International Journal of Reconfigurable and Embedded Systems (IJRES)
Vol. 11, No. 2, July 2022, pp. 115~125
ISSN: 2089-4864, DOI: 10.11591/ijres.v11.i2.pp115-125

Design and performance analysis of efficient hybrid mode multi-ported memory modules on FPGA platform

Druva Kumar Siddaraju, Roopa Munibyrappa
Department of Electronics and Communications, Dayananda Sagar College of Engineering, Bangalore, India

Article Info

Article history:

Received Jan 27, 2022
Revised Feb 11, 2022
Accepted Mar 16, 2022

Keywords:

Bank division with xor
Block random access memories
FPGA device
Multi-ported memories
System-on-chip

ABSTRACT

The multi-ported memories (MPMs) are essential and are part of the parallel computing system for high-performance features. The MPMs are commonly used in most processors and advanced system-on-chip (SoC) for faster computation and high-speed processing. In this manuscript, efficient MPMs are designed using the integration of hierarchical bank division with xor (HBDX) and bank division with remap table (BDRT) approaches. The BDRT approach is configured using remap table with a hash write controlling mechanism to avoid write conflicts. The different multiple read ports are designed using BDX, and HBDX approaches are discussed in detail. The results of 2W4R and 3W4R memory modules are analyzed in detail concerning chip area, operating frequency (MHz), block random access memories (BRAMs), and throughput (Gbps) for different memory depths on virtex-7 field programmable gate array (FPGA). The 2W4R utilizes 2.27% slices, operates at 268 MHz frequency by consuming 64 BRAMs for 16K memory depth. Similarly, the 3W4R uses 2.28% slices, operates at 250 MHz frequency by consuming 96 BRAMs for 16K Memory depth. The proposed designs are compared with existing MPM approaches with better chip utilization (Slices), frequency, and BRAMs on the same FPGA device.

Corresponding Author:

Druva Kumar Siddaraju
Department of Electronics and Communication Engineering, Dayananda Sagar College of Engineering
Shavige Malleshwara Hills, KS Layout, Bengaluru 560 078, Karnataka, India
Email: druva-ece@dayanandasagar.edu

1. INTRODUCTION

Multi-ported memory (MPM) modules are required in parallel computing systems or processors to obtain high performance. The parallel computational system uses the high-end register files along with shared-memory architectures to run the operations. Hence the multiple-parallel write and read ports and high-bandwidth memory modules are required to run the high-end parallel system [1]. There are many systems like vector processors, digital signal processing (DSP) system, chip-multi-processors (CMPs), very long instruction word (VLIW) based processors that rely on MPMs for parallel computations and access. The MPM module is designed using registers and logic elements. But it is feasible and applicable to the low-end memory modules. So, alter the static random-access memory (SRAM) bit cell in MPM to access more ports, which gives more area utilizations as the number of reads and write increases. So, field programmable gate array (FPGA) plays an essential role in customizing the area resources with minimal effort. The FPGA devices are used to build complex applications capable of producing high system speed rather than Application-specific integrated circuits (ASICs). The implementation of MPMs on the ASIC environment performs multiple writes and read operations simultaneously and avoids contention and serialization. The FPGA-based system uses a simple instruction pipeline mechanism and is less affected by instruction-level parallelism (ILP). The advanced FPGAs support only dual-ported RAMs. The user has to convert these to multi-ported memory modules using additional logic elements and block RAMs. The limitations of ILP are analyzed in many existing works to improve the processor system's performance. In real-time, ILP usage on FPGA for MPM is tiny and ineffective [2], [3].

Many approaches are available to design the MPMs in FPGA, including pure logic elements (LE) based, replication approach, banking approach, and multi-pumping approaches. The performance of the MPMs is analyzed using usage of LEs, BRAM usage, maximum operating frequency (Fmax), latency, and throughput [4]. The scalability feature is improved by providing the number of write/read ports and addressable memory space options. The latency and throughput parameters will be improved by utilizing proper unsegmented address space in MPMs [5]. The reliability and portability features will be enhanced by integrating the first-in-first-out (FIFOs) with multi-ports on FPGA [6]. The multi-port shared memory architecture is used to access the parallel data, providing high performance and being suitable for multi-core architectures. High-speed communication will be established only using advanced bus architecture, network on chip (NoC) with shared multi-ported memories [7]. The ternary content addressable memory (TCAM) is used in most high-speed applications but lags with higher power consumption on FPGA. So TCAM with multi-ported static random-access memory (SRAM) and multi-pumping method minimize resource utilization and improve FPGA power [8]. Even though multi-ported SRAM with pseudo features is used extensively in imaging processing applications, especially display drivers [9]. The multi-match packet classification is easily achieved with minimal soft errors using TCAM on FPGA [10]. The high bandwidth memory (HBM) offers parallelism and pipelining solutions with better throughput to most complex systems like neural networks based on a neuromorphic system in real-time scenarios [11]. The MPMs are used in many advanced applications like image processing, radar applications, real-time processing applications, data acquisition system, and multi-processing systems on chip (MPSoC) applications [12]–[14].

In this manuscript, efficient hybrid-mode multi-ported memory modules on the FPGA platform are discussed. The contribution of the work is as: i) The proposed MPMs provide a cost-effective design solution with high system performance for a complex system, ii) The BDRT approach is modified by using remap table with a hash write controller to avoid the write conflicts and (iii) The performance of proposed designs is discussed in detail and compared with existing XOR, live valve table (LVT), and BDRT approaches with better resources improvements. Section 1 discusses the related works on multi-ported memory modules using software and hardware approaches and its findings. The multi-ported memory module using HBDX and BDRT approaches are discussed in detail in section 2. The results and analysis of the proposed design with different memory depths are discussed in detail in section 3. Finally, it concludes the overall work with improvements in section 4.

The existing works related to the MPM modules with different approaches using both software and hardware environments are discussed in this section. Muddebihal and Purdy [15] present the multi-ported memory modules with area-efficient FPGA environment features. The work uses a better hardware mechanism for writing conflict resolution and detection to save BRAM and chip resources. The work lags with frequency features on the FPGA platform and needs to investigate the design on higher-order multi-port banks. Lai and Lin [16] discuss the multi-ported memories on the FPGA platform with an efficient design approach. The method uses a hierarchical approach to save BRAM and saves up to 69% BRAM for 2W4R design than the LVT approach. Abdelhadi and Lemieux [17] present the multi-ported memory compiler module using true dual-port BRAMs on FPGA. The work uses multiple-switched ports for multiple-ported RAM design with optimization. The result is realized with different test cases and improves the BRAM and LE than other approaches. Strollo and Trifiletti [7] present the configurable shared memory architecture for the multiprocessor system with parametrized features. The design reduces the interconnections among processors by sharing the memory locations.

