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

Mr. E. Prabhakar[1], N. Hari Raj Goud[2], M. Varshitha[3], M. Venu Sagar[4], MD. Nawaaz[5]
[1]*Assistant Professor, Dept. of ECE, TKR College of Engineering and Technology*
[2,3,4,5] *Student, Dept. of ECE, TKR College of Engineering and Technology*

Abstract - This paper presents the design and simulation of an efficient DMA Controller with error detection for embedded systems. The Direct Memory Access (DMA) controller enables efficient data transfer without continuous CPU involvement. Conventional DMA systems are limited in handling non-contiguous memory and require repeated processor intervention. This paper presents a Scatter-Gather DMA controller that uses a descriptor-based approach to manage multiple data blocks automatically. The proposed design reduces CPU overhead, improves throughput, and supports efficient data transfer in modern systems.

Index Terms- DMA Controller, Scatter-Gather, ECC, RTL Design, Embedded Systems, Data Transfer.

I. INTRODUCTION

The continuous evolution of high performance embedded systems and System-on-Chip (SoC) architectures has significantly transformed data processing, while also highlighting the "memory wall" problem, where processor speeds increase faster than memory access times, leading to delays and reduced efficiency. In traditional systems, data transfer is handled using Programmed I/O (PIO), where the CPU manages every operation, resulting in high overhead and inefficient resource utilization. To overcome this, Direct Memory Access (DMA) controllers were introduced to enable direct data transfer between memory and peripherals without continuous CPU involvement. The controller autonomously processes these descriptors in sequence, enabling efficient transfer of fragmented data without additional CPU interaction, thereby improving performance and scalability. Furthermore, in modern systems where reliability is critical, data integrity is a major concern due to soft errors that can corrupt memory contents. To ensure fault tolerance, the proposed design incorporates a lightweight Error Correction Code (ECC) mechanism that detects errors during data transfer, preventing incorrect memory operations. Thus, the integration of scatter-gather capability with error detection provides an efficient, reliable, and high-performance solution for modern embedded and real-time applications.

II. METHODOLOGY

The methodology adopted in this work focuses on the design and simulation of an efficient DMA Controller with Error Detection for Embedded Systems.The methodology of the proposed Scatter-Gather DMA (SG-DMA) controller is developed to achieve efficient, autonomous, and reliable data transfer between memory and peripheral devices while minimizing processor dependency.

The proposed Scatter-Gather DMA (SG-DMA) controller is designed to enable efficient data transfer while reducing processor involvement and handling non-contiguous memory effectively. The methodology begins with identifying the limitations of conventional DMA systems, particularly their inability to manage fragmented data without repeated CPU intervention. To overcome this, a descriptor-based approach is used. In this method, the processor initializes a list of descriptors in memory, where each descriptor contains the source address, destination address, transfer size, and a pointer to the next descriptor.

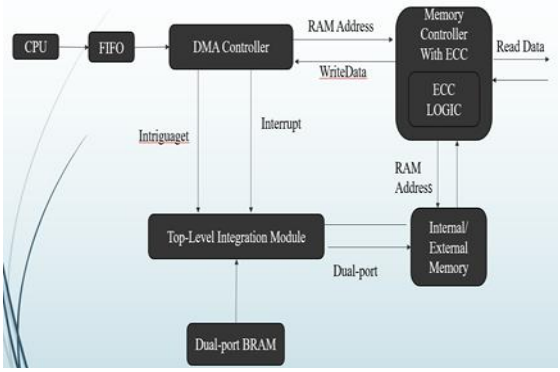

Fig 1: Block Diagram

Once the DMA controller is activated, it automatically fetches the first descriptor and begins the transfer process. A Finite State Machine (FSM) controls the sequence of operations such as descriptor fetching, data reading, data writing, and moving to the next descriptor.

After completing the transfer of one data block, the controller uses the pointer to access the next descriptor and continues the operation without additional processor involvement. This enables continuous transfer of non-contiguous data, improving efficiency and reducing overhead.

To ensure reliable data transfer, an Error Correction Code (ECC) mechanism is included. ECC bits are generated for each data unit and verified during transfer to detect errors. If an error is identified, an error signal is generated to prevent incorrect data transmission.

Overall, this methodology improves system performance by reducing processor load, increasing data throughput, and ensuring reliable data transfer, making it suitable for modern embedded and real-time applications.

