# IMPLEMENTATION AND ANALYSIS ON FIFO USING FPGA

**NANDIGAMA SWABHANU[1], Dr. S. NAGI REDDY[2] ,Dr. SK. UMAR FARUQ[3].**

**1 Pg Scholar, Department of Electronics and Communication Engineering, TEEGALA KRISHNA REDDY Engineering College, Hyderabad.**

**2 Associate Professor, Department of Electronics and Communication Engineering, TEEGALA KRISHNA REDDY Engineering College, Hyderabad.**

**3 Professor, Department of Electronics and Communication Engineering, TEEGALA KRISHNA REDDY Engineering College, Hyderabad.**

## ABSTRACT

Several regions across India are currently experiencing significant energy shortages. This study investigates the implementation of an optimized First-In- First-Out (FIFO) mechanism using Field-Programmable Gate Array (FPGA) technology to enhance environmentally sustainable transmission systems. FIFO, which manages and processes items or data in the order they are received, mirrors the intuitive operation of real-life queues and lines. The proposed design employs the Genesys board as the FPGA hardware, which offers high performance, Gigabit Ethernet connectivity, and design flexibility, making it suitable for highly complex applications. The integration of this optimized FIFO mechanism on the Genesys FPGA board demonstrates potential improvements in the efficiency and sustainability of energy transmission systems, addressing current energy shortfalls effectively.

**Keywords – Xilinx, FPGA, Energy efficiency, Vivado, First in First out (FIFO).**

## INTRODUCTION

The transfer of data over a point-to-point or point-to-multi point communication channel. Examples of such channels are copper wires, optical Fibers, wireless communication channels, storage media and computer buses. Earlier days to achieve the high speed we were using parallel/serial transmission. In parallel transmission, Binary data consisting of 1s and 0s which are transmitted in framing/streaming. By grouping, we can send data n bits at a time instead of one. We use n wires to send n bits at one time. That way each bit has its own wire, and all n bits of one group can be transmitted with each clock pulse from one device to another. Form = 8. Typically, the n number of bits are bundled in a frame with a connector at each end that may be 2:4:8:16:32. The advantage of parallel transmission is speed. All else being equal, parallel transmission can increase the transfer speed by a factor of n over serial transmission. Insignificant disadvantage of parallel transmission is cost. In serial transmission one bit follows another, so we need only one communication channel rather than n to transmit data between two communicating devices.

The advantage of serial over parallel transmission is that with only one communication channel, serial transmission reduces the cost of transmission over parallel by roughly a factor of n Since communication within devices is parallel, conversion devices are required at the interface between the sender and the line (parallel-to-serial) and between the line and the receiver (serial to parallel).

Serial transmission occurs in one of two ways; asynchronous or synchronous. With the rapid development of FPGA technology, FPGA is widely used in the field of communication, medical instrument, consumer electronics, display, portable terminal, and so on. Among various levels of FPGA, the low-and-middle-end FPGA has won a wide market due to its low cost, high performance, technical maturity and short design cycle. As a fundamental storage structure in the design of system based on FPGA, FIFO is usually used as data buffer of digital signal processing system, Communication Bridge between network of different data rates and communication interface between modules in different clock domain. Sometimes, in practice, it is needed to combine more than one FIFO into a new structure as multi-channel FIFO, according to rule of access to it, multi-channel FIFO generally can be classified into four categories as follows:

1) Serial Input and Serial Output FIFO (SISOFIFO), each channel of SISOFIFO is independent, no parallel operation at read port or write port is permitted, i.e., the data of the SISOFIFO is written in channel by channel and read out channel by channel. there is no constraint for access among channels;

2) Serial Input and Parallel Output FIFO (SIPOFIFO), just as its name implies, data of the SIPOFIFO is written in channel by channel but read out at the same time;

3) Parallel Input and Serial Output FIFO (PISOFIFO), the data of the PISOFIFO is written in at the same time but read out channel by channel, namely, the write port should be able to be accessed in parallel;

4) Parallel Input and Parallel Output FIFO (PIPOFIFO), the data of the PIPOFIFO is written in simultaneously and read out simultaneously, in other words, both ports of PIPOFIFO should be able to be accessed in parallel.

