ISSN NO: 9726-001X
Volume 13 Issue 02 2025

JOURNAL OF CURRENT SCIENCE

Efficient VLSI Implementation of Hybrid LDPC-STBC Codes for Enhanced SatelliteCommunication System

K. Pavan Kumar[1], G. Varun [2], M. Vishnuvardhan Reddy [3], Dr. N. Sowmya [4]
[1,2,3] UG Scholar, Department of ECE, St. Martin's Engineering College, Secunderabad, Telangana, India-500100
[4]Assistant Professor, Department of ECE, St. Martin's Engineering College, Secunderabad, Telangana, India-500100
kamtampavan97@gmail.com

Abstract:

The implementation of a Hybrid Low-Density Parity-Check (LDPC) and Space-Time Block Code (STBC) coding system offers a significant enhancement in satellite communication by improving data reliability and reducing error rates. According to recent industry reports, the global satellite communication market is projected to grow at a CAGR of 9.2%, reaching $46.5 billion by 2027. The demand for robust error-correction methods is increasing due to the growing complexity of satellite-based communications, particularly in low-power, high-noise environments. Traditional error-correction methods like Hamming codes struggle to meet the performance requirements of modern communication systems, limiting data throughput and increasing latency.

This research presents a hybrid LDPC-STBC system designed to address the limitations of existing methods. LDPC codes are known for their high error-correcting capabilities, while STBC codes improve reliability through spatial diversity. The system uses LDPC for encoding and decoding, leveraging syndrome calculation for error detection and correction. The STBC component applies recursive systematic convolutional (RSC) codes and Maximum A Posteriori (MAP) process for efficient decoding. The integration of LDPC and STBC encoding and decoding processes provides a highly reliable communication system, especially for satellite-based networks operating in adverse conditions. This hybrid approach enhances both data integrity and transmission efficiency, making it an ideal solution for satellite communication.

Keywords: LDPC, STBC, Maximum A Posteriori, Recursive systematic convolutional codes, data integrity, Hamming codes, Latency, Satellite communication, Spatial diversity.

1.INTRODUCTION

Error correction codes (ECC) are essential tools in the realm of digital communication and data storage, ensuring the integrity and reliability of transmitted or stored information in the presence of errors. At its core, ECC involves the addition of redundant bits to data, enabling the detection and correction of errors that occur during transmission or storage processes. These codes are crucial in mitigating the effects of noise, interference, or hardware faults that can corrupt data. The concept of ECC traces back to the early days of digital communication, where researchers recognized the need for mechanisms to detect and correct errors. Over time, various ECC techniques have been developed, ranging from simple parity checks to more sophisticated algorithms capable of correcting multiple errors and detecting patterns of errors known as bursts.

ECC operates on the principle of redundancy, where additional bits are added to the original data based on mathematical algorithms. These redundant bits enable the receiver to detect errors by comparing the received data with the expected data. In cases where errors are detected, ECC algorithms use redundancy to correct the errors, restoring the original data to its intended state. In addition to enhancing data reliability, ECC techniques are also crucial for increasing data storage density and improving transmission efficiency. By incorporating ECC into storage systems and communication protocols, data was stored more densely and transmitted more reliably, leading to higher performance and lower error rates.

Finally, ECC plays a fundamental role in ensuring the integrity of digital data, enabling robust communication and storage solutions in a wide range of applications, from wireless communication and satellite systems to computer memory and data centers. As digital technology continues to advance, ECC techniques will remain indispensable for maintaining the integrity and reliability of digital information. Error correcting codes operate on the principle of adding redundancy to the original data such that errors was detected and corrected at the receiver end. These codes typically add extra bits (parity bits or redundancy) to the original data based on mathematical algorithms.

ECC algorithms are tailored to strike a delicate balance between hardware complexity, power consumption, and error correction efficiency. Engineers leverage specialized hardware architectures and optimization techniques to design ECC circuits that meet the stringent requirements of modern digital systems. Moreover, ECC modules must seamlessly integrate into larger communication or storage systems, adhering to established protocols and incorporating fault-tolerance mechanisms to handle errors within the ECC circuitry itself.

ECC is crucial for maintaining data reliability and integrity in various digital systems, including telecommunications, data storage, satellite communications, and computer memory. By employing ECC, these systems can operate with higher accuracy and efficiency, even in the presence of adverse conditions that introduce errors. As digital technology continues to advance, ECC techniques will remain indispensable for ensuring the seamless and reliable operation of digital communication and storage systems.

ECC is motivated by the imperative to ensure data integrity, support high-speed data transmission, maximize storage density, minimize energy consumption, adapt to emerging technologies, enhance security, and maintain system reliability in diverse applications, spanning telecommunications, data storage, digital media, and emerging technologies. Figure 1 shows the research motivation with application scenario.

ISSN NO: 9726-001X
Volume 13 Issue 02 2025

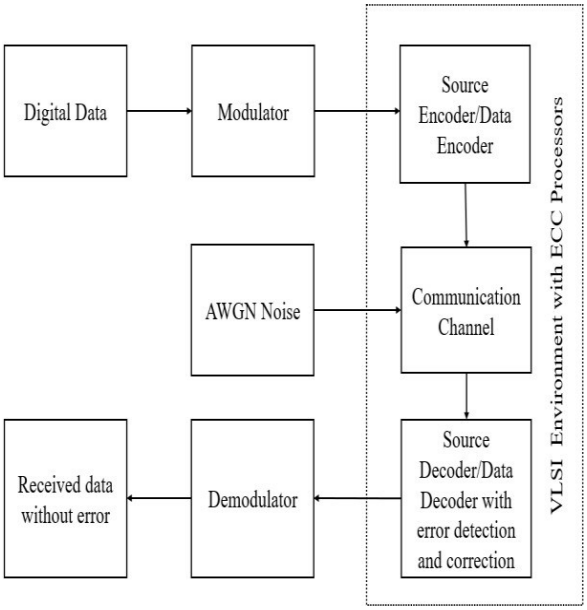

Figure 1.1. Research Motivation.

Digital Data: In the digital data stage of a communication system, analog or digital information is converted into digital format, organized into packets, compressed, if necessary, encrypted for security, and enhanced with ECC.

Modulator: In the modulator stage of a communication system, the digital signal is superimposed onto a carrier wave to facilitate transmission. Modulation techniques like amplitude modulation (AM), frequency modulation (FM), or phase modulation (PM) are employed to encode the digital information onto the carrier wave. This process shapes the characteristics of the carrier wave according to the digital signal, making it suitable for propagation through the communication medium to reach the receiver.

Source encoder/Data encoder: In VLSI with ECC processors, a source encoder compresses data for efficient transmission or storage. ECC processors encode data with ECC for reliable communication. Together, they optimize resource usage and enhance system performance. Source encoding reduces data size, while ECC ensures error detection and correction. These components are integral to achieving efficient and robust data handling in VLSI environments with ECC processors.

