Towards Composing Optimized Bi-Directional Multi-Ported Memories for Next-Generation FPGAs

S. NAVID SHAHROUZI, ARKAN ALKAMIL, AND DARSHIKA G. PERERA, (Senior Member, IEEE)
Department of Electrical and Computer Engineering, University of Colorado at Colorado Springs, Colorado Springs, CO 80918, USA
Corresponding author: Darshika G. Perera (darshika.perera@uccs.edu)

ABSTRACT With the proliferation of embedded computing, there has been a dramatic increase in utilization of FPGAs to accelerate real-time compute/data-intensive applications on embedded platforms. FPGA-based designs typically achieve high speed-performance by leveraging parallelism in computations, which require simultaneous multiple reads/writes from/to the on-chip memory. However, current FPGAs only comprise dual-port on-chip memories, which hinder simultaneous multiple read/write (R/W) operations needed for parallel computing. Although several multi-ported memories are proposed in the literature, these designs become complex due to the extra logic and routing used for techniques/architectures to provide an arbitrary number of R/W ports for simultaneous multiple R/W operations. Furthermore, most of the existing multi-ported memories only provide an arbitrary number of uni-directional R/W ports. There is only one existing multi-ported memory design in the literature that provides bi-directional R/W ports. Unlike uni-directional ones, bi-directional multi-ported memories give more flexibility to the designers, since the number of read and write transactions can be changed as needed on-the-fly at any time. Previously, we introduced unique and efficient uni-directional multi-ported memories, which were created in such a way to significantly reduce the design and routing complexity. In this research work, we introduce novel, unique, and efficient bi-directional multi-ported memory architectures to provide an arbitrary number of bi-directional R/W ports. Our proposed bi-directional multi-ported memories are designed in such a way to dramatically reduce the design and routing complexity by eliminating the circular paths that typically exists in the current multi-ported memories in the literature, while enhancing operating frequency and area-efficiency. To the best of our knowledge, no similar work exists in the literature that provide multi-ported memory designs without the circular paths. Experiments are performed to evaluate the feasibility and efficiency of our bi-directional multi-ported memory designs. These results and analyses illustrate that our bi-directional multi-ported memories are far more efficient compared to the existing ones in the literature. Due to lower design and routing complexity compared to the existing ones, our simplified memories would enable seamless integration to the next-generation FPGAs with minimal design cost.

INDEX TERMS Bi-directional ports, FPGAs, embedded memory, embedded systems, multi-ported memory, on-chip BRAMs.

I. INTRODUCTION

With the advancement of mobile, portable, and embedded computing, FPGAs (Field Programmable Gate Arrays) are becoming increasingly popular among the embedded hardware designers. This is mainly because FPGAs often provide a higher-level of flexibility than ASICs (application-specific-integrated-circuits) as well as higher performance than software running on processors [1]. Unlike ASICs, post-fabrication reprogrammability of FPGAs, allows post-design optimizations and upgrades in applications, and also gives its ability to reuse the same chip/FPGA to process a variety of applications, by changing the on-chip hardware circuitry from one application to another as needed. Since FPGAs are immediately available, the time-to-market is also reduced. Furthermore, the dynamic partial reconfiguration feature of the current FPGAs enables reconfiguring parts of the chip that require modification, while other parts are still operational;
thus achieving significant space savings on chip for complex applications, as illustrated in [2]–[6]. The aforementioned features prolong the useful life of FPGA-based computing systems, while minimizing the long-term cost.

Currently, FPGAs are widely used in many fields, including ASIC prototyping, medical electronics, aerospace, automotive, broadcast, consumer electronics, communications, and defense, to name a few. In recent years, there has been a dramatic increase in utilization of FPGAs, both in the industry and in the academia, to enhance the speedup of many real-time compute/data intensive applications, such as data mining/analytics [2]–[6], machine/deep learning [7], control systems, and image processing, specifically on embedded platforms.

FPGA-based computing systems usually achieve high speed-performance by exploiting fine-grain and coarse-grain parallelism, data parallelism, and pipeline nature of the computations. In order to execute the computations (or tasks) in parallel, it is essential to read/write multiple data/results simultaneously to/from the on-chip memory. However, most of the current state-of-the-art FPGAs consist of on-chip memory blocks with only two ports (i.e., dual ports) for read/write (R/W) operations, for instance, Block RAMs (BRAMs) of Xilinx [8] and Altera [9] FPGAs, which hinder simultaneous multiple read/write (R/W) operations required for parallel computing. As a result, these existing dual-port memories on FPGAs, in their current form, are insufficient to provide the required support in accelerating real-time compute/data intensive applications on embedded platforms. Also, our previous work [2]–[6] on FPGA-based accelerators for compute/data-intensive applications demonstrated that utilizing existing dual-port BRAMs can negatively impact the overall performance of the applications significantly.

An alternative is to employ soft-logic to create register-based memories on FPGAs, as in [10], [11]. However, this approach is not efficient, since it often utilizes significant amount of logic resources on chip, which in turn affects the area and power efficiencies. This is one of the reasons FPGA manufacturers provide dense dual-port on-chip BRAMs as hard-logic with their FPGAs. In this case, increasing the number of ports in the existing hard-logic BRAMs can be costly. Therefore, it is imperative to design and develop efficient multi-ported memories to provide an arbitrary number of R/W ports for simultaneous multiple R/W operations to perform the computations in parallel, in order to enhance the speedup of many real-time compute/data-intensive applications on FPGAs.

From our investigation on existing works (detailed in Section II), it is revealed that several multi-ported memory designs for FPGAs exist in the literature. These existing multi-ported memory designs become complex, comprising complex structures, mainly due to the extra logic and routing needed to create techniques/architectures to provide an arbitrary number of R/W ports for simultaneous multiple R/W operations. These existing multi-ported memories comprise circular paths, which lead to dramatic increase in design and routing complexity. This sheer design and routing complexities hinder seamless integration of these existing multi-ported memories into the next-generation FPGAs and the next-generation FPGA-based CAD tools.

Our aforementioned investigation also reveals that most of the existing works only provide an arbitrary number of uni-directional R/W ports for simultaneous multiple R/W operations. There is only one existing multi-ported memory design in the literature that provides bi-directional R/W ports [11]. Unlike uni-directional multi-ported memories, bi-directional multi-ported memories give more flexibility to the designers, when creating the hardware architectures for a given application. For instance, bi-directional multi-ported memories are not limited to a fixed number of read transactions and a fixed number of write transactions, as in uni-directional ones. Furthermore, with bi-directional multi-ported memories, the designers have the flexibility to change the number of read and write transactions as needed on-the-fly at any time. Also, from our previous work [12], [13], it was observed that certain control systems applications, such as Model Predictive Control, executed on FPGAs, can be dramatically accelerated using bi-directional port multi-ported memories.

All these facts illustrate that it is imperative to especially design and develop bi-directional multi-ported memories to provide an arbitrary number of R/W ports for simultaneous multiple R/W operations, in such a way to eliminate the circular paths that typically exist in the current multi-ported memories in the literature, in order to dramatically reduce the design and routing complexity, which in turn enhances the operating frequency and area-efficiency.