Lai and Huang [18] present the algorithmic multi-ported memory with hierarchical banking architecture on the FPGA platform. The non-table-based MPM is designed using banking architecture to reduce the BRAMs. The different MPMs are realized with varying depths of memory and represent the various hardware resources. Shahrouzi and Perera [19] discuss the memory architecture using FPGA for computer applications. The designed MPM addresses many issues like on-chip area, BRAM utilization, and feasibility for intense computer applications. The memory architecture is built with memory banks, encoder, decoder, and multiplexors modules. The work consumes more clock cycles to produce the read output on the memory module. Patil and Musle [20] present the MPM with fault detection and repair feature using built-in self-test (BIST) on the FPGA platform. The synchronous and asynchronous-based micro-code BIST (MBIST) is used to test the MPM modules. The work analyses only the simulation results to verify the design is working functionally.

Manivannan and Srinivasan [21] present the 2W4R multi-port register file using pulsed latches. The work offers low-power consumption for 2W4R architecture compared to (SRAM) based register file module.

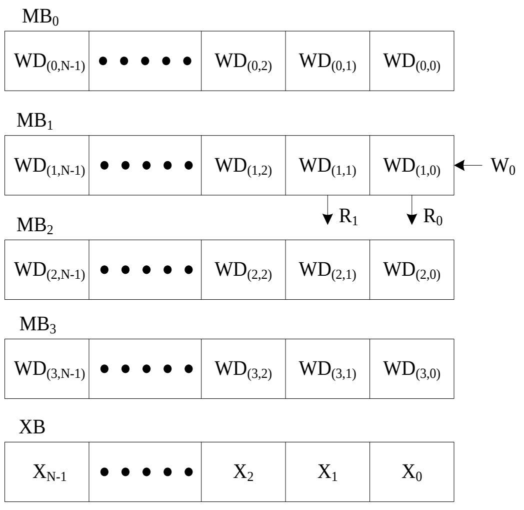
The design work is limited to 8-bit depth for designing the 2W4R multi-port register file. Humenniy *et al.* [22] discuss the shared access memory architecture for data transmission applications. The design uses a software-based approach for shared memory design based on the residue number system with data protection. Ullah *et al*. [23] present the ternary content-addressable memory (TCAM) based multi-ported SRAM module with multi-pumping features on FPGA. The work offers 2.85 times better performance than existing approaches per memory module. But the TCAM based process consumes more BRAMs and hardware resources on FPGA. Navagiri and Muthumanickam [24] elaborate the LVT found MPM module on FPGA. The design uses conventional approaches, consuming more BRAM and other hardware resources than XOR and advanced hybrid approaches. Shahrouzi and Perera [25] present the MPM architectures using the optimized counter mechanism for next-generation advanced FPGAs. The work analyses the routing complexity of MPM using complex and most straightforward architectures. Different MPM architectures like an encoded parallel model, direct data forwarding, encoded forwarding and binary coded forwarding modules are designed using the optimized counter mechanism. Chen *et al.* [26] present algorithm-based MPM using an efficient writing mechanism. The works use remap table-based architecture for multiple writes -ports. Shahrouzi *et al*. [27] discuss the MPMs with composing bi-directional features for advanced FPGAs. The work analyses the uni-directional MPM and bi-directional MPM using the decision-making module (DMM). The work is compared with other conventional approaches with better improvement in BRMAs and other hardware resources. Zhang *et al*. [28] present the XOR-based MPM with high throughput feature using a parallel hash table. The work analyzes the different performance metrics both on Xilinx U250 FPGA and Intel devices.

2. MULTI-PORTED MEMORY MODULE

This section explains the detailed architecture of MPM for both write and read ports approaches. The multiple read ports are designed using Bank division with XOR BDX, and hierarchical BDX (HBDX) approaches. The numerous write ports are created using the Bank division with Remap table BDRT approach. The integration of multiple writes and read ports is used to construct the MPM using BDX, and BDRT approaches.

2.1. Multiple read port approaches

In this work, different read port designs are designed, namely, 1W2R module, 1W2R mode, 4R mode, and HBDX based 1W4R module. The BDX increases the read ports rather than the write port using the conventional XOR approach. The different single-port RAM (1W1R) modules are used in the XOR approach to increase the write ports. In contrast, BDX approaches use many single-port RAMs to increase the read ports. The BDX based 1W2R design example is illustrated in Figure 1. It mainly contains five memory modules: four memory banks (MB) and one XOR bank (XB). The Four MB are represented as MB_0, MB_1, MB_2, and MB_3. Each MB can perform one write, and two read operations parallel. Each MB supports up to N-1 memory depth and stores the data to the corresponding address locations where 'N' denotes the maximum value of memory depth in MB.



MB0
WD(0,N-1) • • • • • WD(0,2) WD(0,1) WD(0,0)
MB1
WD(1,N-1) • • • • • WD(1,2) WD(1,1) WD(1,0)
W0
R1
R0
MB2
WD(2,N-1) • • • • • WD(2,2) WD(2,1) WD(2,0)
MB3
WD(3,N-1) • • • • • WD(3,2) WD(3,1) WD(3,0)
XB
XN-1 • • • • • X2 X1 X0

Figure 1. BDX based 1W2R design example

In this 1W2R example, the write (W_0) value is stored in MB_1 of the 0th location (WD $_{(1,0)}$). Whereas the read the R_0 and R_1 data simultaneously from MB_1 of 0th and 1st locations initially. The XB performs individual XOR operations for all the data values from each MB and is updated in corresponding areas., The XB is represented for the last location (N-1) updation in (1).

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

After XB Updation, the R_0 and R_1 will also be updated as (2) and (3).

$$R_0 = WD_{(1,0)} \quad (2)$$
$$R_1 = WD_{(0,1)} \oplus WD_{(2,1)} \oplus WD_{(3,1)} \oplus X_1 \quad (3)$$

The R_0 is obtained directly from MB_1 of the 0th location, whereas R_1 cannot access it now due to write data conflicts in MB_1. The R_1 recovers from data value by performing the XOR operation using MB_0, MB_2 MB_3, and XB of 1st locations. To access the data from MB3 for any location, the general is represented as (4).

$$WD_{(3,N-1)} = WD_{(0,N-1)} \oplus WD_{(1,N-1)} \oplus WD_{(2,N-1)} \oplus X_{N-1} \quad (4)$$

To access the data from 'n' read using the 1W2R approach is difficult due to duplication and write conflicts mechanism. This causes more area overhead and affects the system performance on FPGA. So HBDX approach is used to these issues with better performance with minor chip overhead.