Communication channel: In the communication channel stage, digital data encoded by the source encoder traverses through a medium encountering noise, such as Additive White Gaussian Noise (AWGN), which can corrupt the signal. The channel distorts and attenuates the transmitted signal. Encoding techniques like pulse code modulation (PCM) ensure data integrity, while noise analysis helps understand and mitigate AWGN's effects, ensuring reliable communication despite environmental disturbances.

source decoder/data decoder: In VLSI with ECC processors, a source decoder decompresses received data to its original format. ECC processors decode data, correcting errors introduced during transmission or storage. They work together to ensure accurate data recovery and integrity. The source decoder restores compressed data to its original state, while ECC decoding rectifies errors, maintaining data fidelity. These components are vital for reliable data retrieval and utilization in VLSI environments with ECC processors.

Demodulator: In the demodulator stage following the source decoder in a communication system, the received modulated signal is separated from the carrier wave. Demodulation reverses modulation, retrieving the original digital data decoded at the source, facilitating further processing or delivery to the destination device for interpretation or display.

Received data without error: In the Received data without error stage of a communication system, the transmitted signal is successfully received without errors. This stage indicates that the demodulated signal matches the original transmitted data, ensuring accurate delivery and enabling subsequent processing or utilization of the information by the recipient device.

2. LITERATURE SURVEY

Singh, S. Pratap, et al. [1] developed Molecular Communication (MC) as a multidisciplinary branch that lay at the junction of nano, bio, and communication technology. MC evolved to serve almost every field of humanity, be it biomedical, environmental, or security against NBC attack. On the other hand, MC lagged in technological demonstration and development level. However, like any communication system, EEC played a vital role in MC systems, improving system performance. Recently, literature presented VLSI implementations of Cyclic Reed–Muller (C-RM) and Hamming code. However, VLSI implementations and demonstrations of Self-Orthogonal Convolution Codes (SOCC) and LDPC codes, which outperformed others, were not available in the literature. Therefore, this literature implemented SOCC and LDPC codes in Tanner EDA Tool followed by presenting the power consumption and delay of respective codes. More specifically, this paper presented the design and implementation of an encoder and decoder of SOCC for MC systems using Tanner EDA Tool and demonstrated the performance in terms of power consumption and delay. In addition, a similar analysis was presented for the LDPC code. It is important to mention that the implemented decoder of each of the LDPC and SOCC codes employed a newly presented MLG circuit in its manuscript. Finally, power consumption and delay of SOCC and LDPC were compared with those of available ECCs in the literature.

Silva, Felipe, et al. [2] explored how the evolution of microelectronics boosted more scalable and complex circuit designs, providing high processing speed and greater storage capacity. However, reliability issues grew significantly as electronic devices scaled down, increasing the fault rate, mainly in critical applications exposed to radiation. Memories were sensitive to charged particles, which corrupt data due to transient effects. ECC were highly applied to mitigate data failures, increasing memory reliability. The matrix region selection code (MRSC) was an ECC designed to correct a high rate of adjacent errors in memory but less effectively for nonadjacent errors. However, MRSC had a 2-D structure that made it challenging to implement in memory where one address was accessed at a time. This article introduced the triple burst error correction based on region selection code (TBEC-RSC), an ECC that used MRSC concepts, converting the MRSC format to a 1-D structure. TBEC-RSC was implemented and evaluated in a 16-bit data version; however, the code was easily extensible to the higher base-2 data words (e.g., 64 bits). Experimental results showed that TBEC-RSC corrected 100% of triple burst errors and more than 40% of 8-bit burst errors.

Chu, Syuna-A Ke, et al. [3] introduced Guessing random additive noise decoding (GRAND) as a recently proposed code-agnostic decoding technique for linear block codes, which attempted to guess the possible error pattern applied on the received word to check if the result was a valid codeword. The GRAND with abandonment (GRANDAB) served as a hard-detection decoder by limiting the number of generated test error patterns. This article presented efficient algorithms to reduce the number of queries for the GRANDAB when the codes were systematic and cyclic. These methods exploited the properties of the syndrome weight and cyclic codes to significantly improve the decoding latency as the GRANAB corrected up to the error-correcting capability of the code. The VLSI

architecture of the novel hard-detection GRANDAB was presented, which provided efficient decoding up to 128 bits and correct up to 3-bit errors at or above the error-correcting capability of the code. The property of syndrome weight was integrated with the dial structure for parallelism, supporting the default mode and the mode of systematic encoding with the known error-correcting capability. When compared to the architecture developed by Abbas et al., the average decoding cycles of two and three errors for the (127, 106) Bose–Chaudhuri–Hocquenghem (BCH) code were improved by 30.90% and 48.63%, respectively. For the (128, 96) cyclic redundancy check (CRC) code, the average decoding cycles of two and three errors were reduced by 45.19% and 65.71%, respectively. At the signal-to-noise ratio (SNR) of 5 dB, the average latencies for decoding (128, 96) CRC and (127, 106) BCH codes were improved by 21.34% and 12.29%. The worst-case latency for decoding the (127, 113) BCH code in the presented design was shorter than that of the work implemented by Riaz et al. with the reduction of 98.02%. The developed hardware architecture was also superior to the original dial-based design by Abbas et al. in terms of area–time (AT) complexity.

Kuo, Yao-Ming, et al. [4] illustrated the recommendation by the Consultative Committee for Space Data Systems (CCSDS) to utilize short-block length Bose–Chaudhuri–Hocquenghem and binary LDPC. Despite the significant error-correction capacity of nonbinary low-density parity-check (NB-LDPC) codes, their consideration had been lacking due to the complexity associated with decoding. Our research focused on assessing the feasibility of implementing NB-LDPC coding for space telecommand link applications by employing an RISC-V soft-core processor in conjunction with a vector coprocessor. The objective of this study was to eliminate the necessity for dedicated decoder hardware, thereby enabling the reconfiguration of the customized general-purpose processor responsible for decoding to handle other crucial onboard tasks. This approach aimed to reduce logic utilization and power consumption by allowing the onboard processor to accommodate additional functionalities. Additionally, we demonstrated a method to accelerate the NB-LDPC decoder over GF (16) using the RISC-V vector extension, achieving a throughput of 8.48 kb/s for the forward–backward implementation of the min–max decoding algorithm. This throughput was found to be compatible with the low-rate and mid-rate telecommand systems recommended by the CCSDS.

