International Journal of Electrical and Computer Engineering (IJECE)
Vol. 11, No. 6, December 2021, pp. 4785~4793
ISSN: 2088-8708, DOI: 10.11591/ijece.v11i6.pp4785-4793

Design and analysis of multiple read port techniques using bank division with XOR method for multi-ported-memory on FPGA platform

Druva Kumar S., Roopa M.
Department of Electronics and Communication Engineering, Dayananda Sagar College of Engineering, India

Article Info

Article history:

Received Oct 20, 2020
Revised May 25, 2021
Accepted Jun 12, 2021

Keywords:

2R1W/4R
BDX approach
FPGA
Multi-port memory
Read ports

ABSTRACT

The multiple read and write operations are performed simultaneously by multi-ported memories and are used in advanced digital design applications on reprogrammable field-programmable gate arrays (FPGAs) to achieve higher bandwidth. The Memory modules are configured by block RAM (BRAMs), which utilizes more area and power on FPGA. In this manuscript, the techniques to increase the read ports for multi-ported memory modules are designed using the bank division with XOR (BDX) approach. The read port techniques like two read-one write (2R1W) memory, hybrid mode approach either 2R1W or 4R memory, and hierarchical BDX (HBDX) Approach using 2R1W/4R memory are designed on FPGA platform. The Proposed work utilizes only slices and look-up table (LUT's) rather than BRAMs while designing the memory modules on FPGA, which reduces the computational complexity and improves the system performance. The experimental results are analyzed on Artix-7 FPGA. The performance parameters like slices, LUT utilization, maximum frequency (Fmax), and hardware efficiency are analyzed by concerning different memory depths. The 4R1W memory design using the HBDX approach utilizes 4% slices and works at 449.697 MHz operating frequency on Artix-7 FPGA. The proposed work provides a better platform to choose the proper read port technique to design an efficient modular multiport memory architecture.

Corresponding Author:

Druva Kumar S.
Department of Electronics and Communication Engineering
Dayananda Sagar College of Engineering
Shavige Malleshwara Hills, KS Layout, Bengaluru 560 078, KA, India
Email: sdktronix@gmail.com

1. INTRODUCTION

Multi-ported memories build high-performance processor designs. In earlier years, the multi-ported RAMs are used in superscalar, vector processors, multicore processors, digital signal processors (DSPs), and later in field-programmable gate arrays (FPGAs). The registers and logic elements are used to build the multi-ported RAMs but applicable for small memories. The Large and complex systems are built on a single on-chip, which needs many operations like message quieting, computation nodes, sharing access, synchronization between processing elements on FPGA. The multi-ported register files and FIFO's are used to increase read and write memory banks simultaneously and process speed from the multiple ports. The FIFO's and register files are not efficient in implementing multi-ported memories on FPGA [1], [2].

The soft processors and hardware accelerators-based FPGAs use mainly multi-ported memories to increasing system performance. The FPGA-based soft processors like RISC processors exploit the

instruction-level parallelism (ILP). The reason is that the resource provides by the FPGA's are inefficient to use multi-ported memories. There are many approaches available in the existing survey to implement the multi-ported memory modules like adaptive logic modules (ALM) based, Replication based, multi-pumping based, line value table (LVT), and banking based approach's [3]-[5]. The multi-ported memory controller (MPMC) is used to transfer a large amount of data by providing storage resource sharing on time. The FIFO-based intellectual property (IP) cores and native port interface are used as memory modules, which are available default in FPGA with customization [6]. The multi-ported memory connects many processing elements (PEs) like processors using simultaneous access and exclusive access based on multi-port RAM modules [7]. The external synchronous dynamic RAM (SDRAM) controller module is connected to the FPGA system. It has a multi-port port design with clock synchronization to improve system performance, interface design for SDRAM and FPGA interconnection, finite state machine logic for data access, and arbitration logic module to provide proper priority to each port of SDRAM. The multi-ported SDRAM provides high-speed data access capability by utilizes adequate bandwidth on FPGA and is used in many real-time applications like image/video processing, data acquisition system, and many more [8]-[12].

The replication technique is one of the conventional methods of the read-port technique, which offers multiple read-ports by duplicating the data values on multiple-block RAM (BRAMs). The replication technique [3] provides an easy-to-design feature and less complex control logic incorporation. The replication technique utilizes more BRAMs while reading the data, affecting system performance and degradation of computational complexity. These problems are overcome using the proposed approach. The proposed multi-port read techniques using the BDX approach offer multiple-read data using simple XOR, higher operating frequencies, and only slice utilization rather than BRAMs on FPGA.