The two-mode approach is designed using either 1W2R or 4R modules, and it is used further as a hybrid approach for the implementation of HBDX design. The 1W2R mode and 4R mode using BDX are illustrated in Figures 2(a) and 2(b), respectively. Figure 2(a) shows that the 1W2R mode works the same as 1W2R with few additional mechanisms. The R_0 reads the data directly from MB_0 of the 0th location. The R_1 reads the data from any banks like MB_1, MB_2, MB_3, and XB with the exact locations. The Read update (R_u) updates the XOR bank by reading the corresponding memory locations. Figure 2(b) illustrates the 4R mode, which does not have to write requests or data and performs only four read operations. The R0 and R2 are accessed simultaneously from MB_0, whereas R_1 and R_3 will access the data from any banks like MB_1, MB_2, MB_3, and XB.

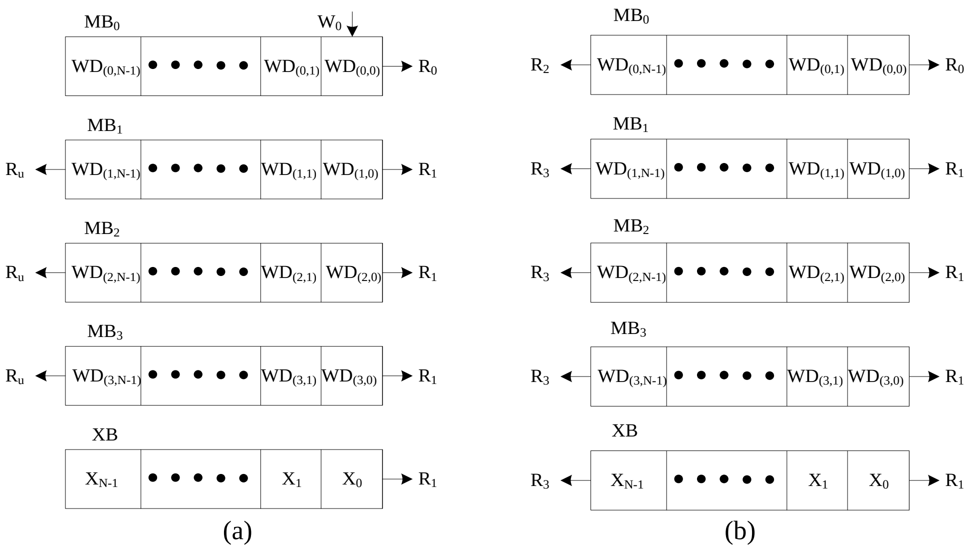


Figure 2. BDX based 1W2R/4R Mode design example (a) 1W2R mode and (b) 4R mode

The duplication and write conflicts are overcome using the HBDX approach using the 1W2R/ 4R mode approach. The HBDX based 1W4R design example as a worst-case scenario is illustrated in Figure 3. The MB_0 is considered 1W2R mode, and other banks like MB_1, MB_2, MB_3 and XB are considered 4R mode.

The data (W_0) is written in MB0 and reads R_0 and R_2 simultaneously with corresponding locations. In contrast, the banks like MB_1, MB_2, MB_3, and XB reads the related memory locations by performing XOR operation to generate the rest of the read outputs like R_2 and R_3. The corresponding updated read values from the banks are XORed to create the final read update (R_u) value.

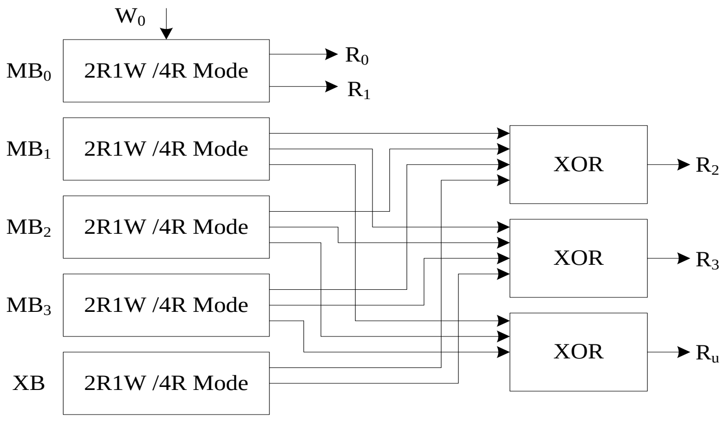
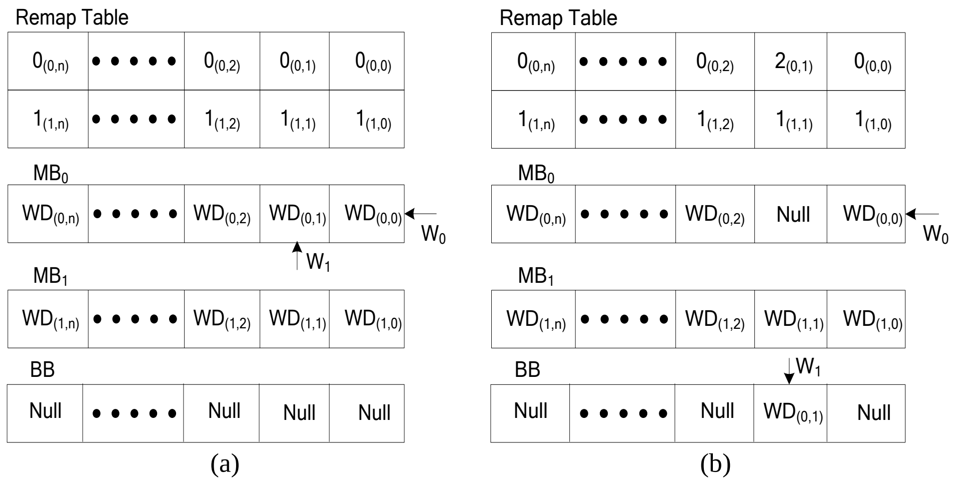


