International Journal of Pure and Applied Mathematics
Volume 119 No. 14 2018, 121-125
ISSN: 1314-3395 (on-line version)
url: http://www.ijpam.eu
Special Issue

ijpam.eu

Design of LVT Based Multiported Memory in FPGA

E.Navagiri[1], T.Sheela[2] T. Muthumanickam[3]

[1]M.E VLSI, [2]Asst. Prof., [3]Professor, Dept. of ECE, VMKVEC, VMRF (Deemed to be University) Salem, Tamilnadu.

Abstract

The Multi-ported memories are challenging to implement with FPGAs since the provided block RAMs typically have two ports only. We present a thorough exploration of the design space of FPGA-based multi-ported memories by evaluating conventional solutions to this problem, and introduce a new design that efficiently combines block RAMs into multi-ported memories with arbitrary numbers of read and write ports and true random access to any memory location, while achieving significantly higher operating frequencies than conventional approaches. Memories with more read/write ports can be extended from the proposed 2R1W/4R memory and the hierarchical 4R1W memory. Compared with previous xor-based and live value table-based approaches, the proposed designs can, respectively of BRAM usage for 4R2W memory designs with 8K-depth. This work proposes Live Value Table-Multipumping based approach that could improve performance in area, speed and increase the memory depth compared with the previous works. For complex multi ported designs, the proposed BRAM-efficient approaches can achieve higher clock frequencies by alleviating the complex routing in an FPGA. For 4R3W memory with 8K-depth, the proposed design can save 53% of BRAMs and enhance the operating frequency by 20%.

***Keywords**: FPGA, memory, multi-port, BRAM.*

1. Introduction

Field-Programmable Gate Arrays (FPGAs) have been broadly used in fast prototyping of complex digital systems. FPGAs contain programmable logic arrays, usually referred to as slices [4]. Slices can be configured into different logic functions. The flexible routing channels can support data transferring between logic slices. In addition to implementing logic operations, if needed, the slices can also be used as storage elements, such as flip-flops, register files, or other memory modules. Due to the increasing complexity of digital systems, there is a growing demand for in-system memory m1odules. Synthesizing a large number of memory modules would consume a significant amount of slices, and would therefore result in an inefficient design. The excessive usage of slices could also pose a limiting factor to the maximum size of a system that can be prototyped on an FPGA. To more efficiently support the in-system memory, modern FPGAs deploy block RAMs (BRAMs) that are hardcore memory blocks integrated within an FPGA to support efficient memory usage in a design. Compared with the storage module synthesized by slices, BRAMs are more area and power efficient while at the same time achieving higher operating frequencies. Multi ported memories, which allow multiple concurrent reads and writes, LVT-based approach would be limited in the depth of memory and in speed because multipumping approach is orthogonal to LVT-approach and LVT approach combines replicated and banking approaches. This work proposesthe LVT-multipumping based approach that could increase the memory depth, increase speed, and reduce the area.The rest of this paper is organized as follows: conventional multi-ported memory approaches and proposed approach are introduced briefly in session II. Session III describes the imple-mentation results of conventional and proposed approaches with Stratix FPGA of Altera that will show the benefit of this work.

2. Multi-Ported Memory Design Approaches

In this section, conventional approaches of designing multi-ported memory are described briefly. Each one will be introduced in the structure of circuits, operation, advantages, dis-advantages and corresponding applications. At the end of this section, proposed approach is presented to reduce most of the conventional approaches' disadvantages, to give effective performance that could satisfy the demand of modern designs.