Grurl, Thomas, Christoph Pichler, et al. [5] discussed the challenges posed by frequent noise effects in real quantum computers due to the fragility of quantum mechanical effects, resulting in errors during computations. Quantum error-correcting codes were proposed as a solution to identify and correct these errors. However, much of the research on quantum error correction had been theoretical or limited to specific hardware models. Furthermore, the development and evaluation of corresponding codes often relied on tedious trial-and-error methods. In their work, they introduced an open-source framework designed to assist engineers and researchers in automatically applying error-correcting codes for a given application, followed by an automatic noise-aware quantum circuit simulation. Case studies illustrated that this approach led to a significantly more efficient implementation and evaluation of error-correcting codes.

Wu, Yujun, Bin Wu, et al. [6] developed the QC-LDPC code, with its excellent error correction performance and hardware friendliness, and was identified as one of the channel encoding schemes by Wi-Fi 6. Shortening, puncturing, or repeating operations were needed to ensure that user data was sent with integer symbols and complete rate matching. Due to the uncertainty of the user data size, the modulation's selectivity, and the difference in the number of spatial streams, the receiver had to deal with more than 106 situations. At the same time, other computationally intensive tasks occupied the time slot budget of the receiver. Typical were demodulation and decoding. Hence, the receiver needed to quickly reverse the demodulated data process. This literature first proposed a co-processing method and VLSI architecture compatible with all code lengths, code rates, and processing parameters. The co-processor separated field and block splicing, simplifying the control logic. There was no throughput rate bottleneck, and the maximum delay was less than 1 μs.

Pokhrel, Nabin Kumar, et al. [7] presented a class of integer codes capable of correcting burst asymmetric errors. The presented codes were constructed with the help of a computer and had the potential to be used in various practical systems, such as optical networks and VLSI memories. In order to evaluate the performance of the proposed codes, the literature analyzed the probability of erroneous decoding for different bit error rates. The presented codes were also analyzed from a rate-efficiency point of view. The obtained results showed that for many data lengths they required less check-bits than optimal burst error correcting codes.

Saini, Madan Lal, et al. [8] proposed: "In digital communication, various single- and double-bit error correcting and detecting codes were available. The efficiency of an error correcting code was evaluated by its error correction capabilities and redundancy. This paper presented new single bit and double bit error correcting codes which had lower redundancy compared to other existing codes. In this literature, the number of parity bits over the message bits for Hamming, BCH, RS Code, and DEC were examined and overhead was calculated. The proposed codes had less parity bits compared to other existing ones and had up to double bit error correction capabilities, minimizing the encoding/decoding time delay."

Ponmalar, VJ Beulah Sherin, et al. [9] developed a convolution code namely LDPC which was proposed and implemented in VLSI architectures (FPGA). In general, a digital communication system suffered from errors due to noise, distortion, and interference during data transmission and various algorithms commonly used to correct the errors. A third-generation wireless communication system used convolutional codes for the transmission of voice and control signals. Whereas in the existing system, the convolution code namely the turbo codes were used with the belief propagation algorithm that played a vital role in VLSI implementation which had low bit error rate and low signal-to-noise ratio but high decoding error rate and high decoding complexity. The tanner graphs were used to construct longer codes from smaller ones to perform row and column operations. The LDPC used the message passing algorithm that acted as the superior performer and iterative decoding technique for low decoding complexity and high decoding rate that helped in parallel processing for updating a large number of messages and transferring between check nodes and bit nodes. The optimized method of Moore's Law was implemented. This attracted public attention as a code for the fourth generation in communication systems and provided high security. The simulation results indicated that our proposed scheme reduce power dissipation by overcoming the soft error process, which was the incorrect switching or an impractical activity of a memory cell and provided an optimal circuit to enhance VLSI Technologies.

Boncalo, Oana, et al. [10] proposed a novel iterative decoding method - Gradient Descent Symbol Update - for real number parity-based ECC, as well as its corresponding hardware architecture. The decoding process was based on the gradient descent optimization technique, as well as binary maximum likelihood error correction decoding. The error correction performance and the convergence rate of the gradient descent symbol update for a parity-based BCH (26,16) code were presented. Furthermore, FPGA implementation results for the corresponding decoder architecture were depicted.

Dutta, Shruti, et al. [11] proposed the applications involving machine learning and neural networks had become increasingly essential in the AI revolution. Emerging trends in Resistive RAM technologies provided high-speed, low-cost, scalable solutions for such applications. These RRAM cells provided efficient and sophisticated memory hardware structures for machine-learning applications. However, it was difficult to achieve reliable multilevel cell storage capacity in these memory technologies due to the

occurrence of soft and hard errors. As these memories store multi-bits per cell, exploring limited magnitude symbols (multi-bit) error correction in RRAM was important. This paper proposed a new syndrome-based double error correcting code that divided the syndromes into groups and used addition and XOR operations to correct double limited magnitude errors in the RRAM cells. The key idea was to use the built-in current summing capability of RRAM cells to perform the addition operations that were used for the error correction, thereby greatly reducing the overhead of the decoding logic needed to implement the ECC. This effectively avoided the need for explicit adder hardware in the decoding logic making it smaller and faster than conventional ECC codes with similar error-correcting capability. Experimental results showed that the proposed code reduced the number of check symbols and significantly reduced the decoder area and power by using the RRAM cells to perform the addition.

Dhandapani, et al. [12] proposed the implementation of a BCH Encoder in the ZYNQ-7000 AP SOC to guarantee that the sensitive data acquired from capsule endoscopy was transmitted without any errors through a wireless medium. The BCH Encoder and Decoder were designed, synthesized, and simulated using Xilinx Vivado. Upon successful simulation, the code was dumped to the ZYNQ-7000, which featured a single-core ARM Cortex™-A9 processor mated with 28 nm Artix-7-based programmable logic. ZYNQ-7000 was proposed because it was the most flexible & scalable platform for maximum reuse and best TTM, also Industry-leading design tools, C/C++, and Open CL design abstractions with the largest portfolio of SW & HW design tools, Soms, design kits, and reference designs. To ensure the safety and security of sensitive encoded data, Encryption was done using the Triple DES algorithm. Triple DES was the sophisticated version of DES. It was a block cipher technique based on Feistel Structure, and it used symmetric-key block cipher which applied the DES algorithm thrice to each data block. TDES comprise a bundle of three keys. By doing this sensitive data were secured and provided no chances for stealing or modifying the data. The abstract should summarize the contents of the paper in short terms.

Ajmal, Muhammad, et al. [13] presented the first memory-less transition bus encoding technique for low power dissipation, crosstalk avoidance, and error correction simultaneously, as was previously introduced by Chee et al. (Des Codes Cryptor 77(2–4):479–491, 2015). They constructed optimal or asymptotically optimal constant weight codes that eliminated each kind of crosstalk. In this article, we constructed the improved asymptotically optimal (n, 4, 3)-IV code for all even orders by using a combinatorial design approach. Furthermore, we showed that an optimal weighted three code avoiding type-III crosstalk was also an optimal code avoiding the crosstalk of type- {III, I}, for each odd order $n \geq 3$.