The significant contribution of the proposed work is highlighted as follows: i) The different Read port techniques are designed using the BDX approach; ii) The proposed design offers less slice area and operates at higher frequencies for reading port techniques on FPGA; iii) The designed read port modules utilize only slices and look-up table (LUTs) rather than BRAMs; iv) The proposed work offers BRAM-less than existing approaches that provide better system performance and less computational complexity in FPGA devices; v) The user can choose the proper read-port technique to continue further to design efficient multi-port memory architecture.

Section 1 describes the review of existing read port techniques and multi-port memories designs on FPGA's. The different read-port techniques using the BDX approach are explained in section 2. Section 3 discusses the experimental results and performance analysis of different read port techniques. Finally, it concludes the overall work with design constraints analysis in section 4.

This section provides a brief overview of existing read-port techniques and different multi-port memory methods on FPGA's. Laforest *et al.* [13] present FPGA-based Multi-ported Memories using the XOR approach, which uses less chip area and utilizes more BRAMs. The XOR-based multi-read and write memory module is implemented using BRAMs, which is relatively better than the line value table (LVT) based designs. The XOR-based approach provides better operating frequency, less utilization of BRAMs, and logic modules than LVT based Approach. Laforest *et al.* [14] describe the soft-multi ported memories on FPGA for conventional solutions. The LVT and XOR-based approaches are improved and evaluate the performance parameters on different FPGA vendors like Altera and Xilinx. Yantir *et al.* [15] describe the shift register-based multi-pumping Approach to overcome the conventional replication and banking-based approaches' drawbacks. The register file-based multi-pumping Approach supports both single and multiple-port numbers for write and reads operations. The multiplexor and shift register-based multi-pumping approaches are analyzed with different performance parameters on FPGA. The Heterogeneous register file-based complete system on a single FPGA chip is designed by Yantir *et al.* [16]. The Heterogeneous register file mainly contains a base register file connected with single and multi-port multi-pumping units followed by processing elements. The Heterogeneous register file-based module improves the frequency and energy consumption resources and reduces the system complexity.

Lin *et al.* [17] present BRAM based multi-port memory design, which supports multiple reads using the XOR-Bank method, and multiple-write using a remap table-based approach. The multiple read technique reduces around 37.5% BRAM utilization than existing replication methods; similarly, the multiple write method reduces around 25% BRAM utilization than the existing LVT-based approach. Nguyen *et al.* [18] explain the configurable multi-port memory controller (MPMC), which offers parallel and pipelining supports for multimedia applications. The MPMC mainly contains frontend-Slave port and configurable modules along with Arbiter modules. The Arbiter module supports both the read and writes to process. The bandwidth utilization of MPMC write and read processes are analyzed with existing similar Memory controller approaches and achieved with better improvements. Abdelhadi *et al.* [19] presented the optimization of data banks and adopted the same in-memory compiler to recover set cover issues using dual-port BRAMs. The data banks are designed using a data-flow graph (DFG) to optimize the switching portions.

The dual-port BRAM's solves the covering problems and analyzes the data dependencies. The results are highlighted for multi-switched write and read for each port by concerning the frequency, BRAM, and logic module utilization. Navid *et al.* [20] explain the embedded-based multi-ported memory using the decision-making module (DMM) for future generation FPGA's. The read/write operations in DMM-based memory are straightforward and easy to evaluate. The DMM-based memory module is compared with XOR, and LVT approaches with better constraints on FPGA.

Charles and Lin [21], Charles and Huang [22] hierarchical banking approaches for multi-ported memory modules improve the design resources like BRAMs and frequency parameters. The hierarchical banking approaches offer efficient utilization of BRAMs, routing complexity reduction for large designs, memory module reduction in Table based approaches, and improved scalable features. Ullah *et al.* [23] discuss the FPGA realization of TCAM based Multiported SRAM approach with multipumping features. First, partition the conventional TCAM table, Build the TCAM Memory and its architecture, and finally, analyze the effect of the multipumping SRAM on hardware resources. Kwan *et al.* [24] present the lossy multi-port memory (LMM) module, which contains memory banks connected parallel into local ring buffers and input ports by Lossy switch networks. The LMM offers better resource utilization and throughput than NoC based systems and LMM4. Jain *et al.* [25] present the single-port memory module using suitable methods and achieves multi-port memory performance. The coding methods are included for memory banks and multi-port memory emulation with best and worst cases to improve memory access. Shahrouzi *et al.* [26] describe the bi-directional multi-ported memories with optimization for future generation FPGA systems. The work provides bi-directional features to multiple- write and read operations simultaneously. The design uses multiple BRAMs for multi-directional multi-port memory designs.

2. READ PORT TECHNIQUES USING BDX APPROACH

The Different read port techniques using the bank division with XOR (BDX) approach are designed and discussed in this section. The overview of the multiple read-port designs using the BDX approach is shown in Figure 1. First, define the five Individual Memory modules: four for memory banks (MB) and one for XOR bank (XB) for designing the multiple read-port techniques. The memory modules Data_width and memory depth are configurable. So the Data_width can be set either 8-bit, 16bit, 32-bit, or 64-bit.