III. IMPLEMENTED DESIGN

The implementation of the proposed the DMA Controller of the advanced Scatter-Gather DMA controller begins with writing the design in Verilog Hardware Description Language. The entire system is divided into modules such as the DMA controller, memory interface, and control logic based on a Finite State Machine. Each module is coded separately and then integrated to form the complete design. The Verilog code defines inputs such as clock, reset, start signal, and descriptor address, and outputs such as memory control signals and completion interrupt. The FSM logic is described using sequential and combinational blocks to control descriptor fetching, read and write operations, and transfer completion. The project is implemented using Verilog HDL and simulated using Xilinx Vivado. The advanced Scatter-Gather DMA controller is designed in a modular way, including components such as the Finite State Machine. (FSM), memory interface, descriptor handling unit, and ECC logic.

These modules are interconnected using signals like memory address, data lines, and control signals for read and write operations. The FSM controls the overall data transfer process by moving through states such as idle, fetch, read, write, and complete.

A Verilog testbench is developed to simulate the system by generating clock and reset signals, initializing memory with descriptors and data, and providing a start signal to trigger the DMA operation. The design is compiled and simulated in Vivado, and waveform analysis is used to verify the behavior of signals such as address updates, data transfer, and FSM transitions. The results show that the DMA controller correctly fetches descriptors, performs data transfer between non-contiguous memory locations, and generates an interrupt signal after completion.

The ECC logic is also tested during simulation, where errors are introduced and successfully detected, ensuring reliable data transfer. Since the implementation is fully simulation-based without FPGA hardware, it provides an efficient and cost-effective way to validate the functionality and performance of the proposed system.

IV.RESULTS AND DISCUSSION

The implementation of the advanced Scatter-Gather DMA controller begins with writing the design in Verilog Hardware Description Language. The entire system is divided into modules such as the DMA controller, memory interface, and control logic based on a FSM.

Each module is coded separately and then integrated to form the complete design. The Verilog code defines inputs such as clock, reset, start signal, and descriptor address, and outputs such as memory control signals and completion interrupt. The FSM

logic is described using sequential and combinational blocks to control descriptor fetching, read and write operations, and transfer completion.

4.1 Project Creation in Xilinx Vivado

After writing the Verilog code, a new project is created in Xilinx Vivado. The target device can be selected, but since this project focuses only on simulation, hardware selection is not critical. The Verilog source files are added to the project under design sources, and a separate testbench file is added under simulation sources. The tool automatically organizes files and prepares them for synthesis and simulation.

4.2 Module Connections and Integration

In the design, all modules are interconnected through defined signals. The DMA controller is connected to the memory interface using signals such as memory address, read enable, write enable, and data lines. Internally, registers are used to store source address, destination address, transfer count, and descriptor pointers. The FSM generates control signals that coordinate data flow between these modules. Proper signal connections ensure that data is read from the correct source, processed, and written to the destination without conflicts.

4.3 Simulation Setup and Testbench

To verify the functionality of the design, a testbench is created in Verilog. The testbench simulates the behavior of memory and provides input signals such as clock, reset, and start.

It also initializes descriptor values and data in a simulated RAM. The testbench generates a clock signal using a periodic toggle and applies reset at the beginning to initialize the system. After that, it triggers the start signal to begin DMA operation. The memory model responds to read and write signals, allowing the DMA controller to perform operations as if it were connected to real hardware.

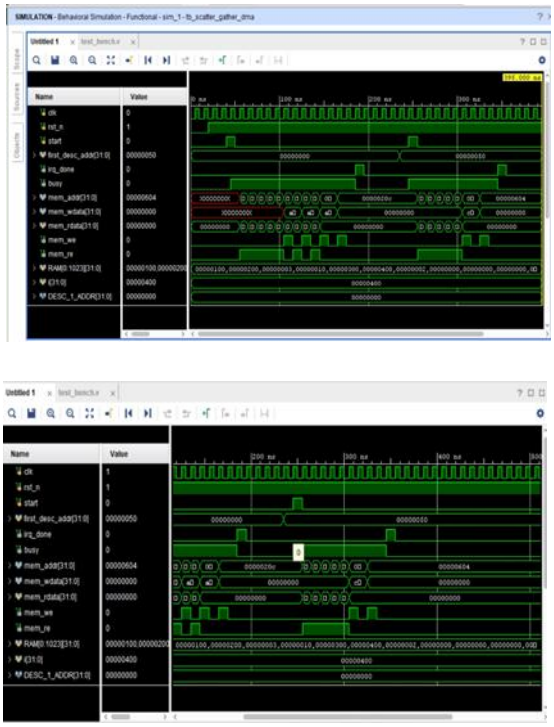

Fig 2: Wave Forms

4.4 Running Simulation

Simulation is executed using the built-in simulator in Xilinx Vivado. When the simulation is run, the tool compiles the design and testbench, and then executes the behavior over time. The waveform window displays signals such as clock, reset, start, memory address, read/write enable, data values, and completion signal.

