LOW POWER DFT IMPLEMENTATION FOR ASYNCHRONOUS FIFO

N. VINOD KUMAR[1], P. MADHU KUMAR[2], AVINASH YADLAPATI[3]

[1]M. Tech, Sree Vidyanikethan Engineering College, Tirupati, J N T University, Ananthapuram

[2]Assistant Professor, Dept.of Elect. and Comm. Engg, Sree Vidyanikethan Engineering College, Tirupati

[3]Senior director, Mirafra technologies, Hyderabad.

ABSTRACT:

Asynchronous FIFO is a memory file which uses synchronization for reading and writing with different clocks, by performing the conditions of overrun and underrun. In essence, the transfer of data from read domain to write domain with different frequencies. To generate overrun and underrun status flags the synchronization takes place with the help of "preceding operation" of both the write and read pointers. In this design the gray code converters are used to reduce switching activity and the low power DFT technique was applied by considering the two phases that is scan insertion and ATPG Simulations. This design is executed by using synthesizable Verilog RTL Code and verified with xilinx ISE simulator.

KEYWORDS: Asynchronous FIFO, synchronization, overrun, underrun, status flags, gray code converter, Design for test, RTL Code.

1. INTRODUCTION:

First in first out is a memory file which allows the data in a queue and it uses the synchronization for transferring the data. In this paper the low power DFT is implemented by using the asynchronous FIFO. FIFO can be differentiated in two ways and they are

1. Synchronous FIFO
2. Asynchronous FIFO

In the synchronous FIFO data is steered into and out of the memory array by two pointers i.e., read pointer and write pointer. After completion of each operation the particular pointer is incremented to allow access the next address pointer in the sequence of an array. By using the single clock the read and write operations are performed by using same frequency.

Where as in asynchronous FIFO two clocks are needed for read and write pointers both the pointers needed to be access with two separate clock frequencies. One clock is used for composing the data in to the memory and another clock for reading the data from the memory.

The concept in this paper is to reduce the power consumption using DFT (Design for testability) by implementing the novel architecture of asynchronous FIFO by synchronizing the read and write pointers and generating the overrun (FIFO_FULL) and underrun (FIFO_EMPTY) conditions with the help of "preceding operation".

2. SCHEMATIC SYMBOL:

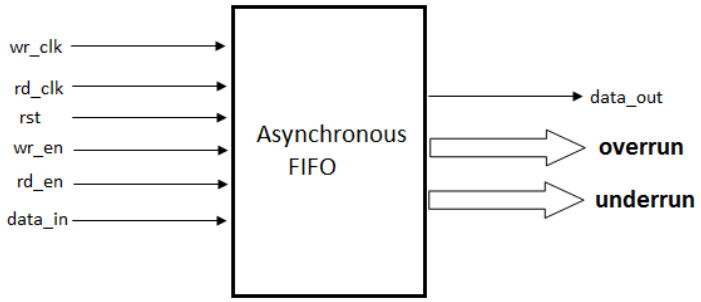

Figure 1: Schematic symbol of asynchronous FIFO

Above figure 1 shows the schematic figure of an Asynchronous FIFO and the input signals are shown below:

- wr_clk
- rd_clk
- Rst
- wr_en
- rd_en
- data_in

The output signals are:

- data_out
- Overrun
- Underrun

In asynchronous FIFO two clocks are used with different frequencies. wr_clk and rd_clk indicates to the write clock and read clock respectively. Whereas the write and read operations are performed by using those two clock signals. Both the read clock and write clocks are used for the write and read operations with different frequency. Rst indicating the resetting signal which is used to reset the FIFO to the known state. Write operation is formed if and only if wr_en is high. Similarly for Read operation also. Data_in is input signal used for writing the data into the memory. And Data_out is the output signal used for reading the data from memory. Here the two status flag generations i.e., Overrun and Underrun conditions which are proposed to prevent the overflow (writing the data inside the FIFO) condition that is requesting the write data after the completion of total address lines in the FIFO. Underrun is proposed to prevent the underflow (reading the data from the FIFO) condition that is requesting the read data after the FIFO is empty.

