IOSR Journal of Computer Engineering (IOSR-JCE)
e-ISSN: 2278-0661,p-ISSN: 2278-8727, Volume 18, Issue 4, Ver. V (Jul.-Aug. 2016), PP 36-41
www.iosrjournals.org

Structural Analysis of RF Architecture using Crossbar Type Topology for Cryptographic Application

Manisha Khorgade[1,1], Pravin Dakhole1,
[1] *Research Scholar, YCCE, Nagpur, India*

Abstract: *The strongly emerging class, Coarse-Grained Re-configurable Architecture (CGRA) is currently receiving attention and also having excellent performance with flexibility in fabrication. An entire range of components are now being offered as choices for system building blocks. The Reconfigurable fabric (RF) will be offered as an important building block for complex system design, CGRA processor. This paper gives an innovative design of reconfigurable fabric (RF) to achieve parallel processing techniques. Cryptographic hash function is a hash function which is considered practically impossible to invert, that is, to recreate the input data from its hash value alone. RF contains set of processing elements (PE) with crossbar type of arrangement for single dimensional processing of encryption technique to achieve reconfigurability. This implemented reconfigurable architecture design is evaluated by analyzing area, power, processing speed(delay to process) and area on various FPGAs using Xilinx 13.4.*

Keywords: *CGRA, Processing Element, Reconfigurable Fabric, crossbar type Modelling, FPGA, crypto graphical application*

I. Introduction

Increasing complexity of wireless and multimedia applications are driving the system designers to innovate continuously. A fundamental trade-off between flexibility and the traditional performance metrics (energy-area-timing) is being deemed important than ever. In such a scenario, the building blocks of modern System-on-Chip (SoC) - under critical review are those with a tunable balance of performance and flexibility. Coarse-Grained Reconfigurable Architecture (CGRA), with strong performance advantage as well as ability to be flexed post fabrication, is one such key building block. Recent design proposals, both academic and commercial, include CGRAs for DSP applications [1] [2] [3] or are completely based upon CGRAs [4] [5]. A Coarse-Grained Reconfigurable Architecture (CGRA) is a processing platform which constitutes an interconnection of coarse-grained computation units (viz. Processing Elements (PEs), Arithmetic Logic Units (ALUs)). These units communicate directly, viz. send-receive like primitives, as opposed to the shared memory based communication used in multi-core processors. CGRAs are a well-researched topic and the design space of a CGRA is quite large. The design space can be represented as a 7-tuple (C;N; T; P;O;M;H) where each of the terms have the following meaning: C - choice of computation unit, N - choice of interconnection network, T - Choice of number of context frame (single or multiple), P - presence of partial reconfiguration, O - choice of orchestration mechanism, M - design of memory hierarchy and H - host-CGRA coupling. Since the traditional application of FPGAs is mostly restricted to prototyping and emulation, design methodology of CGRAs is still an emerging field. As the defining roles of CGRAs in modern systems are getting clearer, design flows are being proposed. Regardless of the approach followed by these design flows, the central challenges of design remain same. The main part of RF unit is functional unit. It is like ALU .In this designing approach, it is considered as processing element(PE). There are various types of placement and routing techniques of reconfigurable fabric unit and processing element .In this paper we will discuss implementation of RF using Mesh type topology to analyze area, time ,throughput and speed for edge detection technique. Paper contains methodology for implementation in section 2, similarly section 3 and 4 will give implementation of PE and RF unit. Section 5 contains operation of system and analysis is studied and results are contained in section 6. Section 7 contains conclusion and after that references.

II. Methodology

The reconfiguration system of DSP processor architecture can be implemented using software and hardware techniques. Reconfigurable architecture has two types as coarse grained (CGRA) and fine grained (FGRA).CGRA works with byte wise input and FGRA works with bitwise input. Both are having their own advantages and disadvantages. Here we are considering CGRA which has reconfigurable fabric (RF) along with control unit and input and output unit.

The RF has the set of processing elements(PE). They are arranged with mesh type of topology. Here PEs are the homogeneous type. It accepts n-bit stream of inputs and n-bit stream of outputs. PEs can be arranged in other type of arrangements also like bus, crossbar. The main task is to schedule given task and distribute it among all PE to perform. During instruction scheduling, a record is maintained to keep track of which resources are used in each time slot. Considering that the scheduler for a CGRA must perform placement of operations, routing information should be recorded by the scheduler. This routing information can be included in the table because PEs is used both for computation and routing. Management of the interconnect network is not essential as all of the connections are dedicated point-to-point connections, meaning that no congestion can occur in the network. So that parallelism can be achieved and scaling with various applications.

