*International Journal for Modern Trends in Science and Technology*
*Volume 12, Special Issue 01*, pages 557-566.
ISSN: 2455-3778 online
Available online at: http://ijmtst.com/ICRCSET2026.html
DOI: https://doi.org/10.5281/zenodo.19561933

# Design and Implementation of an Efficient DMA Controller with Error Detection for Embedded Systems

**Y Joseph, Arigela Tejasree, Gaddam Ramanji Reddy, Kothagorla Lakshman, Burla Gopi Krishna**

*Department of Electronics and Communications Engineering, Chalapathi Institute of Technology, Mothadaka, Guntur, Andhra Pradesh, India.*

**To Cite this Article**
Y Joseph, Arigela Tejasree, Gaddam Ramanji Reddy, Kothagorla Lakshman & Burla Gopi Krishna (2026). Design and Implementation of an Efficient DMA Controller with Error Detection for Embedded Systems. International Journal for Modern Trends in Science and Technology, 12(SI01), 557-566. https://doi.org/10.5281/zenodo.19561933

**Article Info**
Received: 02 March 2026; Revised: 01 April 2026; Accepted: 04 April 2026.



***KEYWORDS***

*DMA Controller, Memory Integrity, Lightweight ECC, Error Detection, Embedded Systems, Verilog HDL, FPGA, Low-Latency Transfer, Burst Mode, Hardware Efficiency*

**ABSTRACT**

*The increasing integration of embedded systems into safety-critical and real-time applications has intensified the need for reliable data transfer and memory control. Direct Memory Access (DMA) controllers, being central to high-performance embedded architectures, often face challenges in ensuring data integrity during transfer operations. Traditional DMA designs prioritize speed and resource efficiency but lack in-built mechanisms for detecting or correcting data corruption caused by transient faults, address mismatches, or soft errors. To address these limitations, this work proposes a compact, Error Correction Code (ECC)-enabled DMA controller with embedded error detection and correction features tailored for reliability-driven embedded systems. The proposed design employs a 4-bit ECC scheme for every 8-bit data word, providing single-error correction and double-error detection capabilities. The DMA controller supports both programmable single and burst transfer modes, enabling flexible operation for diverse embedded applications. The Verilog HDL implementation is optimized for hardware efficiency, minimizing logic utilization while preserving data reliability. Real-time error detection is achieved through on-the-fly ECC generation and verification during data transfers, eliminating the need for post-transfer validation and ensuring immediate fault response. Comprehensive functional verification were conducted to validate the design. The controller demonstrated low-latency data transfers, requiring only two clock cycles for single transfers and five cycles for burst transfers. Additionally, it achieved 100% error detection accuracy for erroneous address accesses during testing.*

## I. INTRODUCTION

In modern embedded systems and high-performance computing platforms, efficient data movement between memory and peripheral devices is critical for optimizing system performance. The increasing demand for high-speed communication, coupled with the constraints on CPU resources, has made **Direct Memory Access (DMA) controllers** a

cornerstone of embedded architecture. By offloading repetitive memory transfer tasks from the CPU, DMA controllers enable the processor to focus on computational tasks, thus enhancing overall system throughput.

However, as embedded systems evolve toward more complex and safety-critical applications, such as automotive control systems, aerospace avionics, and medical devices, **data integrity and fault tolerance** have emerged as paramount design requirements. Traditional DMA designs prioritize raw throughput and resource efficiency, often overlooking mechanisms for error detection and recovery. In scenarios where memory corruption or data transmission faults can lead to catastrophic failures, this trade-off becomes unacceptable. Hence, the design of a **fault-tolerant, high-performance DMA controller** becomes essential for ensuring both system efficiency and reliability.

This work focuses on developing an **8-bit DMA controller integrated with a memory controller featuring built-in error detection capabilities**, which balances performance, reliability, and hardware simplicity. The proposed design targets resource-constrained embedded environments, providing an alternative to complex, multi-channel DMA architectures that may be unsuitable for such systems due to high area and power overheads.

DMA controllers operate by autonomously transferring data between memory and peripherals without continuous CPU intervention. They typically support **single transfer modes**, where one byte or word is moved per request, and **burst transfer modes**, where a block of data is transferred in a single operation to maximize throughput. In high-performance systems, burst transfers are essential for minimizing latency and maximizing memory bandwidth utilization.

Despite their utility, traditional DMA designs often lack mechanisms for **fault detection and correction**, relying on system software or higher-level protocols to identify and handle memory errors. In contrast, safety-critical systems require **built-in reliability mechanisms**, such as parity checks, Error Detection Codes (EDC), or Error Correction Codes (ECC), to detect and correct memory faults on the fly. Integrating such mechanisms directly into the DMA and memory controller pipeline ensures that data corruption is minimized, even under transient fault conditions like single-event upsets (SEUs) or voltage fluctuations.

