6 mins	
							4/2+4/2	16/6+12/6	32/6+12/6
Technology	65 nm	65 nm	65 nm	90 nm	65 nm	90 nm	65 nm	65 nm	65 nm
Result Type	Synthesis	Synthesis	Synthesis	Layout	Synthesis	Synthesis	Layout	Layout	Layout
Input quantize.	6 bits	6 bits	6 bits	5 bits	5 bits	4 bits	6 bits	6 bits	6 bits
Signal proc.	Digital	Digital	Digital	Digital	Digital	TD	TD	TD	TD
Architecture	Parallel	Parallel	Parallel	Parallel	Serial	Serial	Parallel	Parallel	Parallel
Address	No	No	No	Yes	No	No	Yes	Yes	Yes
Num. of Minima	2	2	2	4	6	6	2	6	6
NANDs (gates)	19,349	14,965	8,027	31,017	82,994	26,494	3,936	11,652	13,670
Critical Delays (ns)	2.91	3.57	2.61	4.17	N/A	2.66	1.35	2.48	2.35

[1] All results are either reported or estimated.
[2] All results are normalized to 65 nm technology.
[3] Area in 65 nm = 0.53×(area in 90 nm), power in 65 nm @1.1v = 0.65×(power in 90 nm @1.1v), delay in 65 nm = 0.74×(delay in 90 nm) [41].

performance in terms of area, critical delays, power efficiency and power consumption.

TABLE III Synthesis results of the proposed AE TDP architecture compared with digital architectures.

	Bitonic	[31]	Proposed
Signal Processing	Digital	Digital	TD
Area (mm^2)	0.115	0.043	0.016
Critical Delay (ns)	7.56	3.81	2.29
Power (mW)	38.09	33.8	1.45
Clock Cycles	114	74	48

4.6 Conclusion

The proposed MVA finder architecture combines parallel TS and TDP techniques to effectively determine the 6 smallest values and their corresponding positions in an input vector of 256 quantized real numbers. By utilizing the proposed AETD signal design and an efficient parallel TS-based architecture, the presented method offers significant improvements in terms of latency and complexity compared to the state-of-the-art. These advantages make the proposed architecture promising for construction of high-throughput error-correction decoders.

Chapter 5 A Novel Hybrid Soft Decoder for Efficient Error Correction in Noisy Systems

Yang Ge, Dmitri Truhachev, Abolfazl Zokaei, Xxiaotong Lu, Xiaoting Huang and Kamal El-Sankary,

5.1 Abstract

This chapter introduces a novel hybrid soft decoder architecture designed to enhance error correction in communication systems plagued by high noise. The architecture integrates time-domain components with traditional digital elements, optimizing performance in noisy environments such as 5G, satellite, and optical communications. We call it a "hybrid decoder" because it integrates both digital and time-domain processing techniques to achieve a more efficient and effective decoding process. In traditional decoders, all operations are typically performed in the digital domain. However, in a hybrid decoder, some of these digital components are replaced with time-domain components, such as DTCs, time-domain comparators, and time-based minimum finders. By substituting specific digital components with their time-domain counterparts, such as the Minima Finder and LLR Summation, the proposed decoder achieves significant reductions in power consumption and circuit area. Implemented using TSMC 65-nm CMOS technology and synthesized with tools from Synopsys, the decoder's effectiveness is further validated through comprehensive FPGA-based simulations. Experimental results demonstrate that the proposed hybrid approach not only reduces the physical footprint but also maintains high decoding accuracy, offering a promising solution for future high-noise communication channels.

5.2 Introduction

Modern fiber-optical communication systems operate at speeds of hundreds of gigabits per second per wavelength and require output bit error rates (BERs) below 10^{-15}. Achieving such performance necessitates advanced digital signal processing (DSP) and forward error-correction (FEC) techniques [49].

Convolutional product-like FEC schemes are widely used in fiber-optical communication standards due to their substantial net coding gains, moderate complexity, and manageable latency. Examples of these FEC schemes include the concatenated Hamming/staircase code in 400ZR [50] and the open FEC (oFEC) code [51], which are endorsed by the optical inter-networking forum (OIF) for the open ROADM multi-source agreement schemes targeting 400 Gbps communication. The oFEC is also being considered for future application in 800 Gbps links [52].

Many optical FEC systems employ BCH component codes [53], [54] with double- or triple-error-correcting capabilities because their structures are highly compatible with both Hard Decision Decoding (HDD) and Soft Decision Decoding (SDD) implementations. Both the oFEC and the staircase code [55] used in 400ZR are types of concatenated codes built from BCH component codes. Specifically, the oFEC is crafted from extended (256, 239, 2) double-error-correcting BCH codes and is decoded using a soft-input soft-output (SISO) decoding algorithm that typically involves a Chase II SDD [56] with the output refined by the Pyndiah algorithm [57]. Similarly, most soft BCH decoders are designed and implemented using various forms of the Chase algorithm [56].

To achieve high throughput under constraints of power and latency, an SDD implementation requires numerous HDDs operating in parallel on different test patterns. Alternatively, a hardware-optimized BCH HDD should utilize a technique other than the Chien search, which consumes a significant portion of the circuit area and incurs high latency [59]. While parallel implementations of the Berlekamp-Massey [60], Peterson [61], and inverse-Peterson [62]

algorithms are available, there remains a pressing need for a more efficient decoder capable of reaching speeds and throughputs beyond 800 Gbps under stringent power constraints.

This research introduces a cutting-edge hybrid soft decoder architecture that merges the accuracy of traditional digital decoding with the operational efficiency of time-domain processing (TDP). This novel approach aims to mitigate the high power and area demands of conventional soft decoders by integrating faster, more efficient TDP components. This integration promises not only to enhance decoding performance in challenging noise conditions but also to significantly reduce the power consumption and physical footprint of the decoding process.

While the proposed decoder has been synthesized using the latest technology in the Synopsys Design Compiler and designed for fabrication on a TSMC 65-nm CMOS process, the physical chip is not yet available for direct testing. However, we have thoroughly validated its performance using FPGA-based simulations. These simulations are crucial for assessing the design's functionality, timing, and power characteristics in a realistic environment. By leveraging FPGA platforms, we can emulate the behavior of the proposed decoder, ensuring it meets the stringent requirements of practical optical communication systems. This rigorous FPGA testing provides a reliable proof of concept and verifies that the design will perform as intended when the chip becomes available for fabrication and deployment.

The proposed decoder, synthesized using the latest technology in the Synopsys Design Compiler and fabricated on a TSMC 65-nm CMOS process, embodies

a leap forward in decoder design [45]-[46]. It has undergone rigorous validation through FPGA-based simulations, affirming its capability to meet the demanding conditions of practical optical communication systems [47]-[48].

The rest of this chapter is organized as follows: Section 5.3 presents the conventional hard decision decoder. The proposed soft decision decoder is described in Section 5.4. Section 5.5 presents the operation of the proposed hybrid decoder. Section 5.6 demonstrates the circuit implementations of each component used in the decoder. Section 5.7 and 5.8 presents the synthesis and FPGA results. Section 5.9 compares the proposed architecture with other literatures, followed by the conclusion in Section 5.10.

5.3 Conventional Look-Up Table (LUT)-based Hard Decision Decoder (HDD)

A conventional LUT-based hard decision decoder is a type of error-correcting mechanism used in digital communication systems to ensure data accuracy. The decoder operates within a framework where data is encoded into codewords using a specific error-correcting code, such as a BCH (Bose, Chaudhuri, and Hocquenghem) code. These codes are designed to not only detect errors but also correct them by utilizing redundant bits. The input to the decoder is a received sequence that might have been corrupted during transmission. This sequence is processed in a binary manner, meaning each bit is evaluated as either a 0 or a 1, which is why it's called "hard decision". The process of LUT-based HDD is explained in the following.

The first step in decoding is to calculate the syndrome vector from the

received sequence. The syndrome is a mathematical expression derived by multiplying the received vector by a predetermined parity-check matrix. The outcome helps in determining whether errors are present and, if so, how many. The syndrome is calculated using the parity-check matrix H of the BCH code:

$$s = V_{HD}H \quad (35)$$

where V_{HD} is the received hard decision sequence, and H the parity-check matrix. If s = 0, no errors are present. If s not equals to 0, then errors are presented. The error pattern will be used to identify the error locations.

The LUT is pre-computed and contains mappings from possible syndrome values to error locations. LUT can decode as:

$$Error\ position(s) = LUT(s) \quad (36)$$

The LUT will output either a single error position or two error positions, depending on the syndrome pattern.

Using the information from the LUT, the decoder knows the positions in where errors have occurred. It corrects these by flipping the bits at these positions.

5.4 Proposed Soft Decision Decoder (SDD)

SDD offers several advantages over LUT-based HDD, especially in the areas of error correction effectiveness and adaptability:

1. Better Error Correction: SDD uses Log-Likelihood Ratios (LLRs), providing a nuanced view of each bit's probability of being '0' or '1'. This detailed information enables more precise corrections, particularly in noisy conditions, compared to LUT-HDD, which only considers binary outcomes and may misinterpret weak signals.