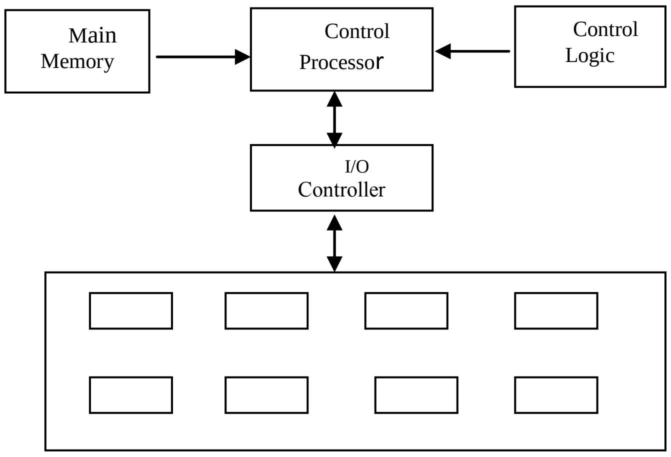

Fig.1. Block diagram of Reconfigurable Processor

In this approach task has been divided and feed into 2*8 arrangement of PEs to achieve parallism. For observing results of this arrangement various applications like image processing having edge detection technique with sobel operator is performed.

III. Implementation of PE

The processing element is the heart of RF unit. They are designed like a RISC. It perform the task for particular application send by control unit.PE is designed like RISC processor to perform certain task like ALU. It is RF unit performing set of functions ask to perform..This PE is taking 8 bit input and 8 bit output, PE is 8 bit. The designing is optimized in such a way that input-output stream can be change into 16 bit,32 bit,64 bit and so on .But it will change processing time and speed of operation. As bit size increases, processing time, speed also increases. Each PE contain input(bit stream),output(bit stream),clock, enable and operation to perform.PE has 2k RAM memory to store bit for small task and task is perform parallel to optimize time and speed. The RTL View of PE is given as follows which define the input and output stream with enable, clock also. The RTL View of PE is given as follows which define the input and output stream with enable, clock also. The RTL View of PE is given as follows which define the input and output stream with enable, clock also.

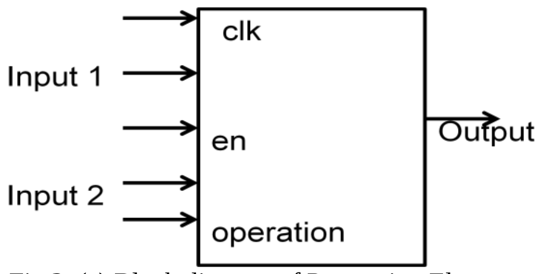

Fig.2. (a) Block diagram of Processing Element ;

IV. Implementation of Reconfigurable Fabric

Reconfiguration means multiple PEs are arranged with their different types of arrangement to perform various tasks at same time .If the decision about reconfiguration is done before running program ,that reconfiguration technique is considered to be static reconfiguration. In this approach 8 PEs,4 PEU and 2 Nodes to control logic are designed. The granularity is coarse grain type with mesh type of arrangement . Mesh-based architectures arrange their processing elements in a rectangular array, featuring horizontal and vertical connections. This structure allows efficient parallelism and a good use of communication resources. However, the advantages of a mesh are traded for the need of an efficient placement and routing step. The quality of this step can have a remarkable impact on the application performance. It is a homogeneous type of reconfigurable component i.e. even granularity.

Depending on the requirement of PEs as per task, scale-in and scale-out of unit decision taken by control node. With the help of this control unit, various applications can be run simultaneously, so with this reconfiguration is also achieved.

4.1 Interconnect topologies

It has been introduced various types of interconnect topologies. For most of the time, communication between processors (and their caches) with memory have been over a standard, shared-bus interconnect. The problem with this approach is that there will be significant contention on the interconnect because only one processor can be using the interconnect at any given time. The motivation for exploring other interconnect topologies is to allow for the possibility of optimizing various aspects of its design - for example, one may choose to optimize the latency or scalability of an interconnect system, or the actual cost of its real-world implementation might be worth optimizing instead.

A. Crossbar type

One type of interconnect topology is called a crossbar as shown in Fig 4(a). A crossbar has every node of the network connected to every other node. Because any node can send simultaneous messages to every other node in the system without conflicts, this network topology is non blocking. Furthermore, the direct connections allow for O(1) latency - each switch in the network directly connects the source node of a message to its destination node, so the message only has to traverse one "hop" in the network in order to be delivered. Crossbars also generally provide high bandwidth.