Similarly, the memory depth would be 8 to 8k depth. In this design, Data_width is set to 8-bit and memory depth varied from 8 to 512 depth. The Artix-7 FPGA is used for implementation, which takes more time to synthesis if memory depth exceeds 512. All the memory modules (4-MB and 1-XB) are initialized to zero. The address generation for five individual memory modules is achieved using the counter, which counts till the last address location of memory depth. They write the input data into any of the four memory banks (MB). Update the XOR bank by XORing the four memory bank's data locations-wise. Apply the BDX approach to read the multiple-read port data using memory modules.

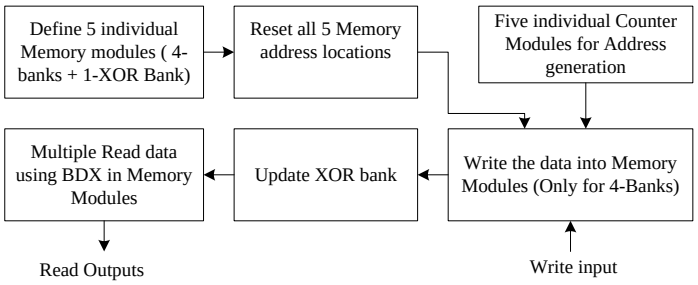

Figure 1. Overview of the multiple read-port designs using BDX approach

Different read port techniques are designed; namely, 2R1W module, 2R1W mode, 4R mode, and hierarchical BDX (HBDX) based 4R1W using 2R1W/4R mode. The different read-port techniques are explained in the following subsections as follows.

2.1. 2R1W memory design

The first read-port technique is divided into two modules: 2R1W memory module (generic) and multiple read ports generations using 2R1W memory modules. The BDX approach increases the read port and is different from the XOR-based approach [13]. The XOR-based approach is used to increase the write

port data from different memory modules, whereas the BDX approach increases the read-port data from different memory banks. An example of 2R1W memory design (generic) using the BDX approach is shown in Figure 2. It mainly contains five memory modules, like four MB and one XB. The memory locations are divided into four memory banks (MB0-MB3), and XOR Bank (XB) is used to add the XOR data values of MBs. The single MB supports two reads and one write (2R1W) simultaneously. Each MB supports memory depth from 0 to N-1. Where 'N' is the memory depth maximum value. The address of the memory locations (WD) is generated using the counter method. Firstly, reset all the memory locations from all the banks. Write the data into the particular memory locations depends on the user's interest.

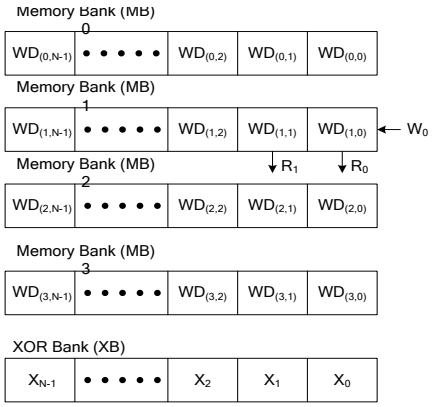

Figure 2. 2R1W memory design using BDX approach example

In this example, the data is written (W_0) into the First memory bank (MB1) of the 0^{th} memory location (WD (1, 0)). The XOR bank (XB) is also a memory module, where the data is stored in each memory location is done by XORing data values of MB's. In general, The XOR bank updation for X_0 and $X_{(N-1)}$ is represented in (1) and (2), respectively.

$$X_0 = WD_{(0,0)} \oplus WD_{(1,0)} \oplus WD_{(2,0)} \oplus WD_{(3,0)} \quad (1)$$

$$X_{(N-1)} = WD_{(0,N-1)} \oplus WD_{(1,N-1)} \oplus WD_{(2,N-1)} \oplus WD_{(3,N-1)} \quad (2)$$

The data value is read from memory locations (WD) of MBs. In this example, two read outputs (R_0 and R_1) of the 2R1W memory module is represented in (3) and (4).

$$R_0 = WD_{(1,0)} \quad (3)$$

$$R_1 = WD_{(0,1)} \oplus WD_{(2,1)} \oplus WD_{(3,1)} \oplus X_1 \quad (4)$$

The memory location of WD (1, 0) data accessed directly from the first Read output (R_0), whereas the second read output (R_1) cannot access them directly due to memory bank conflicts. So the second Read output R_1 has recovered the data value by XORing all the banks (MB-0, 2, 3,) and XOR bank of 1^{st} memory location except MB1. In general, if the data is targeted from First Memory bank (MB1), any of the memory locations is represented in (5).