## LITERATURE SURVEY

In the previous section we mentioned that NoC is composed of elements like FIFO’s (Virtual channels), Interconnects, Arbitration logic etc. The access mechanism in buffer structure plays an important role in the design of NoC applications. In turn, the mechanism influences how efficiently packets share link bandwidth. Buffers are used to store packets or flits when they cannot be forwarded right away onto the output port. Queuing buffers consume the most area and power among building blocks in the NoC [1, 2, 27]. We have investigated related work as follows:

### SHORT FIXED LENGTH QUEUE

In short fixed length queue, each physical channel of router has single queue. The arriving flit is stored at the tail of the queue. The flit present at head is read and forwarded to output channel through crossbar. The single queue buffer has fixed and short length, therefore, upstream router keeps track of free buffers. The flit is forwarded to downstream router only when a free buffer is available. The single queue buffer may lead to the problem of HoL blocking.

### MULTIPLE VARIABLE LENGTH QUEUES

The above discussed buffers are fixed in length. This may lead congestion in the network when there is abnormal traffic pattern. There may be many empty virtual channels that could not be utilized. This situation leads to poor buffer utilization and multi-channel FIFO is depicted in Figure 2. low throughput. To overcome this problem, a variable length multiple queues have been proposed. This mechanism provides better buffer utilization but at the cost of complex circuitry. The design needs a complex data structure to keep track of head and tail of queue.

## PROPOSED SYSTEM

We propose a new solution for the remaining three kinds of multi-channel FIFO to implement them in one Block RAM. As to SISOFIFO, the channels are independent and there are no timing constraints among them, so it is relatively easy to achieve. In the coming sections, we would concentrate our attention on the implementation of SIPOFIFO and PISIFIFO in one Block RAM, it can significantly improve utilization of Block RAM, reduce the cost of product and help to enhance market competitiveness.

BASIC FIFO From the analysis above, we have to implement multi-channel FIFO in one Block RAM and provide write or read operation in parallel to some extent. It's obviously impossible to realize write or read operation at the same time for multiple FIFOs that built in one Block RAM directly. In order to solve the problem of parallel access, it is necessary to place registers into the ports with function of parallel access as data buffer.

It is mainly composed of write control logic, DPRAM, read control logic and input/output buffering registers. The buffering registers are located at the parallel port, i.e., write port for PISOFIFO, and read port for SIPOFIFO. DPRAM can be instantiated from Block RAM with IP core generator.

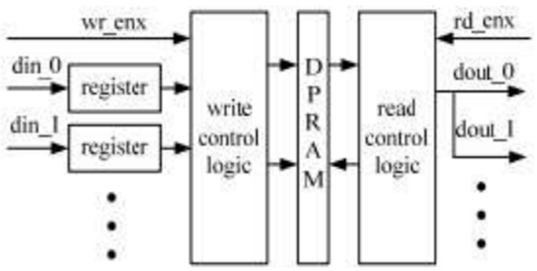
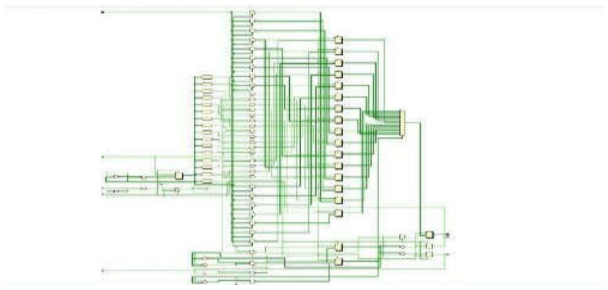


**Fig: General structure of multi-channel FIFO**



## FIFO AND XILINX

**Fig: RTL Diagram** The idea is to replace the oldest page in memory, just like you would replace the oldest item in a queue. In business and accounting, FIFO is used to calculate the cost of goods sold (COGS) and the value of inventory. According to the FIFO LUTs are essential to FPGA logic cells for Boolean operations.

**LUTRAM (LUT RAM):**

principle, the items that were acquired or produced first are assumed to be sold or used first. This method can have an impact on tax liabilities and financial reporting.

In networking, a FIFO buffer (First-In, First-Out buffer) is often used to manage the flow of data packets. Data packets are stored in the buffer in the order they arrive, and they are processed and sent out in the same order[8, 9]. FIFOs are also used in hardware design, especially in digital circuits. These FIFOs are used to manage data flow between different parts of a system, ensuring that data is handled in the order it was received. Overall, the FIFO principle is a straightforward and effective way to manage the order of items or data in various contexts, ensuring that the first item to arrive.

The hardware utilization statistic measures how much of the FPGA's resources a certain design is using. It is usually measured in LUTs, LUTRAMs, IOs, and BUFGs. Efficient FPGA designs need hardware optimization. Let me simply describe various hardware utilization components: Look-up table: LUTs are essential to FPGAs. Any combinatorial logic function may be implemented by this tiny, programmable memory device.