Figure 3. HBDX based 1W4R design example

2.2. Multiple write port approach

The BDRT approach is used to increase the write ports was initially proposed by Lai *et al.* [12]. It is similar to the conventional approaches like LVT and XOR with few modifications. The BDRT approach supports multiple write data by using remap table and extra BRAMs and avoiding replication mechanism. The remap table is used to find the memory location of the updated write data in the memory module. The BDRT based 2W1R memory module example is illustrated in Figure 4. It mainly contains one remap table, two memory banks (MB_0 and MB_1), and one bank buffer (BB). In BB, null entries are stored temporarily, which doesn't contain any valid user data. The written requests W0 and W1 are first analyzed in the remap table to identify the correct memory and its location to store the reported data. Based on the remap table mechanism, the W_0 and W_1 requests go to the bank MB_0 of 0th and 1st memory address locations. But the MB can store one write at a time. So W_0 request is updated in 0th memory location, and W_1 is stored temporarily in BB of 1st location as a null entry with offset 1. Once the remap table is updated, as shown in Figure 4(b), 1st memory location is updated by the remap table, whose identification number is 2 in BB. The BB stores W_1 data in 1st memory location based on the updated remap table in the final stage. The BDRT approach uses less chip area and BRAM the conventional LVT approach, but it needs additional registers to design the remap table.



Remap Table
0(0,n) ••••• 0(0,2) 0(0,1) 0(0,0)
1(1,n) ••••• 1(1,2) 1(1,1) 1(1,0)
MB0
WD(0,n) ••••• WD(0,2) WD(0,1) WD(0,0)
W0
W1
MB1
WD(1,n) ••••• WD(1,2) WD(1,1) WD(1,0)
BB
Null ••••• Null Null Null
(a)
Remap Table
0(0,n) ••••• 0(0,2) 2(0,1) 0(0,0)
1(1,n) ••••• 1(1,2) 1(1,1) 1(1,0)
MB0
WD(0,n) ••••• WD(0,2) Null WD(0,0)
W0
MB1
WD(1,n) ••••• WD(1,2) WD(1,1) WD(1,0)
BB
W1
Null ••••• Null WD(0,1) Null
(b)

Figure 4. BDRT based 2W1R memory module example (a) Initialization stage and (b) Final stage

2.3. Integrated MPM module

The integration of BDRT and HBDX for writing and read ports, respectively, for the nWmR memory module is illustrated in Figure 5. It mainly contains BDRT and HBDX modules, where the memory module is divided into 'd' data banks.

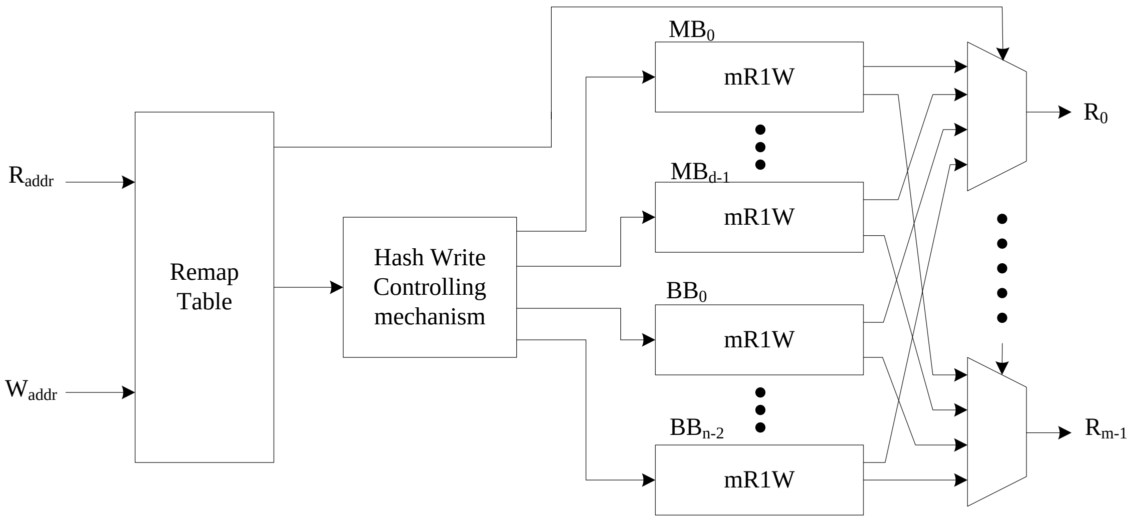


Figure 5. nWmR memory implementation using BDRT with HBDX approach

The BDRT approach provides all the necessary written data using remap table to the HBDX module. The BDRT requires n-1 BBs for write port updation. Additionally, the hash writes control mechanism is incorporated in the BDRT approach for the proper distribution of write data to the corresponding banks in the HBDX module. The 2W4R memory module using BDRT with the HBDX approach is illustrated in Figure 6. For 2W4R implementation, four memory banks and two bank buffers are required. The MBs and BBs are designed using 1W2R or 1W2R mode or 4R mode memory module. In this work, the 1W2R memory module is considered to avoid write conflict for MBs and BBs design. The four MBs (1W2R memory module) and two BBs (1W2R memory module) receive the write data and generate the corresponding read data based on the HBDX approach. The remap table provides the multiplexors' select line to generate the final four outputs (R_0, R_1, R_2, and R_3). For 3W4R memory implementation, only four MBs are used, and three BBs are considered. Only BBs can be increased to solve the write conflicts in the 3W4R memory module.

