Turkish Online Journal of Qualitative Inquiry (TOJQI)
Volume 12, Issue 6, June 2021: 1285-1291

Design and Implementation of BRAM Memory for Reconfigurable Applications

S Logeswari[a], T Madhu Bala[b], B Nivetha[c], and S Karthika[d]

[a,b,c] UG Students, Department of Electronics and Communication Engineering, National Engineering College, Kovilpatti, Tamilnadu, India
[d]Assistant Professor, Department of Electronics and Communication Engineering, National Engineering College, Kovilpatti, Tamilnadu, India

email:[a]logevidhya@gmail.com,[b]tmadhubala992@gmail.com,[c]2000nivethab@gmail.com,[d]skarthikanec@gm ail.com

Abstract

To improve device performance, reconfigurable computing refers to including application-specific hardware in accordance with various applications. Dynamic reconfiguration allows a portion of the hardware to be modified by the device while the rest remains operational. Furthermore, the FPGA configuration registers access by an internal configuration access port(ICAP). By specializing in the machine for a particular application, reconfigurable computing increases system performance. Application reconfiguration can only increase the performance of the system if the time complexity exceeds the initialization period. This ensures that only the device efficiency of quasi-static applications can be enhanced by dynamic reconfiguration. Typical reconfiguration times were obtained in milliseconds and this still holds for most applications, despite ongoing study. This is because of their long period or the delay generated by the process of reconfiguration. Different attempts have been made to increase the system's throughput to rival that of the ICAP controller to overcome these drawbacks and migrate reconfigurable computing to complex applications. Reconfiguration may increase the utilization of the region as well. To demonstrate the feasibility of the proposed BRAM-based architectural design for the reconfiguration of real-time applications and to check the literature's proposed throughput is the only aim of this project.

Keywords:

1. Introduction

Generally speaking, the reconfiguration of an FPGA process greatly reduces device efficiency.

Reconfiguration refers to the use, in conjunction with general-purpose software, of application-specific hardware. The overhead for reconfiguration is the time it takes to create new hardware and the time it takes to migrate these new configurations to the configuration memory of the computer from an external memory location.

The method called hardware-controlled reconfiguration includes previously generating the new hardware, storing the configuration data in the block random access memory (BRAM) of the FPGA, and using a finite state machine based on hardware to facilitate the reconfiguration operation.

A BRAM sometimes called embedded memory or embedded BRAM is a discrete part of an FPGA, which means there are so many of them are present on the chip. They have dedicated blocks of memory and ideal for most memory requirements and they can use multiple blocks for large memories. Each FPGA has a unique number so that we can have more or less BRAM depending on the application. It constructs both single and real dualport RAM and writes and reads synchronously, which is different from distributed RA. They have width and depth that can be adjusted. For storing a large amount of data within the FPGA, block RAMs are used. And for lots of applications, they are useful. There are some of the BRAM features below,

- The on-chip bulk memory needed by many applications is supported.
- Massive aggregate bandwidth provides
- Support the ability to write bytes
- Has integrated logic from First In First Out
- Combined logic for error-correcting

The use of BRAM and a hardware-controlled reconfiguration method therefore, enhances the efficiency of the device. By specializing in the scheme for a particular application, it enhances device performance. A drop in power usage and part count are additional benefits.

2. Proposed System

The bitstream as .coe file is loaded into Block RAM using the Xilinx core generator. Use the 128-bit

configuration selector to pick the configuration. The reconfiguration controller is activated with the aid of a pushbutton after selecting the configuration. The bitstream loaded into the BRAM is then moved to the reconfiguration control. Then it moves to the ICAP (Internal Configuration Access Port) again, which serves as the configuration memory interface. The bitstream is then sent from the configuration memory to the MAC machine. Multiplication and addition are performed here and give the desired output. The configuration selected will be reflected in the LED connected to the MAC unit and the output of MAC will be reflected in the waveform.