$$WD_{(1,N-1)} = WD_{(0,N-1)} \oplus WD_{(2,N-1)} \oplus WD_{(3,N-1)} \oplus X_{N-1} \quad (5)$$

The multiple read ports are designed using several 2R1W memory modules are represented in Figure 3. The write data (W_0) is common to all the 2R1W memory modules. Each 2R1W memory supports only two reads output ports out of k read ports. All the 2R1W modules work simultaneously to generate the k read port outputs. This method has to write conflict and duplication of memory module issues, which consumes more area utilization on FPGA and degrades the system performance. So The HBDX-approach is used to resolve the above issues and is explained in section 2.3.

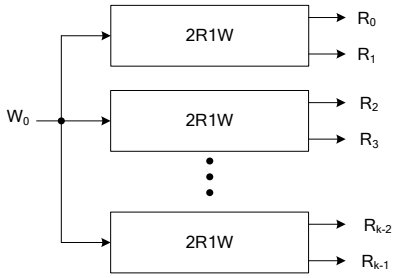

Figure 3. Example design of kR1W memory using multiple 2R1W memory modules

2.2. 2R1W/4R mode memory designs

The two-mode (2R1W/4R) architecture is designed either using a 2R1W or 4R memory module. The two-mode (2R1W/4R) architecture is considered a hybrid approach to implement the HBDX memory module. The two-mode (2R1W/4R) architecture works the same as the 2R1W memory module, as discussed in earlier section III-A. The 2R1W Mode is shown in Figure 4(a), which contains four MBs and one XB. The write data (W0) is stored into a particular target address location and generates the corresponding read data R0 from the exact memory location of MB0.

Similarly, if data is written into any MB-1, 2, 3, and XB, the corresponding read data (R_1) is generated from the particular memory locations. The read update (R_u) updates the XB by reading all other MBs of the exact memory locations. The 4R mode is shown in Figure 4(b). The 4R mode is activated only when there is no write data input or request. The 4R mode works only for four read output ports. In diagram 4(b) shows that The MB0 access two read outputs (R_0 and R_2) directly; simultaneously, the other MBs (1-3) and XB read the particular data values of the exact memory location by XOR operation to generate the R_1 and R_3.

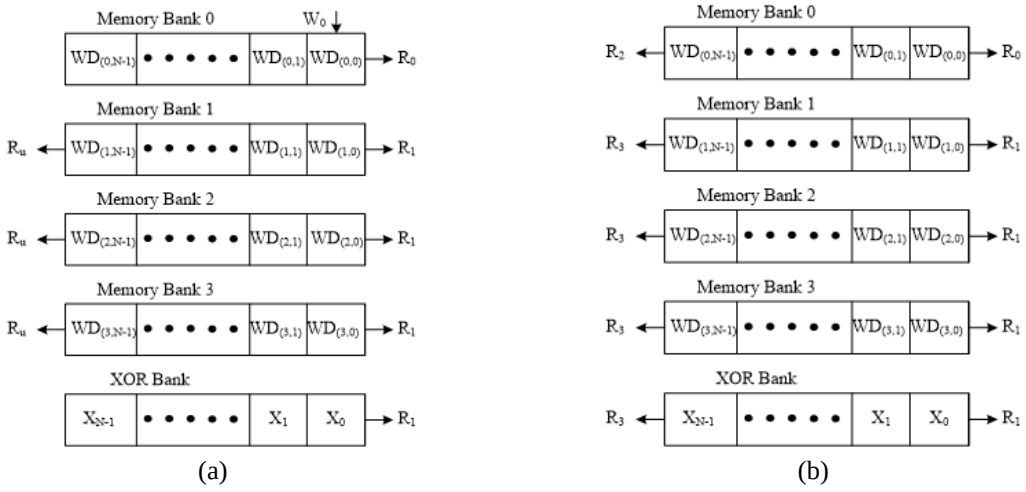

Figure 4. The two-mode (2R1W/4R) architecture; (a) 2R1W mode, (b) 4R mode memory module designs

2.3. 4R1W memory design using HBDX approach

The hierarchical-BDX (HBDX) approach for 4R1W is designed using multiple 2R1W/4R mode memory modules are shown in Figure 5. The HBDX offers cost-effective solutions for multiple read-ports and significantly reduces the area utilization on FPGA. Consider the 4R and 1W ports that want to access the MB0 memory locations as one of the worst-case scenarios, in that MB0 is used as 2R1W Mode, and other banks (MB1-MB3, XB) are considered as 4R Mode. The data request is written (W_0) into the MB0 of target memory locations. Like R_0 and R_1, the two read-port outputs read the data from the target memory locations of MB0. Like MB1-MB3 and XB, the other banks read the exact target memory locations using XOR operation to generate R_2 and R_3 reading port outputs. The read update (R_u) is updated by XORing the remaining read ports of MB's of the target memory location.