Maity, Raj Kumar, et al. [14] addressed the increasing importance of Error Correcting Codes (ECCs) for protecting memories from localized errors, which are primarily caused by radiation-induced soft errors corrupting data stored in a single cell or multiple cells of a memory. Initially, Single Error Correction (SEC) and Single Error Correction-Double Error Detection (SEC-DED) codes were widely used to protect memories against soft errors. However, with the continuous downscaling of technology nodes, the likelihood of multiple errors in memory cells has been increasing. The most common errors in multiple memory cells are single errors and double-adjacent errors. These errors are corrected by employing Single Error Correction-Double Adjacent Error Correction (SEC-DAEC) codes. But designing SEC-DAEC codes poses major challenges, including higher decoding overheads and a higher mis correction rate. In this paper, H-matrices for a new class of SEC-DAEC codes were proposed to reduce the decoding overheads. The proposed codecs were designed and synthesized on an ASIC platform with suitable word lengths for memories. Both the theoretical and synthesis results for the proposed codecs were compared with recently published related works. These comparisons showed that the proposed (24, 16) codec required a maximum of 1.38 times and 1.74 times lower area and delay, respectively, compared to other related codecs of the same word length. Additionally, the proposed 64-bit codec consumed a maximum of 2.75 times less power with respect to existing 64-bit codecs. Thus, the proposed codecs was employed in memories for correcting single and double-adjacent errors with improved area, delay, and power requirements.

Gracia-Morán, L. J. Saiz-Adalid, et al. [15] observed that during these last years, the use of embedded systems has grown exponentially, mainly due to the expansion of the Internet of Things (IoT). Data collected by IoT devices were sent to the cloud to be processed in datacentres. Edge Computing philosophy aimed to change this “passive” behavior of IOT devices. The basic idea was to process data produced by IoT devices closer to where they were created, instead of sending them through long routes. New challenges emerged with the change to the Edge Computing philosophy. One of them was reliability. IoT devices were built with ow-reliable components, reduced weight and volume, and not very high computing and memory capacity for low power consumption.

3. PROPOSED METHODOLOGY

This research represents a Hybrid LDPC-STBC combines LDPC coding for error correction and STBC for diversity gain in wireless communications. LDPC corrects errors efficiently, while STBC enhances signal reliability through diversity. This hybrid approach optimizes spectral efficiency and robustness in modern communication systems.

3.1 LDPC-STBC Encoder

LDPC Encoder operates by partitioning a message into blocks and generating parity bits to enhance error correction capabilities. It begins by creating a parity check matrix with a low density of ones. The message bits are multiplied by this matrix, resulting in parity bits appended to the original message. These parity bits are calculated based on predefined rules dictated by the LDPC code structure. The encoder ensures a balanced distribution of ones in the output codeword, optimizing error correction efficiency. This process provides redundancy for error detection and correction, crucial in reliable communication systems such as wireless networks and storage devices.

STBC is a technique used in multiple-input multiple-output (MIMO) wireless communication systems to improve reliability. In STBC encoding, data symbols are distributed across multiple antennas and transmitted over multiple time intervals. The encoder creates a matrix of symbol vectors, where each vector represents the symbols transmitted simultaneously from different antennas. By carefully designing these matrices, STBC ensures that the transmitted signals have diversity properties, enhancing the system's resilience to fading and interference. At the receiver, these encoded signals are decoded to recover the transmitted symbols, exploiting the diversity introduced by the encoding process to improve reliability.

ISSN NO: 9726-001X
Volume 13 Issue 02 2025

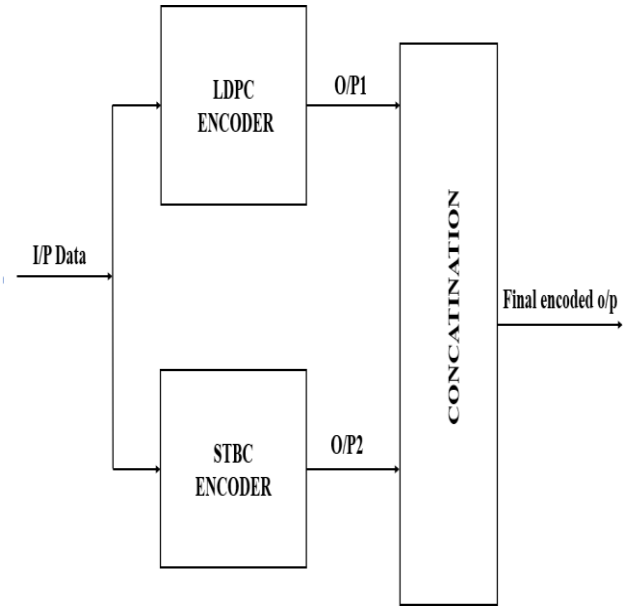

Figure 3.1. LDPC-STBC Encoder

Concatenating LDPC codes with STBC involves encoding data in two stages. Firstly, LDPC encodes the input data using a sparse parity-check matrix to create redundancy for error correction. Secondly, STBC encodes the LDPC-coded symbols across multiple antennas and time slots, exploiting diversity to enhance reliability in wireless communication. The LDPC-coded symbols are spread across the STBC matrix, facilitating robust transmission against fading channels. Concatenating LDPC with STBC thus leverages the error-correction capability of LDPC codes with the diversity gain of STBC, offering improved performance in noisy and fading wireless channels.

3.2 LDPC-STBC Decoder

LDPC-STBC data splitting involves dividing input bits into LDPC codewords. Each codeword undergoes STBC encoding, then the resulting symbols are arranged into blocks for transmission over multiple antennas, facilitating robustness against fading. Decoding at the receiver involves LDPC decoding followed by STBC decoding.

LDPC decoding begins by initializing variable nodes with received symbols. Messages are passed iteratively between variable and check nodes. At each iteration, variable nodes compute check node messages based on received symbols and previous messages. Check nodes then update variable node messages according to parity check equations. This process iterates until either a maximum number of iterations is reached, or all parity check equations are satisfied. Finally, the decoder outputs the estimated codeword based on the variable node messages. This iterative process aims to minimize the discrepancy between received symbols and the estimated codeword, achieving reliable decoding of LDPC codes.