3. Block Diagram

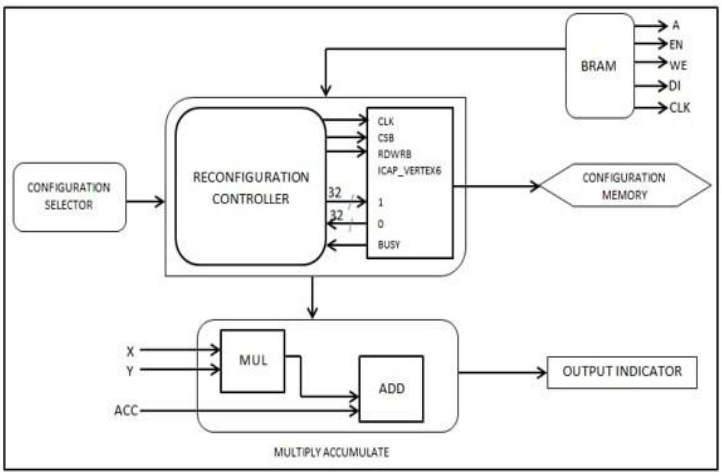

Fig.1 The architecture of MAC with the hardware-based reconfiguration

3.1 Block Ram

BRAM is called Block Random Access Memory. BRAMs are used to store a large amount of data within

your FPGA. They are one of the four commonly recognized elements on an FPGA data table. The remaining DSPs are Flipflops, Look-up Tables, and Optical Signal Processors. The bigger and more expensive the FPGA, the more BRAM that would have on it. A Block RAM is a different part of an FPGA (sometimes referred to as embedded memory, or Embedded Block RAM (EBR)), which means that many of them are available on the device. Depending on your application, you may need more or less BRAM. It comes in a finite size 4/8/16/32 kb and every FPGA has a different amount. There is an adjustable width and depth for them. And for many users, they're very useful.

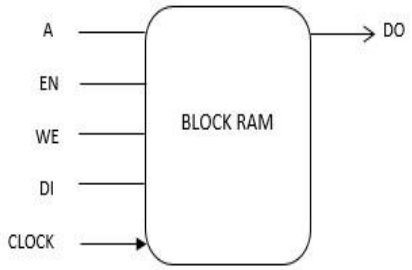

Fig .2 Logic diagram of Block RAM

3.2 Configuration Selector

The 128-bit configuration is selected using the configuration selector. The Dual-in-line Module (DIP)

switch serves as a configuration selector, which is activated by the PUSH button.

3.3 Reconfiguration Controller

The reconfiguration controller is responsible for reconfiguring the frequency parameters which

controlling the pulse width modulation(PWM) and it contains the multi-boot state machine. The external pushbutton is used to trigger the reconfiguration. Its time is measured from the moment the external trigger goes high to when the active-low LED, including the "DONE" state of the configuration process, illuminates. It provides functions for Partial Reconfiguration Designs. The RC pulls a partial bitstream from the memory and delivers it to an internal configuration access port when hardware or software causes events to occur (ICAP). The RC also assists with local decoupling and startup events, customizable per Reconfigurable Partition.

3.4 Internal Configuration Access Port

In Xilinx FPGA, the internal configuration access port (ICAP) is used to allow the user application to

configure the application structure at run time. Application circuits configured on the Xilinx Virtex series FPGAs can reconfigure the FPGA at run time using the on-chip ICAP. Traditional methods used by OPB-based and PLBbased systems to reach ICAP are overcomplicated and rarely reused.

