A Comprehensive Review On Hamming Code Architectures And Their Implementation Across Modern Digital Technologies

[1]Dr S. V. Viraktamath, [2c]Shriraksha Channalli, [3]Praveenakumar Budihal, [4]Preethi S. Awate

[1234]Department. of Electronics and Communication Engineering,
SDM College of Engineering and Technology Dharwad, Karnataka, India

Abstract—Reliable data communication and storage have become essential in modern digital systems, where increasing circuit density and operating speeds make error correction a critical design requirement. Among various error control techniques, hamming code remains one of the most widely adopted solutions due to its simplicity, low redundancy, and effective single-error correction and double-error detection capability. This review presents a comprehensive examination of Hamming code architectures across FPGA, HDL, CMOS, GDI, CNTFET, QCA, and emerging AI-assisted platforms. The analysis covers design methodologies, implementation techniques, and system-level integration strategies derived from a broad collection of existing research works. The study further explores practical applications of Hamming code in communication systems, memory, and storage devices, IoT and automotive systems, biomedical electronics, and intelligent embedded platforms Recent advancements in low power logic families, nanoscale technologies, and machine learning based decoders demonstrate the continuous evolution and adaptability of Hamming-based error control schemes. The Obtained results establish Hamming code as a foundational and future relevant error correction mechanism capable of supporting high- performance, energy-efficient, and fault-tolerant system design across a wide spectrum of emerging digital technologies.

Index Terms—Hamming Code, Error Correction Coding, FPGA Implementation, Low-Power VLSI, Fault-Tolerant Systems

I. Introduction

Error detection and correction are critical components in digital communication and computing systems, ensuring reliable data transfer and storage in the presence of noise, interference, or hardware faults. As modern electronic devices demand high performance, low power consumption, and miniaturized architectures, achieving fault tolerance has become a key design objective. Among various error control techniques, the Hamming code remains one of the most efficient and widely adopted methods due to its simplicity, minimal redundancy, and strong capability for single-bit error correction and double- bit error detection.

Originally introduced by Richard W. Hamming, this code forms the foundation of most linear block codes used in data communication and memory systems. Its mathematical elegance and hardware-friendly structure have made it ideal for implementation across multiple platforms, ranging from FPGA and ASIC architectures to emerging nano and quantum computing technologies. With the continuous evolution of Very Large-Scale Integration (VLSI) and nanotechnology, researchers have focused on optimizing Hamming code architectures to meet modern design challenges such as high speed, low area utilization, and energy efficiency.

In recent years, Hardware Description Language (HDL) based designs using VHDL and Verilog have enabled rapid prototyping and reconfigurable hardware implementations, making the Hamming code highly suitable for real-time error control. Parallel advancements in Gate Diffusion Input (GDI), Complementary Metal-Oxide Semiconductor (CMOS), and Carbon Nanotube Field Effect Transistor (CNTFET) technologies have enhanced the energy efficiency and scalability of these circuits. Additionally, emerging paradigms such as Quantum-dot Cellular Automata (QCA) and reversible logic have extended the scope of Hamming code applications toward nanoscale and low-power computing.

The growing adoption of intelligent systems and adaptive error correction using machine learning further demonstrates the relevance of Hamming code principles in modern electronics. Its applications now extend beyond traditional communication systems to IoT devices, automotive electronics, industrial automation, memory systems, and biomedical instrumentation, where reliable and energy-efficient data integrity is essential.

II. HAMMING CODE FUNDAMENTALS

Hamming codes belong to the class of linear block codes that add redundant parity bits to data bits to detect and correct transmission errors. A Hamming code with n total bits and k data bits follows the relation (1):

$$2^r \geq n + 1 \quad \ldots\ldots\ldots (1)$$

where r is the number of parity bits and n is the total code length. These parity bits occupy positions that are powers of two (1, 2, 4, 8, . . .) within the encoded frame. The encoding process involves generating parity bits using exclusive-OR (XOR) operations over specific data bit positions. The syndrome vector obtained during decoding identifies the position of an erroneous bit, enabling single-bit correction and double-bit detection (SECDED).