2. Lower Error Floor: SDD's probabilistic approach helps achieve a lower error floor, meaning it can continue improving the bit-error rate even at high signal-to-noise ratios. LUT-HDD, lacking this probabilistic insight, limiting its effectiveness in high-quality signal environments.

These features make SDD particularly valuable in scenarios where maintaining high data integrity is crucial and where the operating conditions may vary significantly.

The proposed architecture is designed to enhance the error correction process by incorporating a blend of soft and hard decision mechanisms. The process is explained in the following (as shown in Fig. 5.1).

1. The SDD starts by receiving a vector of Log-Likelihood Ratios (LLRs), which are real numbers representing the probability estimates of the transmitted bits being a ‘0’ or a ‘1’. These LLRs form the basis for both detecting and correcting errors in the transmitted data.

2. From these LLRs, a hard decision sequence is generated. This sequence is derived by examining the sign of each LLR: if the LLR is positive, the corresponding bit in the hard decision sequence is set to '0', and if

negative, it is set to '1'. This sequence is then used to calculate the syndrome by multiplying it with the code's parity-check matrix. The syndrome helps determine the presence and location of errors.

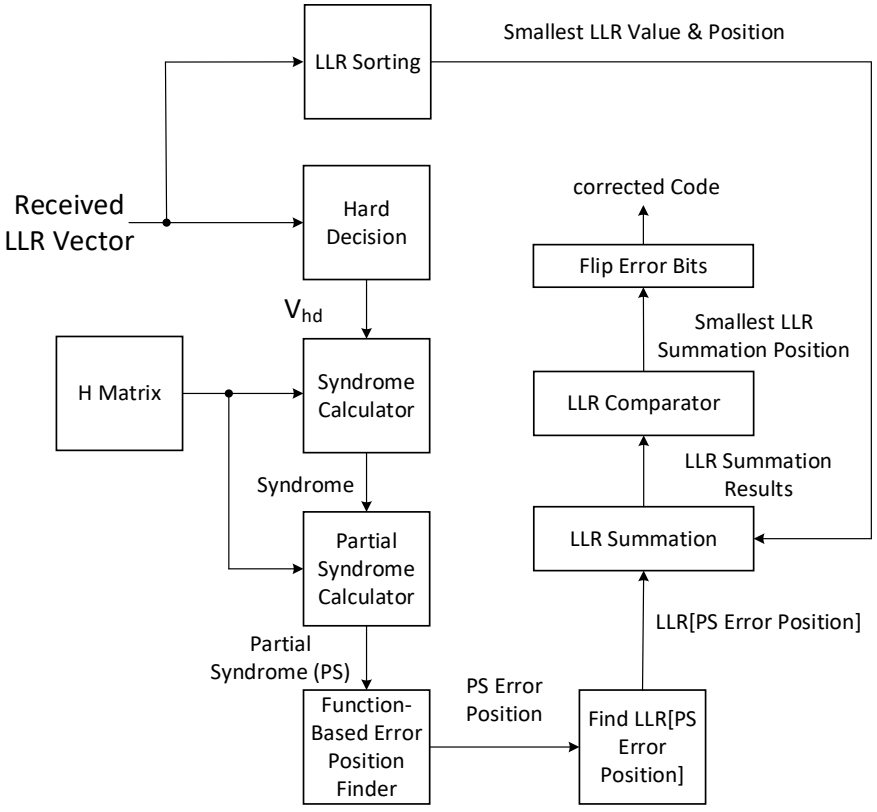


Fig. 5.1. Block diagram of SDD.

3. The decoder identifies positions of the LLRs that have the smallest magnitudes—these positions are more likely to be erroneous since their LLR values indicate less confidence in the decision. The decoder then creates various test patterns by flipping the bits at these uncertain positions in all possible combinations. Each test pattern represents a potential version of the transmitted sequence with different corrections applied.

4. Each test pattern is subjected to hard decision decoding, where the modified sequence (original hard decision sequence XOR with a test pattern) is checked for errors using the parity-check matrix. If a valid codeword is found, it is considered a potential candidate for the original transmitted sequence. The partial syndrome is calculated as:

$$PS = S \oplus h_1 \oplus h_2 \ldots \oplus h_j \quad (37)$$

Where S is the syndrome, h_j denoted as the column of the parity-check matrix H.

After applying the HDD for each test pattern T_P, the modified matrix is decoded as:

$$V_L = HDD(V_{HD} \oplus T_P) \quad (38)$$

5. For each candidate codeword generated from the test patterns, the decoder calculates an analog weight. This weight measures how much the candidate codeword differs from the original hard decision sequence, weighted by the absolute values of the LLRs. The candidate with the smallest analog weight is most likely the correct interpretation of the transmitted data, as it suggests minimal deviation from expected values under the influence of noise. The analog weight $W(V_L)$ is computed to measure deviation from V_{HD}, weighted by the absolute values of the LLRs:

$$W(V_L) = \sum_{i=1}^{k} |K_i|(V_{L,i} \oplus T_{P,i}) \quad (38)$$

where K_i is the value of LLR for the j-th bit. $V_{L,I}$ is the i-th codeword.

6. Finally, the candidate codeword with the lowest analog weight is selected as the final output of the decoder. This approach harnesses the detailed information provided by the LLRs (soft information) while effectively using hard decision corrections to refine the final decision, leading to potentially higher accuracy in error-prone conditions.

This architecture novelly combines the strengths of both soft decision inputs (for their probabilistic insights) and hard decision outputs (for their simplicity and decisiveness), aiming to optimize error correction in digital communication systems.

5.5 Operation of the Proposed Hybrid Decoder

Recall the block diagram of the hybrid soft decoder (Fig. 5.2). The proposed decoder is split into two domains, which are digital domain (green part in Fig. 5.2) and time domain (yellow part in Fig. 5.2). The components such as digital to time converter (DTC), pulse generator and address sampler, are used to connect the digital part and the TD part. The digital domain components are pre-implemented, this proposal will focus on the time domain components.

The pre-implemented digital decoder has already been designed, developed, and tested in prior work. These components are ready to be used without requiring additional design or modification efforts for the current proposal. The pre-implemented digital decoder is established and serve as a foundation or starting point, allowing the focus of the current proposal to shift towards developing and optimizing the time-domain components of the hybrid soft

decoder.

The ultimate objective of the decoder is to find the addresses of received error bits (potentially more than 1), then invert the values of these error bits based on identified addresses. Most common way to decoding soft inputs are digital decoder [15]. However, in this project, we introduce a novel hybrid soft decoder that replaces certain components of the digital decoder with counterparts in TD. By taking the advantages offered by TD, we saw further enhancements in terms of area efficiency, reduced delay, and optimized power consumption compared to conventional digital architecture. The operation process of the TD is explained in the following.

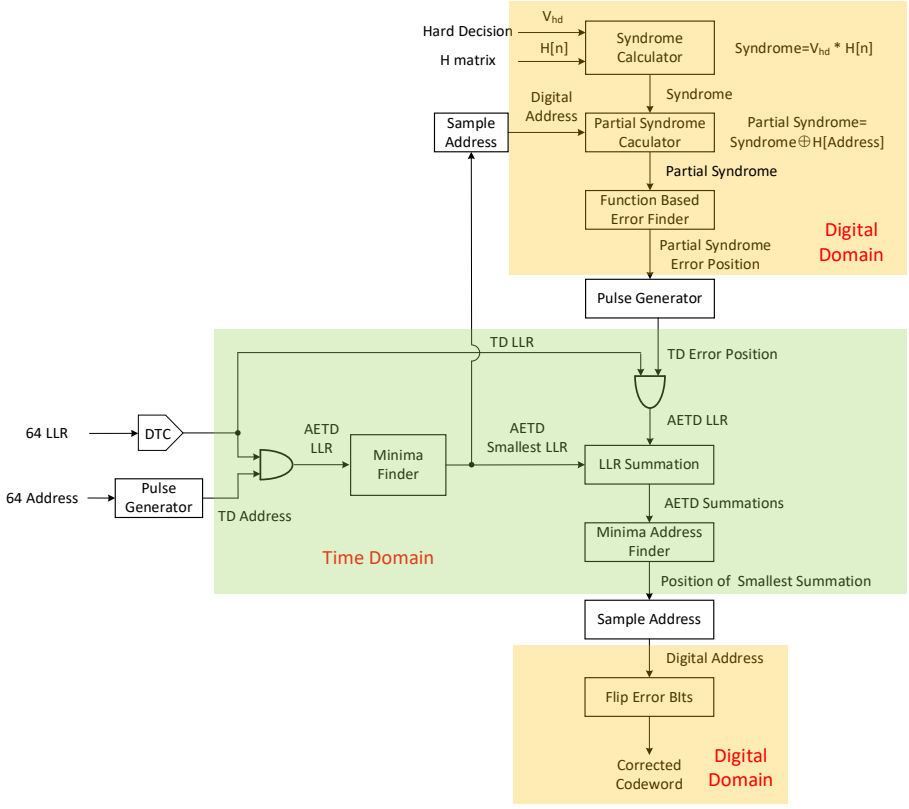


Fig. 5.2. Block diagram of a hybrid soft decoder. digital components are highlighted in yellow, and the time domain components are highlighted in green.