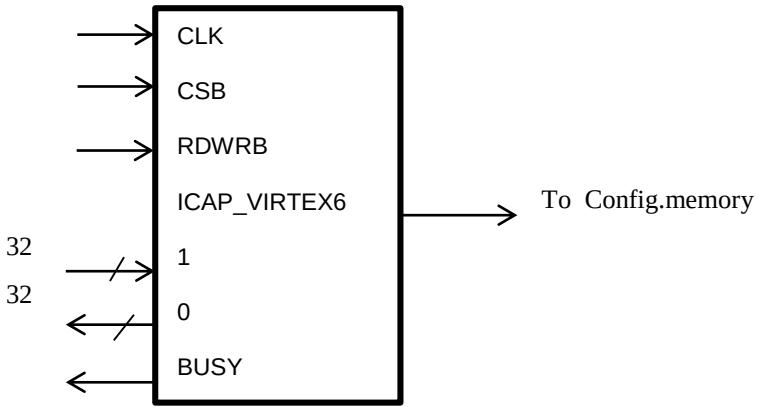

Fig.3 ICAP

3.5 Configuration Memory

For configuring low-cost SRAM, BRAM FPGAs as well as higher-density Xilinx and Altera high performance FPGAs, configuration memory can be used. These memory configuration devices have In-System Programmable (ISP) capabilities and can be used for the most common FPGAs and programmable SoCs as data storage.

3.6 MAC UNIT

A common step that determines the product of two numbers and adds that product to an accumulator is

the multiply-accumulate operation. The hardware unit that operates is known as a multiplier–accumulator (MAC, or MAC unit), the operation itself is also often called a MAC or a MAC Unit operation.

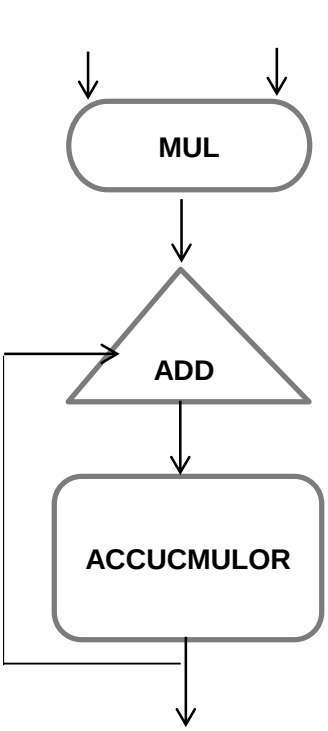

Fig.4 Multiply and Accumulate Unit

3.7 Output Indicator

Indicators, which are variables that transmit information on important aspects of the outcomes of the

simulation, are used to present the output. Performance measures are complementary to impact indicators and related to outcomes. The output indicator is connected to external LEDs and indicates the selected configuration based on the output of the Multiply Accumulate(MAC).

4. Design Flow

4.1 Bram Initialization

Using the reconfiguration information the Block RAM should be initialized and also called the bitstream. However, this faces different issues since the bitstream cannot be fed directly into the BRAM using the Xilinx Core generator, which supports .coe files. The bitstream contains binary data representing the FPGA's configuration bits, while the .coe file is a text-based file containing a header and configuration data for the Block RAM. A hexadecimal format is an example of an unformatted bitstream. The data are not grouped, as can be seen, which makes the data loading process more complicated.

Using Bit Gen the bitstream can be converted into American Standard Code for information interchange(ASCII). As can be seen, the data are grouped into 32-bit sets each representing a configuration command. This ASCII file can be easily be loaded into the BRAM as a .coe file. Alternatively, it can also be loaded into the BRAM on synthesis using the VHDL construct. This construct is capable of reading a text-based file containing the configuration data and initializing the BRAM.

A central component in the proposed reconfiguration architecture is the hardware required to facilitate the reconfiguration process.

4.2 Hardware Based Reconfiguration

Compared to conventional methods, a processor bus like processor local bus (PLB) is required for the use of hardware implemented on the FPGA to control the reconfiguration process is called Hardware controlled reconfiguration(HCR). A state machine is used to control the reconfiguration process. A multi-boot is a feature included in Xilinx FPGAs and the state machine is used for multiboot. In the event of a configuration

malfunction, operating failure, or single event upset, it allows an active program to fall back to a previous good configuration (known as the "golden image") (SEU). It also supports warm boot reconfiguration, which is a subset of fallback reconfiguration that requires only a portion of the system to be reconfigured without impacting the device's reminder.