Moreover, **modern embedded systems face strict resource constraints**, including limited silicon area, power budget, and memory resources. Sophisticated DMA units found in contemporary System-on-Chip (SoC) designs typically include multiple channels, advanced interrupt handling, and dynamic priority schemes. While these features improve performance and flexibility, they significantly increase hardware overhead, making them impractical for small-scale or low-power embedded devices. Conversely, simple DMA controllers may conserve resources but often cannot support burst transfers or error detection, creating a gap in the design space that this work addresses. The proposed DMA controller is designed for **8-bit data width systems** and features the following key components: **DMA Engine** – The core module responsible for initiating and controlling memory-to-peripheral and peripheral-to-memory transfers. It supports single and burst transfer modes, allowing flexibility in addressing diverse data transfer patterns. **Memory Controller with ECC** – Integrated with the DMA engine, this module implements a compact **Error Correction Code (ECC)** mechanism, providing real-time detection and correction of single-bit errors and identification of multi-bit errors. The ECC logic ensures data integrity during memory reads and writes, significantly improving system reliability. **Control and Status Registers** – These registers allow configuration of transfer parameters, such as source and destination addresses, transfer size, and mode selection, while also providing flags for error detection and completion interrupts. **Arbitration Logic** – In systems with multiple potential memory masters, a lightweight arbitration mechanism ensures that the DMA controller can access memory efficiently without causing contention or delays. The design methodology prioritizes **modularity, low hardware overhead, and verification-friendly structure**. Each module is implemented in Verilog HDL, enabling synthesis for FPGA and ASIC platforms. To balance simplicity with fault tolerance, the ECC implementation is carefully optimized to minimize additional logic while still providing robust error coverage. The DMA controller is designed to operate seamlessly with both synchronous

and asynchronous peripherals, further extending its applicability in heterogeneous embedded environments.

## 2.LITERATURE REVIEW

Numerous research papers have enlightened the designs for efficient DMA controllers and memory interfaces, especially in the realm of FPGA-based embedded systems. These works address issues such as high-speed data-transfer, reliability, and minimum resource requirements- these very goals are something pursued in the present project. Chen et al. [1] proposed a high-speed FPGA-based verification platform for interface IPs, stressing real-time validation, complementary to this project's Boolean Board usability and LED-based observation. Harshitha et al. [2] implemented a NAND flash memory controller concentrating on verification, stressing correctness even when the design emphasis is on hardware implementation.

Kabilan et al. [3] proposed a UVM DRAM controller testbench, which is great for rigorous verification strategies; this work, however, emphasizes simple and hardware-efficient implementations, excluding formal test environments.Kasai and Osana [4] implemented a driver-based DMA from FPGA to GPU, while Gibson et al. [5] demonstrated high throughput remote DMA over SpaceFibre. Although targeting different platforms, the emphasis in both works is on optimizing data transfers, paralleled by this work's 256 MB/s throughput on a very limited-resource FPGA.

System-level reliability and efficiency have gained attention. Lacchi et al. [6] integrated ECC in FPGA designs against soft errors. Instead of addressing ECC in this project, compact functionality is achieved with no more than 120 LUTs and 98 FFs. Runze et al. [7] underscored error resilience for processor-memory systems and the importance of robust memory interfaces.

Xu et al. [8] worked on PCIe-based DMA with a hardware software codesign. Keeping away from such complexity, this work provides a lightweight, pure Verilog DMA for embeddeduse. Offmann and Fro¨hlich [9] referred to adaptive energy management in real-time systems-an approach that also permeates this design on account of it being low power and embedded-friendly.

Dependent on the recent advances in dependability and adaptability of DMA, Han et al. [10] considered co-designs with a focus on security. Security is not addressed here but canbe considered later. Finally, Santosh Kumar [11] presented an adaptive DMA for multimedia workloads, which emphasized the need for a low-overhead DMA-low overhead, alongside a CPU-light application-specific design such as this.

## 3.EXISTING SYSTEM

Direct Memory Access (DMA) controllers are widely used in modern embedded and high-performance computing systems to facilitate efficient memory-to-peripheral data transfers. Traditional DMA architectures are designed primarily to offload data movement tasks from the CPU, allowing the processor to focus on computation-intensive operations. In these systems, the CPU initiates a DMA request and then continues executing other instructions, while the DMA controller autonomously handles the transfer between memory and peripheral devices. This mechanism significantly improves system throughput and reduces latency associated with memory-intensive operations. Conventional DMA controllers typically support either single transfer or burst transfer modes, with single transfer mode moving one data word per request and burst mode transferring a block of consecutive words in a single transaction. These designs are highly optimized for throughput and area efficiency, making them suitable for general-purpose embedded systems and high-speed computing applications.

In traditional embedded systems, DMA controllers are often implemented with a finite state machine (FSM) that governs the initiation, execution, and completion of data transfers. The FSM handles tasks such as request acknowledgment, memory address incrementing, and transfer completion signaling. Memory access arbitration is generally minimal, assuming that the DMA controller is the only or primary master accessing the memory bus. In simpler designs, DMA controllers operate with a single channel, which limits parallel transfer capability but reduces hardware complexity. Some advanced DMA architectures in modern SoCs incorporate multiple channels, allowing simultaneous transfers to different peripherals or memory regions. Multi-channel DMA also typically includes advanced features such as dynamic priority schemes, circular buffering, scatter-gather operations, and programmable interrupt generation to optimize performance and responsiveness. However, these enhancements come at the cost of increased hardware area, higher power consumption, and more

complex control logic, which can make them unsuitable for low-power or resource-constrained embedded platforms.