The nature of a hybrid design, particularly in the context of a hybrid soft decoder, refers to its combination of two different technologies or methodologies to leverage the strengths of each. In this case, the hybrid soft decoder integrates both digital and TDP techniques.

The digital domain typically offers high precision, flexibility, and scalability, making it suitable for complex operations that require exact numerical representation and manipulation. On the other hand, the time-domain approach excels in reducing power consumption, area, and latency, which is beneficial for high-throughput, low-power applications like optical communications.

By combining these two domains, the hybrid decoder aims to achieve a balance that maximizes performance while minimizing resource usage. It takes advantage of the precision and flexibility of digital circuits for certain operations while utilizing the efficiency and speed of time-domain circuits for others. This hybrid approach allows for optimized performance in terms of area, power consumption, and latency, making it well-suited for modern, high-performance, low-power applications.

As depicted in Fig. 5.2, the TD part of the proposed decoder receives 64 LLR values in digital format alongside corresponding 64 6-bits digital addresses. LLR, denoting error probability, follows an inverse relationship where smaller LLR values indicate higher error probabilities. The objective is to identify bits with the greatest error probabilities, this work will be done by minimum finder (MF). The MF locates the address of the smallest LLRs. Subsequently, the LLR Summation module processes the Address-Embedded Time-Domain

(AETD) LLR values (refer to the timing diagram in Fig. 4.2). It sums up the LLR pulses of each received AETD signal without altering their embedded addresses. Finally, the TD Address finder receives the AETD summations, compares the TD LLR values, and outputs the corresponding addresses of the smallest summation. This address represents the locations of error bits, serving as important information for the subsequent digital part to correct error bits.

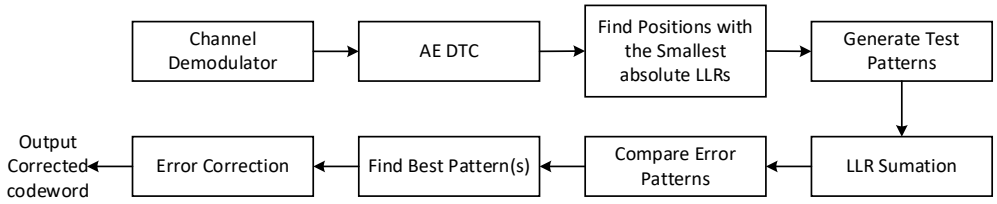


Fig. 5.3. Block diagram of the proposed hybrid soft decoder.

The process of the proposed hybrid decoder is presented in Fig. 5.3. The channel demodulator inputs digital LLRs that pass through an AE DTC converter, this DTC converts the LLR values with their corresponding addresses from digital format into TD. Then the minimum finder identifies positions of the LLRs that have the smallest magnitudes—these positions are more likely to be wrong since their LLR values indicate less confidence in the decision. Subsequently, test patterns are generated according to Chase II algorithm. Test patterns attempt to flip all combinations of 0s and 1s in the p positions where p are the smallest LLRs located (in our design p = 4). For each candidate codeword generated from the test patterns, the decoder calculates an analog weight. Finally, the candidate codeword with the lowest analog weight is selected as the final output of the decoder.

5.6 Circuit Implementations of Time Domain Components

Since all the digital parts (yellow highlighted in Fig. 5.2) are implemented conventionally, this section focuses on the circuit implementation of TD components (green part highlighted in Fig. 4.4).

5.6.1 Concept of the Time Register

In time domain, the signal information represented as the time difference between two edges associated with a single pulse or two consecutive pulses. Therefore, store and retrieve time information precisely is crucial in TDP. The fundamental component of TDP is time register, Fig. 5.4 and Fig. 5.5 show the circuit implementation and the timing diagram, respectively. The operational principle is explained in the following.

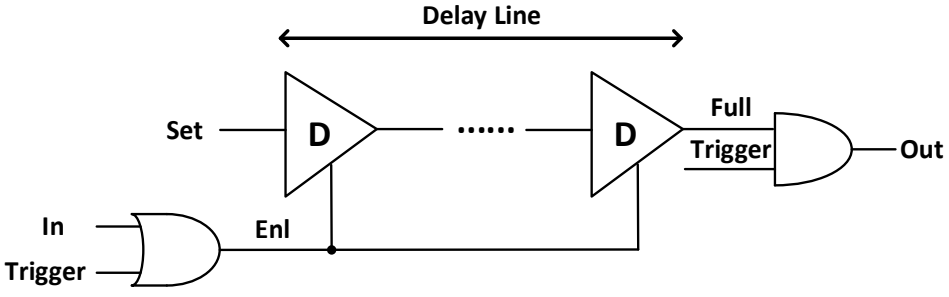


Fig. 5.4. Circuit implementation of time register.

As shown in Fig. 5.4, a delay line consists of several delay cells. Delay cell's on and off are controlled by *Enl*, delay cell turns on (signal can propagate through delay cell) when *Enl* = 1, delay cell turns off (signal can not propagate through delay cell) when *Enl* = 0. Firstly, *Set* goes high while *Trigger* remains at low. Upon propagating a pulse, denoted as *In*, with a width of T_{in} enters the time register as input. *Set* starts propagates through delay line when *In* is high.

Subsequently, when *In* logically reverts to 0, the delay line turns off, holding

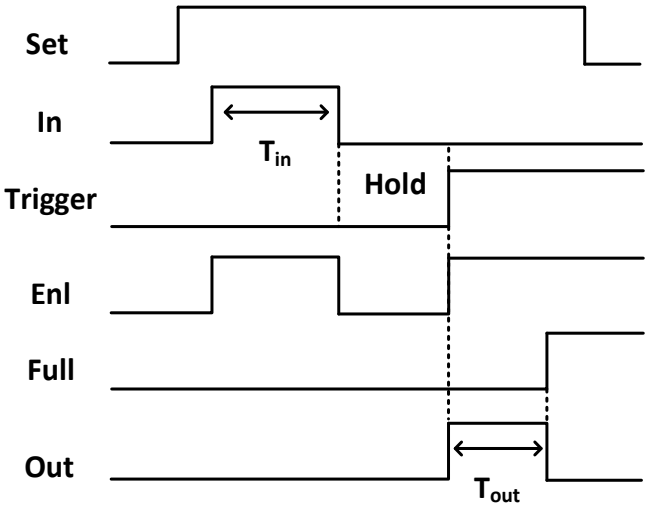


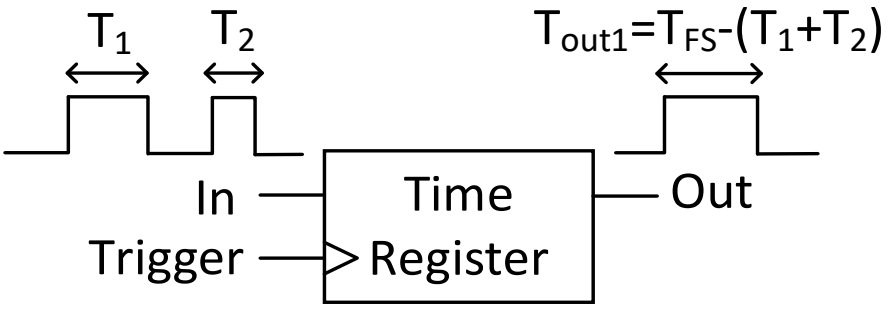
Fig. 5.5. Timing diagram of time register.

the time phase not change. Trigger then flips high, initiating the propagation of the time phase for reading out the time information. Eventually, *Sel* reaches the end of the delay line, triggering *Full* to flip. The time difference between the two rising edges of *Trigger* and *Full* is denoted as T_{out}. Given that the full-scale propagation time T_{FS} remains fixed, T_{out} can be derived as follows:

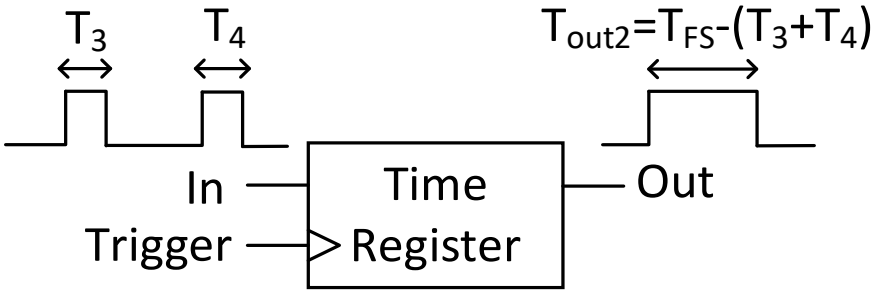
$$T_{out} = T_{FS} - T_{in}$$

Consequently, T_{out} is output corresponding to T_{in}. Finally, *Set* flips down to reset *Full*. After a duration of T_{FS}, *Full* logically returns to 0 in preparation for the subsequent registration cycle.