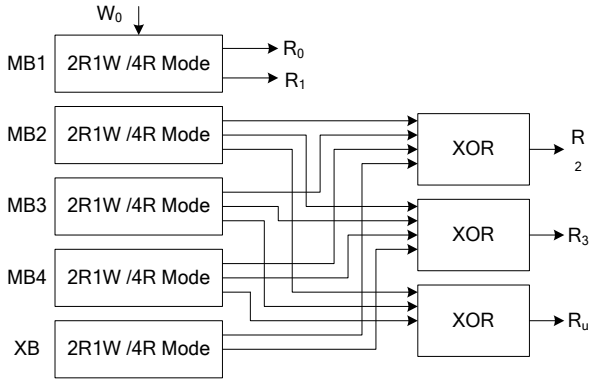

Figure 5. 4R1W memory design using HBDX approach

3. RESULTS AND DISCUSSION

This section provides an experimental analysis for different read port techniques on Artix-7 FPGA. The design provides different performance metrics by concerning different memory depths for each read port technique. All the read port techniques are designed for 8-bit data width with variable memory depth. The designs are configurable to more data width like 32-bit, 64-bit, and more. The read port techniques are designed and synthesized on Artix-7 (XC7A100T) FPGA. The designs are implemented on Xilinx-ISE 14.7 environment using verilog-HDL and also simulated using Modelsim-6.5 simulator. The Artix-7 FPGA Device supports 15850 slices, 101,440 logic cells, 270 block RAM (18Kb), and 240 DSP slices [27]. Each slice contains a minimum of four 6-input LUT's and eight flip-flops (FFs) on Artix-7 FPGA.

The read port memory modules are designed using the BDX approach, which utilizes only Slices, LUT's and offers BRAMless architectures. The performance parameter includes slices, LUT's, maximum operating frequency (Fmax), and Hardware efficiency (Fmax/Slice) are analyzed for different read port techniques and are tabulated in Table 1.

Table 1. Performance analysis of different read port techniques

Resources	Memory Depth	Only BDX_2R1W	BDX_2R1W Mode	BDX_4R Mode	HBDX_4R1W
Slices	256	32	4106	521	2826
	512	34	12292	1511	6121
LUT's	256	27	5297	1091	4684
	512	29	15540	2156	8535
Max. Frequency (Fmax)	256	492.15	486.029	469.091	479.54
	512	480.354	381.804	451.867	449.697
Fmax/Slice	256	15.37	0.118	0.9	0.17
	512	14.12	0.032	0.299	0.074

All the read port designs using Artix-7 FPGA supports till memory depth 512. If the memory depth is >512 and higher, the Artix-7 FPGA consumes more time to synthesize for read-port designs individually. The Area (Slices) v/s Memory depth of Different Read Port techniques is represented in Figure 6. The memory depth varied from 8 to 512 for different read-port designs. The BDX_2R1W utilizes <1% slices for 8 to 512 memory depth. The BDX_2R1W mode utilizes <1% slices from 8 to 64 memory depth, 1% slices for 128 memory depth, 3% slices for 256 memory depth and 9% slices for 512 memory depth. Similarly, BDX_4R mode utilizes <1 % slices for 8 to 256 memory depth and 1% slices for 512 memory depth. The HBDX_4R1W module utilizes <1 % slices for 8 to 128 memory depth, 2 % and 4% slices utilization for 256 and 512 memory depth, respectively, on Artix-7 FPGA.

The area (LUT's) v/s memory depth of different read port techniques is represented in Figure 7. The BDX_2R1W utilizes <1 % LUT's for 8 to 512 memory depth. The BDX_2R1W mode utilizes <1% LUT's for 8 to 64 memory depth, 4%, 8%, and 24% LUT's for 128, 256, and 512 memory depth respectively. Similarly, BDX_4R mode utilizes <1 % LUT's for 8 to 128 memory depth, 1% and 3% LUT's for 256 and 512 memory depth respectively. The HBDX_4R1W module utilizes <1 % LUT's for 8 to 64 memory depth, 2%, 7% and 13% LUT's utilization for 128, 256 and 512 memory depth respectively on Artix-7 FPGA. The area (Slices/LUT's) is directly proportional to the memory depth. If the different read port designs' memory depth increases, the Area utilization also increases on Artix-7 FPGA.