Previously, we introduced unique and efficient uni-directional multi-ported memories ([14]–[16]), which were designed in such a way to significantly reduce the design and routing complexity, which in turn led to increase in operating frequency and area-efficiency. In this research work, our main objective is to create novel, unique, and efficient bi-directional multi-ported memory designs for real-time compute/data-intensive application on embedded platforms, considering the associated constraints, issues, and requirements, including the design/routing complexity and the design cost. In this paper, we make the following contributions:

- We introduce novel, unique, customized, and optimized bi-directional port multi-ported memory architectures, which are designed utilizing true dual-port BRAMs. With our memory architectures, an arbitrary number of bi-directional ports can be provided for simultaneous multiple R/W operations. To facilitate this, we introduce a unique internal architecture for our bi-directional multi-ported memories.
- We also introduce a novel and unique BRAM block distribution scheme, with a unique constraint/condition, to allocate/place the BRAMs among the n bi-directional ports. This BRAM distribution scheme is crucial in
In Section V, we summarize our work and conclude. These comparison results are also presented in Section IV.

We design our bi-directional port multi-ported memory architectures in such a way to dramatically reduce the design and routing complexity by eliminating the circular paths that typically exist in the current multi-ported memories in the literature. To simplify the designs, only the read data from the BRAMs are processed using intermediate combinatorial logic before being forwarded to the read data output ports, while all the other signals are directly forwarded to the BRAMs without incorporating any intermediate logic between modules. To the best of our knowledge, no similar work exists in the literature that provide bi-directional multi-ported memory designs without the circular paths.

Also, our bi-directional multi-ported memories are designed to obtain the highest possible operating frequency with the minimum occupied area on chip, as well as to achieve very high memory depths. Conversely, the only existing bi-directional multi-ported memory design has complex structures, occupies significant area on chip, and can not be used for high memory depths.

We implement different multi-ported memory configurations with varying number of bi-directional ports (from 3 to 8 bi-directional ports) and with varying memory depths (from 2 to 128K). These are implemented using our 4 different DMM modules with varying internal architectures. This leads to 356 bi-directional multi-ported memory configurations. We perform experiments on all 356 configurations, and analyze the maximum operating frequency, occupied slices, BRAM usage, and available memory depths. Our experimental results and analyses illustrate that our bi-directional multi-ported memories are far more efficient compared to the existing ones in the literature.

This paper is organized as follows: In Section II, we present our investigation and analysis of the existing works on multi-ported memories for FPGAs in the published literature. Our proposed multi-ported memory architectures are discussed and presented in Section III. Initially, in this section, we briefly discuss and present our previous work on uni-directional multi-ported memories for FPGAs. Next, we detail our proposed novel, unique, and efficient bi-directional multi-ported memory architectures, which consist of our unique BRAM block distribution scheme for the bi-directional multi-ported memories, as well as our unique internal architecture of our $n$ bi-directional multi-ported memories. Our experimental results and analysis in terms of maximum operating frequency, number of occupied slices, BRAM usage, maximum available memory depths are reported and discussed in Section IV. We evaluate our proposed bi-directional multi-ported memory architectures with the existing multi-ported memories in the literature. These comparison results are also presented in Section IV. In Section V, we summarize our work and conclude.

II. ANALYSIS OF EXISTING WORKS ON MULTI-PORTED MEMORIES ON FPGAS

We investigated the existing research works on multi-ported memory architectures on FPGAs in the published literature. As stated in [10], [11], there are three existing standard techniques for increasing the number of ports in BRAMs on FPGAs, i.e., Replication, Banking, and Multipumping. These three techniques utilized the existing dual-port BRAMs on FPGAs. Replication maintains a replica of the memory to support each additional read port, but can not support more than one write port [10]. Banking divides data across multiple BRAMs (banks), but each R/W port can only access its corresponding memory bank [10], [11]. Multipumping increases the number of R/W ports by adding a multiplexer and registers, and also clocking the memory at a multiple of the external clock, which dramatically reduces the maximum external operating frequency [10], [11].

There are several existing memory designs, which are often used in conjunction with the soft processors on FPGAs including Multithreaded [17], [18], VLIW [19]–[22], Application-Specific [23], and MicroBlaze (from Xilinx [24]) and Nios (from Altera [25]) processors. These memories have strict limitations, to accommodate the needs of the pre-designed soft processors. For example, these memories require an arbitrary number of R/W ports for simultaneous R/W operations at any given time, which is accomplished by incorporating techniques that add extra logic and routing to these designs.

We also investigated several recent research works on multi-ported memory designs for FPGAs [10], [11], [26]–[29], [29]–[35]. These memory designs can be configured to have an arbitrary number of R/W ports, and are implemented using the existing dual-port BRAMs on FPGAs. In addition, specific techniques are developed to determine the memory bank that has the most recently written (or updated) data. In [10], the proposed multi-ported memory was designed using replication, coupled with a technique called Live Value Table (LVT), to track the memory bank that contains the “live” or most recently updated value for each memory location. Another design for multi-ported memory was introduced based on the XOR operations and also used replication [29]. A detailed comparison of the LVT-based [10] and XOR-based [29] designs can be found in [11]. The LVT-based multi-ported memory (in [10]) was used to implement a multi-ported cache memory in [31] with few modifications. In [27] and [28], modular multi-ported memories were designed using techniques called the invalidation-based LVT (I-LVT) and switched-ports respectively. In [26], a memory compiler was introduced for the design in [28] to optimize the multi-ported memory by making and solving a set cover problem. In [32]–[35], multi-ported memories were designed utilizing a technique similar to the XOR operations. These memory designs ( [32]–[35]) introduced a new banking structure in order to reduce the BRAM utilization. In [30], a new write scheme was introduced for the proposed multi-ported memory design. However, similar to the memory designs in [11],
Multi-Ported Memory (a)

Multi-Ported Memory (b)

FIGURE 1. Illustrating design and routing complexity of multi-ported memories using (a) complex structure vs. (b) simplified structure [14].

[29], this multi-ported memory (in [30]) requires forwarding circuitry, which was not mentioned or discussed in the paper [30]. A multi-ported memory design, composed of a “Data Distribution Scheme”, was introduced in [16] to accelerate compute-intensive applications on FPGAs. However, this memory design does not support simultaneous R/W operations, since the proposed data distribution scheme was not followed properly. With the aforementioned existing memory designs, the routing complexity increases significantly with the increasing number of R/W ports as well as with the increasing memory depths (capacity). In addition, various techniques incorporated to determine the most recently written data, add extra logic and routing, thus dramatically increasing the design complexity.

It should be noted that all the aforementioned memories are uni-directional port multi-ported memory designs. From these existing architectures and techniques, only the LVT-based technique was utilized to compose bi-directional multi-ported memory designs in the literature [11], [31]. In this case, the LVT-based technique typically utilizes many resources (Logic Elements (LEs) or Slices) on chip, thus making it infeasible to use for multi-ported memories with high memory depths [11], [31], since the occupied area (i.e., slices or LEs) on chip increases, with the increasing memory depths.

The above investigation indeed illustrates that the existing multi-ported memory designs become complex, mainly due to the extra logic and routing needed to create techniques to provide an arbitrary number of R/W ports. This sheer design and routing complexity hinders FPGA manufacturers (e.g., Xilinx and Altera) from employing these multi-ported memories with their next-generation FPGAs. Furthermore, from our previous work on FPGA-based designs [2]–[6], it was observed that using the existing dual-port BRAMs on FPGAs can have a significant negative impact on the speed-performance of real-time compute/data-intensive applications (e.g., data mining, machine learning, control systems) on embedded platforms.