5.6.2 Operation of the time register in the proposed hybrid decoder



(a)



(b)

Fig. 5.6. Operation of time register used in the proposed hybrid decoder. (a) Find the time value T_{out1}. (b) Find the time value T_{out2}.

Fig. 5.6 shows the operation process used in the proposed hybrid decoder, Fig. 5.6 (a) and (b) are two examples to find specific time values T_{out1} and T_{out2}. In the proposed hybrid soft decoder, we have the requirement to compare LLR summations in TD. In Fig. 5.6 (a), the LLRs are in the TD format as the input enter time register, ex., T_1 and T_2 represent two TD LLRs. After the time register, T_{out1} is computed as $T_{out1} = T_{FS} - (T_1 + T_2)$. Similarly, in Fig. 5.6 (b), T_{out2} is calculated as $T_{out2} = T_{FS} - (T_3 + T_4)$. In order to identify which summation ($T_1 + T_2$ or $T_3 + T_4$) is bigger, we only need a simple comparison of the pulse width of the outputs from each time register. Apparently, in the example of Fig. 5.6, T_{out2} exhibits a bigger pulse width, indicating that $T_1 + T_2$

has the bigger summation. These outputs from each time register are subsequently fed into the following minimum address finder for further processing.

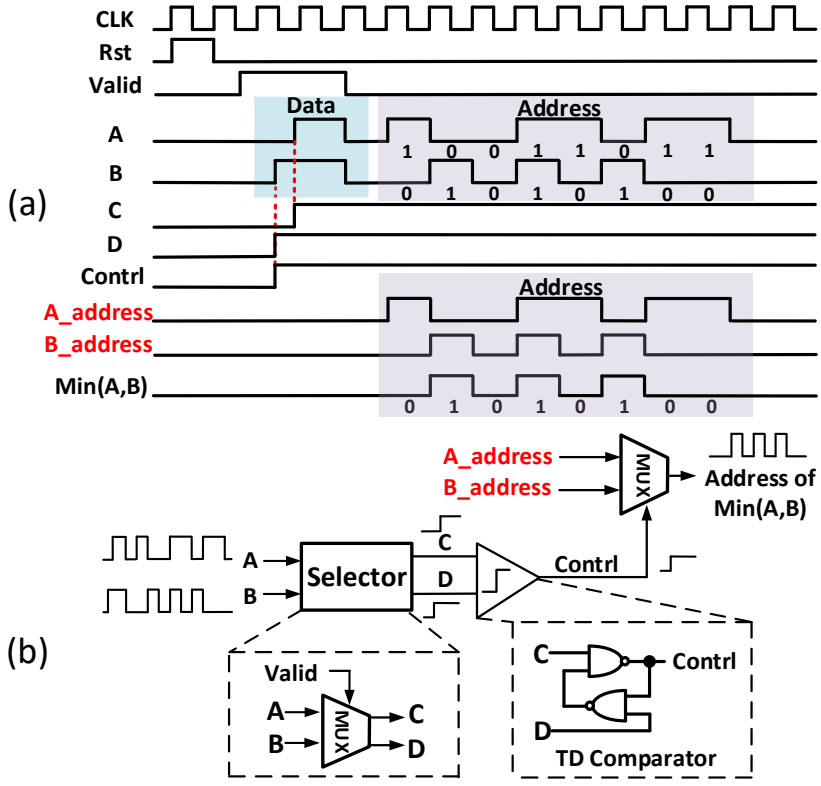


Fig. 5.6. Process of finding address of two AETD LLRs. (a) Timing diagram. (b) Block diagram.

5.6.3 Time-Domain Address Finder

In Fig. 5.6, the process for determining the address of the minimum LLR is illustrated. Signals *A* and *B* represent input AETD LLR summations, and the output is the address corresponding to the smaller summation.

Referencing Fig. 4.2 utilized in the MVA finder, we employ the same architecture while substituting the input signals of the MUX from *A_lag* and *B_lag* to *A_address* and *B_address*. Since our primary concern is identifying the address of the smallest LLR summation, the actual summation value is not

required. Consequently, only the address is extracted and forwarded to the subsequent digital part for error correction.

5.7 Synthesis Results and Discussions

Table IV highlights the synthesis results for the **entire** proposed hybrid decoder, comparing them directly with the pre-implemented digital decoder. This table evaluates the performance of both decoders based on three key parameters: area, delay, and power consumption, offering a thorough assessment of their overall operational efficiencies.

The hybrid decoder, a focal point of this comparison, demonstrates superior efficiency across all measured parameters. It occupies an area of only 0.099 um^2, which is significantly less than the 0.192 um^2 required by the pre-implemented digital decoder. This smaller area suggests that the hybrid decoder is more compact, making it suitable for applications where space is at a premium.

In terms of operational speed, the hybrid decoder achieves a delay of just 1.89 ns, which is less than half the 4.10 ns recorded for the pre-implemented digital decoder. This improvement in delay indicates that the hybrid decoder can process signals much faster, enhancing the throughput of the system in which it is deployed.

Power efficiency is another area where the hybrid decoder excels, consuming only 0.43 mW of power compared to the 1.18 mW used by the pre-implemented digital decoder. This reduction in power consumption not only

makes the hybrid decoder more energy-efficient but also reduces the overall operational costs and heat generation, which is beneficial for maintaining the longevity and reliability of the system.

Table IV

Synthesis results of the proposed hybrid decoder.

Signal Process	Area (mm^2)	Delay (ns)	Power (mW)
Hybrid	0.099	1.89	0.43
Pre-implemented digital decoder	0.192	4.10	1.18

5.8 FPGA Verification

5.8.1 Ports Description

Fig. 5.8 provides a detailed visual representation of the input and output ports for the BCH decoder, categorizing them distinctly with inputs on the left side and outputs on the right. This arrangement helps in clearly understanding the flow of data into and out of the decoder. Below is a further breakdown of these ports as listed in Table V, which includes the type, size, and specific role of each port in the decoder's functionality.

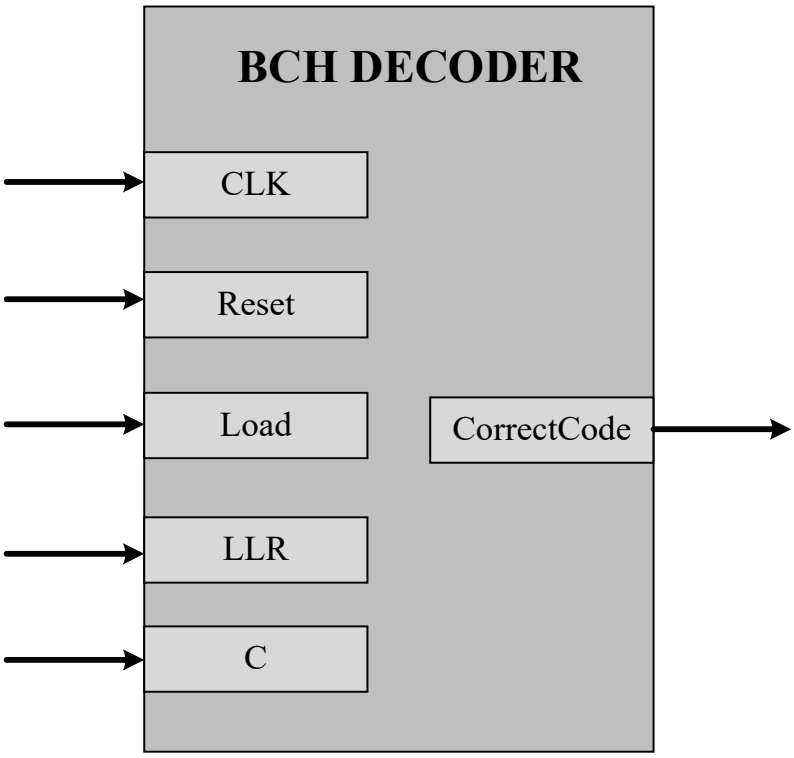


Fig. 5.8. BCH decoder ports diagram.

This table provides data paths and control signals that drive the decoding operations within the BCH decoder. The clock (CLK) and reset signals are fundamental for timing and control, ensuring that the decoder operates synchronously and can be reset to a known state as needed. The load signal is particularly important as it indicates when the incoming data is valid and ready for processing.

The LLR inputs are critical for the soft decoding process, where the likelihood of each bit being a '0' or '1' is considered, rather than making binary decisions based on hard thresholds. This approach typically results in higher fidelity decoding, especially in noisy conditions. The hard decision outputs (C) and the corrected outputs (CorrectCode) provide interfaces for further error correction stages or for interfacing with other system components, ensuring

that the decoded data is reliably transmitted or stored.