3. Implementation and Working of an Asynchronous FIFO:

This paper discusses the detail architecture of Asynchronous FIFO which is different from the other designs. Because the entire architecture was designed using only with the help of mux's and flip flops.

Before designing the FIFO, one should study and clearly understand the read and write pointers in a FIFO, the entire operation is dependent on the pointers itself. Here these pointers address memory locations. So, it is better to consider pointers and shift them accordingly instead of shifting the data that is to be written or read from one address location to another address location. So, there is a reduction in circuitry and complexity. This makes an efficient design.

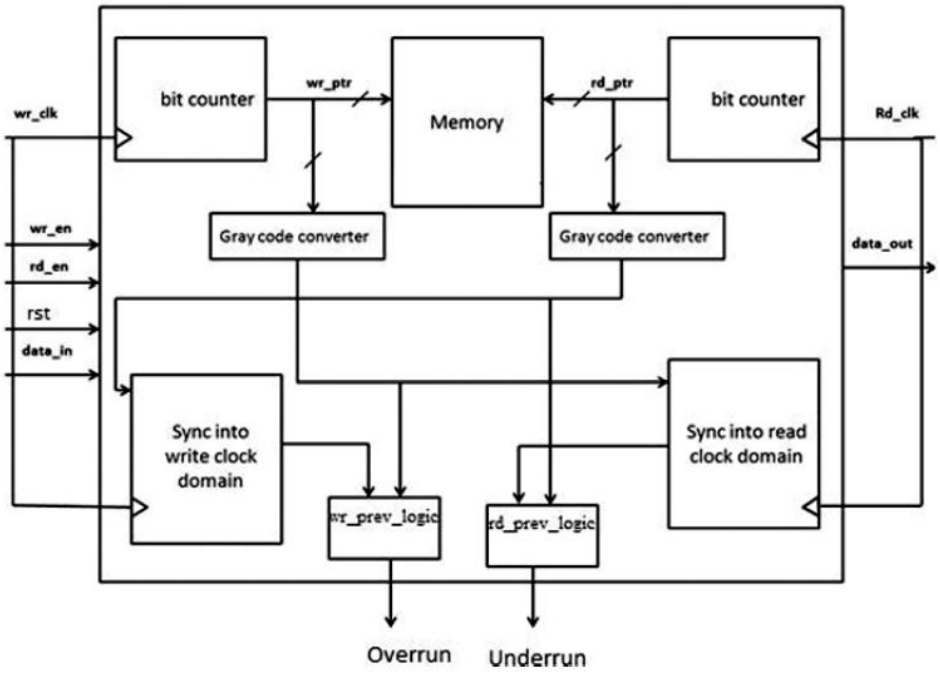

Figure 2: Architecture of an Asynchronous FIFO

Firstly the read and write pointers are placed at the initial positions of the address locations and then these pointers are incremented according to the assertion of read enable and write enable signals. In this paper the memory specified was "n" number of address locations and of m-bit wide data. After completion of read operation, the read pointer will immediately points to its next address location, similarly for write operation too. After addressing all the "n" number of address locations the pointer comes to its initial state depending upon the enable signals. If the data is continuously written into the memory and the performance of read operation is not occurred, then the write pointer will come to its initial state and will remain in the same state, even though the write enable signal is asserted. Similarly, it is for read pointer too.

- In Synchronous FIFO, the write and read operations will takes place only with the help of single clock with same frequency. The execution of read and write operations are sequential which means the occurrence of write operation follows read operation immediately. By performing this read and write operations continuously, The FIFO initially will be Empty at Reset and once the write happens in the location the read follows. So, always we have to reset the FIFO to read operation. Since, the read and write happens in the single clock and the transfer of data will be slow in the Synchronous FIFO.

- In Asynchronous FIFO for performing the read and write operations there will be two separate clocks are needed, those may be operated at different clock frequencies. In this case the transition of data will be faster with the synchronization of read and write clocks viz., will happen by using the two stage synchronizer, this is also called as Dual-flop synchronizer. Primarily the data synchronization in high speed data transfer takes place in Asynchronous FIFO.