After the above investigation and analysis, as shown in Figure 1, we divided the existing multi-ported memory designs (in the literature) into two categories [14]: complex structures versus simplified structures. These multi-ported memories were distinguished based on the design and routing complexity of the memory designs as individual entities, and considering the extra design and routing complexity needed to use these memories for embedded applications.

The complex structure, as shown in Figure 1(a), comprises a circular path. As illustrated, the Input Signals (such as address, control, and write-port data signals) to the BRAMs (i.e., On-Chip BRAM module in Figure 1) have to go through additional circuitry (i.e., Extra Logic & Routing module in Figure 1). The Output Signals (such as read-data output signals) from the BRAMs must also be forwarded to the output via this additional circuitry. Typically, the on-chip BRAMs, utilized to create the multi-ported memories, are distributed throughout the FPGA. Thus, to employ these multi-ported memories for embedded applications, the signals among the modules must also be propagated throughout the chip. This leads to a dramatic increase in the design and routing complexity of these embedded applications. When used as individual entities, the multi-ported memory designs with complex structures attain high maximum operating frequency. However, when integrated to an embedded application on the same FPGA, these multi-ported memory designs achieve low maximum operating frequency. Furthermore, in some cases, the embedded applications employing these multi-ported memories (with complex structures), might not fit into the chip (or FPGA) due to this sheer routing complexity.

The simplified structure, as depicted in Figure 1(b), does not consist of a circular path. As demonstrated, the Input Signals go directly to the BRAM, and not via the extra logic and routing circuitry as in the complex structure. In addition, only the read-data output signal from the BRAM is forwarded to the output via the extra logic and routing circuitry module (in Figure 1(b)). This significantly reduces the design and routing complexity of these multi-ported memories, while simplifying the design process. This also lessens the design and routing complexity of the embedded applications that employ these multi-ported memories (with simplified structures), since fewer signals are being propagated throughout the chip. The simplified structures also lead to high maximum operating frequency and high area-efficiency.

Most of the existing multi-ported memory designs, [10], [11], [26]–[29], [29]–[35], discussed above, comprise complex structures, and only our previous works, [5], [14], [15], for uni-directional multi-ported memories, fall into simplified category. From our above investigations and to the best of our knowledge, we did not find any multi-ported memory designs that eliminated the circular paths and comprised simplified structures in the published literature, similar to ours.

III. OUR PROPOSED MULTI-PORTED MEMORY ARCHITECTURES

In this section, we discuss and present our unique multi-ported memory architectures, which comprise simplified structures. First, we summarize our previous work on uni-directional port multi-ported memory designs, and then discuss and present our novel, unique, customized, and optimized bi-directional port multi-ported memory designs in detail (in Section III.A).
Previously, in [5], [14], [15], we introduced unique uni-directional port multi-ported memory designs, all of which comprised simplified structures. Our proposed uni-directional port multi-ported memories were generic and parameterized: thus can be configured to have an arbitrary number of R/W ports (in this case, \( m \) number of write ports and \( n \) number of read ports), varying number of memory banks, and varying memory capacities. The top-level architecture of our uni-directional port multi-ported memory design (for \( mW/nR \)) is shown in Figure 2 [14]. As illustrated, our designs comprised of \( m \) number of memory banks, and each bank comprised \( n \) number of simple dual-port BRAMs.

For our proposed multi-ported memory designs, we introduced a unique Decision Making Module (DMM), to find the last written data (or most recently written data). The DMM module: checks the counter values \( C_i \) for \( i = 1 \) to \( m \) of each BRAM in a column, for every read port; extracts the last written data value \( D_i \) from the outputs of the BRAMs; and forwards this data value to the read data output port \( R_{out} \).

With our unique DMM modules, only the read output data signal from the BRAMs would go to this module. Analogues to Figure 1(a), all the other signals such as write data, write addresses, read addresses would be forwarded directly to the BRAMs, without incorporating any extra logic between the modules. This dramatically reduced and simplified the design and routing complexity of our proposed multi-ported memory designs.

The DMM module was one of the major modules in our multi-ported memory designs. In addition, the DMM module was the only user-designed portion, which comprised logic that was combinatorial in nature, whereas all the other user-designed parts comprised signals/interconnects. Also, for our bi-directional port multi-ported memories, the internal architecture of the DMM module depends on the total number \( n \) of ports, whereas for our uni-directional port multi-ported memories [14], the internal architecture of the DMM module depended on the number of write \( (W) \) ports. Hence, creating customized and optimized internal architectures for the DMM module was crucial.

As mentioned above, we introduced four different unique, customized, and optimized internal architectures (as 4 models) for our DMM modules for varying number of R/W ports (as in Figure 3). In [14], we utilized these 4 models to create our uni-directional port multi-ported memory designs. The detailed description of the internal architectures of these four models can be found in [14]. We performed experiments and analysis to evaluate our uni-directional port multi-ported memories (with these 4 models) in terms of maximum frequency, number of occupied slices, and BRAM usage. We also evaluated our uni-directional port multi-ported memories (with these 4 models) with the existing uni-directional port multi-ported memories in the published literature in [10], [11], [29]. From our experimental results and analyses [14], it was evident that our proposed uni-directional port multi-ported memory designs, with the customized and optimized internal architectures (4 models) of the DMM modules [14], achieved higher maximum operating frequency and lower number of occupied slices compared to the existing ones in the literature [10], [11], [29]. For an example, at the memory depths of 1024, the maximum operating frequency of our proposed design (with 8W/16R uni-directional multi-ported memory configuration, utilizing Model 4) [14] was higher than the existing LVT-based 8W/16R un-directional multi-ported design by 36% [10], [11].

**A. OUR NOVEL BI-DIRECTIONAL MULTI-PORTED MEMORY ARCHITECTURES**

In this sub-section, we discuss and present our novel, unique, and efficient multi-ported memory designs with arbitrary
number of bi-directional ports for simultaneous multiple R/W operations. Conversely, in our previous work [14], [15], we only introduced unique multi-pored memory models and designs, which consisted of arbitrary number of uni-directional R/W ports for simultaneous multiple R/W operations. Our proposed bi-directional multi-ported memories (presented in this section) are designed in such a way to eliminate the circular paths that typically exists in the current multi-ported memories in the literature. In this case, to simplify the memory designs, only the read data signals from the BRAMs are processed using intermediate combinatorial logic before being forwarded to the read data output ports, while all the other signals, including write data, read and write addresses are directly forwarded to the write ports and R/W address ports of the BRAMs respectively without incorporating any intermediate logic between modules. This has led to a dramatic reduction in the design and routing complexity compared to the existing ones in the literature, while enhancing the operating frequency and area-efficiency.

Compared to the uni-directional multi-ported memories, bi-directional multi-ported memories have more advantages, including providing far more flexibility to the designers when being employed in embedded applications. In this case, bi-directional multi-ported memories are not limited to a fixed number of read transactions and a fixed number of write transactions, as in uni-directional ones. As a result, the amount of read and/or write transactions can be configured and varied, as needed on-the-fly at any time, based on the requirements of the applications. Advantages of bi-directional multi-ported memories over uni-directional ones are detailed with several examples in Section IV.D. Furthermore, from our previous work, it was observed that utilizing bi-directional multi-ported memories (compared to the uni-directional ones) can dramatically increase the speed-performance of certain embedded applications such as model predictive control applications [12], [13] and data mining/analytics applications [36], [37].