Table V

BCH Ports Description

Ports Name	Type	Size (bits)	Description
CLK	In	1	Clock
Reset	In	1	Reset
Load	In	1	Input code valid
LLR	In	640	Log-Likelihood ratio for each bit of BCH code
C	In	64	Hard decision channel output
CorrectCode	Out	64	Corrected BCH output code

5.8.2 Verification Process and Testbench Setup

Fig. 5.9 illustrates a verification process for an FPGA-based system using MATLAB and a PC for simulations and interfacing. Here's a breakdown of each component and its role in the system:

1. **Random Bits Generator (MATLAB)**: This component generates random binary data, serving as the input to the Encoder. It simulates the data that would typically be processed by the system in a real-world scenario.
2. **Encoder (MATLAB)**: The Encoder receives the random bits and encodes them, preparing the data for transmission through a noisy channel. This step is crucial for testing the resilience and effectiveness

of the encoding scheme against noise.

3. **Noise Generator (MATLAB)**: Following the encoding process, the data passes through a Noise Generator which simulates various types of communication noise or interference that might affect the data during transmission. This component is essential for creating realistic test conditions to evaluate the decoder's performance.
4. **Output Codewords:** The encoded and noise-added data is then fed into an FPGA, specifically into two components.
5. **Decoder**: The proposed (256,239,2) BCH decoder processes the noisy data to recover the original information. The decoder's ability to correct errors and retrieve the original data is a key focus of the verification process.
6. **ILA** (Integrated Logic Analyzer): This tool is used for debugging and performance analysis within the FPGA. It captures and displays data from the FPGA circuits during the decoding process, allowing for real-time monitoring and troubleshooting.
7. **PC**: The PC interfaces with the FPGA, likely using software tools to configure the FPGA, initiate the verification process, and analyze the output data from the FPGA. It may also be used to adjust parameters in the MATLAB simulation based on the results observed from the FPGA outputs.

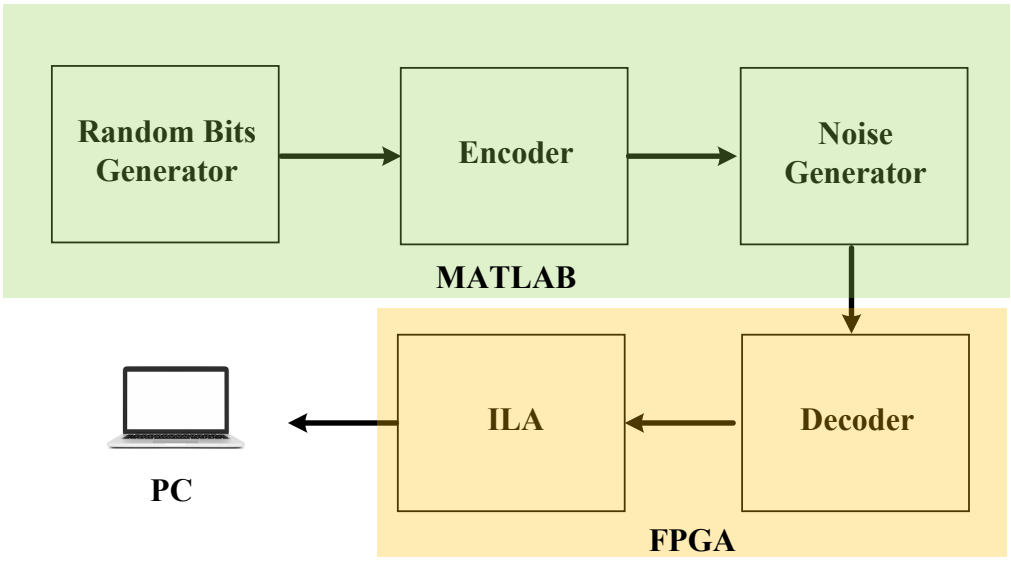


Fig. 5.9. Block diagram verification process.

Overall, this setup forms a closed-loop system where data is generated, encoded with errors introduced, and then processed by the FPGA for error correction. The effectiveness of the decoding process can be evaluated based on how well the original data is recovered, with the ILA providing insights into the FPGA's internal operations and the PC facilitating control and analysis of the entire process.

Fig. 5.10 illustrates our FPGA testbench setup used for debugging and analyzing the decoder. The FPGA board serving as the primary hardware platform for testing. A JTAG interface is employed to program and debug the FPGA, enabling data transfer and real-time configuration connected to a PC. An ILA is embedded within the FPGA design to capture and analyze internal signals, the results of which are displayed on the attached monitor. The monitor shows graphical waveforms of the FPGA's internal signals, provides a detailed view of the internal signal states and timing diagrams.

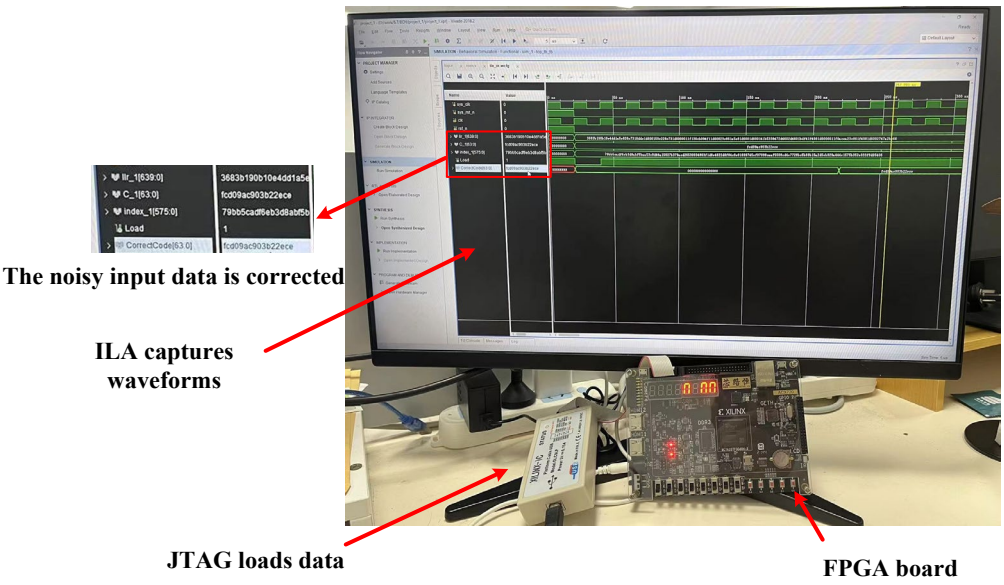


Fig. 5.10. FPGA testbench setup.

5.8.3 FPGA Results

In this research, the FPGA design was utilized by using the XILINX Artix-7 family of devices within the Vivado 2020.2 environment. The design optimization focused on enhancing performance metrics while ensuring efficient resource utilization, critical for real-time processing applications.

The proposed architecture achieved a maximum clock frequency of 425 MHz, indicating its capability for high-speed operations essential for data-intensive applications. And the power consumption is 205 mW.

Table VI

Resource used of the proposed (256,239,2) decoder

Resource	Used	Available	Utilization
FF	12,587	332,800	4%
LUT	56,240	332,800	17%
BRAM	2	100	2%
DSP	18	90	20%
Bonded IOBs	10	300	3%
Number of slices	120	900	13%

The resource utilization Table VI for the proposed (256,239,2) decoder on an FPGA shows a generally low to moderate use of available resources. Notably, the use of Flip-Flops (FF) and Block RAM (BRAM) is particularly low, at 4% and 2% respectively, suggesting that the decoder's implementation does not heavily rely on extensive state storage or large-scale data buffering. This indicates a design that efficiently manages memory resources.

On the other hand, Look-Up Tables (LUTs) and Digital Signal Processors (DSPs) show higher utilization rates at 17% and 20% respectively. This reflects a moderate to significant engagement of logic processing and arithmetic operations within the decoder, critical for its function. The utilization of 13% of the available slices further supports the notion of a moderately complex decoder that effectively balances performance with resource use.

Overall, the resource deployment across the FPGA for this decoder design is cost-effective and efficient. It optimizes the hardware capabilities without overwhelming the system, suggesting a well-balanced approach to handling decoding tasks within the given resource constraints.

The bit error rate (BER) over varying (SNR) for both FPGA results of BCH decoder, theoretical BCH decoder, HDD, and uncoded are shown in Fig. 5.11. Derived from the MATLAB while sending 1000 frames over 6 samples.

Fig. 5.11 illustrates the Bit Error Rate (BER) performance of various decoding schemes across a range of Signal-to-Noise Ratios (SNR) measured in dB. Each curve represents a different coding or decoding approach, providing insight into their respective efficiencies and effectiveness in error correction.

The blue dashed line in Fig. 5.11, representing the HDD, shows a gradual decrease in BER as the SNR increases. This curve is indicative of traditional hard decision decoding techniques, where decisions on bit values are made strictly based on whether received signals are above or below a certain threshold. While the performance improves with increasing SNR, it consistently exhibits higher error rates compared to more sophisticated BCH code implementations.

The red cross line in Fig. 5.11, which denotes the performance of the standard BCH decoder, is noticeably below the HDD line, demonstrating better error correction capability. The BCH code, known for its ability to correct multiple errors within a codeword, leverages algebraic methods to provide more reliable data transmission, particularly at higher SNR levels where its error-correcting advantages become more pronounced.