Hamming codes are widely preferred for:

- High-speed memory and cache systems.
- Wireless and wired digital communication.
- VLSI fault-tolerant architectures
- Low-power embedded control systems

Enhanced versions, including extended Hamming codes and hybrid parity–Hamming combinations, offer greater flexibility for multi-bit fault tolerance while maintaining manageable hardware complexity.

III. DESIGN METHODOLOGIES AND IMPLEMENTATION TECHNIQUES OF HAMMING CODE

Error correction and detection play a vital role in digital communication and memory systems. To ensure reliable data transfer and storage, several researchers have implemented Hamming codes and their derivatives using diverse hardware and software tools. This chapter discusses the various implementation methodologies adopted across FPGA, HDL, CMOS, GDI, CNTFET, QCA, and machine learning platforms. Each implementation has been designed to meet specific requirements such as power efficiency, area optimization, speed improvement, and reliability enhancement.

1) FPGA and HDL-Based Implementations:

Field Programmable Gate Arrays (FPGAs) are widely preferred for rapid prototyping and real-time system implementation of error control codes. The Hamming code algorithm, being simple and linear, is effectively realized using FPGA-based hardware description languages such as VHDL and Verilog HDL. The design [1] demonstrated the realization of the Hamming algorithm using Verilog HDL on a Xilinx Spartan-3E FPGA board. The system performed single-bit correction and double-bit detection with low propagation delay and minimum utilization of logic resources. This implementation proved that FPGA platforms can be used for high-speed and low-cost real- time data communication systems.

The architecture presented [2] utilized FPGA architecture to implement a VLSI version of the Hamming code algorithm. The design used an (11,7,1) Hamming structure that enabled efficient encoding and decoding. The FPGA configuration improved the overall re- liability and reduced delay compared to conventional microcontroller-based systems. A similar approach was discussed [3][4], where the authors implemented the algorithm using VHDL and (Digital Schematic Editor and Simulator) DSCH tools to design a parity-based Hamming encoder and decoder. The design incorporated a single error detection and correction mechanism with minimum redundancy.

A detailed encoding and decoding model were developed using VHDL, which introduced a matrix-based parity generation scheme. The parity bits were generated dynamically depending on the message length, and the system achieved faster correction capability. Similarly,[5] and [6] introduced a (7,4) Hamming encoder and decoder using VHDL for educational and embedded system applications. The 64-bit extended Hamming code presented [7] improved the bit correction capacity while minimizing propagation delay. These FPGA-based systems are compatible with Xilinx simulation tools such as Vivado and ISE Design Suite, ensuring that the error correction process can be verified before hardware implementation.

Another design [8] demonstrated Verilog-based realization of the (7,4) Hamming code using linear block encoding with a generator matrix. This model ensured that each data word was encoded into a code word with adequate redundancy, achieving reliable decoding without retransmission. The design was proven on an FPGA platform with a high success rate and reduced latency. To enhance efficiency, integrated clock gating within the FPGA circuit to minimize dynamic power consumption, achieving a balance between reliability and energy efficiency.

To support the architectural overview of error control mechanisms, the general structure of the Hamming code system is illustrated in Fig. 1. The diagram summarizes the flow of data through the encoder and decoder blocks, highlighting the parity generation, syndrome calculation, and correction stages commonly adopted in FPGA and HDL-based implementations.

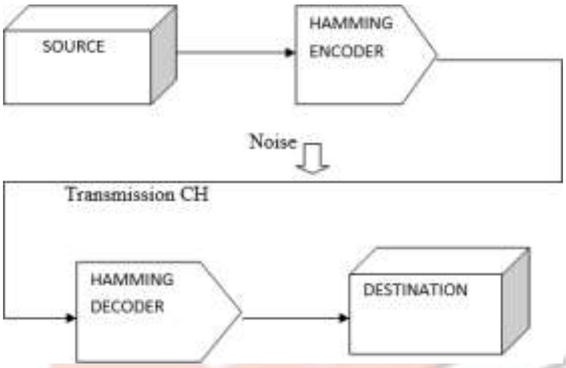
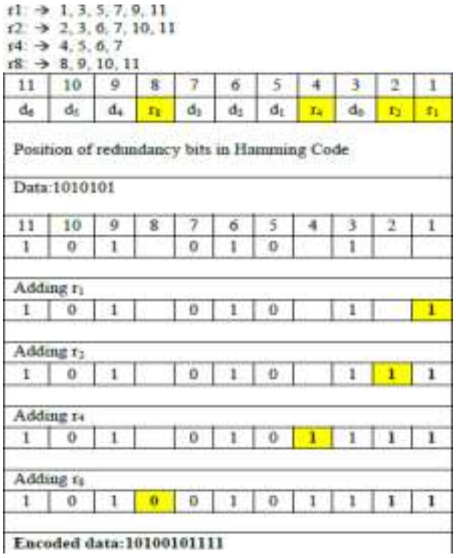