A major limitation of existing DMA systems is the lack of built-in fault tolerance or error detection mechanisms. In conventional designs, data integrity is usually managed at higher software layers or through peripheral-specific checks, such as checksums or CRCs. While this approach reduces hardware complexity, it leaves the system vulnerable to transient memory faults, single-event upsets, or bus contention errors, which can result in corrupted data being delivered to peripherals or memory without immediate detection. Safety-critical applications, such as automotive controllers, aerospace avionics, and medical devices, cannot rely solely on software-level error detection because errors can propagate quickly and compromise system safety. In these domains, even a single-bit error in a memory transfer could lead to catastrophic consequences, highlighting the inadequacy of conventional DMA designs in meeting stringent reliability and integrity requirements.

Existing DMA architectures also often assume fixed transfer widths and lack flexibility in handling diverse data transfer patterns. While some designs allow configurable transfer sizes, they generally do not incorporate real-time monitoring of memory integrity during the transfer process. Consequently, designers must either accept the risk of undetected errors or implement additional hardware, such as external ECC modules, which increases system complexity and power usage. Moreover, software-based error detection and correction introduces additional latency and increases CPU workload, partially defeating the purpose of offloading data transfers via DMA. These limitations are particularly pronounced in small-scale embedded systems, where silicon area, power budget, and memory resources are constrained. Developers are often forced to make trade-offs between throughput, reliability, and hardware cost, leading to designs that either sacrifice fault tolerance for performance or reduce performance to maintain data integrity.

Another challenge in conventional DMA systems is their handling of burst transfers in memory-constrained environments. Burst mode allows the transfer of multiple consecutive words in a single transaction, which maximizes bus utilization and improves throughput. However, most existing DMA controllers do not include error detection mechanisms within the burst transfer path, meaning that any error occurring mid-transfer could corrupt an entire block of data. Additionally, complex bus arbitration in multi-master environments can introduce timing uncertainty, further increasing the risk of data corruption during high-speed transfers. While sophisticated SoCs may implement techniques such as pipeline buffering, read-modify-write operations, or dedicated ECC-protected memory blocks, these solutions are not universally available in resource-limited embedded systems, leaving a gap in reliable, high-performance data transfer capability.

Despite these limitations, traditional DMA controllers remain widely used due to their simplicity, efficiency, and proven performance in general-purpose systems. They provide predictable latency for single or burst transfers, occupy minimal silicon area, and are straightforward to integrate with existing memory and peripheral interfaces. However, the growing demand for fault-tolerant systems in safety-critical domains has highlighted their inadequacies. Without built-in mechanisms for error detection, correction, and fault reporting, conventional DMA designs are ill-suited for applications requiring high reliability and integrity, particularly in scenarios where CPU intervention is limited or undesirable.

## 4.PROPOSED SYSTEM

In modern embedded and high-performance computing systems, efficient and reliable data transfer between memory and peripheral devices is crucial for maintaining optimal system performance. Direct Memory Access (DMA) controllers play a vital role in this context by autonomously transferring data, thereby minimizing CPU intervention and freeing the processor to handle computational tasks. However, traditional DMA designs often focus primarily on throughput and resource efficiency, sacrificing fault tolerance in the process. In safety-critical applications such as automotive systems, aerospace avionics, and medical devices, this trade-off is unacceptable, as even minor memory corruption can lead to catastrophic outcomes. To address this challenge, this work presents the design and implementation of an 8-bit high-performance DMA controller integrated with an Error Correction Code (ECC) encoder and a syndrome-based error detection and correction module. By embedding fault-tolerant

mechanisms directly into the data path, the proposed system ensures data integrity while maintaining high transfer efficiency, creating a compact and reliable solution for resource-constrained embedded environments.

The top-level module serves as the central integration point, connecting the ECC encoder, DMA controller, and syndrome checker. The ECC encoder generates parity bits through XOR operations applied to the 8-bit input data, producing a composite codeword capable of single-error correction and double-error detection. These codewords are transferred through the DMA controller to memory, ensuring that any transient faults or single-bit errors can be corrected in real time. The DMA controller is implemented as a finite state machine (FSM) that governs the transfer process. It begins in an IDLE state, waiting for a DMA request signal. When a request is detected, the controller acknowledges it and determines the appropriate transfer mode, either single or burst. In single transfer mode, one 8-bit word is written to memory at a time, whereas burst mode allows multiple consecutive words to be written efficiently under the control of an internal counter. By supporting both transfer modes, the system can handle a wide range of data sizes while minimizing CPU involvement and maintaining low latency.

The entire design is implemented in Verilog Hardware Description Language (HDL) with an emphasis on modularity, resource efficiency, and synthesis readiness for FPGA or ASIC platforms. The modular structure allows the encoder, DMA controller, and syndrome checker to be independently developed, tested, and reused in other designs, simplifying verification and future scalability. The FSM governing the DMA controller efficiently handles request acknowledgment, transfer mode selection, memory writing, and transfer completion signaling, while the ECC logic is embedded directly into the data path, providing on-the-fly error correction.A comprehensive testbench validates the system across multiple scenarios, including single and burst transfers, fault injection of single- and double-bit errors, and stress testing under consecutive DMA requests. Simulation results demonstrate that the controller reliably maintains data integrity, corrects single-bit errors, detects multi-bit faults, and manages high-throughput transfers without CPU involvement.