2.1 Adaptive Logic Modules-ALM based approach

The first method is to implement Multi-Ported Memory by ALMs (Adaptive Logic Mod-ules) which are the basic building block of logic in the Stratix III architecture [2] and provide advanced features with efficient logic utilization. This design approach uses D locations memory (equivalent to log2D address width) with D (m-to-1) multiplexers (m is the number of write ports) in front of memory to select the suitable memory locations to write data and n (n is the number of read ports) (D-to-1) multiplexers behind memory to select suitable data to read from memory. The problem is that a very large area of ALMs will be used to implement 2 groups of multiplexers and memory; they will reduce the operation frequency and make the CADs tool take a long time to implement place & route in FPGAs. So, this method is not effective to implement multi ported memory when the number of ports as well as the data depth is large [11], [12]. Figure 1 illustrates the operation of multi ported memory using ALMs with m Write Ports and n Read Ports. In this figure, only the data signals are shown (the address, clock, enable signals are ignored).

Implementing multi-ported memory using ALMs usually takes more resource than using specific RAM blocks, because ALMs are usually configured to implement logic, arithmetic, and register functions, thus not good in both area and speed performance to implement memory [13]. RAM blocks would be concentrated to implement multi-ported memory.

2.2 Replicated approach

The replicated method as in Figure 2 is used to expand the number of read ports of memory by adding some extra copies of RAM blocks with the number of copies based on the number of read ports. This method is only used when the memory have only one write port

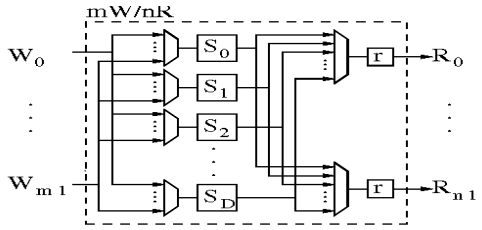

Fig. 1 mW/nR Multi-Ported Memory using ALM

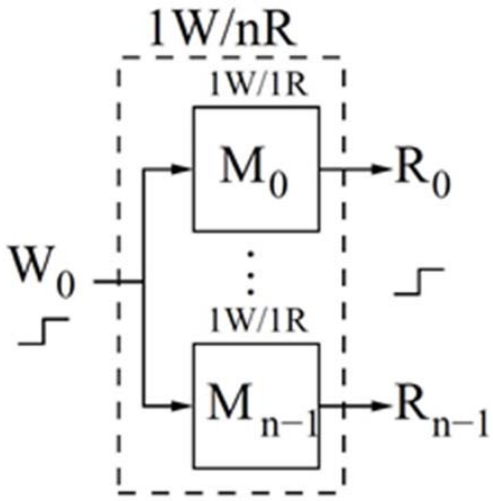

Fig. 2 1W/nR Multi-Ported Memory using Replicated approach

2.3 Banking approach

Compared with the replicated method, the banking method as in Figure 3 is different and can support an arbitrary number of read ports and write ports. Memory will be divided into m sections (corresponding to m write ports). So that, each write port and read port can only access to only one corresponding section. Therefore, this method does not support fully multi-ported memory because a write port cannot access to any location and similarly with the read port. And finally, this method must be combined with any other methods to create the truly multi-ported memory.

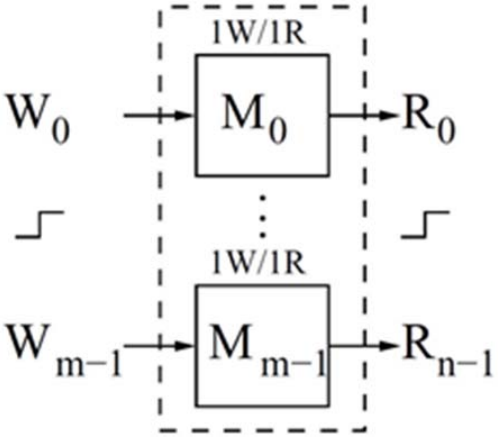

Fig. 3 mW/nR Multi-Ported Memory using Banking approach

2.4 Multipumping approach

This approach uses an external clock that is multiple speed of system clock (that called multipump factor) to control the memory location access process as in Figure 4. The temporary registers and multiplexers will be added to the circuit before and after the memory to multiplex the read/ write operations. For instance, a 1W/1R can support to 2 write ports and 2 two read ports with the multipump factor is 2. Therefore, multipumping approach helps to reduce the area of the design and is suitable for some applications that don't need high speed. On the contrary, the main disadvantage of this design is the reduction of the operation speed to multipump factor time, for example, multipumping with multipumping factor 2 will reduce the operation speed by two times.