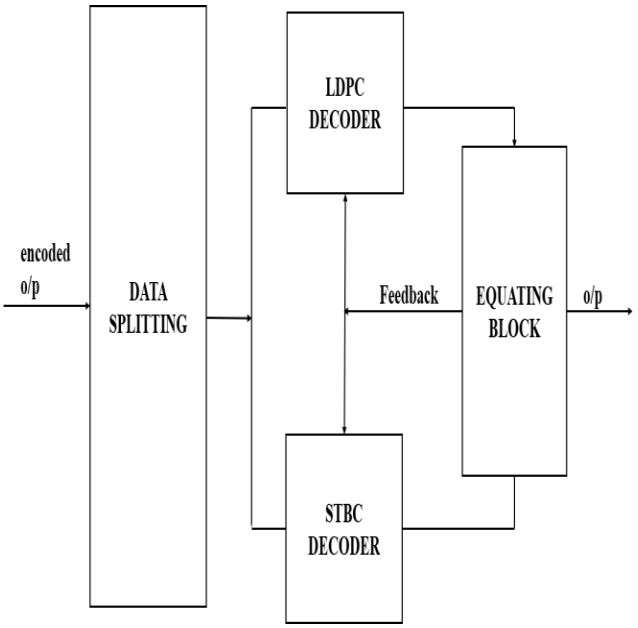

Figure 3.2. LDPC-STBC Decoder

STBC decoding involves recovering transmitted symbols from multiple antennas to improve reliability in wireless communication. The decoder utilizes received signals from different antennas, applies matrix operations such as maximum likelihood or minimum mean square error, and estimates the original transmitted symbols. By exploiting spatial diversity, STBC decoding enhances signal integrity, mitigating fading effects, and enhancing overall communication performance.

In LDPC-STBC decoding, feedback involves iteratively refining soft information about transmitted symbols. The decoder utilizes soft information from received symbols to update the reliability of symbols, incorporating feedback from previously decoded symbols. Equated output combines LDPC decoding with STBC decoding, jointly optimizing symbol detection. It iteratively refines soft information using LDPC decoding techniques while exploiting STBC diversity. This process enhances symbol reliability, mitigating errors introduced by fading channels and noise. Ultimately, equated output produces improved symbol estimates by leveraging both LDPC and STBC decoding advantages in a synergistic manner.

Applications:

- **Secure Communication:** LDPC-STBC coding ensures reliable and secure communication links between military personnel, vehicles, and command centers. This is crucial for transmitting sensitive information, such as troop movements, mission plans, and intelligence data, while mitigating the risk of interception or jamming by adversaries.
- **Battlefield Surveillance:** Satellite communication systems equipped with LDPC-STBC coding enable real-time transmission of surveillance data from unmanned aerial vehicles (UAVs), drones, and reconnaissance satellites. This data includes high-resolution imagery, video feeds, and sensor readings, allowing military commanders to monitor enemy activities, assess battlefield conditions, and make informed decisions in tactical situations.
- **Coordination of Forces:** Hybrid LDPC-STBC coding facilitates seamless coordination and interoperability among different branches of the military, including army, navy, air force, and special operations forces. By providing reliable communication links between dispersed units, command centers, and allied forces, LDPC-STBC-coded satellite systems enhance situational awareness, command

ISSN NO: 9726-001X

Volume 13 Issue 02 2025

JOURNAL OF CURRENT SCIENCE

and control, and joint operations in complex military environments.

- **Navigation and Positioning:** Satellite communication systems with LDPC-STBC coding support precise navigation and positioning capabilities for military vehicles, aircraft, and troops. Global Navigation Satellite Systems (GNSS), such as GPS, Galileo, and GLONASS, utilize satellite communication links to provide accurate positioning, timing, and navigation information in both civilian and military applications, enabling precise targeting, navigation, and mission planning.
- **Emergency Response and Disaster Relief:** In disaster situations or humanitarian crises, military forces often play a critical role in providing emergency response and disaster relief operations. Satellite communication systems with LDPC-STBC coding facilitate communication between military units, relief organizations, and government agencies, enabling coordination of rescue efforts, distribution of aid supplies, and evacuation of affected populations in remote or inaccessible areas.

4. EXPERIMENTAL ANALYSIS

Figure 4.1 shows the existing simulation results for N=32.It can correct only one bit. It cannot correct more than 1 bit. Input applied is 3546.The encoded data is 122331.The number of error bits in the data after the data transmitted is 3. Here, it is not corrected accurately

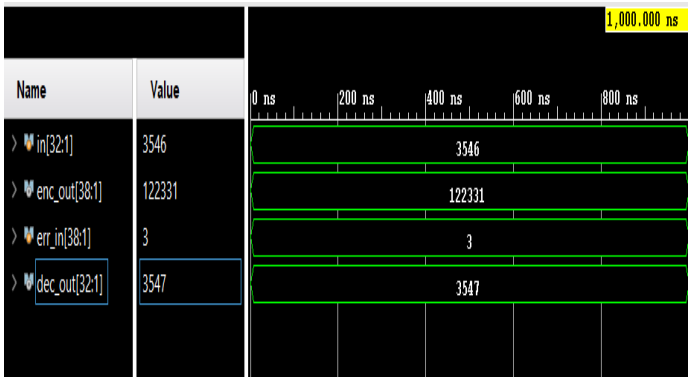

Figure 4.1. Existing Simulation Results for N=32

Figure 4.2 shows the existing area measurements for N=32. Here, 104 number of look up tables (LUTs) are used out of available 134600, which consumes 0.08% of utilization, 140 number of IO are used out of 500, which consumes 28.00% of utilization.

Resource	Estimation	Available	Utilization %
LUT	104	134600	0.08
IO	140	500	28.00

Figure 4.2. Existing Area for N=32

Power

Figure 4.2 Shows Existing power measurements for N=32. Here, the total power is 34.692 W, Static power includes PL Static is 0.301 W, Dynamic power includes signal is 4.427 W, Logic is 2.272 W and I/O is 27.693 W

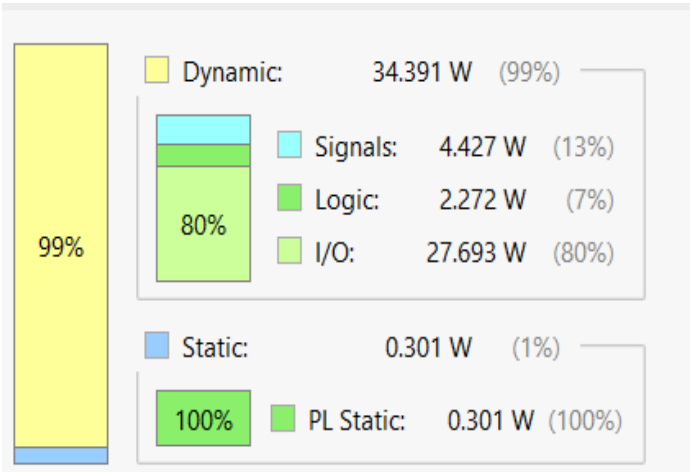

Figure 4.3. Existing Power for N=32

Setup Delay

Figure 4.4 shows Existing Setup delay for N=32. Here, Total Delay is 18.142, maximum Logic Delay is 4.522, maximum Net Delay is 13.620.