Figure 1. Block Diagram of Hamming Code Encoder–Decoder [2]

The encoder module generates parity bits based on the positional parity-check matrix and appends them to the original data bits to form the codeword. During transmission or storage, any single-bit error is identified by computing the syndrome vector inside the decoder. The syndrome value indicates the exact bit position requiring correction, enabling single-error correction and double-error detection capabilities. This architectural structure is consistently observed across Verilog, VHDL, and FPGA based realizations reported in the literature, demonstrating its reliability and simplicity for hardware implementation.

To illustrate how parity bits are generated in the (11,7) Hamming code, the redundancy bit assignment method is presented in Fig. 2. Each redundant bit is responsible for monitoring a predefined set of bit positions, enabling unique error identification during decoding.



r1: → 1, 3, 5, 7, 9, 11
r2: → 2, 3, 6, 7, 10, 11
r4: → 4, 5, 6, 7
r8: → 8, 9, 10, 11

11	10	9	8	7	6	5	4	3	2	1
d_6	d_5	d_4	r_8	d_3	d_2	d_1	r_4	d_0	r_2	r_1

Position of redundancy bits in Hamming Code

Data:1010101

11	10	9	8	7	6	5	4	3	2	1
1	0	1		0	1	0		1		

Adding r_1

1	0	1		0	1	0		1		1

Adding r_2

1	0	1		0	1	0		1	1	1

Adding r_4

1	0	1		0	1	0	1	1	1	1

Adding r_8

1	0	1	0	0	1	0	1	1	1	1

Encoded data:10100101111

Figure 2. Redundancy Bit Calculation for (11,7) Hamming Code During Encoding [1]

This structured parity-bit distribution is fundamental for achieving single-error correction capability. The placement of r1, r2, r4, and r8 ensures that each data bit participates in a unique combination of parity-check equations, thereby supporting efficient hardware implementation in FPGA and HDL-based systems.

2) CMOS, GDI, and CNTFET Hardware Designs:

As technology scales down, CMOS implementations of Hamming code have been replaced by more efficient and low-power logic families such as G (GDI) and Carbon Nanotube Field Effect Transistor (CNTFET). The GDI technique proposed [9] showed a significant reduction in transistor count and static power consumption compared to conventional CMOS logic. The encoder and decoder circuits designed using GDI logic achieved reduced switching delay and improved signal integrity. The work confirmed that power dissipation could be minimized without compromising error correction accuracy.

Traditional CMOS-based designs, such as [10] and [11], focused on enhancing timing performance by using transmission gates and domino CMOS structures. These approaches allowed designers to reduce propagation delay while achieving higher frequency operation. The proposed systems successfully implemented SECDED (Single Error Correction and Double Error Detection) logic, ensuring fault tolerance in high-speed circuits. At the nanoscale level, [12] introduced CNTFET-based Hamming encoder and decoder structures. CNTFET devices are advantageous due to their excellent current carrying capability and low leakage characteristics. The CNTFET implementation provided ultra-low power operation, making it suitable for modern VLSI circuits where leakage control is a challenge.

The hybrid GDI–CNTFET design achieved high reliability, lower supply voltage operation, and minimal layout area, which is essential for portable and battery-powered IoT devices. These hardware implementations collectively indicate a clear transition from traditional CMOS logic toward advanced low-power technologies like GDI and CNTFET. Such evolutions are critical to sustaining the demand for high-speed and energy-efficient VLSI architectures in future systems.