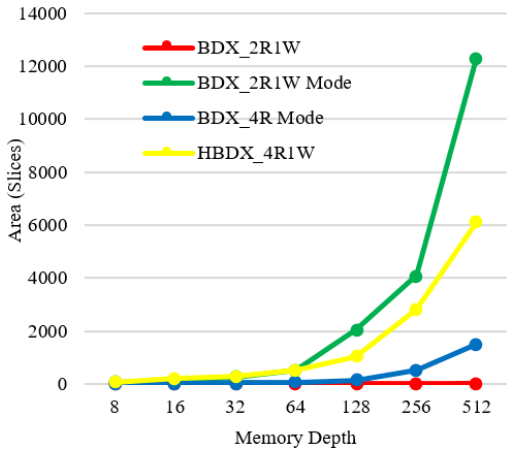

Figure 6. Area (slices) v/s memory depth of different read port technique

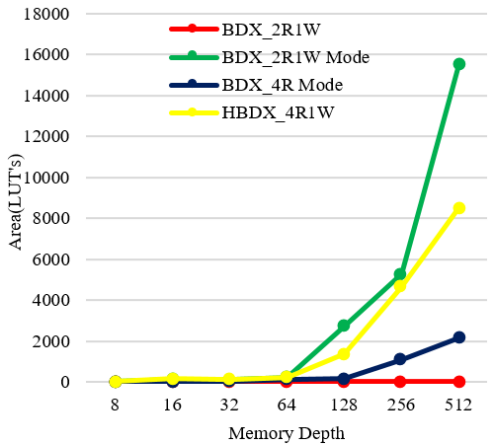

Figure 7. Area (LUT's) v/s memory depth of different read port techniques

The maximum operating frequency (Fmax) v/s memory depth of different read port techniques is represented in Figure 8. The frequency (Fmax) is inversely proportional to the memory depth. If the different read port designs' memory depth increases, the frequency (Fmax) decreases on Artix-7 FPGA. The frequency (Fmax) value differ based on the clock utilization in different read port designs. The BDX_2R1W Mode operates at less Fmax than other read port designs. Because the BDX_2R1W Mode checks all the possible memory locations from the 5 five Banks (four MB and one XB) to read the data and utilizes more FPGA areas. The Fmax of BDX_2R1W Mode degraded around 20% than the other designed read port techniques. The hardware efficiency is calculated using operating frequency (Fmax) and Slice utilization on Artix-7 FPGA. The (Fmax/Slice) v/s memory depth of different read port techniques is represented in Figure 9.

The Fmax is decreasing drastically by increasing the memory depth of reading port techniques. The BDX_2R1W module provides better hardware than other read port techniques. The HBDX_4R1W module provides 0.17 and 0.074 efficiencies for 256 and 512 memory depth, respectively. The hardware efficiency provides benefits to the users to choose better read port designs for different multi-port designs.

Figure 8. Max. frequency v/s memory depth of different read port techniques

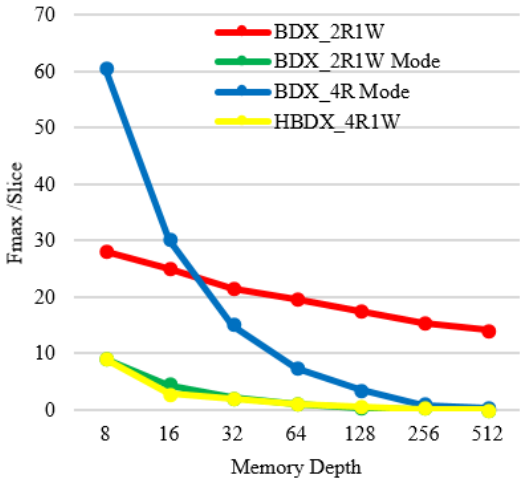

Figure 9. (Fmax/slice) v/s memory depth of different read port techniques

4. CONCLUSION

In this article, efficient multiple read-port techniques are designed using the BDX approach for modular multi-ported memories. The proposed read-port techniques offer BRAMless architectures to improve the Chip area, performance and reduce hardware complexity. The BDX approach is used to increase

the read port data from different data banks rather than increase the write port data. The different read-port techniques like 2R1W, Hybrid (2R1W/4R) mode, and 4R1W hierarchical BDX (HBDX) based approaches are designed and discussed in detail. The proposed work utilizes only slices and LUT's on FPGA rather than BRAMs, which improves the system performance and provides high operating frequencies. The different hardware resource parameters like Slices, frequency, and efficiency are analyzed by concerning the different memory depths on Artix-7 FPGA. The 2R1W utilizes <1% slices and operates at 480.35MHz, whereas 2R1W Mode utilizes 9% slices and operates at 381.80 MHz, 4R Mode utilizes 1% slices and operates at 451.86 MHz, and 4R1W using HBDX utilizes 4% slices and operates at 449.69 MHz for 512 memory depth on Artix-7 FPGA. The hardware efficiency (Fmax/slice) is relatively better for reading port techniques like BDX_2R1W Mode, BDX_4R, and HBDX by obtaining 0.032, 0.299, 0.074, respectively, for memory depth 512. In the future, the proposed read-port techniques are used to design an optimized Multi-port memory module by incorporating suitable write-port techniques.