As mentioned in Section II, we investigated the existing works on bi-directional multi-ported memory designs for FPGAs. This investigation revealed that, thus far, there is only one existing multi-ported memory design in the published literature that provides bi-directional R/W ports [11]. This existing bi-directional multi-ported memory design was based on the LVT technique [11], [31]. In this case, the bi-directional multi-ported memory design was achieved by changing the organization of the BRAMs. Similar to the uni-directional multi-ported memories (in Section II), this LVT-based bi-directional multi-ported memories comprised a complex structure and occupied a large area on chip. Furthermore, with this LVT-based bi-directional multi-ported memory design, the occupied area on chip increased with the increasing memory depth, thus these memory designs can not be used for high memory depths.

In addition, we examined other existing techniques utilized for uni-directional multi-ported memories to determine whether these techniques can be modified and utilized for bi-directional multi-ported memories. Apart from the LVT-based designs, most of the uni-directional multi-ported memories utilized XOR-based techniques, which comprised complex algorithms/structures and forwarding circuitry. This sheer design and routing complexities of the XOR-based and other techniques for existing uni-directional memory designs (detailed in Section II) hinder them from being modified and utilized for bi-directional multi-ported memory designs.

Considering the aforementioned constraints and issues associated with the existing bi-directional multi-ported memories, in this research work, we design our bi-directional multi-ported memories in such a way to have simplified structures and without any circular paths. This is analogous to our previous uni-directional multi-ported memories in [5], [14], [15]. Our bi-directional multi-ported memories are designed and developed efficiently, to obtain the highest possible operating frequency with the minimum occupied area on chip, as well as to achieve very high memory depths. It should be
FIGURE 4. Individual BRAM block utilized for our bi-directional multi-ported memory design.

FIGURE 5. BRAM block distribution scheme for 3 to n bi-directional port multi-ported memories.

In this sub-section, we introduce a novel and unique BRAM block distribution scheme, with a novel and unique constraint/condition, to allocate/place the BRAMs among the n bi-directional ports of our multi-ported memory designs. In this case, the distribution (or the organization) of the BRAM blocks among the ports is imperative to enhance the performance of the bi-directional multi-ported memories. Figure 5 illustrates how the BRAMs are distributed among the ports for our bi-directional multi-ported memories. This figure specifically shows the distribution of the BRAMs for our multi-ported memory designs with varying (i.e., 3 to n) number of bi-directional ports. With our unique BRAM distribution scheme, it is important to satisfy one crucial constraint/condition, i.e., each port must have one and only one BRAM block shared with every other port. As in Figure 5, our BRAM distribution scheme allocates the BRAMs among the ports in such a way to satisfy this vital constraint/condition, and uses \((nX(n−1)/2)\) number of individual BRAMs to create our unique bi-directional multi-ported memories.

During the bi-directional port write mode, a specific port writes the input data to the BRAM blocks that are associated (or annotated) with the same port numbers. For instance, as illustrated in Figure 5, Port 1 will write to the BRAM blocks indexed with (1,2),(1,3),...,(1,n). Furthermore, as shown in Figure 4, the current counter value and the write data are concatenated and written to the corresponding individual BRAM block through Din port.

During the bi-directional port read mode, a specific port reads the output data from the BRAMs that are associated (annotated) with the same port numbers. For instance, as illustrated in Figure 5, Port 2 will read from the BRAMs indexed with (1,2),(2,3),...,(2,n). This read data is then sent to the our unique Decision Making Module (DMM), which is created to determine the last written data among all the BRAMs that are associated (annotated) with specific port number, i.e., specific R/W port number. In our design, we assign one DMM module per bi-directional port. Figure 4 demonstrates the single annotation of read operation on an individual BRAM block.

Next, we explain how our proposed BRAM block distribution with the aforementioned read/write operations would noted that the only existing LVT-based bi-directional multi-ported memory [11] in the literature can not be used for high memory depths (especially for memory depths > 512), and occupies significant area on chip. The analysis and comparison between our proposed bi-directional multi-ported memories and the existing LVT-based memories are presented in Section IV.E.

1) OUR NOVEL BRAM BLOCK DISTRIBUTION SCHEME FOR BI-DIRECTIONAL MULTI-PORTED MEMORIES

Our proposed bi-directional multi-ported memories are designed utilizing true dual-port BRAMs [8] on FPGAs with two bi-directional ports (e.g. A and B ports) for read and/or write operations. As illustrated in Figure 4, i and j are the number of two distinct ports of the true dual-port BRAM. The internal architecture of our multi-ported memory with n (arbitrary) number of bi-directional ports is shown in Figure 7, and detailed in Section III.A.2.
lead to the bi-directional multi-ported memory architecture, shown in Figure 7. As mentioned above, we place a novel and unique constraint/condition for our proposed BRAM block distribution. Based on this constraint/condition, each port has access to the data written by other ports as well as to the data written by its own port. This means that if a specific port wants to read from the multi-ported memory, that port has the access to all the data written by other ports as well as the data written by its own port. In this case, that specific port reads data from the BRAM blocks that are associated (or annotated) with the specific port number, i.e., specific R/W port number. Next, all the read data from these ports are forwarded to the DMM module, which has the same annotation. From these read data, the DMM module determines the last written data (or most recently written data), which is then sent to the read data output of our multi-ported memory (i.e., Dout port in Figure 7). During the read operation, since every port has access to all the data written by any other ports, DMM module only has to distinguish which data is written recently and sends this data to the read data output port for the selected bi-directional R/W port.

2) NOVEL INTERNAL ARCHITECTURE OF OUR N BI-DIRECTIONAL MULTI-PORTED MEMORIES

In this sub-section, we introduce novel and unique internal architecture of our generalized multi-ported memory design with n number of bi-directional ports, which is shown in Figure 7. As demonstrated in Figure 7, each port has an address bus (Addr_i), which provides the address of the memory location for read/write data of our multi-ported memory design. Each port also has separate data buses for writing data (Din) and reading data (Dout). Write enable signal (WE) informs the memory whether a request is a write request (WE = 1'b1) or a read request (WE = 1'b0). The hardware designers need to consider these signals based on the requirements of the application.

As depicted in Figure 7, our proposed bi-directional multi-ported memory architecture comprises m = (nX(n−1)/2) number of true dual-port BRAMs. Each of these BRAMs consists of two independent ports (Ports A and B). Each port also consists of its own signals, for instance, Port A comprises AddrA, DinA, DoutA, and WE_A. Furthermore, in our internal architecture (Figure 7), n number of our unique DMM modules are assigned to the n number of ports. As mentioned above, we use 3 separate busses for address (Addr_i), read data (Dout), and write signals (Din and WE). Although not shown in Figure 7, our bi-directional multi-ported memory design also comprises a counter.