Name	Slack	Levels	Routes	High Fanout	From	To	Total Delay	Logic Delay	Net Delay	Requirement	Source Clock	Destination Clock
Path 1	∞	6	5	6	err_in[36]	dec_out[28]	18.142	4.522	13.620	∞	input port clock	
Path 2	∞	6	5	6	err_in[36]	dec_out[29]	18.013	4.524	13.489	∞	input port clock	
Path 3	∞	6	5	6	err_in[36]	dec_out[31]	18.001	4.516	13.485	∞	input port clock	
Path 4	∞	6	5	6	err_in[36]	dec_out[27]	17.731	4.520	13.211	∞	input port clock	
Path 5	∞	6	5	6	err_in[36]	dec_out[30]	17.714	4.516	13.198	∞	input port clock	
Path 6	∞	6	5	32	err_in[36]	dec_out[25]	17.713	4.535	13.177	∞	input port clock	
Path 7	∞	6	5	15	err_in[26]	dec_out[22]	17.662	4.500	13.162	∞	input port clock	
Path 8	∞	6	5	32	err_in[36]	dec_out[24]	17.627	4.539	13.088	∞	input port clock	
Path 9	∞	6	5	32	err_in[36]	dec_out[18]	17.615	4.535	13.080	∞	input port clock	
Path 10	∞	6	5	32	err_in[36]	dec_out[15]	17.585	4.557	13.028	∞	input port clock	

Figure 4.4. Existing Set Up delay for N=32

Hold Delay

Figure 4.5 shows Existing Hold delay for N=32. Here, Total Delay is 3.716, maximum Logic Delay is 1.917, maximum Net Delay is 1.798.

Name	Slack	Levels	Routes	High Fanout	From	To	Total Delay	Logic Delay	Net Delay	Requirement	Source Clock	Destination Clock
Path 11	∞	4	3	4	err_in[19]	dec_out[14]	3.716	1.917	1.798	-∞	input port clock	
Path 12	∞	4	3	3	err_in[12]	dec_out[8]	3.731	1.871	1.860	-∞	input port clock	
Path 13	∞	5	4	32	err_in[10]	dec_out[10]	3.738	1.916	1.822	-∞	input port clock	
Path 14	∞	4	3	2	err_in[10]	dec_out[6]	3.754	1.930	1.824	-∞	input port clock	
Path 15	∞	4	3	32	err_in[4]	dec_out[4]	3.772	1.896	1.876	-∞	input port clock	
Path 16	∞	4	3	32	err_in[4]	dec_out[16]	3.778	1.879	1.899	-∞	input port clock	
Path 17	∞	4	3	32	err_in[4]	dec_out[2]	3.784	1.916	1.867	-∞	input port clock	
Path 18	∞	4	3	3	err_in[17]	dec_out[12]	3.807	1.849	1.958	-∞	input port clock	
Path 19	∞	5	4	11	err_in[17]	dec_out[9]	3.817	1.904	1.914	-∞	input port clock	
Path 20	∞	5	4	11	err_in[17]	dec_out[5]	3.819	1.897	1.922	-∞	input port clock	

ISSN NO: 9726-001X

Volume 13 Issue 02 2025

Figure 4.5. Existing Hold delay for N=32

4.2 Proposed Results

Figure 4.6 shows the existing simulation results for N=32.It can correct up to N bit. The number of redundant bits is 16. Here the input applied is 113.The codeword is 485331304561.The error which occurred after the data is transmitted is 5476.Finally the data is corrected without any errors.

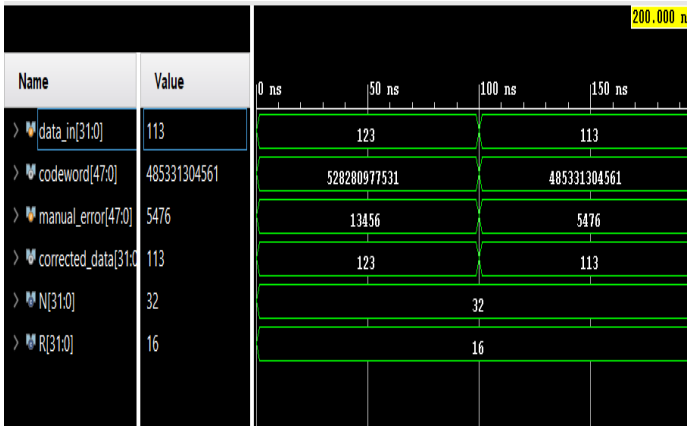

Figure 4.6. Proposed Simulation Results for N=32

Figure 4.7 shows the Proposed area measurements for N=32. Here, 6 number of look up tables (LUTs) are used out of available 134600, which consumes 0.01% of utilization, 112 number of IO are used out of 500, which consumes 22.40% of utilization.

Resource	Estimation	Available	Utilization %
LUT	6	134600	0.01
IO	112	500	22.40

Figure 4.7. Proposed Area for N=32

Power

Figure 4.8 Shows Existing power measurements for N=32. Here, the total power is 17.134 W, Static power includes Device Static is 0.165 W, Dynamic power includes signal is 0.895 W, Logic is 0.047 W and I/O is 16.027 W

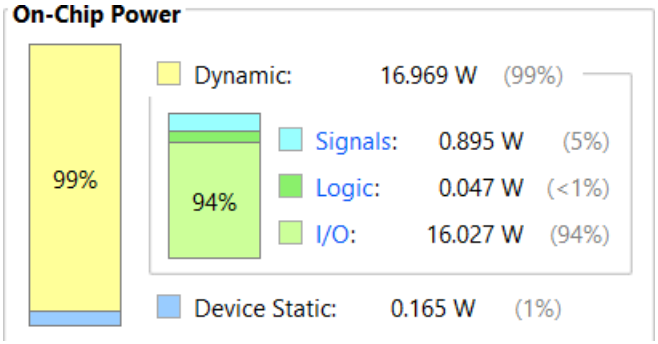

Figure 4.8. Proposed Power for N=32

Setup Delay

Figure 4.9 shows Existing Setup delay for N=32. Here, Total Delay is14.766, maximum Logic Delay is 4.270, maximum Net Delay is 10.496.