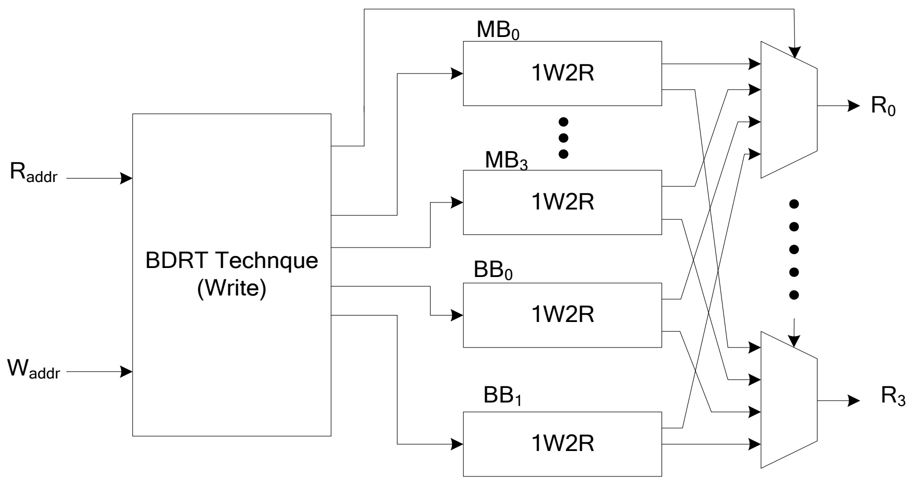


Figure 6. 2W4R memory module using BDRT with HBDX approach

3. RESULTS AND DISCUSSION

This section explains detailed implementation results of MPMs on Virtex-7 FPGA. The Results of multiple read ports and integration of HBDX with the BDRT approach for 2W4R and 3W4R memory modules are discussed. For all the MPMs designs, the standard data width of 32-bit is considered. The performance parameters like chip area utilization (Slices and LUTs), maximum obtained frequency (Fmax) in MHz, BRAMs, and throughput (Gbps) are analyzed by concerning different memory depths are presented for 2W4R and 3W4R memory designs. The Virtex-7 FPGA with XC7V585T device is considered for implementation with a package of FFG1761 [29]. The virtex-7 FPGA has enormous resources and supports high-performance features for complex designs in real-time scenarios. The virtex-7 FPGA mainly contains 582K logic cells, 92K of slices, 729K of configurable logic blocks (CLBs), 795 of BRAMs, 126000 of DSP slices, 850 of Input-Output blocks (IOBs), and 6938Kb of distributed RAMs. The performance analysis of multiple-read ports using the BDX approach on virtex -7 FPGA is tabulated in Table 1. The multiple read ports like 1W2R, 1W2R mode, 4R using BDX approaches, and 1W4R using HBDX approach memory modules are implemented. The multiple read ports designs using 8K and 16K memory depth are realized in this work. The chip area utilization of multiple-read ports on virtex-7 FPGA is illustrated in Figure 7.

Table 1. Performance analysis of multiple-read ports using BDX approach on Virtex -7 FPGA

Resources	BDX_1W2R		BDX_1W2R Mode		BDX_4R Mode		HBDX_1W4R	
	8K	16K	8K	16K	8K	16K	8K	16K
Slices	96	96	96	96	81	41	459	405
LUTs	96	96	65	65	80	40	241	185
BRAMs	16	32	16	32	16	16	32	48
Fmax (MHz)	1759.01	1759.01	1463.05	1463.05	1266.22	1272.14	1223.54	1223.54

The graphical representation of the BRAM utilization and obtained frequency (Fmax) for multiple read ports is shown in Figure 8(a) and Figure 8(b), respectively. The 1W2R module utilizes only 96 slices and 96 LUTs for both 8K and 16K memory depth on virtex-7. The 1W2R uses 16 and 32 BRAMs for 8K and 16K, respectively, and operates 1759 MHz. The 1W2R mode module utilizes only 96 slices for both 8K and 16K memory depth and works with a frequency of 1463.05 MHz on virtex-7. The 1W2R mode module utilizes 16 and 32 BRAMs for 8K and 16K memory depth. The 4R mode module using BDX utilizes only 81 slices and 41 slices for both 8K and 16K memory depth, respectively. The 4R mode module utilizes only 16 BRAMs for 8K and 16K memory depth, respectively. The 4R mode module operates with 1266.22 MHz and 1272.14 MHz for 8K and 16K memory depth, respectively, on Virtex-7 FPGA. The 1W4R module using the HBDX approach utilizes only 459 slices and 405 slices for both 8K and 16K memory depth and operates with a frequency of 1223.54 MHz on virtex-7. The 1W4R module using the HBDX approach utilizes 32 and 48 BRAMs for both 8K and 16K memory depth, respectively.

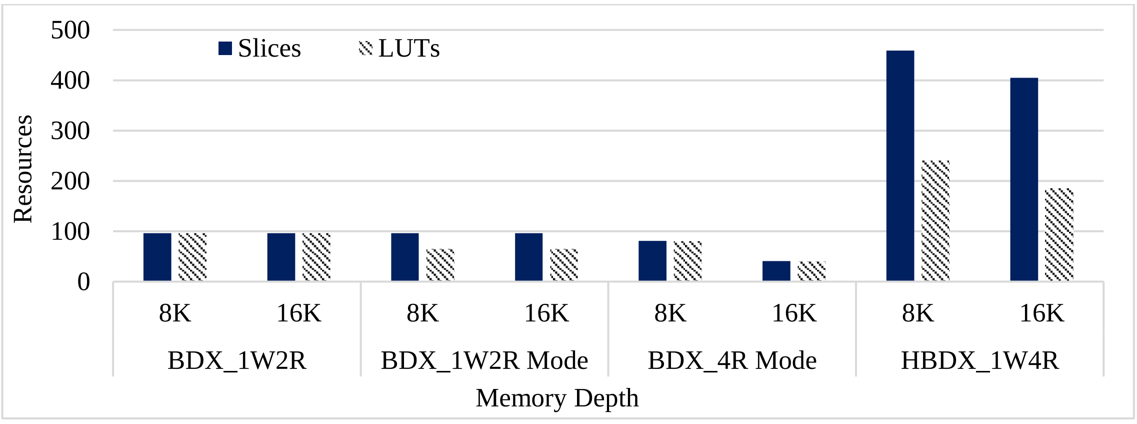
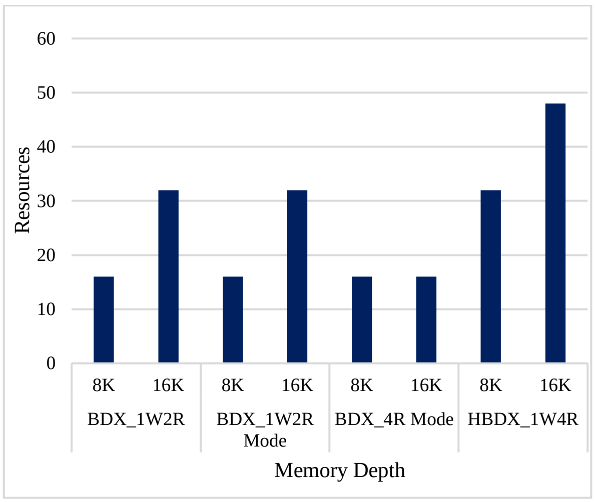
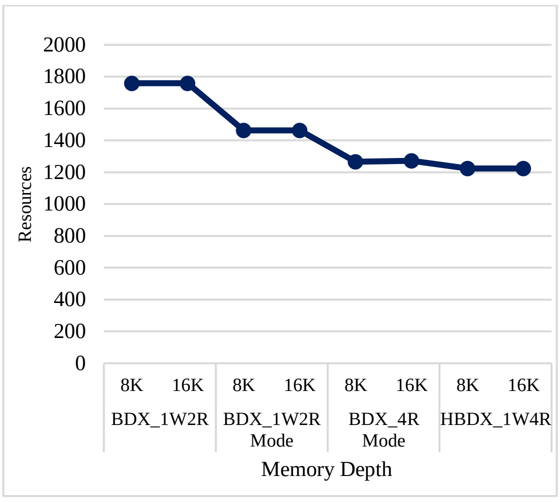