5.Result

5.1.Simulation Result

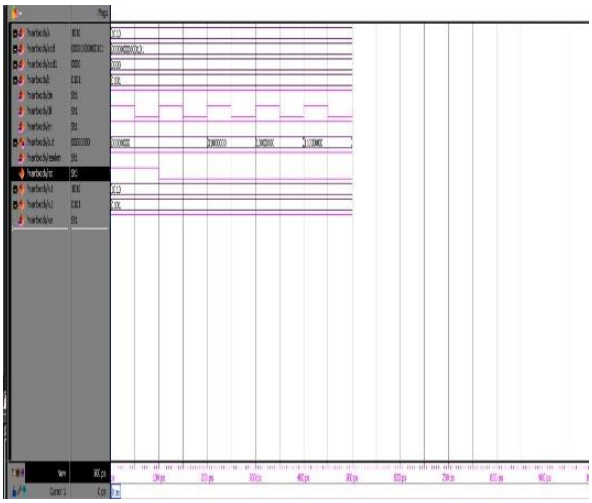

Fig.5 Simulation Result of Main Block

5.2.Result From Cadence

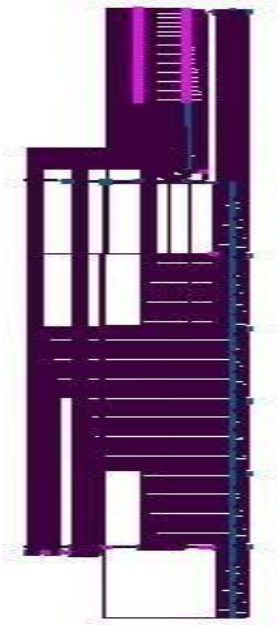

Fig.6 RTL View of Main Block

5.3.Comparision Of Existing And Proposed Work

Table 6.1. Criteria observed for Proposed Work

Parameter / Blocks	Area(in terms of cell area)	Power(in terms of nw)	Delay(in terms of ps)
Main Block	18686	42228.200	483
BRAM	2058	115661.82	322
Memory	32408	545601.18	509
MAC unit	611	32197.66	248
Multiplier	3864	34079.43	431
Ripple Carry Adder	576	111785.90	206
Full Adder	82	1565.75	611
Accumulator	95	1545.56	108

Table 6.2 Comparison of Existing work and Proposed work(Area)

Blocks	Existing work(Area (in terms cell area))	Proposed work(Area (in terms cell area))
BRAM	20500	19387
Memory	34789	23065
MAC unit	6785	5678
Multiplier	1478	1323
Ripple Carry Adder	3099	2734
Full Adder	2372	2364
Accumulator	6574	6543

Table 6.3 Comparison of Existing and Proposed work(Power)

Blocks	Existing work(Area (in terms cell area))	Proposed work(Area (in terms cell area))
BRAM	622895.2	543273.5
Memory	41237.74	32567.9
MAC unit	51383.12	41283.74
Multiplier	3456.89	2656.56
Ripple Carry Adder	1432.02	1256.34
Full Adder	723.67	645.82
Accumulator	362.78	275.63

Table 6.4 Comparison of Existing and Proposed work(Delay)

Blocks	Existing work(Area (in terms cell area))	Proposed work(Area (in terms cell area))
BRAM	8456	789
Memory	9456	8517
MAC unit	632	576
Multiplier	863	786
Ripple Carry Adder	518	428
Full Adder	812	775
Accumulator	745	624

The observation from above table 6.1 to 6.4 shows that area, power and delay were less compared to the existing work. Thus in the proposed work, it can be achieved by replacing it with BRAM structure which will decrease the usage of logic gates. This leads to a decrease in power, area, and delay in reconfigurable computing which reduces power, area consumption, and delays in RISC architecture compared to the existing one.