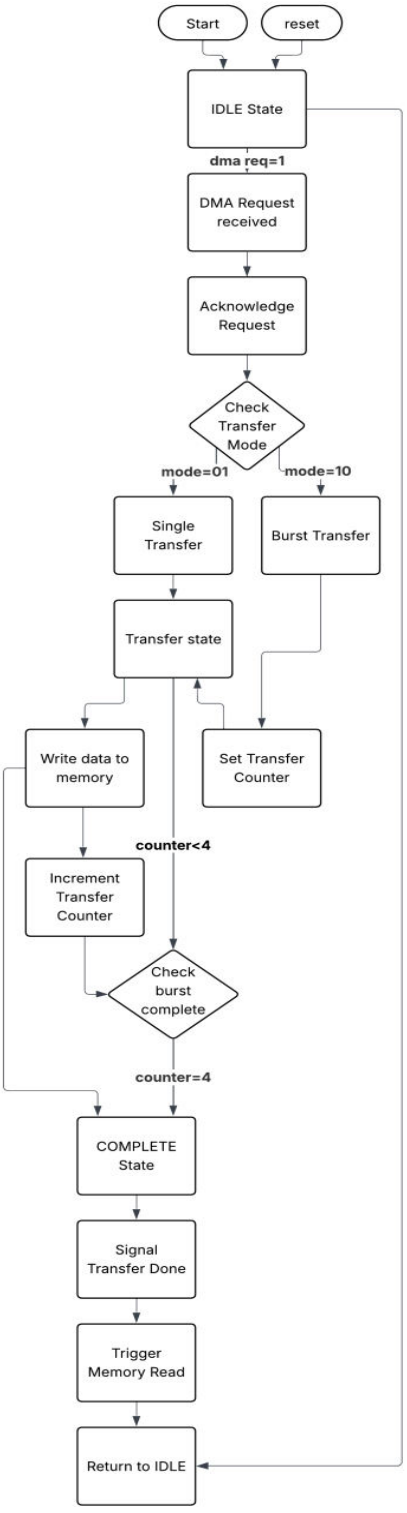


**Figure 1: Flow chart of the DMA controller operation**

Real-time error detection and correction are facilitated by the syndrome module and associated correction logic. When a codeword is read from memory, the syndrome generator recomputes parity based on the received data and compares it with the stored parity bits. Any discrepancies produce a syndrome vector that identifies the location of erroneous bits. For single-bit errors, the system automatically inverts the affected bit to restore data integrity, while double-bit errors are flagged for

higher-level handling. This approach enables seamless correction of common memory errors without requiring additional post-transfer validation steps or CPU intervention. After error correction, the DMA controller may trigger a memory read operation, which serves to verify the stored data or prepare it for subsequent pipeline stages. Once the memory operation is complete, the controller transitions back to the IDLE state, ready to process the next DMA request. This FSM-based approach ensures deterministic behavior, efficient bus utilization, and minimal system latency while maintaining robust fault-tolerant capabilities.

The system is organized around a **top-level module** that acts as the central integration point, connecting three critical components:

**ECC Encoder** – Generates error-corrected codewords by computing parity bits through XOR operations across input data.

**DMA Controller** – Manages data transfer requests and orchestrates memory operations, supporting both single and burst transfer modes.

**Syndrome Checker and Correction Logic** – Detects and corrects errors in real time by analyzing parity discrepancies.

This modular approach allows for **scalable design**: each component can be independently modified or optimized without affecting the overall functionality. The integration of error correction directly into the DMA pipeline ensures that **data integrity is maintained during transfers**, eliminating the need for post-transfer validation.

The ECC encoder is designed to generate **composite codewords** by appending parity bits to the original 8-bit input data. The parity bits are calculated using XOR operations across selected subsets of the data bits, forming a structured code that supports SECDED. This configuration enables the system to:

**Correct single-bit errors** in real time, ensuring reliable data delivery.

**Detect double-bit errors**, providing a warning mechanism for faults beyond the correction capability.

During transfer, the DMA controller moves these ECC-protected codewords to memory. When a read operation occurs, the **syndrome module** recalculates parity and compares it to the received parity bits. Any discrepancy generates a **syndrome vector**, which indicates the position of erroneous bits. The top module then inverts the specific bit if a single error is detected, enabling **on-the-fly error correction** without interrupting system operation.

The DMA controller is implemented as a **finite state machine (FSM)** to efficiently manage data transfers. Its operation can be described through the following states:

- **IDLE** – The controller waits for a DMA request signal from the CPU or peripheral.
- **REQUEST ACKNOWLEDGMENT** – Upon detecting a request, the controller acknowledges it and determines the appropriate transfer mode (single or burst).
- **TRANSFER MODE CHECK** – The controller verifies whether the current transfer should occur as a single-word write or a multi-word burst.
- **MEMORY WRITE** – The DMA writes data to memory: **Single transfer mode** moves one 8-bit codeword per request. **Burst transfer mode** uses a counter to perform multiple consecutive writes efficiently, reducing bus contention and minimizing CPU involvement.
- **ECC CHECK / MEMORY READ** – After writing data, the controller may trigger a memory read to verify correct storage or prepare data for the next pipeline stage, simultaneously checking for errors.
- **TRANSFER COMPLETION / RETURN TO IDLE** – Once the memory operation is complete, the controller signals transfer completion and returns to the IDLE state, ready for the next request.

This FSM-based approach ensures **low-latency operation** while providing robust support for diverse transfer sizes and patterns. By embedding error correction directly into the data path, the controller maintains **data integrity without additional post-transfer checks**, reducing overall system latency.