Next, we explain the process for the read and write operations of our proposed bi-directional multi-ported memory design for Port 1 (in Figure 7).

a: DURING THE READ OPERATION

Initially, the address of the memory location of the data to be read is provided in Addr_1, and the WE_1 is set to 0 (zero) for a read request. In the next active edge of the clock, the read data is captured from Dout_1. In this case, via the address bus, Addr_1 is forwarded to the Addr of the BRAMs allocated to Port 1 (based on our proposed BRAM block distribution.
scheme in Figure 5). Furthermore, via the write bus, \( WE_1 \) is forwarded to the WE of the BRAMs allocated to Port 1. In addition, via the read bus, \((n−1)\) \( Dout \) values of the BRAMs (allocated to Port 1) are forwarded to the DMM1 module (as shown in Figure 7). Then the DMM1 module determines which data is the most recently written data (out of \((n−1)\) \( Dout \) values) and forwards that value to the \( Dout_1 \).

It should be noted that each individual BRAM block has 2 ports: Port A and Port B. In order to determine which BRAM ports are assigned to the input ports of our bi-directional multi-ported memory, we can use our proposed BRAM block distribution scheme shown in Figure 5. For instance, if a BRAM block is annotated with (3,5); that means Port A is assigned to input port 1 and Port B is assigned to input port 5 of our bi-directional multi-ported memory design.

b: DURING THE WRITE OPERATION
Initially, the address of the memory location of the data to be written is provided in \( Addr_1 \), the \( WE_1 \) is set to 1 (one) for a write request, and the write data is provided in \( Din_1 \). Similar to the read operation, in this case also, via the address bus, \( Addr_1 \) is forwarded to the \( Addr \) of the BRAMs allocated to Port 1 (based on our proposed BRAM block distribution scheme in Figure 5). Furthermore, via the write bus, \( WE_1 \) is forwarded to the WE of the BRAMs allocated to Port 1. In addition, via the write bus, \( Din_1 \) (i.e., the write data) is concatenated with the counter signal, and is forwarded to the \( Din \) of the BRAMs (allocated to Port 1). In the next active edge of the clock, the \( Din_1 \) data value is written to the BRAMs (allocated to Port1).

IV. EXPERIMENTAL RESULTS AND ANALYSIS
We perform experiments to evaluate and illustrate the feasibility and efficiency of our proposed bi-directional multi-ported memory architectures. We use Xilinx Virtex-6 XC6VHX380T FPGA [38], [39], which consists of large on-chip logic resources (59,760 slices) and 768 36Kb on-chip BRAMs, and has the highest available speed grade for Virtex-6 FPGAs. Xilinx ISE 14.7 is used to synthesize and implement our bi-directional multi-ported memories on the FPGA, whereas ModelSim SE and Xilinx ISim are used to verify the results and functionalities of the designs. We simulate each design with various test cases. In our bi-directional multi-ported memories, all the modules are designed in Verilog, while BRAMs are generated using Xilinx IP core generator.

Initially, we create a C++ code to generate our memory designs automatically. This has led to the implementation of 356 individual designs, with approximately 144,000 lines of Verilog codes. As detailed above, the total number of bi-directional multi-ported memory configurations is 356. In this case, we implement each configuration 5 times (which led to 1,780 implementations), and the average experimental values are presented and discussed in the following sub-sections.

The experimental results for our proposed bi-directional port multi-ported memory configurations using four models of 128K. These are implemented using our 4 different DMM models with varying internal architectures as in [14]. For all our bi-directional multi-ported memory designs, the word-size of the memory and the counter size are 32-bit and 4-bit, respectively, and are constants. As detailed above, the total number of bi-directional multi-ported memory configurations is 356. In this case, we implement each configuration 5 times (which led to 1,785 implementations), and the average experimental values are presented and discussed in the following sub-sections.

The experimental results for our proposed bi-directional port multi-ported memory configurations using four models
with different DMM modules for varying number of bi-directional ports for memory depths of 2, 512, 1024, 2048, 4096, 8192, and 16384 are presented in Tables 1, 2, 3, 4, 5, 6, and 7 respectively. As in column 1 of these tables, the number of bi-directional ports is varied from 3 to 8. For each bi-directional multi-ported memory configuration as well as for each model, we obtain the maximum operating frequency ($F_{\text{max}}$), the number of occupied slices on chip, and the number of 36Kbit BRAMs used. As shown in Table 7, we could not obtain the experimental results for the memory depth of 16384 with 8 bi-directional port multi-ported memory configurations. Furthermore, as shown in Tables 1 to 7 (row 1), for 3 bi-directional port multi-ported memory configurations, all 4 models produce the same results in terms of the maximum operating frequency and the number of occupied slices.

### TABLE 6. Bi-directional port multi-ported memory configurations with memory depth of 8192.

| No. of Ports | Model 1 | Model 2 | Model 3 | Model 4 | BRAMs |
|--------------|---------|---------|---------|---------|-------|
|              | Fmax    | Slices  | Fmax    | Slices  |       |
| 3            | 107.34  | 226     | 107.34  | 226     | 48    |
| 4            | 91.90   | 507     | 93.31   | 513     | 88.17 | 209   | 127.78 | 210   | 96    |
| 5            | 97.49   | 915     | 102.56  | 953     | 95.67 | 904   | 110.59 | 948   | 150   |
| 6            | 74.11   | 1172    | 77.16   | 1405    | 66.58 | 1308  | 101.10 | 1065  | 240   |
| 7            | 77.88   | 1773    | 80.67   | 1841    | 89.29 | 1921  | 94.67  | 2037  | 356   |
| 8            | 69.73   | 2300    | 67.50   | 2555    | 72.84 | 2472  | 91.99  | 3231  | 469   |

### TABLE 7. Bi-directional port multi-ported memory configurations with memory depth of 16384.

| No. of Ports | Model 1 | Model 2 | Model 3 | Model 4 | BRAMs |
|--------------|---------|---------|---------|---------|-------|
|              | Fmax    | Slices  | Fmax    | Slices  |       |
| 3            | 114.24  | 392     | 114.24  | 392     | 114.24 | 392   | 96     |
| 4            | 106.12  | 759     | 100.12  | 759     | 100.12 | 759   | 138.75 | 793   | 192   |
| 5            | 81.30   | 1024    | 77.16   | 1428    | 89.25  | 1469  | 102.32 | 1456  | 320   |
| 6            | 65.96   | 2199    | 64.05   | 2199    | 64.59  | 2101  | 87.56  | 2283  | 480   |
| 7            | 69.38   | 2952    | 62.45   | 3029    | 64.41  | 3074  | 80.47  | 3475  | 672   |

### A. TIMING ANALYSIS – MAXIMUM OPERATING FREQUENCY

The maximum operating frequency ($F_{\text{max}}$) is one of the main criteria for performance analysis for multi-ported memory designs. As discussed in Section II (existing works), all the existing multi-ported memories can perform up to a certain frequency threshold. Also, multi-ported memory architectures typically add extra logic and routing to the design, which in turn would lead to the reduction in maximum operating frequency. Hence, it is imperative to perform timing analysis on our proposed bi-directional multi-ported memory configurations as well as to illustrate the efficiency of our designs in terms of maximum operating frequency.