3) QCA and Reversible Logic Implementations:

QCA technology represents a futuristic approach for implementing error correction circuits at the nanoscale. QCA operates by manipulating electron positions within quantum dots to represent binary data, thereby eliminating conventional current flow. The work [13] designed a compact and energy-efficient Hamming encoder using reversible logic gates such as Feynman and Fredkin gates. This QCA-based implementation achieved over 80 percent area reduction and 66 percent fewer cells than equivalent CMOS circuits.

Reversible logic offers an additional advantage by minimizing energy dissipation due to its one-to-one mapping between input and output states. The integration of reversible logic within the QCA framework [14] optimized parity-bit generation and minimized computation overhead. This model provided efficient single-bit error correction and was successfully synthesized for nanoscale implementation. Such innovations pave the way for highly dense, energy-efficient, and eco-friendly computing platforms.

4) *Machine Learning and Advanced ECC Techniques*:

The emergence of artificial intelligence and machine learning has opened new possibilities for adaptive and intelligent error correction mechanisms. The study [15] used neural networks to replace traditional decoding algorithms. Machine-learning-based decoders achieved improved accuracy in identifying error patterns without relying on conventional iterative logic. The system was trained using multiple noisy input–output pairs, enabling the decoder to predict and correct errors dynamically. This approach was particularly effective for complex codes where analytical decoding becomes computationally expensive.

Parallel to these developments,[16] explored Low- Density Parity-Check (LDPC) encoders that outperform conventional Hamming codes in long-distance communication. These advanced ECC schemes employ large, sparse parity-check matrices that reduce redundancy while enhancing correction capability. Such adaptive and learning-based error correction mechanisms are expected to define the next generation of self-correcting communication systems.

IV. APPLICATION DOMAINS AND TECHNOLOGICAL INTEGRATION OF HAMMING CODE

Error correction codes have extensive applications across communication systems, memory devices, and intelligent embedded hardware. The Hamming code remains an integral part of these applications due to its efficiency and simplicity.

1) Communication and Telemetry Systems:

Hamming code has been widely adopted in digital communication systems to ensure reliable data transmission over noisy channels. The designs [17], [18] implemented FPGA- based architectures for UART and CAN bus communication systems. These systems demonstrated robust single- bit correction and double-bit detection, ensuring reliable communication even in high-noise environments. By integrating Hamming and Manchester coding, the system achieved efficient synchronization and minimized transmission delay.

Telemetry and satellite communication systems require high levels of redundancy control. The concatenated ECC system proposed in combined Hamming, Fire, and Reed–Solomon codes to handle both random and burst errors efficiently. The simulation results confirmed a re- markable improvement in error correction under AWGN channel conditions. The application of residue number system-based encoding in [19], [20] further demonstrated that combining arithmetic and ECC principles could improve reliability and error localization.

2) Memory and Storage Applications:

Error correction in memory systems is essential to prevent data corruption due to transient faults and radiation effects. [21], [22] Hybrid parity and Hamming architectures were used in SRAM and DRAM arrays to minimize error rate and enhance system stability. The 64-bit and 128-bit ECC frameworks presented in these studies provided fault- tolerant data retention with negligible delay overhead. The systems used SECDED capability to maintain the accuracy of stored data even in radiation-prone environments.

Row-based Hamming architectures [23] achieved efficient burst-error correction with reduced parity bits. These designs were highly suitable for high-capacity memory modules that demanded minimal redundancy. Similarly, [24] implemented encode–compare and de- code–compare cache structures that allowed error detection without introducing significant timing delay. Such implementations have proven vital in data centers and aerospace memory systems, where reliability is paramount.

3) IoT, Automotive, and Industrial Control Systems:

In modern IoT and industrial systems, data integrity is crucial due to the distributed nature of devices. The FPGA- based fault-tolerant systems described [25], [26], and [27] provide continuous error detection and correction capabilities. Automotive systems benefit significantly from ECC integration because of their safety-critical nature. The implementation in embedded Hamming logic into real-time automotive control hardware, ensuring data integrity in harsh operational environments.

Clock-gating techniques discussed [28] have been used to reduce power consumption in continuously operating industrial systems. These systems demonstrate that energy-efficient ECC modules can be seamlessly integrated into industrial automation, robotics, and intelligent sensor networks.