REFERENCES

[1] A. Abdelhadi, and G. G. F. Lemieux, "Modular switched multi-ported SRAM-based memories," *ACM Transactions on Reconfigurable Technology and Systems (TRETS)*, vol. 9, no. 3, pp. 1-26. 2016, doi: 10.1145/2851506.

[2] H. Trang, "Design Choices of Multi-Ported Memory for FPGA, "*Special Information and Communication Technology edition (JICT*), no. 2, 2013.

[3] C. E. LaForest, and J. G. Steffan, "Efficient multi-ported memories for FPGAs," *Proceedings of the 18th annual ACM/SIGDA international symposium on Field programmable gate arrays*, 2010, pp. 41-50, doi: 10.1145/1723112.1723122.

[4] M. G. Alp, H. E. Yantir, A. Yurdakul, and Smail Niar, "Application-specific multi-port memory customization in FPGA," *2014 24th International Conference on Field Programmable Logic and Applications (FPL)*, 2014, pp. 1-4, doi: 10.1109/FPL.2014.6927426.

[5] K. R. Townsend, O. G. Attia, P. H. Jones, and J. Zambren, "A scalable unsegmented multi-port memory for FPGA-based systems," *International Journal of Reconfigurable Computing*, vol. 2015, pp. 1-11, 2015, Art. no. 11, doi: 10.1155/2015/826283.

[6] D. Sarika, and R. A. Pagare, "FPGA-based data transfer using the multi-port memory controller," In *International Conference for Convergence for Technology-2014*, 2014, pp. 1-3, doi: 10.1109/I2CT.2014.7092230.

[7] S. Elio, and A. Trifiletti, "A shared memory, parameterized and configurable in FPGA, for use in multiprocessor systems," *2016 MIXDES-23rd International Conference Mixed Design of Integrated Circuits and Systems*, 2016, pp. 443-447, 2016, doi: 10.1109/MIXDES.2016.7529783.

[8] L. Yang, and G. Liu, "Implementation of multi-port memory access arbitration logic in high-speed image system," *2015 8th International Congress on Image and Signal Processing (CISP)*, 2015, pp. 949-953, doi: 10.1109/CISP.2015.7408015.

[9] M. Shared, T. T. Mutlugun, and W. S.-De, "OpenCL computing on FPGA using multi-ported," *2015 25th International Conference on Field Programmable Logic and Applications (FPL)*, 2015, pp. 1-4, doi: 10.1109/FPL.2015.7293983.

[10] W. Jianmin, Y. Zhang, J. Zhou, P. Jia, and X. He, "Realization of Multi-port SDRAM Controller in LXI Data Acquisition System," *2013 International Conference on Information Science and Cloud Computing Companion*, 2013, pp. 515-520, doi: 10.1109/ISCC-C.2013.105.

[11] A. El Ansari, A. Mansouri, and A. Ahaitouf, "An energy-aware system-on-chip architecture for intra prediction in HEVC standard," *International Journal of Electrical and Computer Engineering (IJECE*), vol. 9, no. 6, pp. 5084-5094, 2019, doi: 10.11591/ijece.v9i6.pp5084-5094.

[12] J. A. Zakwan, I. N. Mahzan, A. S. R. A Subki, and W. H. Wan Hassan, "Improve the performance of the digital sinusoidal generator in FPGA by memory usage optimization," *International Journal of Electrical & Computer Engineering (IJECE)*, vol. 9, no. 3, pp. 1742-1749, 2019, doi: 10.11591/ijece.v9i3.pp1742-1749.

[13] C. E. LaForest, M. G. Liu, E. R. Rapati, and J. G. Steffan, "Multi-ported memories for FPGAs via XOR," *FPGA 2012-Proceedings of the ACM/SIGDA international symposium on Field Programmable Gate Arrays*, 2012, pp. 209-218, doi: 10.1145/2145694.2145730.

[14] C. E. Laforest, Z. Li, T. O'rourke, M. G. Liu, and J. G. Steffan, "Composing multi-ported memories on FPGAs," *ACM Transactions on Reconfigurable Technology and Systems (TRETS)*, vol. 7, no. 3, pp. 1-23, 2014, Art. no. 16, doi: 10.1145/2629629.

[15] Y. H. Erdem, S. Bayar, and A. Yurdakul, "Efficient implementations of multi-pumped multi-port register files in FPGAs," *2013 Euromicro Conference on Digital System Design*, 2013, pp. 185-192, doi: 10.1109/DSD.2013.28.