The yellow circle line in Fig. 5.11 depicts the BCH decoder implemented on a FPGA, aligning closely with the standard BCH curve. This alignment suggests that the FPGA implementation effectively replicates the theoretical

performance of BCH decoding, confirming the practical applicability and effectiveness of BCH codes in hardware-based systems. The FPGA's ability to perform parallel processing likely contributes to this high level of performance, showcasing the suitability of BCH codes for real-world applications requiring robust error correction.

Finally, the purple triangle line in Fig. 5.11 represents the uncoded transmission, serving as a baseline for comparison. As expected, this curve shows the highest BER across all SNR levels. The absence of any error correction coding in the uncoded scenario results in the poorest performance, highlighting the significant impact of sophisticated coding techniques like HDD and BCH on enhancing data integrity in noisy environments.

In summary, the graph clearly demonstrates the effectiveness of BCH codes over traditional HDD and uncoded methods, with FPGA implementations showing that hardware integration can achieve performances close to theoretical predictions. The distinction between the curves emphasizes the importance of selecting appropriate error correction coding techniques to minimize data transmission errors, particularly in applications where reliability is critical.

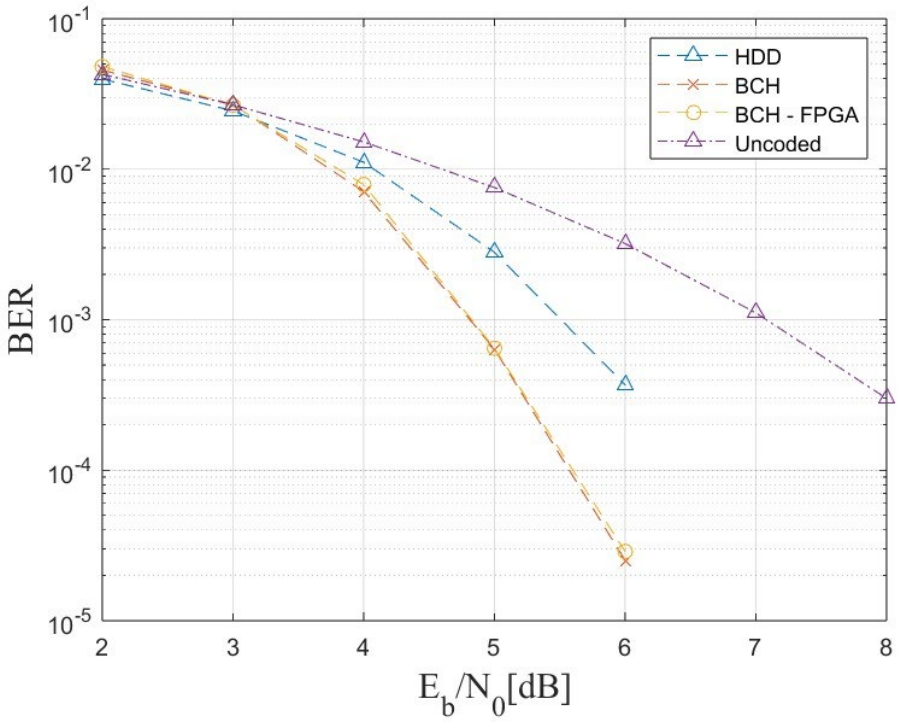


Fig. 5.11. BER vs SNR for BPSK modulation.

Fig. 5.12 presents the circuit diagram of the proposed hybrid BCH decoder post FPGA verification, illustrating its architecture and the flow of signal processing.

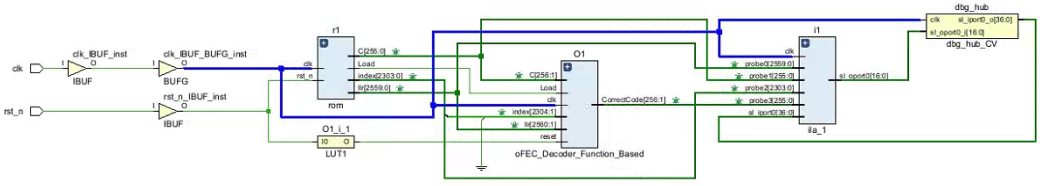


Fig. 5.12. Circuit of the proposed BCH decoder after FPGA verifying.

5.9 Comparison with other State-of-arts

Table VII

Comparison of synthesis results between the proposed **hybrid soft BCH decoder** and recent literatures

	Proposed	Pre-implemented digital decoder	[51]	[52]	[53]	
BCH Code	(256,239,2)	(256,239,2)	(127,113,2)	(63,51,2)	(79,64,2)	
Architecture	Combinational Logic	Combinational Logic	GRAND	Peterson	Peterson	LUT
Power (mW)	0.43	1.18	4.41	2.35	1.30	1.17
Area (mm^2)	0.099	0.192	1.250	0.011	0.003	0.010
Delay (ns)	1.89	4.10	-	2.35	3	3

Table VII provides a detailed comparison of synthesis results for the proposed hybrid soft BCH decoder against both a pre-implemented digital decoder and several recent innovations in the field, as cited in the literature. The table is meticulously organized to evaluate these technologies based on their power consumption, area, and delay—critical metrics that influence the efficiency and suitability of digital decoders in practical applications.

The proposed hybrid Soft BCH decoder showcases significant advancements in minimizing power consumption and circuit area while achieving a remarkably low delay. It consumes only 0.43 mW, considerably less than the pre-implemented digital decoder and other cited works, which range from 1.17 to 4.41 mW. Additionally, its circuit area is just 0.099 um^2, which is notably smaller than others, with the nearest competitor occupying 0.010 um^2. The operational speed of the proposed decoder, reflected by a delay of 1.89 ns, is also competitive, particularly compared to the 4.10 ns of the pre-implemented

decoder.

The decoder's performance underscores its potential in high-speed, low-power digital communication environments. By integrating efficient combinatorial logic architecture, the proposed decoder not only surpasses the traditional digital decoder configurations but also stands out against recent scholarly contributions that utilize various decoding architectures such as GRAND, Peterson, and LUT methods. Each of these methods, represented in the table, offers different benefits in terms of power, area, and delay, yet the proposed hybrid Soft BCH decoder appears to offer an optimal balance among these parameters.

This synthesis analysis confirms the proposed decoder’s superiority in technical specifications, making it a compelling choice for future implementations in systems where power efficiency, compactness, and speed are paramount. The comparative data provided in the table highlights the technological advancements achieved by the proposed decoder, positioning it as a leading solution in the realm of error-correcting decoders.

5.10 Conclusion

The development and implementation of the hybrid soft decoder, integrating time-domain components with traditional digital processing elements, marks a significant advancement in the field of digital communications, particularly in environments characterized by high noise. This novel architecture successfully addresses the primary challenges traditionally associated with soft decoders, namely high-power consumption and extensive silicon area requirements, through its innovative use of time-domain processing.

The hybrid decoder demonstrates substantial improvements in decoding efficiency, evidenced by its ability to operate at a maximum clock frequency of 525 MHz while maintaining minimized resource utilization and power efficiency. By replacing specific digital components with time-domain equivalents, the decoder not only reduces the physical footprint but also enhances the overall reliability and accuracy of the decoding process.

This project highlights the potential of hybrid decoding solutions to address the evolving demands of modern communication systems, paving the way for the development of more efficient, robust, and scalable communication technologies. Although the chip is not yet available, the successful FPGA implementation and thorough validation of the hybrid decoder demonstrate its applicability and effectiveness, establishing it as a promising solution for enhancing error correction in next-generation digital communication infrastructures.

Chapter 6 Conclusion and Future Plan

6.1 Conclusion

In this thesis, a STSD architecture with single TSV and single DCDL was proposed to synchronize the clocks between multiple dies. The performance of our architecture was verified theoretically and experimentally regarding mismatch/finite resolution of delay lines, buffer mismatch, and TSV delay. By using the single-TSV and single-delay-line structure, compared to the state-of-arts, the proposed STSD has the smallest lock-in time, lowest power and chip area. STSD can operate at a range from 250 MHz to 1 GHz, with a maximum residual skew of 23 and 11 ps in Die2 and Die3, respectively. In addition, STSD is designed with standard cells in a TSMC 65-nm CMOS process, thus, the proposed approach can be easily adapted to different processes.

The proposed MVA finder architecture combines parallel TS and TDP techniques to effectively determine the 6 smallest values and their corresponding positions in an input vector of 256 quantized real numbers. By utilizing the proposed AETD signal design and an efficient parallel TS-based architecture, the presented method offers significant improvements in terms of latency and complexity compared to the state-of-the-art. These advantages make the proposed architecture promising for construction of high-throughput error-correction decoders.

the hybrid soft decoder presents a promising advancement in error correction technology, leveraging a strategic integration of time-domain components within a digital decoder framework. Through this innovative approach, the decoder achieves notable enhancements in various performance metrics,

including area efficiency, delay reduction, and power consumption. By replacing select digital components with counterparts operating in the time domain, the decoder effectively mitigates the limitations inherent in traditional digital decoders. The synthesis results prove the efficacy of the hybrid decoder, highlighting its potential to significantly improve error correction capabilities in communication systems and beyond. This innovative solution gives the way for further advancements in decoding methodologies, offering greater efficiency and adaptability in addressing errors in complex data transmission scenarios.