Figure 7. Chip area utilization of multiple-read ports on virtex-7 FPGA



(a)



(b)

Figure 8. BRAM and frequency parameter analysis on virtex -7 FPGA for multiple read ports (a) BRAM utilization v/s memory depth and (b) fmax (MHz) v/s memory depth

The performance analysis of 2W4R and 3W4R memory modules on Virtex-7 FPGA is illustrated in Figure 9. The chip area (Slices and LUTs), frequency, BRAMs, and throughput parameters are considered concerning different memory depths for performance analysis. The 2W4R memory module utilizes 1.15% of slices, 12% of LUT's for 8K memory depth, 2.27% of slices, 29% of LUT's for 16K memory depth on virtex-7 FPGA as shown in Figure 9(a). In contrast, the 3W4R memory module utilizes 1.165% of slices, 20% of LUT's for 8K memory depth, 2.28% of slices, and 36% of LUT's for 16K memory depth Virtex-7 FPGA as shown in Figure 9(b). The 2W4R memory module operates at 292.9 MHz for 8K and 267.3 MHz for 16K. In contrast, the 3W4R memory module operates at 270.1 MHz for 8K and 250.9 MHz for 16K memory depth on virtex-7 FPGA, as illustrated in Figure 9(c).

The throughput is calculated using latency, frequency, and input data width parameters. The latency of 2.5 clock cycles is utilized for both 2W4R and 3W4R memory designs. The 2W4R memory module works at 3.74 Gbps for 8K and 3.42 Gbps for 16K. The 3W4R memory module operates at 3.45 Gbps for 8K and 3.2 Gbps for 16K memory depth on Virtex-7 FPGA, as illustrated in Figure 9(d). The 2W4R memory module utilizes the BRAMs of 32 for 8K and 64 for 16K. In contrast, the 3W4R memory module uses the BRAMs of 48 for 8K and 96 for 16K memory depth on Virtex-7 FPGA, as illustrated in Figure 9(e). The BRAMs utilization increases exponentially as the memory width increases for both the 2W4R and 3W4R memory modules.

The design approach, FPGA device, resources utilization (Slices), Fmax, and BRAM parameters for different memory depths are considered for comparison. The comparison of the proposed 2W4R memory module with existing approaches on the same virtex-7 FPGA is tabulated in Table 2. The proposed 2W4R using the BDRT approach operates at a better operating frequency of 7.8% for 8K and 2.98% for 16K depth than the existing 2W4R using the XOR approach [30]. Similarly, The BRAMs utilization of 60% for 8K and 60% for 16K depth than the existing 2W4R using XOR approach [30]. The proposed 2W4R using the BDRT approach utilizes less area overhead of 93.11% for 8K and 92.92% for 16K depth than the existing 2W4R using the LVT approach [4]. Similarly, The BRAMs utilization of 50% for 8K and 50% for 16K depth than the existing 2W4R using LVT approach [4]. The drastic area reduction of 94.33% for 8K and 95.71% for 16K then the existing BDRT based 2W4R system [16]. Even the operating frequency of the proposed 2W4R modules works at 50.85% for 8K and 52.23% for 16K faster than the existing BDRT based 2W4R approach [16]. The bi-directional MPM is designed by [27] using a decision-making module (DMM) on Virtex-6 FPGA. The proposed design of the 2W4R module utilizes less BRAM of 33.33% for 8K and 33.3% for 16K than the existing approach [27].

In contrast, the comparison of the proposed 3W4R memory module with existing approaches is tabulated in Table 3. The proposed 3W4R memory design provides better Fmax and BRAMs than the existing 3W4R based XOR approach [30]. Similarly, the proposed 3W4R memory design utilizes less area and BRAMs, operates at better frequency than the existing 3W4R based LVT approach [4]. The proposed 3W4R memory design uses less area and works at better frequency than the existing 3W4R based LVT approach [4] on Virtex-7 FPGA. The 2W4R and 3W4R memory using BDRT [16] consume fewer BRAMs than the proposed work, consuming more slices and LUTs than the proposed designs. The DMM-based 3W4R module [27] utilizes BRAMs of 96 and 192 for 8K and 16K memory depths. The proposed 3W4R module utilizes less BRAM of 50% for 8K and 50% for 16K than the existing approach [27]. The MPM designs for different depths are analyzed. If the memory depth is increased, the slices, BRAMs utilization will increase by decreasing the operating frequency of the FPGA. Overall, the proposed 2W4R and 3W4R memory modules utilize less chip area, BRAMs, and provide better-operating Frequency on FPGA hardware than similar approaches.