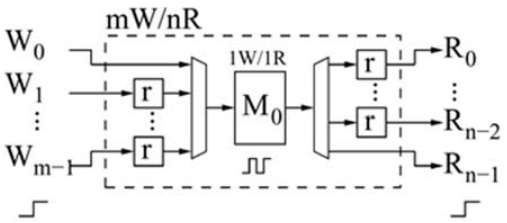

Fig. 4 mW/nR Multi-Ported Memory using Multipumping approach

2.5 Live Value Table-LVT approach

The LVT approach [10] was proposed from a form of indirection through a structure called the Live Value Table (LVT), which is itself a small multi-ported memory implemented in reconfigurable logic similar to Figure 1.

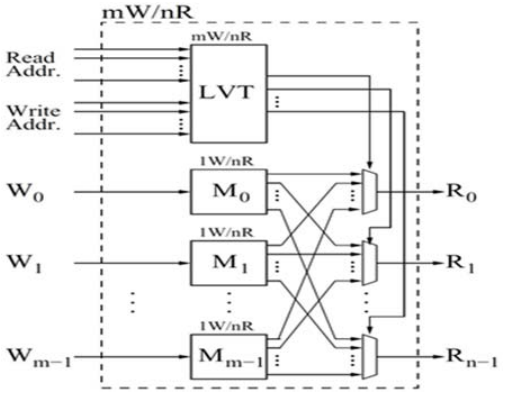

Fig. 5 mW/nR Multi-Ported Memory using LVT approach

This approach allows a banked design to behave like a true multi-ported design by directing reads to appropriate banks based on which bank holds the most recent or "live" write value as in Figure 5.

2.6 Proposed approach: LVT-Multipumping based approach

A novel approach is proposed to increase the performance of design by combining LVT approach, multipumping approach and additional circuitry to solve data contention, and re-duction of the operation speed due to multipump factor time. In this design, each write port will write data to corresponding memory bank being same as the banking approach. The read ports are constructed

similar to the replicated approach. The number of extra copied RAM blocks in each bank based on the number of read ports. Therefore, the mW/nR memory will be designed with m x n of the 1W/1R memory. Besides, the circuit takes responsibility to solve the data contention which banks are storing the newest data for each memory locations. From those results, each output port will select the suitable extra bank for reading each memory locations. Intuitively, this proposed approach makes the multi-ported memory have high operating speed, nearly equal to the RAM block speed; area cost is smaller than LVT-based approach when the number of ports increases because the resource equal mxn times of the real requirement of RAM blocks is taken and multipumping approach is used.

3. 1W/2R memory

To evaluate the performance of different methods to design multi-ported memory, a 1W/2R memory is chosen because it is simple, commonly used and supported by the most FPGA devices [3],[4]. In the Figure 6, the parameters of area and speed of different implemented methods are indicated with each point shown as the depth of memory (for example 32, 64, 128 and 256 bit). These results are used to discuss the conventional techniques for building multi-ported memories on FPGAs.

Figure 6 shows the comparison performance of different methods to implement 1W/2R memories: Pure-ALM (only use available logic elements), From these results would be the best approach because it costs little resource and obtains a high speed close to the speed of RAM block. This approach does not use different logic elements besides RAM block, but the disadvantage is that it is only applied when the number of write port is 1. For clear presentation and deep analysis in 1W/2R, results of area (ALMs) and speed (MHz) in each approach are given in Table 1-Table 5. In these tables, first column describes the memory depth from 32 to 256 bit. Next columns describe the area and speed