6.2. Future Plan

Building on the achievements detailed in this thesis, future work could focus on several enhancements and expansions. Initially, exploring the scalability of the STSD architecture to accommodate more than three dies could be a starting point. Testing the design with an increased number of dies while aiming to maintain or improve metrics such as lock-in time, power, and chip area could prove beneficial. Further research might also investigate implementing the STSD and hybrid soft decoder architectures using smaller process nodes than the TSMC 65-nm CMOS process, potentially at 45-nm or 28-nm, to exploit the benefits of scaling in reducing power consumption and enhancing performance.

Integrating the STSD architecture and hybrid soft decoder into larger system-on-chip (SoC) platforms would be beneficial, examining interactions with other system components and optimizing the interface for broader applications, particularly in fields requiring high data integrity and speed. Additionally, refining the algorithms used in the MVA finder could reduce computational

overhead and improve efficiency. The application of machine learning techniques to dynamically predict and optimize sorting and selection processes offers a promising area for algorithmic enhancements.

Extending testing to real-world applications such as high-speed communication systems and error-sensitive data processing units would provide invaluable feedback on performance in practical scenarios and highlight areas for improvement. Conducting detailed studies on the energy efficiency of the hybrid soft decoder across various operating conditions could also be undertaken to make the technology feasible for energy-constrained applications. Lastly, assessing the robustness of the proposed designs against environmental variabilities such as temperature fluctuations and electromagnetic interference would ensure reliability across diverse operational conditions.

Developing design automation tools that facilitate the design, simulation, and implementation of STSD and hybrid decoding architectures could make the technology more accessible to other researchers and practitioners, thus contributing significantly to the advancement of digital communications and signal processing technologies.

References

[1] J. Park, M. Cheong and S. Kang, "R2-TSV: A Repairable and Reliable TSV Set Structure Reutilizing Redundancies," in IEEE Transactions on Reliability, vol. 66, no. 2, pp. 458-466, June 2017, doi: 10.1109/TR.2017.2681103.

[2] Sandhu, T. S., & El-Sankary, K. (2019). Supply-insensitive digitally controlled delay lines for 3-D IC clock synchronization architectures. IEEE Transactions on Very Large Scale Integration (VLSI) Systems, 27(6), 1480-1484.

[3] J. Ke, S. Huang, C. Tzeng, D. Kwai and Y. Chou, "Die-to-Die Clock Synchronization for 3-D IC Using Dual Locking Mechanism," in IEEE Transactions on Circuits and Systems I: Regular Papers, vol. 60, no. 4, pp. 908-917, April 2013, doi: 10.1109/TCSI.2012.2215394.

[4] C. Chung and C. Hou, "All-digital delay-locked loop for 3D-IC die-to-die clock synchronization," Technical Papers of 2014 International Symposium on VLSI Design, Automation and Test, 2014, pp. 1-4, doi: 10.1109/VLSI-DAT.2014.7834902.

[5] T. S. Sandhu and K. El-Sankary, "A Mismatch-Insensitive Skew Compensation Architecture for Clock Synchronization in 3-D ICs," in IEEE Transactions on Very Large Scale Integration (VLSI) Systems, vol. 24, no. 6, pp. 2026-2039, June 2016, doi: 10.1109/TVLSI.2015.2496312.

[6] Ching-Che Chung, Chi-Yu Hou, “An all-digital delay-locked loop for 3-D ICs die-to-die clock deskew applications”, Microelectronics Journal, Volume 70, 2017, Pages 63-71.

[7] M. Sadi, S. Kannan, L. England and M. Tehranipoor, "Design of a digital IP for 3D-IC die-to-die clock synchronization," 2017 IEEE International Symposium on Circuits and Systems (ISCAS), 2017, pp. 1-4, doi: 10.1109/ISCAS.2017.8050431.

[8] C. -Y. Yang, M. -S. Li and A. -J. Chuang, "A Wide-Range Folded-Tuned Dual-DLL-Based Clock-Deskewing Circuit for Core-to-Core Links," in IEEE Transactions on Very Large Scale Integration (VLSI) Systems, vol. 29, no. 5, pp. 883-894, May 2021, doi: 10.1109/TVLSI.2021.3056506.

[9] K. N. Dang, A. B. Ahmed, A. B. Abdallah and X. -T. Tran, "A Thermal-Aware On-Line Fault Tolerance Method for TSV Lifetime Reliability in 3D-NoC Systems," in IEEE Access, vol. 8, pp. 166642-166657, 2020, doi: 10.1109/ACCESS.2020.3022904.

[10] K. N. Dang, M. C. Meyer, A. B. Ahmed, A. B. Abdallah and X. Tran, "A Non-Blocking Non-Degrading Multiple Defects Link Testing Method for 3D-Networks-on-Chip," in IEEE Access, vol. 8, pp. 59571-59589, 2020, doi: 10.1109/ACCESS.2020.2982836.

[11] M. M. Navidi and G. Byun, "Comparative analysis of clock distribution networks for TSV-based 3D IC designs," Fifteenth International Symposium on Quality Electronic Design, 2014, pp. 184-188, doi: 10.1109/ISQED.2014.7783323.

[12] S. J. Park, N. Natu and M. Swaminathan, "Analysis, Design, and Prototyping of Temperature Resilient Clock Distribution Networks for 3-D ICs," in IEEE Transactions on Components, Packaging and Manufacturing Technology, vol. 5, no. 11, pp. 1669-1678, Nov. 2015, doi: 10.1109/TCPMT.2015.2482947.

[13] R. Kuttappa, B. Taskin, S. Lerner and V. Pano, "Resonant Clock Synchronization with Active Silicon Interposer for Multi-Die Systems," in IEEE Transactions on Circuits and Systems I: Regular Papers, vol. 68, no. 4, pp. 1636-1645, April 2021, doi: 10.1109/TCSI.2021.3059228.

[14] T. Ni et al., “LCHR-TSV: Novel low cost and highly repairable honeycomb-based TSV redundancy architecture for clustered faults,” IEEE Trans. Comput.-Aided Design Integr. Circuits Syst., vol. 39, no. 10, pp. 2938–2951, Oct. 2020, doi: 10.1109/TCAD.2019.2946243.

[15] Z. Gong and R. Rashidzadeh, “TSV extracted equivalent circuit model and an on-chip test solution,” IEEE Trans. Comput.-Aided Design Integr. Circuits Syst., vol. 35, no. 4, pp. 679–690, Apr. 2016, doi: 10.1109/TCAD.2015.2474411.

[16] S. Wang, K. Chakrabarty, and M. B. Tahoori, “Defect clustering-aware spare-TSV allocation in 3-D ICs for yield enhancement,” IEEE Trans. Comput.-Aided Design Integr. Circuits Syst., vol. 38, no. 10, pp. 1928–1941, Oct. 2019, doi: 10.1109/TCAD.2018.2864291.

[17] S. Chen, Q. Xu, and B. Yu, “Adaptive 3D-IC TSV fault tolerance structure generation,” IEEE Trans. Comput.-Aided Design Integr. Circuits Syst., vol. 38, no. 5, pp. 949–960, May 2019, doi: 10.1109/TCAD.2018.2824284.

[18] Q. Min, E.-P. Li, and J.-M. Jin, “Lumped 3-D equivalent thermal circuit model for transient thermal analysis of TSV array,” in Proc. IEEE Int. Conf. Comput. Electromagn. (ICCEM), Mar. 2019, pp. 1–3, doi: 10.1109/COMPEM.2019.8779007.

[19] S. Lin and D. J. C. Jr., Error Control Coding. Pearson, 2004.

[20] “Implementation agreement 400ZR,” OFI Internetworking Forum, Mar. 2020.

[21] “Open ROADM MSA 3.01 W-port digital specification (200G-400G),” Acacia Commun. Inc. 3 Mill and Main, Sep. 2019.

[22] Li, Shizhong, Kamal El-Sankary, Alireza Karami, and Dmitri Truhachev. "Area-and power-efficient staircase encoder implementation for high-throughput fiber-optical communications." IEEE Transactions on Very Large Scale Integration (VLSI) Systems 28, no. 3 (2019): 843-847.

[23] D. Truhachev, K. El-Sankary, A. Karami, A. Zokaei, and S. Li, “Efficient implementation of 400 Gbps optical communication FEC,” IEEE Trans. Circuits Syst. I, Reg. Papers, vol. 68, no. 1, pp. 496–509, Jan. 2021.

[24] Zokaei, A., Truhachev, D., & El-Sankary, K. (2022). Memory optimized hardware implementation of open FEC encoder. IEEE Transactions on Very Large Scale Integration (VLSI) Systems, 30(10), 1548-1552.

[25] K. R. Duffy, W. An, and M. Medard, “Ordered reliability bits guessing random additive noise decoding,” IEEE Trans. Signal Process., vol. 70, pp. 4528–4542, 2022.