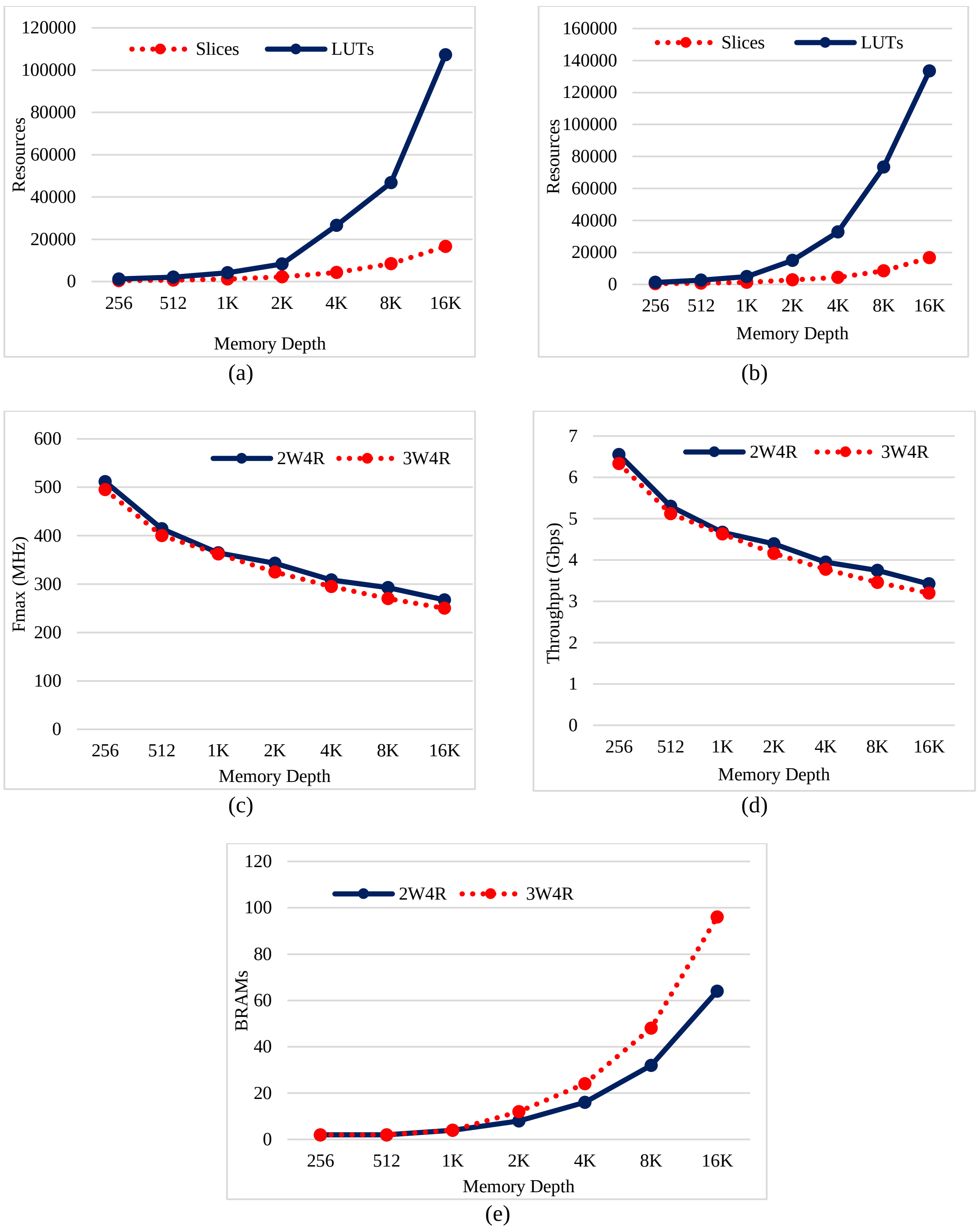


Figure 9. Performance analysis of 2W4R and 3W4R memory modules on virtex-7 FPGA (a) Area v/s memory depth (2W4R), (b) Area v/s memory depth (3W4R), (c) Fmax v/s memory depth, (d) Throughput (Gbps) v/s memory depth and (e) BRAMs v/s memory depth

Table 2. Comparison of proposed 2W4R memory module with existing approaches

Design works	Approach	FPGA	Slices (%)	Fmax 8K	BRAMs	Slices (%)	Fmax 16K	BRAMs
Ref [30]	XOR	Virtex-7	0.8	270	80	0.4	260	160
Ref [4]	LVT	Virtex-7	16.7	149	64	32.1	116	128
Ref [16]	BDRT	Virtex-7	20.3	144	30	53	128	60
Ref [27]	DMM	Virtex-6	0.8	108	48	0.9	115	96
Proposed	BDRT	Virtex-7	1.15	293	32	2.27	268	64

Table 3. Comparison of proposed 3W4R memory module with existing approaches

Design works	Approach	FPGA	Slices (%)	Fmax 8K	BRAMs	Slices (%)	Fmax 16K	BRAMs
Ref [30]	XOR	Virtex-7	0.6	220	144	0.9	199	288
Ref [4]	LVT	Virtex-7	35.4	111	96	66.4	86	192
Ref [16]	BDRT	Virtex-7	38.4	121	40	71.2	94	80
Ref [27]	DMM	Virtex-6	0.9	127.78	96	1.1	138.73	192
Proposed	BDRT	Virtex-7	1.165	270	48	2.28	250	96

4. CONCLUSION

In this manuscript, efficient multi-ported memory modules are designed and implemented on FPGA. The proposed design MPMs offer cost-effective design solutions in terms of resources and performance than the existing MPMs approaches. The multiple read ports are designed using BDX and HBDX approach; The multiple write ports are created using the BDRT approach with remap table. The remap table is configured with a hash write controlling mechanism to avoid the write conflicts in the BDRT approach. The 2W4R and 3W4R memory modules are implemented and analyzed the results. The multiple read port techniques using BDX and HBDX approaches are discussed with performance realization. The integration of BDRT with the HBDX approach is used to construct the MPM modules. The proposed 2W4R and 3W4R utilize less chip area and BRAMs and operate better than the existing approaches. The 2W4R memory module uses 60% and 50% fewer BRAMs than the existing XOR and LVT for 16K memory depth. The proposed 3W4R memory module utilizes 66.6% and 50% fewer BRAMs than the existing XOR and LVT approach for 16K memory depth. The proposed 2W4R and 3W4R memory modules operate at a better Frequency of 52.23% and 62.4%, respectively, than the existing BDRT approach for 16K memory depth.

REFERENCES

[1] A. M. S. Abdelhadi and G. G. S. Lemieux, "Modular switched multiported SRAM-Based memories," *ACM Transactions on Reconfigurable Technology and Systems*, vol. 9, no. 3, p. pp 1–26, 2016, doi: 10.1145/2851506.

[2] H. Trang, "Design choices of multi-ported memory for FPGA," *Le Quy Don Technical University-Section on Information and Communication Technology*, no. 2, pp. 80–91, 2013.

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

[4] C. E. LaForest and J. G. Steffan, "Efficient multi-ported memories for FPGAs," in *Proceedings of the 18th Annual ACM/SIGDA International Symposium on Field Programmable Gate Arrays*, 2010, pp. 41–50, doi: 10.1145/1723112.1723122.

[5] K. R. Townsend, O. G. Attia, P. H. Jones, and J. Zambreno, "A scalable unsegmented multiport memory for FPGA-based systems," *International Journal of Reconfigurable Computing*, vol. 2015, no. 11, pp. 1–12, 2015, doi: 10.1155/2015/826283.

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

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

[8] A. Santhosh and S. Agrawal, "Multipumping-enabled multiported SRAM based efficient TCAM design," in *2020 4th International Conference on Electronics, Materials Engineering & Nano-Technology (IEMENTech)*, 2020, pp. 1–6, doi: 10.1109/IEMENTech51367.2020.9270059.