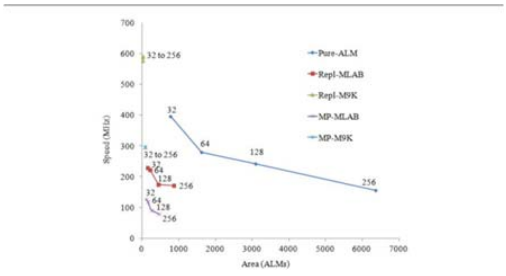

Fig. 6 Comparison Performance of Different 1W/2RMemories

1W/1R memory can be internally clocked at 2X the external frequency to give the illusion of being a 2W/2R memory. A multipumped design must also include multiplexers and registers to temporarily hold the addresses and data of pending reads and writes, and must carefully deÞne the semantics of the or-dering of reads and writes. While reasonably straight-forward, the drawback of a multipumped design is that each increase in the num-ber of ports dramatically reduces the maximum external operating frequency of the memory.

3.1 2R1W/4R(An Efficient Two-Mode Memory):

To brand new perspective of using a 2R1W module as either a 2R1W or a 4R module. This new way of using the 2R1W module is denoted as 2R1W/4R. This hybrid module canimplement HBDX in an efficient way, this paper introduces a support either 2R and 1Wor 4R. when there is no write request. In this case, the design can support up to four conflicting reads. Consider one of the worst cases when all the four reads (R0 to R3) are going to bank 0. As illustrated in Fig. 8(b), R0 and R2 access bank 0 directly. At the same time, R1 and R3 can retrieve the target data by XOR-ing the values at the same offset of the other data banks as well as the XOR-bank. when there is no write request. In this case, the design can support up to four conflicting reads. Consider one of the worst cases when all the four reads (R0 to R3) are going to bank 0. As illustrated in Fig. 8(b), R0 and R2 access bank 0 directly. At the same time, R1 and R3 can retrieve the target data by XOR-ing the values at the same offset of the other data banks as well as the XOR-bank.

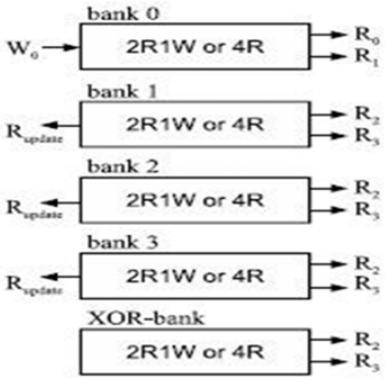

Fig. 7 HBDX 4R1W implemented with 2R1W/4R modules.

2R1W/4R Module: Fig. 2 illustrates a design scheme that can support more read ports by replicating the 2R1W module. However, this design scheme could significantly increase the usage of the limited BRAMs on an FPGA. To achieve a more BRAM-efficient design, this paper proposes HBDX, which adopts a hierarchical structure that organizes the 2R1W to achieve 4R1W without replicating the 2R1W module. To further enhance the design, HBDX in this section leverages the 2R1W/4R scheme introduced in the previous section as the basic building module to implement a 4R1W module. In this 4R1W design, each basic building block is a 2R1W module of the BDX scheme introduced in Fig1. According to the 2R1W/4R memory proposed in the previous section, this 2R1W module can be used as either a 2R1W or a 4R module.

3.2 Multi-ported memory 2W/4R

The implementation results of speed and area for 2W/4R memory configuration in different approaches are presented in Figure 7. The area (ALMs) in LVT-based approach of [10], and this work with LVT-multipumping approach is clearly better than Pure-ALM approach, especially when using M9K RAM blocks in FPGA. In using MLAB block, the approach in this work could help the designer to obtain the memory depth up to 256 bits that is larger than one in of up to 64 bits. This work approach in M9K implementation has the lowest area and highest speed because it has lower interconnection requirements and not the additional logic circuits similar with the ALM or LVT-MLAB implementations. When memory depth increases from 32 to 256, the speed in this work with M9K implementation decreases from 584 MHz to 343 MHz, thus this approach is suitable for the high

speed applications, for instance, FIFO and the registers for Instruction Level Parallelism (ILP).

In implementation result of area, the total area of design should be divided into different areas depending on the function of each block to help the designer to define which blocks affect significantly to the total area of design, and to help designer to decide which block should be optimized. This detail in area for different 2W/4R memories.