Figure 8 (generated from Tables 1 to 7) demonstrates the graphs of maximum operating frequency (in MHz) versus memory depths with varying number of bi-directional ports for Models 1, 2, 3, and 4 respectively. Typically, with the increasing number of bi-directional ports as well as with the increasing memory depths, the maximum operating frequency is expected to decrease. This statement is correct in high-level scenario, for instance, when considering the maximum operating frequency ($F_{\text{max}}$) for 3 versus 8 bi-directional ports, as shown in Figure 8 (and Tables 1 to 7). Conversely, there are discrepancies in these graphs; for instance, when considering $F_{\text{max}}$ for 4 versus 5, and 6 versus 7 bi-directional ports. As stated in [11], [15], these discrepancies (or inconsistencies) might be due the way the FPGA-based CAD tools (in our case, Xilinx ISE Design Tools) interpret and realize the internal architecture of our proposed bi-directional multi-ported memory configurations on Virtex-6 FPGA, thus impacting the operating frequency of our designs.

As illustrated in Figure 8 (and from Tables 1 to 7, row 1), our bi-directional port multi-ported memory configurations have the same maximum operating frequencies for all 4 models with the same memory depths. For instance, $F_{\text{max}}$ is 157MHz with the memory depth of 2, and 114MHz with the memory depth of 16K for all 4 models. Also, from Tables 1 to 7, Model 4 has the highest maximum operating frequencies for our 4 to 8 bi-directional port multi-ported memory configurations (with varying memory depths) for all 4 models. For instance, for 4 bi-directional port multi-ported memory configurations, maximum operating frequency varies from 116MHz to 173MHz for Model 4, whereas for other models maximum operating frequency varies from 88MHz to 1155MHz. This demonstrates that Model 4 is a better option for these bi-directional multi-ported memory configurations. Furthermore, maximum operating frequencies are quite similar for Models 1, 2, and 3 for most of the 4 to 8 bi-directional port multi-ported memory configurations with varying memory depths. This might also be due to the way CAD tools interpret and realize our designs on the FPGA (as stated in [11], [15]), making it difficult to discern these three models.

### B. SPACE ANALYSIS – NUMBER OF OCCUPIED SLICES

Since multi-ported memory architectures add extra logic and routing to the design, another important evaluation criterion is the space (or area) analysis in terms of number of occupied slices on chip. Figure 9 (generated from Tables 1 to 7) demonstrates the graphs of number of occupied slices versus memory depths with varying number of bi-directional ports for Models 1, 2, 3, and 4 respectively. Typically, with the increasing number of bi-directional ports as well as with the increasing memory depths, the number of occupied slices is expected to increase. This statement is true for all 4 models, especially for increasing number of bi-directional ports. For instance, as illustrated in Figure 9, the number of occupied slices increases with the increasing number of bi-directional ports for all 4 models. Also, at a glance, the number of occupied slices is the same for the varying memory depths from 2 to 4096 for all 4 models, which contradicts the aforementioned statement. These inconsistencies are minor (thus negligible) and might also be due to the way CAD tools interpret and realize our designs on the FPGA, as stated in [11], [15].
same number of occupied slices for all 4 models with the same memory depths. For instance, the number of occupied slices is 73 for memory depth of 2 (Table 1, row 1), and 392 for memory depth of 16K (Table 7, row 1) for all 4 models. Furthermore, the number of occupied slices is quite similar for all 4 models for most of the 4 to 8 bi-directional port multi-ported memory configurations with varying memory depths. These results and analysis illustrate that our Model 4 is a better option for most of our bi-directional multi-ported memory configurations, not only in terms of maximum operating frequency but also in terms of number of occupied slices.

C. ANALYSIS OF BRAM UTILIZATION

Figure 10 demonstrates the graph of number of BRAMs versus the memory depths for our bi-directional multi-ported memory configurations. As shown in Tables 1 to 7, the number of BRAM usage is same for all the models with varying memory depths. As illustrated in Figure 10, the BRAM utilization remains constant for the memory depths of 2 to 1024 for the same number of bi-directional ports. However, the BRAM utilization increases with the increasing memory depths from 1024 to 16K. Furthermore, the BRAM utilization increases with the increasing number of bi-directional ports. For instance, from Table 1, for the memory depth of 2, the BRAM utilization is 6 for 3 bi-directional port, and 56 for 8 bi-directional port multi-ported memory configurations.

D. ANALYSIS OF MAXIMUM AVAILABLE MEMORY DEPTHS

Table 8 illustrates the maximum memory depths that our proposed multi-ported memory designs can be housed in Xilinx Virtex-6 XC6VHX380T FPGA. As shown, this table contains maximum memory depth information for not only our proposed $n$ bi-directional port (i.e., $n$ number of bi-directional ports) multi-ported memory configurations, but also for our proposed $n \times m$ uni-directional port multi-ported memory configurations. For the latter, $n \times m$ means either $nR/mW$ memory or $mR/nW$; for instance, when $n = 6$ then $6 \times 3$ means $6R/3W$ or $3R/6W$. In column 1, $n$ denotes the maximum number of simultaneous R (reads) and/or W (writes) for bi-directional port multi-ported memories as well as for uni-directional port multi-ported memories. For our $n$
bi-directional port multi-ported memories, the \( n \) number of bi-directional ports varies from 3, 4, 5, 6, 7, and 8 (Table 8, column 1). Furthermore, as illustrated in Table 8 (column 2), the maximum available memory depths for our \( n \) bi-directional port multi-ported memory configurations vary based on the number of bi-directional ports; for instance, for 3, 4, 5, 6, 7, and 8 bi-directional port multi-ported memories, maximum available memory depths vary from 128K, 64K, 38K, 26K, 18K, and 14K, respectively.

As demonstrated in Table 8, if \( n = 3 \), then we can either have 3 bi-directional port multi-port memory configuration or any one of the 3X1, 3X2, and 3X3 uni-directional port multi-ported memory configurations. In this case, for 3 bi-directional port multi-port memory, the maximum available memory depth is 128K (row 1, column 2). For 3X1, 3X2, and 3X3 uni-directional port multi-ported memory configurations, the maximum available memory depths vary (or decrease) from 128K, 64K, and 43K (row 1, columns 3, 4, 5) respectively. Hence, from Table 8, we can distinguish bi-directional port and uni-directional port multi-ported memory configurations into two categories: one in green vs. yellow and another in blue vs. yellow. In this case, considering the same rows, the maximum available memory depth is higher for bi-directional port multi-ported memory configurations (in yellow cells) compared to the uni-directional port multi-ported memory configurations in green cells. However, the maximum available memory depth is lower for bi-directional port multi-ported memory configurations (in yellow cell) compared to the uni-directional port multi-ported memory configurations in blue cells.

Based on the above two categories, the application designers could have higher maximum available memory depths by selecting bi-directional ports for the former, and by selecting uni-directional ports for the latter, which typically depends on the requirements of the application. For instance, if the designer needs a maximum number of 5 reads (5R) and no more than 1 write (1W), then it is better to use 5X1 (5R/1W) uni-directional port multi-ported memory configuration (row 3, column 3), since this configuration gives the maximum memory depth of 77K. However, if the number of simultaneous writes increases to 2 or if the maximum number of writes is not decided, then it is better to use 5 bi-directional port multi-ported memory configuration, which provides the maximum memory depth of 38K (row 3, column 2).