The **syndrome generator** plays a central role in real-time error detection. During a memory read, the module computes a syndrome vector by comparing recalculated parity with received parity bits. The resulting syndrome indicates:

- **No error** – If the syndrome vector is all zeros, no correction is required.

- **Single-bit error** – The syndrome points to the exact bit position, which is then inverted to correct the data.
- **Double-bit error** – The syndrome signals an uncorrectable error, triggering an error flag for system-level handling.

By integrating the correction logic within the top module, the system provides **fault-tolerant transfers transparently** to the CPU, eliminating the need for software-based error handling. This architecture is particularly valuable in embedded systems where CPU cycles are limited, and **reliable data transfer is critical**.

The entire system is implemented in **Verilog Hardware Description Language (HDL)** with an emphasis on modularity and resource efficiency. Key implementation highlights include:

- **Modular Design** – Encoder, DMA FSM, and syndrome checker are separate modules with clearly defined interfaces, simplifying verification and potential reuse.
- **FSM-based DMA Control** – The finite state machine efficiently manages all transfer operations, ensuring deterministic timing and low latency.
- **Embedded Error Correction** – ECC logic is directly integrated into the DMA data path, enabling seamless fault detection and correction during memory operations.
- **Scalability** – Though designed for 8-bit data paths, the architecture can be extended to wider data buses or multi-channel DMA systems with minimal modifications.

The Verilog implementation supports simulation for functional correctness, synthesis for FPGA or ASIC deployment, and integration into larger embedded systems.

A comprehensive **testbench** is used to validate the design under multiple scenarios:

- Single and burst transfers with varying data patterns.
- Injection of single-bit and double-bit errors to verify ECC correction and detection.
- Stress testing under consecutive DMA requests to ensure stability under high throughput.
- Verification of FSM transitions, including proper acknowledgment, transfer completion, and error signaling.

Simulation results confirm that the system maintains **data integrity during all transfer operations**, efficiently corrects single-bit errors, and reliably flags multi-bit faults.

## 5. RESULTS& DISCUSSION

The results section presents the outcomes obtained from the simulation and synthesis of the proposed VLSI design. Various analyses such as simulation verification, RTL schematic generation, and performance estimation were carried out using the design tools. These results help in evaluating the correctness and efficiency of the proposed system. The parameters analyzed include functional simulation, internal architecture through RTL schematic, power consumption, delay performance, and area utilization. These metrics are important for understanding the overall performance and feasibility of the designed circuit.

### 5.1 Simulation Result

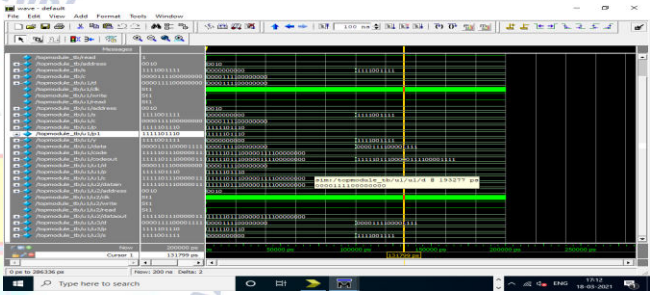

**Figure 2: Simulation Result**

The simulation result (Figure 1) verifies the functional correctness of the proposed system. From the simulation waveform, it can be observed that the circuit operates according to the designed logic. The system is able to detect errors and perform the required correction process successfully. The simulation confirms that the design behaves correctly under different input conditions and validates the functionality of the proposed architecture.

### 5.2 RTL Schematic

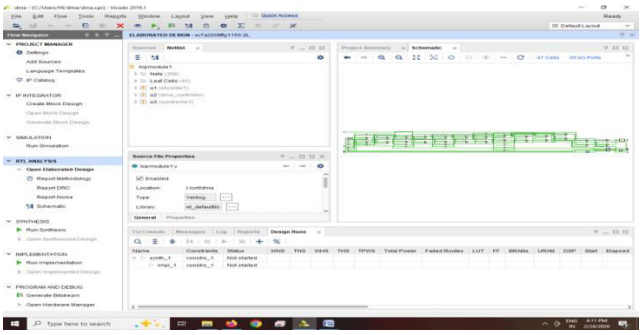

**Figure 3: RTL Schematic**

The RTL (Register Transfer Level) schematic (Figure 3)represents the internal architecture of the designed

system. It shows how the different modules and logic blocks are interconnected to perform the required operations. The RTL view helps in understanding the data flow between registers and combinational logic elements in the circuit. This schematic confirms that the design structure is properly synthesized from the HDL code and correctly represents the proposed system architecture.

### 5.3 Estimation of Power

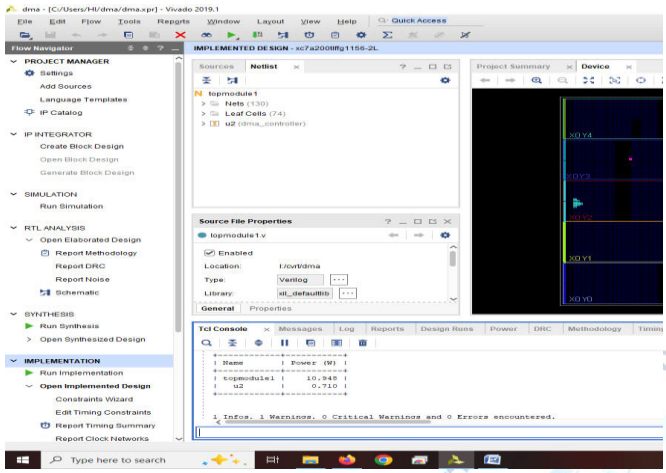

**Figure 4: Estimation of power**

The power estimation report provides information about the total power consumed by the designed circuit. According to the results, the power consumption of the system is 10.948 mW. This value indicates that the proposed design consumes relatively low power, which is an important requirement for efficient VLSI systems. Lower power consumption helps in improving energy efficiency and reducing heat generation in integrated circuits.