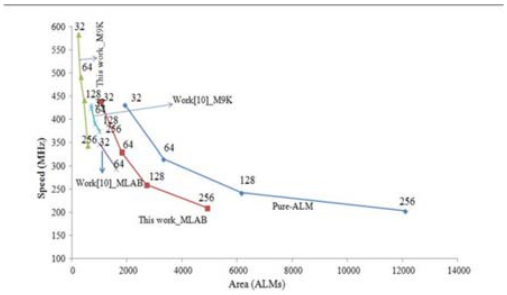

Fig. 8 Performance comparison of Different 2W/4R

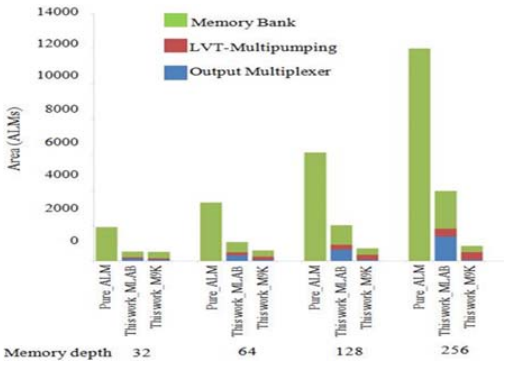

Fig. 9 Area detail in different approaches for different 2W/4R

4. Conclusion

Multi-ported memory takes a more and more important role on the applications that need sharing, queuing, synchronization between functional units. It helps to increase the process speed and to allow write and read memory banks from multiple ports at the same time. However, most FPGA devices only support 2 ports. Some conventional approaches such as ALM, banking, replication, multipumping retain disadvantages in timing, area, expanding the memory ports . . . The LVT-multipumping based approach in this paper is introduced to increase the performance of multi-ported memory in area and speed. This work could help the multi-ported memory designer to choose which approach is suitable for the specific application. One problem when increasing significantly number of memory ports is to take much RAM blocks to construct the required multiple ports memory. Potential work should be done to decrease the number of RAM blocks and still satisfy all memory requirements. By exploiting the 2R1W/4R as the building block, this paper proposes a hierarchical design of 4R1W memory that requires 33% fewer BRAMs than the previous designs based on replication. Memories with more read/write ports can be extended from the proposed 2R1W/4R memory and the hierarchical 4R1W memory. Compared with XOR-based and LVT-based approaches, the proposed designs can, respectively, reduce up to 53% and 69% of BRAM.

A multipumped design must also include multiplexers and registers to temporarily hold the addresses and data of pending reads and writes, and must carefully the semantics of the or-dering of reads and writes. While reasonably straight-forward, the drawback of a multipumped design is that each increase in the num-ber of ports dramatically reduces the maximum external operating frequency of the memory.

References

[1] Lange, H., Koch, A., Memory Access Schemes for Configurable Processors, International Conference on Field-Programmable Logic, 2000

[2] SAGHIR, M., AND NAOUS, R. A ConÞgurable Multi-ported Register File Architecture for Soft Processor Cores. In ARC 2007: Proceedings of the 2007 International Workshop on Applied Reconfigurable Computing (March 2007), Springer-Verlag, pp. 14Ð25.

[3] SAGHIR, M. A. R., EL-MAJZOUB, M., AND AKL, P. Datapath and ISA Customization for Soft VLIW Processors. In ReConFig 2006: IEEE International Conference on Reconfigurable Computing and FPGAs (Sept. 2006)

[4] Xilinx. 7 Series FPGAs Configurable Logic Block User Guide, accessed on May 30, 2016. [Online]. Available: http://www.xilinx.com/

[5] support/documentation/user_guides/ug474_7Series_CLB.pdf

[6] Xilinx. Zynq-7000 All Programmable SoC Overview, accessed on May 30, 2016. [Online]. Available: http://www.xilinx.com/ support/documentation/data_sheets/ds190-Zynq-7000-Overview.pdf