Let’s consider another example, where the designer needs 7 simultaneous reads (7R) and 7 simultaneous writes (7W), then the designer needs 14 simultaneous R and W transactions. In this case, if the designer selects 7X7 (i.e., 7R/7W) uni-directional port multi-ported memory configuration (row 5, column 9), then the maximum available memory depth is 8K. However, if the designer uses 7 bi-directional port multi-ported memory configuration, then maximum memory depth of 18K (row 5, column 2) can be leveraged. Then the issue is: 7 bi-directional port multi-ported memory configuration can only support 7 simultaneous transactions at a time. Typically, the application designers prefer to have higher available memory depths for their designs, since it gives them more flexibility to deal with the stored data. Also, having more data on the chip would lead to higher application performance, since off-chip RW transactions lead to high memory access latency, which in turn would impact the application performance. In order to address the above issue as well as to accommodate the application designers’ requirements, it is possible to pipeline the design and add a WAIT state (or a wait clock cycle). In this case, 7 reads can be performed in the first clock cycle and then 7 writes can be performed in the next clock cycle. In certain scenarios, the application designers would be able to conceal the WAIT state by overlapping it with a pipeline stage that takes more than one clock cycle.

From the aforementioned results and analysis, it is evident that bi-directional port multi-ported memories, in most scenarios, provide higher maximum available memory depths, while providing much higher flexibility for the application designers with selecting/modifying the number of simultaneous R/W ports, compared to the uni-directional port multi-ported memories. These features are imperative for the applications designers (both in the industry and in academia) to accelerate the real-time compute/data-intensive applications on embedded platforms.

### E. COMPARISON WITH EXISTING MULTI-PORTED MEMORIES

As mentioned in Section II and Section III.A, the only existing bi-directional port multi-ported memory is the LVT-based design proposed in [11], [31]. As detailed in Section II, LVT-based multi-ported memories have several issues, including very high design and routing complexity, and dramatic increase in occupied area on chip with the increasing memory depths. For instance, for high memory depths (> 8K), LVT-based designs exceed the available area on chip. In [11], the existing LVT-based bi-directional multi-ported memory designs were realized on a different FPGA device from a different FPGA vendor (i.e., on Altera Stratix FPGA) compared to ours on Xilinx Virtex-6 FPGA: thus, making it difficult to perform a fair comparison between these two designs. However, in [11], the LVT-based uni-directional multi-ported memory designs were realized on Xilinx Virtex-6 FPGA, and the experimental results were obtained and presented.

In general, the bi-directional multi-ported memories are more complex than the uni-directional multi-ported memories. Hence, the former occupies more area on chip compared to the latter, when utilizing the same memory.
designs/techniques. For instance, with our counter-based memory design, the number of occupied slices is higher for the proposed bi-directional multi-ported memories (in this paper) compared to our uni-directional ones proposed in [14]. This is typically true for the LVT-based memory design/technique; thus, the LVT-based bi-directional multi-ported memories should occupy more area on chip compared to the uni-directional ones [11]. Therefore, we decide to compare our proposed bi-directional multi-ported memories with the LVT-based uni-directional multi-ported memories [11] realized on the same Virtex-6 FPGA in terms of number of area and memory depths.

Figure 11 demonstrates the comparison of our proposed 3 to 8 bi-directional port multi-ported memories and the LVT-based 2R/4W uni-directional port multi-ported memory. As detailed in [11], [14], the 2R/4W multi-ported memory is the simplest uni-directional multi-ported memory design. As illustrated, even the simplest LVT-based uni-directional port multi-ported memory (i.e., 2R/4W) occupies higher area on-chip, especially for memory depths > 1024 [11], compared to our proposed bi-directional memory designs. Furthermore, the number of occupied slices increases dramatically with the increasing memory depths (i.e., for memory depths > 512); and the area for the LVT-based memory designs with memory depths higher than 8K exceeds the available area on chip. These experimental results and analyses clearly illustrate that our bi-directional multi-ported memories are far more efficient than even the simplest LVT-based uni-directional multi-ported memories in terms of area.

Also, from the experimental results and analysis in [11], it is obvious that the existing LVT-based bi-directional port multi-ported memory designs can not be employed for higher memory depths, especially for memory depths > 512 ([11], [31]), since these memory designs occupied significant amount of on chip resources. For instance, for 4 bi-directional port multi-ported memory designs (with the memory depth of 2048), the occupied area was 32768 ALMs ([11]) for the LVT-based design, whereas the occupied area is 159-203 slices for our design. Furthermore, our bi-directional port multi-ported memory can be employed for a wide range of memory depths (for memory depths ranging from 2 to 16384), compared to the existing LVT-based design.

The aforementioned results and analysis indeed demonstrate that our proposed bi-directional multi-ported memory designs are far more superior compared to the only existing LVT-based bi-directional multi-ported memory design, especially in terms of area and memory depths.

V. CONCLUSION AND FUTURE WORK
In this paper, we introduced novel, unique, and efficient bi-directional multi-ported memory architectures, in order to provide an arbitrary number \( n \) of bi-directional ports for simultaneous multiple R/W operations. To facilitate this, we introduced a unique internal architecture for the bi-directional multi-ported memory. We also introduced a unique BRAM block distribution scheme, with a unique constraint/condition, to allocate/place the BRAMs among the \( n \) bi-directional ports efficiently. Our proposed bi-directional multi-ported memories were designed in such a way to eliminate the circular paths that typically exists in the current multi-ported memories in the literature. To simplify the memory designs, only the read data signals from the BRAMs were processed using intermediate combinatorial logic before being forwarded to the read data output ports, while all the other signals, including write data, R/W addresses were directly forwarded to the write ports and R/W address ports of the BRAMs respectively, without incorporating any intermediate logic between modules. This has led to a dramatic reduction in design and routing complexity compared to the existing ones in the literature, while enhancing the operating frequency and area-efficiency. To the best of our knowledge, no similar work exists in the literature that provide bi-directional multi-ported memory designs without the circular paths.

We implemented different multi-ported memory configurations with varying number of bi-directional ports (from 3 to 8 bi-directional ports) and varying memory depths (from 2 to 128K), using our 4 different unique DMM models with varying internal architectures, which lead to 356 memory configurations. Next, we performed experiments on all 356 configurations to evaluate the feasibility and efficiency of our proposed bi-directional multi-ported memories. From our experimental results and analysis, it was observed that Model 4 achieved the highest maximum operating frequency, whereas the number of occupied slices was quite similar for all 4 models for most of the memory configurations. These illustrated that our Model 4 was a better option for most of our bi-directional multi-ported memory configurations, not
only in terms of maximum operating frequency but also in terms of number of occupied slices.

Our experimental and analysis also demonstrated that bi-directional multi-ported memories often provided higher maximum available memory depths, while providing much higher flexibility for the application designers with selecting (and modifying) the number of simultaneous R/W ports, compared to the uni-directional multi-ported memories. These features are imperative to accelerate the real-time compute/data-intensive applications on embedded platforms.

Apart from our proposed bi-directional multi-ported memories, there is only one existing bi-directional multi-ported memory design [11], based on the LVT technique, in the literature. Unlike our proposed memory designs, the LVT-based memories comprised complex structures, leading to higher design and routing complexity. Furthermore, bi-directional LVT-based memories occupied significant area on chip, and can not be used for high memory depths (especially for memory depths > 512 [11]). Conversely, our proposed bi-directional port multi-ported memory can be employed for a wide range of memory depths (for memory depths ranging from 2 to 16384). Our experimental results and analysis demonstrated that our proposed bi-directional multi-ported memory designs were far more superior compared to the only existing LVT-based bi-directional multi-ported memory design, especially in terms of area and memory depths.