[9] H. M. Lam, F. Guo, H. Qiu, M. Zhang, H. Jiao, and S. Zhang, "Pseudo multi-port SRAM circuit for image processing in display drivers," *IEEE Transactions on Circuits and Systems for Video Technology*, vol. 31, no. 5, pp. 2056–2062, 2021, doi: 10.1109/TCSVT.2020.2979046.

[10] K. Sakthi and P. Nirmal Kumar, "Efficient soft error resiliency by multi-match packet classification using scalable TCAM implementation in FPGA," *Microprocessors and Microsystems*, vol. 74, pp. 1–6, 2020, doi: 10.1016/j.micpro.2019.102985.

[11] B. U. Pedroni, S. R. Deiss, N. Mysore, and G. Cauwenberghs, "Design principles of large-scale neuromorphic systems centered on high bandwidth memory," in *2020 International Conference on Rebooting Computing (ICRC)*, 2020, pp. 90–94, doi: 10.1109/ICRC2020.2020.00013.

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

[13] S. Memory, T. T. Mutlugun, and S. De Wang, "OpenCL computing on FPGA using multiported," in *2015 25th International Conference on Field Programmable Logic and Applications (FPL)*, 2015, pp. 1–4, doi: 10.1109/FPL.2015.7293983.

[14] J. Wang, Y. Zhang, J. Zhou, P. Jia, and X. He, "Realization of multi-port SDRAM controller in LXI data acquisition system," in *2013 International Conference on Information Science and Cloud Computing Companion*, 2013, pp. 515–520, doi: 10.1109/ISCC-C.2013.105.
[15] A. A. Muddebihal and C. Purdy, "Design and implementation of area efficient multi-ported memories with write conflict resolution," in *2015 IEEE 58th International Midwest Symposium on Circuits and Systems (MWSCAS)*, 2015, pp. 1–4, doi: 10.1109/MWSCAS.2015.7282041.
[16] B. C. C. Lai and J. L. Lin, "Efficient designs of multiported memory on FPGA," *IEEE Transactions on Very Large Scale Integration (VLSI) Systems*, vol. 25, no. 1, pp. 139–150, 2017, doi: 10.1109/TVLSI.2016.2568579.
[17] A. M. S. Abdelhadi and G. G. F. Lemieux, "A Multi-ported memory compiler utilizing true dual-port BRAMs," in *2016 IEEE 24th Annual International Symposium on Field-Programmable Custom Computing Machines (FCCM)*, 2016, pp. 140–147, doi: 10.1109/FCCM.2016.45.
[18] B. C. Lai and K. 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.
[19] S. N. Shahrouzi and D. G. Perera, "An efficient FPGA-based memory architecture for compute-intensive applications on embedded devices," in *2017 IEEE Pacific Rim Conference on Communications, Computers and Signal Processing (PACRIM)*, 2017, pp. 1–8, doi: 10.1109/PACRIM.2017.8121901.
[20] S. R. Patil and D. B. Musle, "Implementation of BIST technology for fault detection and repair of the multiported memory using FPGA," in *2017 International conference of Electronics, Communication and Aerospace Technology (ICECA)*, 2017, pp. 43–47, doi: 10.1109/ICECA.2017.8212849.
[21] T. S. Manivannan and M. Srinivasan, "A 4-READ 2-WRITE Multi-port register file design using pulsed-latches," in *2018 Second International Conference on Electronics, Communication and Aerospace Technology (ICECA)*, 2018, pp. 262–267, doi: 10.1109/ICECA.2018.8474898.
[22] P. Humenniy, O. Volynskyy, I. Albanskiy, and A. Voronych, "Designing a shared access memory and its application in data transmission and protection systems," in *2018 14th International Conference on Advanced Trends in Radioelecrtronics, Telecommunications and Computer Engineering (TCSET)*, 2018, pp. 143–147, doi: 10.1109/TCSET.2018.8336174.
[23] I. Ullah, Z. Ullah, and J. A. Lee, "Efficient TCAM design based on multipumping-enabled multiported SRAM on FPGA," *IEEE Access*, vol. 6, pp. 19940–19947, 2018, doi: 10.1109/ACCESS.2018.2822311.
[24] E. Navagiri, T. Sheela, T. Muthumanickam, M. E. Vlsi, and A. Prof, "Design of LVT based multiported memory in FPGA," *International Journal of Pure and Applied Mathematics*, vol. 119, no. 14, pp. 121–125, 2018, [Online]. Available: http://www.ijpam.eu.
[25] S. N. Shahrouzi and D. G. Perera, "Optimized counter-based multi-ported memory architectures for next-generation FPGAs," in *2018 31st IEEE International System-on-Chip Conference (SOCC)*, 2018, pp. 106–111, doi: 10.1109/SOCC.2018.8618500.
[26] B. Y. Chen, B. E. Cher, and B. C. Lai, "Efficient write scheme for algorithm-based multi-ported memory," *2019 International Symposium on VLSI Design, Automation and Test (VLSI-DAT)*, pp. 1–4, 2019, doi: 10.1109/VLSI-DAT.2019.8741927.
[27] S. N. Shahrouzi, 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.
[28] R. Zhang, S. Wijeratne, Y. Yang, S. R. Kuppannagari, and V. K. Prasanna, "A High throughput parallel hash table on FPGA using XOR-based Memory," in *2020 IEEE High performance extreme computing conference (HPEC)*, 2020, pp. 1–7, doi: 10.1109/HPEC43674.2020.9286199.
[29] Xilinx, "7-series-product-selection-guide," *Xilinx*, 2021. https://www.xilinx.com/content/dam/xilinx/support/documents/selection-guides/7-series-product-selection-guide.pdf (accessed Nov. 09, 2021).
[30] C. E. LaForest, M. G. Liu, E. R. Rapati, and J. G. Steffan, "Multi-ported memories for FPGAs via XOR," in *Proceedings of the ACM/SIGDA international symposium on Field Programmable Gate Arrays*, 2012, pp. 209–218, doi: 10.1145/2145694.2145730.

BIOGRAPHIES OF AUTHORS

Druva Kumar Siddaraju 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 10 years of Academic service. He has published 5 research articles in reputed peer reviewed journals, and he presented papers in 4 international conferences. He can be contacted at email: druva-ece@dayanandasagar.edu.

Roopa Munibyrappa 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, and 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 31 years of Academic Service. She can be contacted at email: surajroopa@gmail.com.