By analyzing these waveforms, it is possible to verify whether the FSM transitions correctly through states like IDLE, FETCH, READ, WRITE, and COMPLETE.

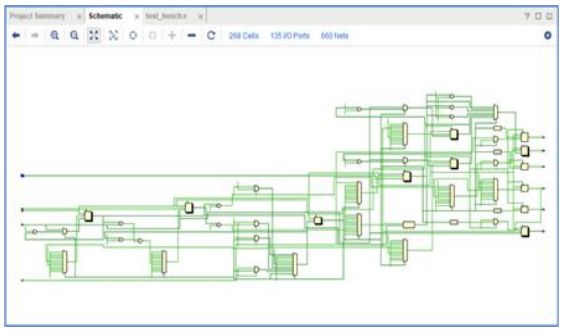

Fig 3: Schematic Diagram

4.5 Error Detection Verification

The ECC mechanism is also tested during simulation by introducing faults in the data. When an error

occurs, the syndrome calculation detects it, and an error signal is generated. Overall, the implementation process involves writing Verilog code, creating a project in Vivado, connecting modules properly, developing a testbench, and running simulations to verify the design. The results confirm that the Scatter-Gather DMA controller works efficiently, handling non-contiguous memory and providing reliable data transfer.

Overall, the implementation process involves writing Verilog code, creating a project in Vivado, connecting modules properly, developing a testbench, and running simulations to verify the design. The results confirm that the Scatter-Gather DMA controller works efficiently, handling non-contiguous memory and providing reliable data transfer.

V. CONCLUSION

This paper presented the design and simulation of an efficient DMA Controller with error detection for embedded systems. The advanced Scatter-Gather DMA controller developed in this project provides an efficient and reliable solution for high-speed data transfer by overcoming the limitations of conventional DMA systems. By using a descriptor-based approach, it successfully handles non-contiguous memory blocks in a continuous manner without requiring frequent CPU intervention, thereby reducing processing overhead and improving system performance. The Finite State Machine ensures smooth control of operations such as descriptor fetching, data transfer, and completion, while automatic address updating and descriptor chaining enhance efficiency. The integration of ECC logic further improves reliability by detecting errors during data transfer and preventing corrupted data from being written into memory. The design is implemented using Verilog HDL and verified through simulation in Xilinx Vivado, where results confirm correct functionality, proper signal timing, and efficient operation under different conditions. Overall, the proposed system achieves low latency, high throughput, and reliable performance, making it suitable for modern embedded and real-time applications.

REFERENCES

[1] Intel Corporation, “Direct Memory Access (DMA) Controller Architecture,” Intel Technical Manual, 2018.
[2] ARM Ltd., “AMBA AXI and DMA Controller Design Guide,” ARM Documentation, 2019.
[3] Xilinx Inc., “AXI DMA v7.1 LogiCORE IP Product Guide,” Xilinx Documentation, 2020.
[4] Altera (Intel FPGA), “DMA Controller Core User Guide,” Altera Documentation, 2017.
[5] S. Goel and R. Kumar, “Design and Implementation of DMA Controller Using Verilog,” International Journal of VLSI Design, 2018.
[6] M. Patel and K. Shah, “High Performance DMA Controller for Embedded Systems,” International Journal of Computer Applications, 2019.
[7] P. Gupta and S. Sharma, “Efficient Data Transfer Using DMA in Embedded Systems,” International Journal of Engineering Research, 2017.
[8] R. Singh and A. Verma, “Scatter-Gather DMA Architecture for High-Speed Systems,” IEEE Conference on Embedded Systems, 2020.
[9] S. Reddy and K. Rao, “FPGA-Based DMA Controller Design,” International Journal of Electronics and Communication Engineering, 2018.
[10] Y. Zhang et al., “High Throughput DMA Engine for SoC Applications,” IEEE Transactions on Very Large Scale Integration (VLSI) Systems, 2019.
[11] T. Kim and H. Lee, “Low Latency DMA Controller Design for Real-Time Systems,” IEEE Embedded Systems Letters, 2021.
[12] N. Patel and J. Mehta, “Design of Scatter-Gather DMA Controller Using Verilog,” International Journal of Advanced Research in Electronics, 2020.
[13] K. Ramesh and V. Kumar, “Performance Analysis of DMA Controller in Embedded Systems,” International Journal of Computer Science, 2017.
[14] S. Banerjee et al., “Efficient Memory Access Using DMA Techniques,” IEEE Access, 2021.
[15] J. Smith and L. Brown, “DMA Controller Design for High-Speed Data Transfer,” IEEE Conference on Computer Architecture, 2018.