Due to lower design and routing complexity, our simplified bi-directional port multi-ported memory architectures will enable seamless integration to the next-generation FPGAs and FPGA-based CAD tools with minimal design cost. As future work, we are planning to create specific architectures in the form of hard logic and interconnects, which will further facilitate the integration of our proposed multi-ported memory architectures with the next-generation FPGAs. These attempts will enhance the overall performance of the compute/data-intensive applications. Furthermore, we will incorporate HDL code optimization techniques (as in [40]) to further improve the performance of our multi-ported memory architectures, in terms of area-efficiency and maximum operating frequency. Utilizing our simplified, optimized, as well as highly flexible bi-directional multi-ported memories, application hardware designers will be able to significantly accelerate the real-time compute/data-intensive applications on embedded platforms.

REFERENCES

[1] D. G. Perera and K. F. Li, “Analysis of single-chip hardware support for mobile and embedded applications,” in Proc. IEEE Pacific Rim Conf. Commun., Comput. Signal Process. (PACRIM), Aug. 2013, pp. 366–376.

[2] D. G. Perera and K. Fan Li, “FPGA-based reconfigurable hardware for compute intensive data mining applications,” in Proc. Parallel, Grid, Cloud Internet Comput., Oct. 2011, pp. 100–108.

[3] D. G. Perera, “Chip-Level and reconfigurable hardware for data mining applications,” Ph.D. dissertation, Dept. Elect. Comput. Eng., Univ. Victoria, BC, Canada, May 2012.

[4] S. N. Shahrouzi and D. G. Perera, “Dynamic partial reconfigurable hardware architecture for principal component analysis on mobile and embedded devices,” EURASIP J. Embedded Syst., vol. 2017, no. 1, Dec. 2017, Art. no. 25.

[5] S. N. Shahrouzi, “Optimized embedded and reconfigurable hardware architectures and techniques for data mining applications on mobile devices,” Ph.D. dissertation, Dept. Elect. Comput. Eng., Univ. Colorado Colorado Springs, Colorado Springs, CO, USA, Dec. 2018.

[6] S. N. Shahrouzi and D. G. Perera, “Optimized hardware accelerators for data mining applications on embedded platforms: Case study principal component analysis,” Microprocessors Microsystems, vol. 65, pp. 79–96, Mar. 2019.

[7] M. A. Mohsin and D. G. Perera, “An FPGA-based hardware accelerator for K-Nearest neighbor classification for machine learning on mobile devices,” in Proc. 9th Int. Symp. Highly-Efficient Accel. Reconfigurable Technol., Jun. 2018, pp. 1–7.

[8] Xilinx. (Dec. 2012). LogiCORE IP Block Memory Generator PG058 (v7.3). [Online]. Available: https://www.xilinx.com/support/documentation/ip_documentation/blk_mem_gen/v7_3/pg058-blk_mem_gen.pdf

[9] Xilinx. TriMatrix Embedded Memory Blocks in Stratix IV Devices, Stratix IV Device Handbook. Accessed: Dec. 1, 2011. [Online]. Available: https://www.altera.com/en_US/pdfs/literature/hb/stratix-iv/stx4_siv51003.pdf

[10] C. E. LaForest and J. G. Steffan, “Efficient multi-ported memories for FPGAs,” in Proc. IEEE 18th Annu. ACM/SIGDA Int. Symp. Field Program. Gate Arrays (FPGA), 2010, pp. 41–50.

[11] C. E. LaForest, Z. Li, T. O’Rourke, M. G. Liu, and J. G. Steffan, “Composing multi-ported memories on FPGAs,” ACM Trans. Reconfigurable Technol. Syst., vol. 7, no. 3, pp. 1–23, Aug. 2014.

[12] A. K. Madsen and D. G. Perera, “Efficient embedded architectures for fast-charge model predictive controller for battery cell management in electric vehicles,” EURASIP J. Embedded Syst., vol. 2018, no. 1, pp. 1–36, Dec. 2018.

[13] A. K. Madsen and D. G. Perera, “An optimized FPGA-based hardware accelerator for physics-based EKF for battery cell management,” in Proc. IEEE Int. Symp. Circuits Syst., (ISCAS), Seville, Spain, May 2020, pp. 1–5.

[14] S. N. Shahrouzi and D. G. Perera, “Optimized counter-based multi-ported memory architectures for next-generation FPGAs,” in Proc. 31st IEEE Int. System-on-Chip Conf. (SOCC), Sep. 2018, pp. 106–111.

[15] S. N. Shahrouzi and D. G. Perera, “An efficient embedded multi-ported memory architecture for next-generation FPGAs,” in Proc. IEEE 28th Int. Conf. Application-specific Syst., Architectures Processors (ASAP), Jul. 2017, pp. 83–90.

[16] S. N. Shahrouzi and D. G. Perera, “An efficient FPGA-based memory architecture for compute-intensive applications on embedded devices,” in Proc. IEEE Pacific Rim Conf. Commun., Comput. Signal Process. (PACRIM), Aug. 2017, pp. 1–8.

[17] B. Fort, D. Capalija, Z. G. Vranesic, and S. D. Brown, “A multithreaded soft processor for SoPC area reduction,” in Proc. 16th Annu. IEEE Symp. Field-Program. Custom Comput. Mach., Apr. 2006, pp. 131–142.

[18] R. Moussali, N. Ghanem, and M. A. R. Saghir, “Supporting multithreading in configurable soft processor cores,” in Proc. Int. Conf. Compil., Archit., Synth. Embedded Syst. (CASES), 2007, pp. 155–159.

[19] A. K. Jones, R. Hoare, D. Kusic, J. Fazekas, and J. Foster, “An FPGA-based VLIW processor with custom hardware execution,” in Proc. Arch. Synth. Syst. Des. 15th Int. Symp. Field-Program. Gate Arrays (FPGA), 2005, pp. 107–117.

[20] J. A. Fisher, “Very long instruction word architectures and the ELI-512,” ACM SIGARCH Comput. Archit. News, vol. 11, no. 3, pp. 140–150, Jun. 1983.

[21] M. A. R. Saghir and R. Naous, “A Configurable Multi-Ported Register File Architecture for Soft Processor Cores,” in Proc. Int. Workshop Appl. Reconfigurable Comput. (ARC), Mar. 2007, pp. 14–25.

[22] M. R. Saghir, M. El-Majrouh, and P. Aki, “Datapath and ISA customization for soft VLIW processors,” in Proc. IEEE Int. Conf. Reconfigurable Comput. FPGA’s ResConFij, Sep. 2006, pp. 1–10.

[23] P. Yiannacouras, J. G. Steffan, and J. Rose, “Application-specific customization of soft processor microarchitectures,” in Proc. Int. Symp. Field Program. Gate Arrays (FPGA), 2006, pp. 201–210.

[24] Xilinx. MicroBlaze Processor Reference Guide. Accessed: Mar. 1, 2020. [Online]. Available: https://www.xilinx.com/support/documentation/sw_manuals/xilinx11/mh_ref_guide.pdf