[16] Y. H. Erdem, and A. Yurdakul, "An efficient heterogeneous register file implementation for FPGAs," *2014 IEEE International Parallel & Distributed Processing Symposium Workshops*, 2014, pp. 293-298, doi: 10.1109/IPDPSW.2014.40.

[17] L. J.-Liang, and L. Bo-Cheng Charles, "BRAM efficient multi-ported memory on FPGA," *VLSI Design, Automation, and Test (VLSI-DAT)*, 2015, pp. 1-4, doi: 10.1109/VLSI-DAT.2015.7114526.

[18] N. X.-Thuan, H.-T. Nguyen, and P. C.-Kha, "Parallel pipelining configurable multi-port memory controller for multimedia applications," *2015 IEEE International Symposium on Circuits and Systems (ISCAS)*, 2015, pp. 2908-2911, doi: 10.1109/ISCAS.2015.7169295.

[19] A. M. S. Abdelhadi, and G. G. F. Lemieux, "A multi-ported memory compiler is utilizing true dual-port BRAMs," *2016 IEEE 24th Annual International Symposium on Field-Programmable Custom Computing Machines (FCCM)*, 2016, pp. 140-147, doi: 10.1109/FCCM.2016.45.
[20] S. S. Navid, and D. G. Perera, "An efficient embedded multi-ported memory architecture for next-generation FPGAs," *2017 IEEE 28th International Conference on Application-specific Systems, Architectures and Processors (ASAP)*, 2017, pp. 83-90, doi: 10.1109/ASAP.2017.7995263.
[21] L. Bo-C. Charles, and J.-L. Lin, "Efficient designs of multi-ported memory on FPGA," *IEEE Transactions on Very Large Scale Integration (VLSI) Systems*, vol. 25, no. 1, 139-150, 2016, doi: 10.1109/TVLSI.2016.2568579.
[22] L. Bo-C. Charles, and K.-H. Huang, "An efficient hierarchical banking structure for algorithmic multi-ported memory on FPGA," *IEEE Transactions on Very Large Scale Integration (VLSI) Systems*, vol. 25, no. 10, pp. 2776-2788, 2017, doi: 10.1109/TVLSI.2017.2717448.
[23] U. Inayat, Z. Ullah, and J.-A. Lee, "Efficient TCAM design based on multipumping-enabled multi-ported SRAM on FPGA," *IEEE Access*, vol. 6, pp. 19940-19947, 2018, doi: 10.1109/ACCESS.2018.2822311.
[24] B. P. Y. Kwan, G. C. T. Chow, T. Todman, W. Luk, W. Xu, "Lossy Multiport Memory," *2018 International Conference on Field-Programmable Technology (FPT)*, 2018, pp. 250-253, doi: 10.1109/FPT.2018.00046.
[25] J. Hardik, M. Edwards, E. R. Elenberg, A. S. Rawat, and S. Vishwanath, "Achieving Multi-Port Memory Performance on Single-Port Memory with Coding Techniques," *2020 3rd International Conference on Information and Computer Technologies (ICICT)*, 2020, pp. 366-375, doi: 10.1109/ICICT50521.2020.00065.
[26] S. S. Navid, A. Alkamil, and D. G. Perera, "Towards Composing Optimized Bi-Directional Multi-Ported Memories for Next-Generation FPGAs," *IEEE Access*, vol. 8, pp. 91531-91545, 2020, doi: 10.1109/ACCESS.2020.2994882.
[27] Xilinx, "7 Series FPGAs Data Sheet: Overview, Product Specification, DS180 (v2.6.1)," 2020, [Online]. Available: https://www.xilinx.com/support/documentation/data_sheets/ds180_7Series_Overview.pdf

BIOGRAPHIES OF AUTHORS

Druva Kumar S., is currently pursuing a Ph.D. from Visvesvaraya Technological University, Belgaum. His main research interests are VLSI Design and Embedded Systems. He is presently working as an Assistant Professor in the Department of Electronics and Communication Engineering, Dayananda Sagar College of Engineering, Bengaluru, with 8 years of Academic service. He has published 4 research articles in reputed journals, and he presented papers at 3 international conferences.

Roopa M., completed a Ph.D. degree in May 2016 from Gulbarga University, Kalaburgi. In Applied Electronics Division with the Specialization of VLSI and Embedded Systems. Presently working as a professor in the Department of Electronics and Communication Engineering, Dayananda Sagar College of Engineering, Kumaraswamy Layout, Bangalore. Presented papers at both National and International Conferences. Guided many UG and PG projects. Having Professional Body membership in LMISTE, MIETE, MIEEE. Member of BOS and BOE. I started Guiding Ph.D. students also and having the credit of one UG and 2 PG degrees. She completed 30 years of Academic Service.