4) Biomedical and Smart Embedded Applications:

The integration of Hamming and parity-based ECCs in biomedical applications is an emerging research direction. The molecular communication system proposed [29] incorporated Hamming encoding at the biological interface, enabling reliable molecular-level data transfer. This system achieved high fault tolerance, ensuring error-free molecular signaling.

Further, [30] developed IoT-enabled biomedical monitoring systems using FPGA-based LDPC and Hamming codes for real-time data transmission. These systems achieved significant power savings while ensuring reliable physiological data transfer. The architecture [31] applied adjacent error detection and correction (AEDAC) for radiation-sensitive medical electronics, providing additional fault coverage. Such biomedical applications confirm that ECC techniques are not only valuable in traditional electronics but also critical in next-generation healthcare and nano-bio systems.

V. CHALLENGES AND FUTURE SCOPE.

Despite the maturity of Hamming code theory, modern design constraints introduce several open challenges:

- Scalability in Nano-Devices: At deep sub-micron levels, conventional CMOS faces variability and leakage issues. Future work must focus on adaptive ECCs compatible with CNTFET and QCA technologies [9], [13].
- Power-Delay Trade-off: Portable and IoT systems demand low-energy operation without compromising speed. Logic-level optimizations like GDI and domino CMOS can balance efficiency and accuracy [10], [11].
- Machine-Learning-Based Decoding: Integrating neural and transformer models for adaptive decoding remains a promising direction [15]. However, model complexity and on-chip inference cost need minimization for real- time feasibility.
- Multi-Level Error Protection: Combining Hamming, Reed–Solomon, and LDPC frameworks [16], [18] could yield hierarchical ECC systems for high-reliability mem- ory and aerospace communication.
- Standardization and Reconfigurability: Future VLSI and FPGA platforms must support reconfigurable ECC blocks to adapt dynamically to varying error rates and environ- mental conditions.

Research should emphasize co-optimization of hardware architecture, coding theory, and AI-driven fault prediction, ensuring robust, power-efficient, and scalable ECC implementations.

VI. CONCLUSION

The evolution of Hamming code continues to exemplify the harmony between theoretical innovation and practical realization in modern digital system design. As one of the most enduring and efficient error correction techniques, it has demonstrated remarkable adaptability across generations of technology—from FPGA, VHDL, and Verilog-based implementations to energy-efficient CMOS, GDI, and CNTFET architectures. These developments have not only enhanced data reliability but also optimized power consumption, area utilization, and processing speed. The advancement of QCA, reversible logic, and AI-assisted error correction frameworks further extends the applicability of Hamming code into the realms of nanoscale and intelligent computing. Collectively, the diverse methodologies and applications reviewed reaffirm Hamming code as a foundational and forward-looking solution for achieving reliability, fault tolerance, and efficiency in next-generation communication and computing systems.

REFERENCES

[1] R. Hosamani and A. S. Karne, “Design and implementation of Hamming code on FPGA using Verilog,” Int. J. Eng. Adv. Technol., vol. 4, no. 2, pp. 180–181, 2014.

[2] V. Y. M. Vijayakumara, B. K. Byregowda, S. P. Kumar, S. Raghav, M. Nataraja, and S. N. Sheshappa, “A VLSI implementation of Hamming code algorithm using FPGA architecture”, SSRG Int. J. VLSI Signal Process., vol. 7, no. 2, 2020.

[3] D. Mokara, S. Naidu, and A. K. Gupta, “Design and implementation of Hamming code using VHDL and DSCH,” Int. J. Latest Eng. Res. Appl., vol. 2, no. 11, pp. 33–40, 2017.

[4] J.-J. Shan and J. Zhou, “Design of encoding and decoding of Hamming code based on VHDL,” in Proc. IEEE Int. Conf. Comput. Sci. Manage. Technol. (ICCSMT), 2020, doi: 10.1109/ICCSMT51754.2020.00056.

[5] U. S. Sani and I. Shanono, “Design of (7,4) Hamming encoder and decoder using VHDL,” in Proc. Bayero Univ. Kano Conf., 2015.