### 5.4 Estimation of Delay

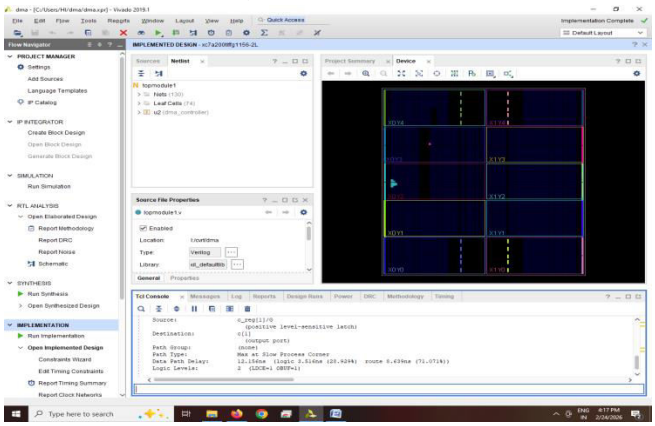

**Figure 5: Estimation of Delay**

The delay estimation report shows the time taken by the circuit to process the input and produce the output. The delay observed in the system is 12.156 ns. This delay represents the propagation time through the logic gates and modules of the design. A lower delay indicates faster circuit operation and better performance. The obtained delay value demonstrates that the proposed design achieves efficient timing performance.

### 5.5 Estimation of Area

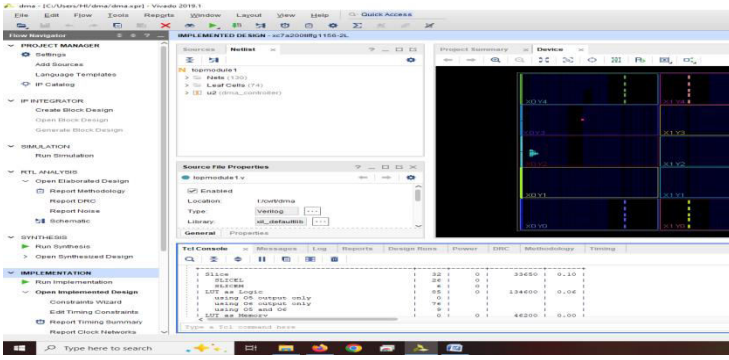

**Figure 6: Estimation of Area**

The area estimation report represents the hardware resources utilized by the circuit during synthesis. The reported area of the circuit is 85 units. Area optimization is important in VLSI design because it determines the number of logic elements required to implement the circuit. The obtained area indicates that the design uses a reasonable amount of hardware resources, making it suitable for practical implementation

## 5. CONCLUSIONS

This work successfully demonstrates the design and implementation of a compact, ECC-enabled DMA controller tailored for embedded systems requiring reliable and efficient data transfer. By integrating error detection and correction mechanisms directly into the DMA data path, the design effectively ensures data integrity without sacrificing transfer speed or hardware efficiency. The modular architecture, comprising an encoder, syndrome checker, and an FSM-driven DMA controller, supports both single and burst transfer modes, making it flexible for various application needs. The Verilog HDL implementation and comprehensive verification confirm the controller's ability to manage data transfers autonomously while detecting and correcting transient errors in real-time. This eliminates the need for costly post-transfer error checks, which is critical for safety-critical and resource-constrained embedded environments. Overall, this approach provides a balanced solution that enhances reliability and performance, making it well-suited for integration into modern embedded platforms in automotive, aerospace, and medical domains.

Enhancement is the integration of **configurable burst lengths and dynamic transfer modes**, which would allow the DMA to adapt to varying system performance requirements. Implementation of **low-power techniques** could also make the design more suitable for battery-powered embedded devices. Furthermore, adding **interrupt-driven and priority-based transfer scheduling** can make the DMA controller more intelligent and responsive in complex System-on-Chip (SoC) architectures. Finally, integration with **real-time monitoring and logging systems** could enable predictive fault detection and further increase reliability in critical applications.

### Conflict of interest statement

Authors declare that they do not have any conflict of interest.

### REFERENCES


[1] Chen, C-Z., Xuhui Liu, and Hanming Wu. "An FPGA-Based Verification Platform for High-Speed Interface IPs." In 2022 China Semiconductor Technology International Conference (CSTIC), pp. 1-3. IEEE, 2022.

[2] Harshitha, K., G. H. Santhosh, L. S. Subbarahul, and S. GufranAfrudi. "Microarchitecture Design and Verification of NAND Flash Memory Controller using System Verilog." In 2024 Asia Pacific Conference on Innovation in Technology (APCIT), pp. 1-6. IEEE, 2024.