V. Operation

It has 3 bit operation mode ($B_2B_1B_0$) to perform depending upon the values of bits from MSB (B2) – LSB (B0). For selection of controller C1 and C2, B2 (MSB) is selected .It means value of MSB decides which controller is selected. If the value of B2 is 0, controller C1 is selected and if B2 is 1, Controller C1 and C2 are selected. C1 has 2 PEU. each and PEU has 2 PES, similar with C2 also. So if C1 selected, it will select 2PEU with 4 PEs. If C1 and C2 are selected, it will select 4 PEU with 8 PEs. If B1 bit is selected and bit is 0, C2 is selected which select PEU0 (2 PEs).If B1 is 1,C2 is selected which select PEU0 and PEU 1 (4 PEs). For third bit (B0), it will select controller C1. If B0 is 0, C1 is selected with PEU0 (2 PEs) and if B0 is 1, it will select PEU0, PEU1 (4 PEs).

Table 1. Bitwise distribution with controller, PEU &PE selection

Bit	Bit Value	Controller	PEU	PEs
B2	0	C1	--	--
	1	C1 & C2		
B1	0	C2	PEU0	2
	1	C2	PEU0,PEU1	4
B0	0	C1	PEU0	2
	1	C1	PEU0,PEU1	4

The following table give the core distribution for the input bit as explained in operation. As there are four cores, the allotted task is scheduled and distributed to the respective core and that will be get executed. For selection of one core delay is more and if four cores are selected delay automatically get reduced.

Table 2: Relationship of no of cores and task performed

Sr No	Input Bit	No of cores selected	Task performed	
1	00	1 core	Complete task	100%
2	01	2 core	Task divided in 2 equal parts	50%- core1
				50 % - core2
3	10	3 core	Task divided in 3 equal parts	1/3 part –core1
				1/3 part –core2
				1/3 part-core3
4	11	4 core	Task divided in 4 equal parts	¼ part – core1
				1/4 part –core2
				¼ part –core3
				¼ part –core4

VI. Results

The design is implemented and evaluate for cryptography application with hashing technique using xoring method. Processing is done for various bit streams sizes like 16bits, 32bits, 64bits,. The architecture is designed and implemented on Xilinx 13.4.To evaluate and observe the performance of architectreal desing it is dumped in various FPGA. Various parameters are considered for evaluations are time, area, power, speed and throughput.

6.1 Hashing function

A cryptographic *hash function* is a hash function which is considered practically impossible to invert, that is, to recreate the input data from its hash value alone. The input data is often called the *message*, and the hash value is often called the *message digest* or simply the *digest*. The ideal cryptographic hash function has four main properties as it is easy to compute the hash value for any given message, it is infeasible to generate a message from its hash also it is infeasible to modify a message without changing the hash.

Hashing is the transformation of a string of characters into a usually shorter fixed-length value or key that represents the original string. Hashing is used to index and retrieve items in a database because it is faster to find the item using the shorter hashed key than to find it using the original value. It is also used in many encryption algorithms. A database search mechanism would first have to start looking character-by-character across the name for matches until it found the match (or ruled the other entries out). But if each of the names were hashed, it might be possible (depending on the number of names in the database) to generate a unique four-digit key for each name.

The hash function is used to index the original value or key and then used later each time the data associated with the value or key is to be retrieved. Thus, hashing is always a one-way operation. There's no need to "reverse engineer" the hash function by analyzing the hashed values. In fact, the ideal hash function can't be derived by such analysis. A good hash function also should not produce the same hash value from two different inputs. If it does, this is known as a collision. A hash function that offers an extremely low risk of collision may be considered acceptable. Following Fig 5 shows the delay calculation for the proposed architecture .Here number of cores used like core one to core four. This implementation is done for hashing techniques using xoring method.

Cryptographic Application used hashing Techniques i.e. one way encryption technique by using XORing method implementation is done with the cross bar for the proposed design. Here the following Fig 3 illustrate that when we are giving input bit stream of 16 bit,32 bit or 64 bit ,it will gives us the relationship for the number of core used to the delay in thousand nanosecond.