[26] R. Hadavian, D. Truhachev, K. El-Sankary, H. Ebrahimzad, and H. Najafi, “Ordered reliability direct error pattern testing (ORDEPT) algorithm,” IEEE Global Commun. Conf., Dec. 2023.

[27] S. Hwang, S. Moon, J. Jung, D. Kim, I.-C. Park, J. Ha, and Y. Lee, “Energy-efficient symmetric BC-BCH decoder architecture for mobile storages,” IEEE Trans. Circuits Syst. I, Reg. Papers, vol. 66, no. 11, pp. 4462–4475, 2019.

[28] B. Park, S. An, J. Park, and Y. Lee, “Novel folded-KES architecture for high-speed and area-efficient BCH decoders,” EEE Trans. Circuits Syst. II, Exp. Briefs, vol. 64, no. 5, pp. 535–539, 2017.

[29] D. Jo, S. Kwon, and D.-J. Shin, “Blind reconstruction of BCH codes based on consecutive roots of generator polynomials,” IEEE Communications Letters, vol. 22, no. 5, pp. 894–897, 2018.

[30] C.-H. Yang, T.-Y. Huang, M.-R. Li, and Y.-L. Ueng, “A 5.4 μw soft decision BCH decoder for wireless body area networks,” IEEE Trans. Circuits Syst. I, Reg. Papers, vol. 61, no. 9, pp. 2721–2729, 2014.

[31] S. Choi, H. K. Ahn, B. K. Song, J. P. Kim, S. H. Kang, and S.-O. Jung, “A decoder for short BCH codes with high decoding efficiency and low power for emerging memories,” IEEE Trans. Very Large Scale Integr. (VLSI) Syst., vol. 27, no. 2, pp. 387–397, 2019.

[32] R. Pyndiah, “Near-optimum decoding of product codes: block turbo codes,” IEEE Trans. Commun., vol. 46, no. 8, pp. 1003–1010, 1998.

[33] C.-L. Wey, M.-D. Shieh, and S.-Y. Lin, “Algorithms of finding the first two minimum values and their hardware implementation,” IEEE Trans. Circuits Syst. I, Reg. Papers, vol. 55, no. 11, pp. 3430–3437, 2008.

[34] Y. Lee, B. Kim, J. Jung, and I.-C. Park, “Low-complexity tree architecture for finding the first two minima,” IEEE Trans. Circuits Syst. II, Exp. Briefs, vol. 62, no. 1, pp. 61–64, 2015.

[35] X. Zhu and G. He, “An area time-efficient structure to find the approximate first two minima for Min-Sum-based LDPC decoders,” IEEE Trans. Circuits Syst. II, Exp. Briefs, vol. 63, no. 8, pp. 793–797, 2016.

[36] T. X. Pham and H. Lee, “Efficient first four minimum values finder for NB-LDPC decoders with compressed messages,” IEEE Trans. Circuits Syst. II, Exp. Briefs, vol. 69, no. 3, pp. 1024–1028, 2022.

[37] J. L. J. Tian, S. Song and Z. Wang, “Optimized trellis-based Min-Max decoder for NB-LDPC codes,” IEEE Trans. Circuits Syst. II, Exp. Briefs, vol. 67, no. 1, pp. 57–61, 2020.

[38] M.-R. Li, C.-H. Yang, and Y.-L. Ueng, “A 5.28-Gbps LDPC decoder with time-domain signal processing for ieee 802.15.3c applications,” IEEE J. Solid-State Circuits, vol. 52, no. 2, pp. 592–604, 2017.

[39] Z. Chen and J. Gu, “High-throughput dynamic time warping accelerator for time-series classification with pipelined mixed-signal time-domain computing,” IEEE J. Solid-State Circuits, vol. 56, no. 2, pp. 624–635, 2021.

[40] F. E. Yuan, CMOS Time-Mode Circuits and Systems: Fundamentals and Applications (1st ed.). CRC Press, 2016.

[41] A. Stillmaker and B. Baas, “Scaling equations for the accurate prediction of CMOS device performance from 180nm to 7nm,” Integration, the VLSI Journal, vol. 58, Feb. 2017.

[42] H. Y. B. Yong Kong and I. C. Park, “Efficient sorting architecture for successive-cancellation-list decoding of polar codes,” IEEE Trans. Circuits Syst. II, Exp. Briefs, vol. 63, no. 7, pp. 673–677, 2016.

[43] Johnson, A., & Smith, B. (2018). "Energy Efficiency in Communication Systems." IEEE Transactions on Communications, 66(12), 4729-4741.

[44] Lee, C., & Kim, D. (2020). "Area-Efficient Designs of FEC Decoders for Optical Communications." Journal of Optical Communications and Networking, 12(4), A1-A10.

[45] Davis, R., & Patel, M. (2017). "Hybrid Decoding: Bridging the Gap Between Digital and Time-Domain Processing." IEEE Transactions on Very Large Scale Integration (VLSI) Systems, 25(8), 2275-2287.

[46] Thompson, H., & Garret, M. (2018). "Improving Decoding Techniques Through Hybrid Systems." Journal of Digital Signal Processing, 28(1), 55-64.

[47] Wang, F., et al. (2022). "FPGA-Based Testing of Communication Decoders." IEEE Transactions on Industrial Electronics, 69(11), 10854-10862.

[48] Singh, A., & Rao, J. (2019). "On the Validation of Communication Systems with FPGA." Journal of Communications Technology and Electronics, 64(12), 1239-1246.

[49] Y. Ge, T. S. Sandhu, D. Truhachev and K. El-Sankary, "A Single-TSV and Single-DCDL Approach for Skew Compensation of Multi-Dies Clock Synchronization in 3-D-ICs," in IEEE Transactions on Very Large Scale Integration (VLSI) Systems, vol. 31, no. 4, pp. 567-577, April 2023, doi: 10.1109/TVLSI.2023.5244489.

[50] Y. Ge, D. Truhachev, A. Zokaei, X. Lu, X. Huang and K. El-Sankary, "Efficient Address-Embedded Time-Domain Implementation of Minima Finders for Soft Error-Correction Decoders," in IEEE Transactions on Circuits and Systems II: Express Briefs, doi: 10.1109/TCSII.2024.4401701.

[51] A. Riaz, V. Bansal, A. Solomon, W. An, Q. Liu, K. Galligan, K. Duffy, M. Medard, R. Yazicigil, “Multi-code multi-rate universal maximum likelihood decoder using GRAND,” European Solid State Circuits Conference (ESSCIRC), Grenoble, France, pp. 239-246, September 2021.

[52] C. Yang, T. Huang, M. Li, Y. Ueng, “A 5.4μW Soft-Decision BCH Decoder for Wireless Body Area Networks,” IEEE Trans. Circuits Syst. I, Reg. Papers, vol. 61, no. 9, pp. 2721-2729, September 2014.

[53] S. Choi, H. Ahn, B. Song, J. Kim, S. Kang, S. Jung, “A decoder for short BCH codes with high decoding efficiency and low power for emerging memories,” IEEE Transactions On Very Large Scale Integration (VLSI) Systems, vol. 27, no. 2, pp. 387-397, February 2019.

[54] L. Zhang, K. Tao, W. Qian, W. Wang, J. Liang, Y. Cai, and Z. Feng,“Real-time FPGA investigation of interplay between probabilistic shaping and forward error correction,” J. of Lightwave Tech.. vol. 40, no. 5, pp. 1339-1345, March 2022.

[55] “Implementation agreement 400ZR,” OFI Internetworking Forum, March 2020. [Online]. Available: https://www.oiforum.com/wp-content/uploads/ OIF-400ZR-01.0_reduced2.pdf

[56] “Open ROADM Proposal Draft,” Acacia Communications, June 2019. [Online]. Available: www.openRoadm.org

[57] C. Condo, “Iterative soft-input soft-output decoding with ordered reliability bits GRAND,” IEEE Globecom Workshops, pp. 510-515, Rio de Janeiro, Brazil, December 2022.

[58] A. Hocquenghem, “Codes correcteurs d’erreurs,” Chiffres (in French), vol. 2, pp. 147–156, 1959.

[59] R. C. Bose and D. K. Ray-Chaudhuri, “On a class of error correcting binary group codes,” Inform. and Control, vol. 3, pp. 68–79, Mar. 1960.

[60] B. P. Smith, A. Farhood, A. Hunt, F. R. Kschischang, and J. Lodge, “Staircase codes: FEC for 100 Gb/s OTN,” J. of Lightwave Tech., vol. 30, no. 1, pp. 110–117, January 2012.

[61] D. Chase, “Class of algorithms for decoding block codes with channel measurement information,” IEEE Trans. On Inform. T., vol. 18, no. 1, pp. 170-182, January 1972.

[62] R. M. Pyndiah, “Near-optimum decoding of product codes: Block turbo codes,” IEEE Trans. Commun., vol. 46, no. 8, pp. 1003-1010, Aug. 1998,

[63] D. Kim, I. Yoo, I. Park, “Fast low-complexity triple-error-correcting BCH decoding architecture,” IEEE Transactions On Circuits And Systems II: Express Briefs, vol. 65, no. 6, pp. 764-768, June 2017.