[6] A. H. Saleh, “Design of Hamming code for 64-bit single error detection and correction using VHDL,” Diyala J. Eng. Sci., vol. 8, no. 3, pp. 22–37, 2015.

[7] A. H. Saleh, “Design of Hamming encoder and decoder circuits for (64,7) and (128,8) codes using VHDL,” J. Sci. Eng. Res., vol. 2, no. 1, 2015.

[8] M. A. Muneeb and N. S., “Verilog implementation of Hamming code for error control coding,” Int. J. Res. Eng. Sci., vol. 10, no. 1, pp. 69–73, 2022.

[9] S. V. Mythry et al., “Low-power implementation of Hamming-code encoder and decoder using GDI logic,” in Proc. IEEE Int. Conf. Comput., Commun. and Control (IC4), 2024.

[10] N. Shep and P. H. Bhagat, “Implementation of Hamming code using VLSI,” Int. J. Eng. Trends Technol., vol. 4, no. 2, pp. 186–188, 2013.

[11] S. G. P., S. S. Das, and V. J., “Implementation of [7,4] Hamming code,” NIT Karnataka, Tech. Rep., 2015.

[12] G. Subhasri and N. Radha, “VLSI design of parity check code with Hamming code for error detection and correction,” in Proc. IEEE ICICCS, 2019.

[13] P. B. Siddaiah, M. Puttaswamy, and N. Kamat, “Compact and energy- efficient QCA-based Hamming encoder for error detection and correc- tion,” Theoretical Appl. Electr. Eng., vol. 21, no. 2, 2023.

[14] D. Y. V., K. M., and M. S., “A comprehensive review of low- density parity-check encoder techniques,” Inge´nierie des Syste`mes d’Information, vol. 27, no. 1, 2022.

[15] Y. Yuan, “Improving the decoding of error correction codes via machine learning,” M.S. thesis, Eindhoven Univ. Technol., 2024.

[16] K. Kalaichevi, “Design of Hamming code encoding and decoding based on clock gating technique,” Kronika Journal, vol. 25, no. 6, pp. 585–590, 2025.

[17] R. Hosamani et al., “FPGA-based UART system integrating Hamming code,” Int. J. Eng. Res. Technol., 2019.

[18] S. P. Singh et al., “VLSI implementation of Hamming code for molecular communication,” in Lecture Notes in Electrical Engineering, Springer, 2021.

[19] M. A. Muneeb et al., “Modified SPA-based Hamming decoder for IoT,” Int. J. Eng. Sci., 2022.

[20] S. V. Mythry et al., “Low-power RNS-integrated ECC for embedded systems,” in Proc. IEEE IC4, 2024.

[21] G. Subhasri and N. Radha, “Hybrid parity–Hamming design for memory reliability,” in Proc. IEEE ICICCS, 2019.

[22] N. Shep et al., “Memory-optimized VLSI ECC,” Int. J. Eng. Trends Technol., 2013.

[23] Leena, S. Gandhi, and J. Khurana, “Implementing (7,4) Hamming code using CPLD,” Int. J. Eng. Res. Technol., 2013.

[24] A. H. Saleh, “Error detection in cache architectures using ECC,” Diyala J. Eng. Sci., 2015.

[25] V. Y. M. Vijayakumara et al., “FPGA architecture for automotive error control,” SSRG Int. J. VLSI Signal Process., 2020.

[26] U. S. Sani et al., “VHDL-based fault-tolerant IoT system,” in Proc. BUK Conf., 2015.

[27] S. G. P. et al., “Embedded Hamming code for automotive control,” NIT Karnataka, 2015.

[28] K. Kalaichevi, “Clock-gated Hamming encoder for energy-efficient designs,” Kronika Journal, 2025.

[29] S. P. Singh et al., “Molecular communication ECC design,” in Lecture Notes in Electrical Engineering, Springer, 2021.

[30] D. Y. V. et al., “Biomedical LDPC-based ECC systems,” Inge´nierie des Syste`mes d’Information, 2022.

[31] P. B. Siddaiah et al., “Adjacent error detection and correction for biomedical systems,” Adv. Electr. Electron. Eng., 2023.