Name	Slack ^1	Levels	Routes	High Fanout	From	To	Total Delay	Logic Delay	Net Delay	Requirement	Source Clock	Destination
Path 1	∞	4	3	4	data_in[7]	corrected_data[0]	14.766	4.270	10.496	∞	input port clock	
Path 2	∞	2	1	4	data_in[9]	corrected_data[9]	13.225	3.972	9.253	∞	input port clock	
Path 3	∞	2	1	2	data_in[0]	codeword[32]	13.050	3.902	9.148	∞	input port clock	
Path 4	∞	2	1	4	data_in[14]	corrected_data[14]	12.989	3.972	9.016	∞	input port clock	
Path 5	∞	2	1	4	data_in[10]	corrected_data[10]	12.719	3.974	8.745	∞	input port clock	
Path 6	∞	2	1	4	data_in[8]	corrected_data[8]	12.617	3.960	8.658	∞	input port clock	
Path 7	∞	2	1	4	data_in[5]	corrected_data[5]	12.597	3.973	8.625	∞	input port clock	
Path 8	∞	2	1	4	data_in[1]	corrected_data[1]	12.541	3.988	8.553	∞	input port clock	
Path 9	∞	2	1	4	data_in[4]	corrected_data[4]	12.433	3.954	8.479	∞	input port clock	
Path 10	∞	2	1	2	data_in[23]	codeword[23]	12.373	3.908	8.465	∞	input port clock	

Figure 4.9. Proposed Set Up delay for N=32

Hold Delay

Figure 4.10 shows Existing Setup delay for N=32. Here, Total Delay is 2.471, maximum Logic Delay is 1.573, maximum Net Delay is 0.718.

Name	Slack ^1	Levels	Routes	High Fanout	From	To	Total Delay	Logic Delay	Net Delay	Requirement	Source Clock	Destination Cloc
Path 11	∞	2	1	4	data_in[11]	codeword[43]	2.471	1.753	0.718	-∞	input port clock	
Path 12	∞	2	1	4	data_in[12]	codeword[44]	2.486	1.768	0.718	-∞	input port clock	
Path 13	∞	2	1	4	data_in[13]	codeword[45]	2.489	1.771	0.718	-∞	input port clock	
Path 14	∞	2	1	4	data_in[15]	codeword[47]	2.494	1.787	0.707	-∞	input port clock	
Path 15	∞	2	1	4	data_in[11]	codeword[11]	2.797	1.744	1.053	-∞	input port clock	
Path 16	∞	2	1	4	data_in[12]	codeword[12]	2.814	1.761	1.053	-∞	input port clock	
Path 17	∞	2	1	4	data_in[15]	codeword[15]	2.826	1.783	1.043	-∞	input port clock	
Path 18	∞	2	1	4	data_in[13]	codeword[13]	2.850	1.796	1.053	-∞	input port clock	
Path 19	∞	2	1	4	data_in[14]	codeword[46]	2.937	1.746	1.191	-∞	input port clock	
Path 20	∞	2	1	4	data_in[2]	codeword[34]	3.131	1.778	1.352	-∞	input port clock	

Figure 4.10. Proposed Hold delay for N=32

4.3 Performance Comparisons

The proposed method significantly outperforms the existing method across various metrics. In terms of LUT (Look-Up Table) usage, it reduces it by 94.2%, indicating greater efficiency. IO (Input/Output) increases by 37%, suggesting enhanced connectivity. Total power consumption drops by 48%, with static power decreasing by 8.6% and dynamic power by 48.5%, indicating substantial energy savings. However, there's a trade-off in terms of total delay, which increases by 54.7%, primarily due to logic delay rising by 72.84% and net delay by 63.03%. This trade-off underscores a shift towards power optimization at the expense of increased delay.

Table 4.1. Performance comparison of existing and proposed methods for N=16

Metric	Existing Method	Proposed Method	Comparison
LUT	52	3	94.2%
IO	70	96	37%
TOTAL POWER	17.346W	8.985 W	48%
STATIC POWER	0.150W	0.137 W	8.6%
DYNAMIC POWER	17.195W	8.848 W	48.5%
TOTAL DELAY	9.701	15.011	54.7%
LOGIC DELAY	2.261	3.908	72.84%
NET DELAY	6.81	11.103	63.03%

The proposed method demonstrates significant improvements over the existing method across various metrics. It achieves a 94% reduction in LUT (Look-Up Table) usage and a 20% decrease in IO (Input/Output) requirements, indicating enhanced efficiency and resource utilization. Total power consumption is reduced by 50.6%, with static power dropping by 45.18% and dynamic power by 50.65%, showcasing substantial energy savings. Additionally, there's an 18.6% decrease in total delay, with logic delay improving by 5.57% and net delay by 22.93%. These results highlight the effectiveness of the proposed method in optimizing resource usage, reducing power consumption, and improving overall performance compared to the existing method.

Table 4.2. Performance comparison of existing and proposed methods for N=32

Metric	Existing Method	Proposed Method	Comparison
LUT	104	6	94%
IO	140	112	20%
TOTAL POWER	34.692 W	17.134 W	50.6%
STATIC POWER	0.301 W	0.165 W	45.18%
DYNAMIC POWER	34.391 W	16.969 W	50.65%
TOTAL DELAY	18.142	14.766	18.6%
LOGIC DELAY	4.522	4.270	5.57%
NET DELAY	13.620	10.496	22.93%

The proposed method yields substantial improvements over the existing method across key metrics. It achieves a reduction of 93.75% in LUT (Look-Up Table) usage and a 13.33% decrease in IO (Input/Output) requirements, indicating enhanced efficiency and resource utilization. Total power consumption is notably reduced by 62.97%, with static power dropping by 64.28% and dynamic power by 62.96%, demonstrating significant energy savings. Moreover, there's a considerable 62.48% decrease in total delay, with logic delay improving by 56.39% and net delay by 64.5%. These results highlight the effectiveness of the proposed method in optimizing resource utilization, reducing power consumption, and improving overall performance compared to the existing method.

Table 4.3. Performance comparison of existing and proposed methods for N=64

Metric	Existing Method	Proposed Method	Comparison
LUT	208	13	93.75%
IO	240	208	13.33%
TOTAL POWER	69.384W	25.69 W	62.97%
STATIC POWER	0.602W	0.215 W	64.28%
DYNAMIC POWER	68.782W	25.475 W	62.96%
TOTAL DELAY	36.284	13.611	62.48%
LOGIC DELAY	9.044	3.944	56.39%
NET DELAY	27.24	9.667	64.5%

5. CONCLUSION

The hybridization of LDPC and STBC codes presents a promising avenue for enhancing satellite communication systems, particularly in terms of improving error correction capabilities and mitigating the effects of fading channels. Through the comprehensive review and analysis conducted in this paper, several important insights have emerged.

Furthermore, the integration of LDPC-STBC coding within satellite communication systems offers significant practical advantages. By enhancing the robustness of transmissions against channel impairments, such as multipath fading and interference, the proposed scheme enables reliable communication over long-distance satellite links. This is particularly important for applications requiring high data rates and stringent error rate requirements, such as multimedia streaming, remote sensing, and telemedicine.

Moreover, the scalability and flexibility of LDPC-STBC coding make it well-suited for future satellite communication standards and deployments. As communication technologies continue to evolve, there is a growing need for efficient and adaptable error control techniques that can accommodate changing channel conditions and network requirements. The inherent parallelism and low-complexity decoding algorithms of LDPC codes, combined with the diversity and coding gain of STBC schemes, make the hybrid approach a compelling solution for next-generation satellite systems.