LUTRAM extends LUT to store data and execute logical operations. It's great for designing with little memory. Input/output (IO) refers to the FPGA's input/output pins. These pins let the FPGA connect to external components. Connecting an FPGA to the outside world requires efficient IO pin use. The BUFG (Buffer Element) ensures the clock signal is distributed across the FPGA. For dependable FPGA functioning and synchronization, BUFGs must manage and distribute clock signals. Reports and tools.

in boundary condition. The performance metrics of the buffer/FIFO are latency and throughput and data loss. The Latency is the time taken by the buffer to receive the data from instantiation and the time taken by the buffer to send the data to the output port. Latency may occur due to propagation delay of data or transmission delay from input port to output port of the buffer.

The throughput is the amount of data successfully transmitted in a given time period. This is computed based on the time period of write clock (Wclk_i) and the read clock (Rclk_i). The variation in these clocks like clock skew, clock drift and the clock distortion have an impact on wr_en_i and rd_en_i duration and variation in time period in storing and retrieving data in the buffer. Since writing and reading of data in the buffer/FIFO is depended on clocks

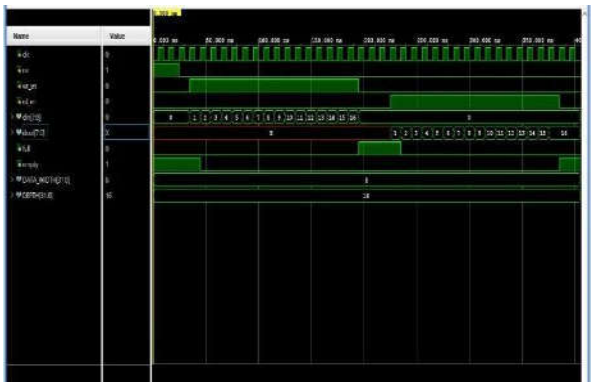


**Fig: simulation result**

## CONCLUSION

The performance and safety analysis is carried out by varying the frequency in inputs clocks (wr_clk_i and rd_clk_i) that is carried delayed clocks will lead to delay in the entire process of the module that impact on other modules connected to its output port. Data loss may occur with distorted clock pulses.

## FUTURE SCOPE

As CMOS dynamic random-access memory (DRAM) processes and voltages continue to scale, DRAM array access latency remains relatively constant. Faster logic resulting from process scaling is offset by timing latency associated with constant die size for new generation array densities. The benefits of process scaling are also countered by rising interconnect resistance and voltage scaling. All totaled, the benefits of process scaling are not fully realized and DRAM array access have remained fairly constant for sub-micron processes.

## REFERENCES

[1] S. Gandhare and B. Karthikeyan, "Survey on FPGA architecture and recent applications," in 2019 International Conference on Vision Towards Emerging Trends in Communication and Networking (ViTECoN), 2019: IEEE, pp. 1-4.

[2] B. Zhang, R. Kannan, and V. Prasanna, "Boostgcn: A framework for optimizing gcn inference on fpga," in 2021 IEEE 29th Annual International Symposium on Field-Programmable Custom Computing Machines (FCCM), 2021: IEEE, pp. 29-39.

[3] R. J. Hayne, "Translating the Instructional Processor from VHDL to Verilog," in 2018 ASEE Annual Conference & Exposition, 2018.

[4] T. Wang, C. Wang, X. Zhou, and H. Chen, "An overview of FPGA based deep learning accelerators: challenges and opportunities," in 2019 IEEE 21st International Conference on High Performance Computing and Communications; IEEE 17th International Conference on Smart City; IEEE 5th International Conference on Data Science and Systems (HPCC/SmartCity/DSS), 2019: IEEE, pp. 1674-1681.

[5] D. Koch, N. Dao, B. Healy, J. Yu, and A. Attwood, "FABulous: An embedded FPGA framework," in The 2021 ACM/SIGDA International Symposium on Field-Programmable Gate Arrays, 2021, pp. 45-56.

[6] G.-M. Sung, L.-F. Tung, H.-K. Wang, and J.-H. Lin, "USB transceiver with a serial interface engine and FIFO queue for efficient FPGA-to-FPGA communication," IEEE Access, vol. 8, pp. 69788-69799, 2020.

[7] V. Yadav, S. Kashyap, R. Pandey, and J. Madan, "Impact of Doping Variation on the Performance of Sb 2 S 3 based Solar Cell using Device Simulations," in 2023 IEEE Devices for Integrated Circuit (DevIC), 2023: IEEE, pp. 52-55.

[8] S. Parker et al., "Impact of FIFO work arrangements on the mental health and wellbeing of FIFO workers," 2018.