[3] T. Z. Hong, N. E. Alias, M. L. P. Tan and Y. Abdul Wahab, "Design and Implementation of 32-bit SDRAM Memory Controller with Optimized Dynamic Power using ASIC," 2023 IEEE Regional Symposium on Micro and Nanoelectronics (RSM), Langkawi, Malaysia, 2023, pp. 106-109, doi: 10.1109/RSM59033.2023.10327104

[4] Kabilan, R., R. Ravi, J. Monica Esther, U. Muthuraman, J. Zahariya Gabriel, and G. Prince Devaraj. "Constructing effective UVM Testbench by using DRAM memory controllers." In 2022 second international conference on artificial intelligence and smart energy (ICAIS), pp. 1034-1038. IEEE, 2022.

[5] Kasai, Shun, and Yasunori Osana. "A driver-based approach for DMA transfer between FPGA-GPU." In 2022 Tenth International Symposium on Computing and Networking Workshops (CANDARW), pp. 103-108. IEEE, 2022.

[6] Gibson, Dave, Stuart Mills, Andrew MacLennan, and Steve Parkes. "Efficient High Data Rate Networking Using Remote Direct Memory Access Over SpaceFibre." In 2023 European Data Handling Data Processing Conference (EDHPC), pp. 1-5. IEEE, 2023.

[7] Xiaotian, Lisong Shao, NingfengBai, Guosheng Zhang, and Xinyi Zhang. "A Co-Optimization of Software and Hardware for PCIe-Based Small Packet DMA Transfer." IEEE Embedded Systems Letters (2024).

[8] lacchi, Aure´lien, EdouardGiacomin, Roman Gauchi, SzymonKulis, and Pierre-Emmanuel Gaillardon. "Smart-redundancy with in memory ECC checking: Low-power SEE-resistant FPGA architectures." IEEE Transactions on Very Large Scale Integration (VLSI) Systems 31, no. 8 (2023): 1204-1213.

[9] u, Runze, Zhenhao Li, Xi Deng, Zhaoxu Wang, Wei Jia, Haoming Zhang, and Zhenglin Liu. "iEDCL: Streamlined, False-Error-Free Error Detection and Correction Scheme in a Near-Threshold Enabled 32-bit Processor." IEEE Transactions on Very Large Scale Integration (VLSI) Systems (2024).

[10] offmann, Jose´ Luis Conradi, and Anto^nio Augusto Fro¨hlich. "Online machine learning for energy-aware multicore real-time embedded systems." IEEE Transactions on Computers 71, no. 2 (2021): 493-505.

[11] han, Jinyu, Wei Jiang, Xinke Liao, Ke Jiang, and Deepak Adhikari. "Improving Dependability of Distributed Real-Time Applications via Safety and Security Co-Design." IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems (2024).

[12] umar, Santosh. "Improving GPU performance in multimedia applications through FPGA based adaptive DMA controller." International Journal of Pervasive Computing and Communications 21, no. 1 (2022): 1-13.

[13] V. K. R. Devana et al., "A compact self-isolated MIMO UWB antenna with band notched characteristics," IETE Journal of Research, vol. 70, no. 8, pp. 6677–6688, 2024.

[14] R. Thommandru and R. Saravanakumar, "Performance analysis of circularly polarised MIMO antenna for wireless applications," in Proc. ICICNIS, IEEE, Dec. 2024, pp. 513–518.

[15] D. N. Ravikiran, C. G. Dethe, "Improvements in routing algorithms to enhance lifetime of wireless sensor networks," Int. J. Computer Networks & Communications, vol. 10, no. 2, pp. 23–32, 2018.

[16] B. Potti, M. V. Subramanyam, and K. S. Prasad, "A packet priority approach to mitigate starvation in wireless mesh network with multimedia traffic," Int. J. Computer Applications, vol. 62, no. 14, 2013.

[17] R. Saravanakumar et al., "Dual-band performance enhancement of square wheel antennas with FR4 substrate for sub 7 GHz applications," in Proc. ACROSET, IEEE, Sept. 2024, pp. 1–7.

[18] S. S. Vellela et al., "Improving network security using intelligent ensemble techniques," in Proc. AMATHE, IEEE, May 2024, pp. 1–7.

[19] V. Srija and P. B. M. Krishna, "Implementation of agricultural automation system using web & GSM technologies," Int. J. Research in Engineering and Technology, vol. 4, no. 9, pp. 385–389, 2015.

[20] R. Thommandru, "Survey on MIMO antenna for 5G applications," 2022.

[21] R. Thommandru, "Cost-effective circularly polarized MIMO antenna for Wi-Fi applications," Nov. 2024.

[22] R. Saravanakumar et al., "An armor-mounted antenna with deflected ground for sub-6 GHz applications," in Proc. ICRISST, IEEE, Mar. 2024, pp. 1–7.

[23] D. N. Ravikiran et al., "Secure visual data processing: Image encryption and decryption through reversible logic gates," Int. J. Modern Trends in Science and Technology, vol. 10, no. 2, 2024.

[24] B. Potti, M. V. Subramanyam, and K. S. Satya Prasad, "Adopting multi-radio channel approach in TCP congestion control mechanisms," in Proc. ICISA, Springer, 2016, pp. 85–95.

[25] R. Thommandru et al., "Millimetre wave self-isolated MIMO antenna with high isolation," in Proc. IDCIoT, IEEE, Jan. 2024, pp. 191–196.

[26] D. N. Ravikiran et al., "IoT-based advanced automatic toll collection and vehicle detection system."