3.1. Gray Code Converter:

Here the synchronization of read and write pointers will takes only with the help of "Gray Code Converter" and the other characteristics if Gray code converter will be comes to an importance when representing the successive binary numbers, for each binary increment there is reflection only with one bit change, thus reducing the switching action and hence power. This lower switching action accomplishes to less glitch formation and thus reducing any metastability.

DECIMAL	BINARY	GRAY
0	000	000
1	001	001
2	010	011
3	011	010
4	100	110
5	101	111
6	110	101
7	111	100

Figure 3: Gray code sequence conversion

When comparing the two pointers (representing 5-bit read and write address) reduction of metastability will comes to an importance, the possibility of interpreting a signal transition from 1 to 0 or 0 to 1 as 0's and 1's respectively. The number of transitions will be reduced by the combinational logic (xnor gate as an equivalency check) will become easier.

3.2. Synchronizer:

In general, the overrun and underrun conditions are generated based on the read and write positions pointers. Here the synchronizer circuit performs the mechanism of by reading the gray code write pointer with read clock domain and the gray code read pointer with write clock domain. Hence there is a need of synchronizing circuit and the above figure 4 depicts the synchronization mechanism.

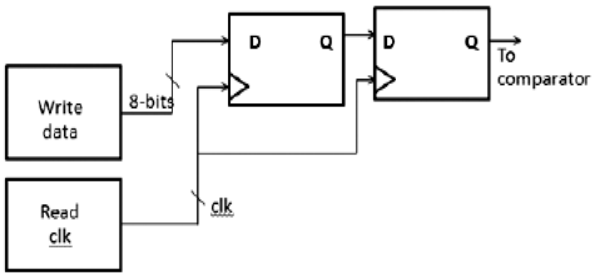

Figure 4: synchronizer circuit

Here a one bit data is synchronized using two flip flops. The read data is synchronized with write clock and write data is synchronized with read clock.

3.3. Generation of Overrun and Underrun Conditions:

- During the write operation, the first step is to check whether the FIFO is Full or not. If the FIFO is full, there is no scope of write operation and write pointer incrimination, this results the overrun condition. In other words there is a continues write operation and no read operation is performed then the overrun condition is occurred.

- During the read operation, the first step is to check whether the FIFO is EMPTY or not. If the FIFO is Empty, there is no scope of read operation and read pointer incrimination, this results the underrun condition. In other words there is a continues read operation and no write operation is performed then the underrun condition is occurred.

Figure 5: Overrun Status flag generation

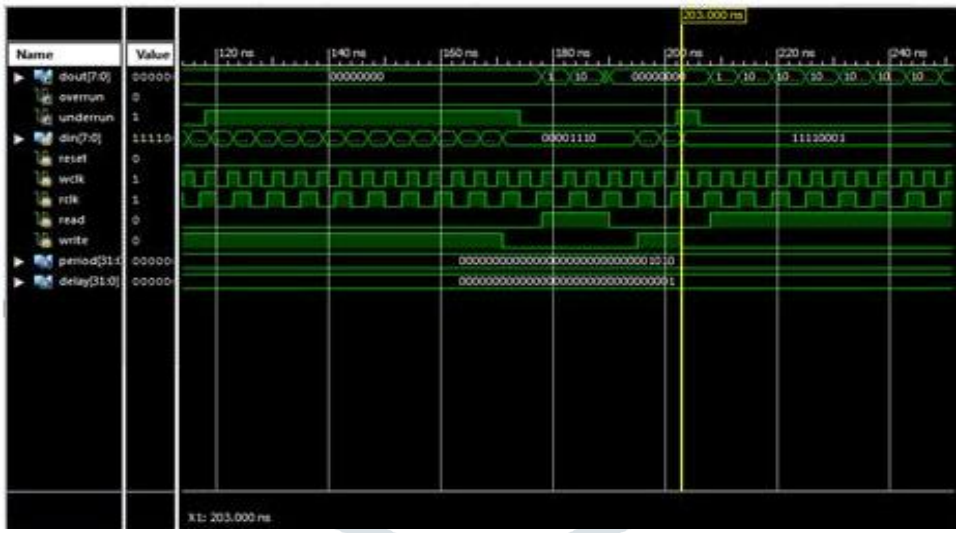

Figure 6: Underrun Status flag generation

From figure 5 and figure 6, it is clear that the overrun and underrun conditions are generated by last operation logic and the comparison of gray code pointers using a comparator with synchronization of read pointer with write clock and the write pointer with read clock pointer. And both the comparisons between read and write gray code pointers are done as follows:

- If the last operation is write, and both the pointers are equal. Then overrun status flag is asserted.
- If the last operation is read, and both the pointers are equal, Then underrun status flag is asserted

3.4. Scan Insertion and ATPG Simulations:

The Scan Insertion and ATPG Simulations are introduced out in the existing architecture, the paper will dividing the scan clock for even and odd scan chains and thereby reducing the power constraints using the Low Power DFT (design for test). The power constraints will be considering into three appropriate conditions. Firstly, the two techniques are applied to different locations in the implemented architecture

and thereby considering the double edge triggering method. Then the power will be reduced from previous existing to DET method to an extent of 0.00032W to 0.00027W. Whereas from DET method to considering the frequency by halving the total hierarchy power will be utterly diminishes to an greater extent of 0.00009

Device Utilization Summary		
Logic Utilization	**Used**	
Number of Slice Flip Flops	8	
Number of 4 input LUTs	23	
Logic Distribution		
Number of occupied Slices	12	
Number of Slices containing only related logic	12	
Number of Slices containing unrelated logic	0	
Total Number of 4 input LUTs	23	
Number of bonded IOBs	14	
Number of BUFGMUXs	1	

Table 1: device utilization summary

METHODS	OUTPUT POWER RESULTS	TOTAL HIERARCHY
EXISTING METHOD APPLIED WITH SCAN - INSERTION AND ATPG TECHNIQUE	Total Quiescent Power 0.081 Total Dynamic Power 0.013 Total Power 0.094	Name: Hierarchy total Power (W)*: 0.00032
CONSIDERING DOUBLE EDGE TRIGGERING	Total Quiescent Power 0.081 Total Dynamic Power 0.012 Total Power 0.094	Name: Hierarchy total Power (W)*: 0.00027
CONSIDERING FREQUENCY/2	Total Quiescent Power 0.081 Total Dynamic Power 0.000 Total Power 0.081	Name: Hierarchy total Power (W)*: 0.00009

Table 2: Power comparison by considering three techniques for proposed architecture

4. Conclusion:

Asynchronous FIFO considering overrun and underrun conditions was implemented using Xilinx ISE. This implementation was novel since overrun and underrun conditions helped in generating status flags with the help of a signal i.e., previous operation. The Gray code counter employees in uses a comparator along with the preceding operation for the generation of overrun and underrun status flags. Continuous writing of data into the memory, continuous reading of data from memory and simultaneous reading and writing from the memory are verified with different test cases. All these test cases are verified using Xilinx ISE Simulator. The work involves employing scan insertion and ATPG flow. The power analysis of the double edge triggered FIFO was done with reference to available Asynchronous FIFO with Scan insertion and ATPG. It was observed that 15.62% of power reduction. It was also compared by halving the frequency with reference to double edge triggered. It was observed that there was 66.6% of power reduction. Finally the power was reduced by 71.87%.

The future scope will be by implementing the design by using the existing techniques like BIST for the self-testing analysis and the reduction for power consumption the CPF (common power format) and the UPF(unified power format) will be preferred in the synopsis tool for better results.

REFERENCES:

[1] http://icslwebs.ee.ucla.edu/dejan/classwiki/images/9/97/Lec-15 _Multi-Vdd.pdf.
[2] http://www.engr.iupui.edu/~skoskie/ECE362/lecture_notes/LNB25_html/text12.html.
[3] https://www.techwalla.com/articles/difference-between-synchronous-and-asynchronous-data-transfer.
[4] http://citeseerx.ist.psu.edu/viewdoc/download?doi=10.1.1.519.8623&rep=rep1&type=pdf
[5] A Murali, K Hari Kishore, D Venkat Reddy, "Integrating FPGAs with Trigger Circuitry Core System Insertions for Observability in Debugging Process", Journal of Engineering and Applied Sciences, ISSN No: 1816-949X, Vol No.11, Issue No.12, December 2016, pp:2643-2650.
[6] https://www.researchgate.net/publication/224333768_A_robust_ultralow_power_asynchronous_FIFO_memory_with_self-adaptive_power_control.
[7] https://web.eic.nctu.edu.tw/lpsoc/courses/MS2017Spring/supplemental/20Subthreshold%20ASynchronous%20FIFO.pdf
[8] http://ijcsit.com/docs/Volume%205/vol5issue02/ijcsit20140502241.pdf.
[9] G. Ramesh, V. Shivraj Kumar, K. Jeevan Reddy," Asynchronous FIFO Design with Gray Code Pointer for High Speed AMBA AHB Compliant Memory controller", IOSR Journal of VLSI and signal processing (IOSR-JVSP), volume: 1, Issue 3, Nov-Dec 2012, pp: 32-37.
[10] https://www.xilinx.com/support/documentation/application_notes/xapp258.pdf
[11] https:// www.oocities.org/deepakgeorge2000/vlsi_book/Asynch1.pdf
[12]Mahesh Mudavath, K Hari Kishore, D Venkat Reddy, "Design of CMOS RF Front-End of Low Noise Amplifier for LTE System Applications Integrating FPGAs" Asian Journal of Information Technology, ISSN No: 1682-3915, Vol No.15, Issue No.20, December 2016 pp: 4040-4047.
[13] S Nazeer Hussain, K Hari Kishore "Computational Optimization of Placement and Routing using Genetic Algorithm" Indian Journal of Science and Technology, ISSN No: 0974-6846, Vol No.9, Issue No.47, December 2016, pp: 1-4.
[14] K Hari Kishore, K Akhil, G Viswanath, N Pavan Kumar " Design and Implement of 8X8 Multiplier using 4-2 Compressor and 5-2 Compressor", International Journal of Reconfigurable and Embedded Systems, ISSN 2089-4864, Volume 5, Number 3 , November 2016, pp. 131-135.
[15] P Bala Gopal, K Hari Kishore "An FPGA Implementation of On Chip UART Testing with BIST Techniques", International Journal of Reconfigurable and Embedded Systems, ISSN 2089-4864, Volume 5, Number 3 , November 2016, pp. 176-182.
[16] N Bala Gopal, K Hari Kishore "Analysis of Low Power Low Kickback Noise in Dynamic Comparators in Pacemakers" Indian Journal of Science and Technology, ISSN No: 0974-6846, Vol No.9, Issue No.44, November 2016 ,pp: 1-4.
[17] N Bala Gopal, Kakarla Hari Kishore "Reduction of Kickback Noise in Latched Comparators for Cardiac IMDs" Indian Journal of Science and Technology, ISSN No: 0974-6846, Vol No.9, Issue No.43, November 2016, pp: 1-6.
[18] Nidamanuri Sai Charan, Kakarla Hari Kishore "Recognization of Delay Faults in Cluster Based FPGA Using BIST" Indian Journal of Science and Technology, ISSN No: 0974-6846, Vol No.9, Issue No.28, July 2016, pp: 1-7.
[19] Avinash Yadlapati, Hari Kishore Kakarla "Validating Advanced Extensible Interface Protocol using Randomized Verification Environment" Research Journal of Applied Sciences, Engineering and Technology, ISSN No: 2040-7459, Vol No.13, Issue No.1, July 2016, page: 42-47.
[20]Sravya Kante, Hari Kishore Kakarla, Avinash Yadlapati," Design and Verification of AMBA AHB-Lite protocol using Verilog HDL" International Journal of Engineering and Technology, E-ISSN No: 0975-4024, Vol No.8, Issue No.2, April-May 2016, pp:734-741.
[21] N Bala Gopal, Kakarla Hari Kishore "Reduction of Kickback Noise in Latched Comparators for Cardiac IMDs" Indian Journal of Science and Technology, ISSN No: 0974-6846, Vol No.9, Issue No.43, November 2016, pp: 1-6.
[22] S Nazeer Hussain, K Hari Kishore "Computational Optimization of Placement and Routing Using Genetic Algorithm" Indian Journal of Science and Technology, ISSN No: 0974-6846, Vol No.9, Issue No.47, December 2016, pp: 1-4.