6.Conclusion And Future Work

In this project, an area and power-efficient of MAC with hardware-based reconfiguration architecture are presented. The proposed architecture is designed and several circumstances are taken into scrutiny like area, power, and latency. To evaluate power consumption, area, and delay, the proposed architecture is enforced on a Cadence. The obtained results are effectively reducing the power consumption, area, and latency.

Thus the proposed method of architecture using configuration selector 128-bit configuration is selected. The configuration selector will be reflected in the LED connected to the MAC unit and the output of MAC will be reflected in the waveform**.**

Significant improvement in area and delay is obtained. For demanding future applications we are currently considering four main ideas: more aggressive pipelining, better contention resolution, memory coherence, and use of dual-port RAM blocks.

References

[1] Le Roux, Rikus & Schoor, George & Van Vuuren, Pieter (2015). Block RAM-based architecture for real-time reconfiguration using Xilinx® FPGAs. South African Computer Journal. 56. 10.18489/sacj.v56i1.252.

[2] Kevin R.Townsend, Osama G.Attia, Phillip H.Jones and Joseph Zambareno(2019) "A Scalable Unsegmented

[3] Multiport Memory for FPGA-Based Systems" in International Journal of Reconfigurable Computing.10.1155/2015/826283

[4] Abdelhadi, Ameer & Lemieux, Guy (2016). Multi-Ported Memory Compiler Utilizing True Dual-port BRAMs. 10.1109/FCCM.2016.45.

[5] C.E.Lafaorest, J.L.Lin, and B.C.Lai, M.G.Liu, E.Rapati, and J.G.Steffan(2017) "BRAM based multi-ported memory designs on FPGA" in Integration.10.1109/VLSI-DAT.2015.7114526

[6] Huang, Kejie & Ha, Yajun & Zhao, Rong & Kuma, Akash & Lian, Yong(2014). A Low Active Leakage and High-Reliability Phase Change Memory (PCM) Based Non-Volatile FPGA Storage Element. Circuits and Systems I: Regular Papers, IEEE Transactions on. 61. 10.1109/TCSI.2014.2312499.

[7] Roman Iwanczuk and Steven P. Young(1998). "STRUCTURE AND METHOD FOR LOADING WIDE FRAMES OF DATA FROM A NARROW INPUT BUS", 09/128,964.

[8] Raymond C. Pang, Steven P. Young, Trevor J. Bauer(2018). "Block Ram with Configurable Data width And parity for use in Field Programmable Gate Array"

[9] Le Roux, Rikus & Schoor, George & Van Vuuren, Pieter(2015). Block RAM-based architecture for real-time reconfiguration using Xilinx® FPGAs. South African Computer Journal. 56. 10.18489/sacj.v56i1.252.

[10] Bonamy, Robin & Chillet, Daniel & Bilavarn, Sebastien & Sentieys, Olivier(2012). Power Consumption Model for Partial Dynamic Reconfiguration. 2012 International Conference on Reconfigurable Computing and FPGAs, ReConFig 2012. 10.1109/ReConFig.2012.6416772.

[11] Dessouky, Ghada & Klaiber, Michael & Bailey, Donald & Simon, Sven(2014). Adaptive Dynamic On-Chip Memory Management for FPGA-based Reconfigurable Architectures. Conference Digest - 24th International Conference on Field Programmable Logic and Applications, FPL 2014. 10.1109/FPL.2014.6927471

[12] Garcia, P., Bhowmik, D., Stewart, R., Michaelson, G., & Wallace, A. M. (2019). Optimized Memory Allocation and Power Minimization for FPGA-Based Image Processing. Journal of Imaging, 5(1), [7]. https://doi.org/10.3390/jimaging5010007.