REFERENCES

[1] Singh, S. Pratap, Ruchi Rai, Shashank Awasthi, Dinesh Kumar Singh, and M. Lakshmanan. "VLSI Implementation of Error Correction Codes for Molecular Communication." Wireless Personal Communications 130, no. 4 (2023): 2697-2713.

[2] Silva, Felipe, Alan Pinheiro, Jarbas AN Silveira, and César Marcon. "A Triple Burst Error Correction Based on Region Selection Code." IEEE Transactions on Very Large-Scale Integration (VLSI) Systems (2023).

[3] Chu, Shao-I, Syuan-A Ke, Sheng-Jung Liu, and Yan-Wei Lin. "An Efficient Hard-Detection GRAND Decoder for Systematic Linear Block Codes." IEEE Transactions on Very Large-Scale Integration (VLSI) Systems (2023).

[4] Kuo, Yao-Ming, Mark F. Flanagan, Francisco Garcia-Herrero, Oscar Ruano, and Juan Antonio Maestro. "Integration of a Real-Time CCSDS 410.0-B-32 Error-Correction Decoder on FPGA-Based RISC-V SoCs Using RISC-V Vector Extension." IEEE Transactions on Aerospace and Electronic Systems (2023).

[5] Grurl, Thomas, Christoph Pichler, Jurgen Fus, and Robert Wille. "Automatic Implementation and Evaluation of Error-Correcting Codes for Quantum Computing: An Open-Source Framework for Quantum Error Correction." In 2023 36th International Conference on VLSI Design and 2023 22nd International Conference on Embedded Systems (VLSID), pp. 301-306. IEEE, 2023.

[6] Wu, Yujun, Bin Wu, and Xiaoping Zhou. "High-Performance QC-LDPC Code Co-Processing Approach and VLSI Architecture for Wi-Fi 6." Electronics 12, no. 5 (2023): 1210.

[7] Pokhrel, Nabin Kumar, Pankaj Kumar Das, and Aleksandar Radonjic. "Integer codes capable of correcting burst asymmetric errors." Journal of Applied Mathematics and Computing 69, no. 1 (2023): 771-784.

[8] Saini, Madan Lal, Vivek Kumar Sharma, and Ashok Kumar. "An Efficient Single and Double Error Correcting Block Codes with Low Redundancy for Digital Communications." In 2023 International Conference on Advancement in Computation & Computer Technologies (InCACCT), pp. 827-831. IEEE, 2023.

[9] Ponmalar, VJ Beulah Sherin, and M. Ruth Jenila. "Low Density Parity Check used in VLSI Circuits Providing Optimal Circuits."

[10] Boncalo, Oana, and Alexandru Amaricai. "Gradient Descent Iterative Correction Unit for Fixed Point Parity Based Codes." In 2023 IEEE International Symposium on Defect and Fault Tolerance in VLSI and Nanotechnology Systems (DFT), pp. 1-4. IEEE, 2023.

[11] Dutta, Shruti, Sai Charan Rachamadugu Chinni, Abhishek Das, and Nur A. Touba. "Highly Efficient Layered Syndrome-based Double Error Correction Utilizing Current Summing in RRAM Cells to Simplify Decoder." In 2023 IEEE International Symposium on Defect and Fault Tolerance in VLSI and Nanotechnology Systems (DFT), pp. 1-4. IEEE, 2023.

[12] Dhandapani, N., M. Z. Mohamed Ashik, Kalthi Reddy Bhargav, N. Achyuth, and Deepa Jose. "VLSI Implementation of BCH Encoder with Triple DES Encryption for Baseband Transceiver." In Mobile Radio Communications and 5G Networks: Proceedings of Third MRCN 2022, pp. 329-341. Singapore: Springer Nature Singapore, 2023.

[13] Ajmal, Muhammad, Masood Ur Rehman, and B. G. Rodrigues. "Improved asymptotically optimal error correcting codes for avoidance crosstalk type-IV on-chip data buses." Computational and Applied Mathematics 42, no. 4 (2023): 150.

[14] Maity, Raj Kumar, Jagannath Samanta, and Jaydeb Bhaumik. "An Improved Single and Double-Adjacent Error Correcting Codec with Lower Decoding Overheads." Journal of Signal Processing Systems (2023): 1-13.

[15] Gracia-Morán, J., L. J. Saiz-Adalid, J. C. Baraza-Calvo, D. Gil-Tomás, and P. J. Gil-Vicente. "Comparison of the overheads provoked by the inclusion of different Error Correction Codes in Embedded Systems."

[16] Hou, Yuqi, Xi Liu, Lei Tang, Sheng Zhong, and Hangzai Luo. "Low Redundancy Two-Dimensional Matrix-Based HVDB Code for Double Error Correction." In 2023 6th International Conference on Communication Engineering and Technology (ICCET), pp. 11-16. IEEE, 2023.

[17] Cohen, Alejandro, Rafael GL D'Oliveira, Ken R. Duffy, Jongchan Woo, and Muriel Médard. "AES as Error Correction: Cryptosystems for Reliable Communication." IEEE Communications Letters (2023).

[18] Mondal, Somnath, Sachin Patkar, and T. K. Pal. "Hardware implementation of Ring-LWE lattice cryptography with BCH and Gray coding based error correction." In 2023 36th International Conference on VLSI Design and 2023 22nd International Conference on Embedded Systems (VLSID), pp. 1-6. IEEE, 2023.

[19] Alnajjar, Khawla A., Abdallah YI Abushawish, and Sam Ansari. "Hardware-Based Error Correction Systems for Hamming Codes: a Review of the Literature." In 2023 International Conference on Smart Applications, Communications and Networking (SmartNets), pp. 1-8. IEEE, 2023.

[20] Choe, Jeongwon, and Youngjoo Lee. "A 2.35 Gb/s/mm 2 (7440, 6696) NB-LDPC Decoder over GF (32) using Memory-Reduced Column-Wise Trellis Min-Max Algorithm in 28nm CMOS Technology." In 2023 IEEE Symposium on VLSI Technology and Circuits (VLSI Technology and Circuits), pp. 1-2. IEEE, 2023.

[21] Shaligram, and Varaganti Priyanka. "An Instinctive Error Encoder and Decoder Using XOR for Space Applications." International Journal of Research in Engineering, Science and Management 6, no. 3 (2023): 105-107.

[22] Tang, Yok Jye, and Xinmiao Zhang. "Generalized Integrated Interleaved Codes for High-Density DRAMs." IEEE Transactions on Circuits and Systems II: Express Briefs (2023).

[23] Kim, Moon-Seok, Sungho Kim, Sang-Kyung Yoo, Bong-Soo Lee, Ji-Man Yu, Il-Woong Tcho, and Yang-Kyu Choi. "Error reduction of SRAM-based physically unclonable function for chip authentication." International Journal of Information Security (2023)