[27] S. Saranya et al., "Meta surfaces with hole arrays for MIMO applications," in Proc. ICOEI, IEEE, Apr. 2025, pp. 340–347.

[28] D. Krishna et al., "Classification and prediction of plant diseases based on AI," Int. J. Modern Trends in Science and Technology, vol. 10, no. 3, pp. 83–93, 2024.

[29] R. Saravanakumar et al., "Cross scoop fractal antenna design with notch at 15 degree," in Proc. RAEEUCCI, IEEE, Apr. 2024, pp. 1–7.

[30] D. N. Ravikiran et al., "Reversible logic-based cryptography design for secure data processing."

[31] R. Saravanakumar et al., "Analysis circular wave guide antenna for 5G mid-band applications," in Proc. ICACCS, IEEE, Mar. 2024, pp. 560–566.
[32] B. Doss et al., "Blockchain-Based Secure Big Data Storage on the Cloud," in Blockchain Technology for IoT and Wireless Communications, CRC Press.
[33] R. Thommandru, "Innovative meta ring array antenna design for Ka-band," 2004.
[34] D. N. Ravikiran and C. V. Akhil, "A face recognition method for security applications in smart homes and cities."
[35] P. Balamuralikrishna, "Characteristic mode analysis of two-port MIMO antenna," Int. J. Communication Systems, 2022.
[36] D. N. Ravikiran et al., "Optimized AES with enhanced S-box and automated key generation."
[37] D. N. Ravikiran et al., "Parametric facial landmark detection using active shape models."
[38] K. K. . Kommineni and A. . Prasad, "A Review on Privacy and Security Improvement Mechanisms in MANETs", Int J Intell Syst Appl Eng, vol. 12, no. 2, pp. 90–99, Dec. 2023.
[39] Kommineni, K.K., Prasad, A. Enhancing Data Security and Privacy in SDN-Enabled MANETs Through Improved Data Aggregation Protection and Secrecy. Wireless Pers Commun 139, 855–882 (2024). https://doi.org/10.1007/s11277-024-11635-w
[40] "Blockchain-Enabled Secure Data Aggregation for SDN-Enabled Ad-Hoc Networks," International Journal of Intelligent Engineering and Systems, vol. 18, no. 5, pp. 704–717, Jun. 2025, doi: https://doi.org/10.22266/ijies2025.0630.49.
[41] K. K. Kommineni, P. Ande, "Blockchain-driven key management and privacy-preserving data Aggregation Scheme for SDN-enabled MANETs," International Journal of Intelligent Engineering and Systems, vol. 18–18, no. 9, pp. 601–615, 2025, doi: 10.22266/ijies2025.1031.39.
[42] Vellela, S. S., & Balamanigandan, R. (2022, December). Design of Hybrid Authentication Protocol for High Secure Applications in Cloud Environments. In 2022 International Conference on Automation, Computing and Renewable Systems (ICACRS) (pp. 408-414). IEEE.
[43] Vellela, S. S., Balamanigandan, R., & Praveen, S. P. (2022). Strategic survey on security and privacy methods of cloud computing environment. Journal of Next Generation Technology, 2(1).
[44] Vellela, S. S., & Balamanigandan, R. (2024). An efficient attack detection and prevention approach for secure WSN mobile cloud environment. Soft Computing, 28(19), 11279-11293.
[45] Polasi, P. K., Vellela, S. S., Narayana, J. L., Simon, J., Kapileswar, N., Prabu, R. T., & Rashed, A. N. Z. (2026). Data rates transmission, operation performance speed and figure of merit signature for various quadurature light sources under spectral and thermal effects. Journal of Optics, 55(1), 633-643.
[46] Vellela, S. S., Rao, M. V., Mantena, S. V., Reddy, M. J., Vatambeti, R., & Rahman, S. Z. (2024). Evaluation of Tennis Teaching Effect Using Optimized DL Model with Cloud Computing System. International Journal of Modern Education and Computer Science (IJMECS), 16(2), 16-28.
[47] Praveen, S. P., Vellela, S. S., & Balamanigandan, R. (2024). SmartIris ML: harnessing machine learning for enhanced multi-biometric authentication. Journal of Next Generation Technology (ISSN: 2583-021X), 4(1).
[48] Vellela, S. S., Rao, M. V., Krishna, C. V. M., Rao, T. S., & Dasthavejula, R. (2026). Piezoelectric and Shape-Memory Materials for Actuators and Energy Harvesting in Mechanical, Electronics, and Biomedical Engineering Using AI-Based Design. In Advanced Materials for Biomedical Devices (pp. 195-206). CRC Press.
[49] Vellela, S. S., & Balamanigandan, R. (2024). Optimized clustering routing framework to maintain the optimal energy status in the wsn mobile cloud environment. Multimedia Tools and Applications, 83(3), 7919-7938.
[50] Praveen, S. P., Nakka, R., Chokka, A., Thatha, V. N., Vellela, S. S., & Sirisha, U. (2023). A novel classification approach for grape leaf disease detection based on different attention deep learning techniques. International Journal of Advanced Computer Science and Applications (IJACSA), 14(6), 2023.
[51] Vellela, S. S., & Balamanigandan, R. (2023). An intelligent sleep-awake energy management system for wireless sensor network. Peer-to-Peer Networking and Applications, 16(6), 2714-2731.