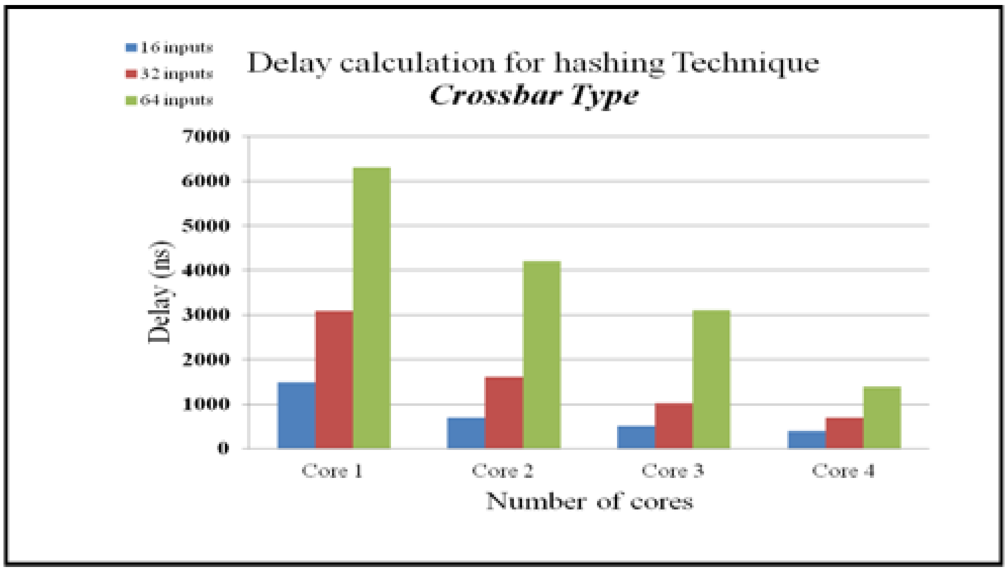

Fig 3 Delay calculation for cross bar type topology

The input to the proposed architecture changes from 16 bit inout,32 bit input and 64 bit input. As we are changing number of cores from one core to four cores respectively the delay time and accordingly speed also changes for the type of cross bar interconnecting topologies as shown in Table 4.

Table 4 : No of cores used and delay taken for processing for cross bar topology

Cryptography-Crossbar Type		
Input	**Core used**	**Delay (ns)**
16	**00**	**1484ns**
	01	**700ns**
	10	**505ns**
	11	**400ns**
32	**00**	**3095ns**
	01	**1600ns**
	10	**1011ns**
	11	**700ns**
64	**00**	**6300ns**
	01	**4198ns**
	10	**3106ns**
	11	**1395ns**

Similarly we have tried to calculate the power delay for all four cores for the input bit stream of 16 bit and 32 bit. As we are changing number of cores from one core to four cores respectively the power in watts increases up till core 3 and if four cores are used it will work with parallism it will reduced the power for 32 bit stream input for both type of cross bar interconnecting topologies as shown in Table 5 below. This power calculation is done by selecting the FPGA of Spartan 3E kit using power analyzer.

Table 5 : Cryptography power calculation(32 bit) Spartan3E

Sr.No	Value		
	Parameter	16 Bit	32 Bit
1	Core 1	0.01523w	0.01649w
2	Core 2	0.01797w	0.03569w
3	Core 3	0.02279w	0.01936w
4	Core 4	0.02091w	0.01139w

Also we have tried to calculate area occupied by the proposed architecture of reconfigurable fabric using FPGA. While dumping on the FPGA using Spartan 3E and Spartan 6 kit are selected. This family give the area covered with number of slice register, flip flops, LUTs, IOBs, Maximum number of path covered i.e. wire length. Also we have tried to count the CPU time for processing , Maximum frequency and also memory usage with these FPGA hardware board as shown below in Table 6 .So practically we have implemented the proposed design on software as well as hardware to give the comparison and new idea about implementation of RF for CGRA design.

Table 6: Area Calculations : Cryptography with Mesh type Arrangement

Sr No	Parameter	Cryptography			
		(16 Inputs)		(32 inputs)	
		Spartan6	Spartan3E	Spartan6	Spartan3E
1	Slice Register	337	333	469	2735
2	Flip-flop	1027	333	1796	461
3	LUTs	1069	2718	1779	5230
4	IOBs	261	261	517	517
5	Minimum Period	5.918ns	18.179ns	5.948ns	18.615ns
6	Maximum Frequency	168.98MHz	55.009MHz	168.136MHz	53.72MHz
7	Total number of paths/Destination ports	27593/337	1320294/381	3079856/469	3079856/513
8	Total Real time to Xst completion	10.00Sec	26.00sec	12.00sec	44.00sec
9	Total CPU time to Xst completion	9.87Sec	26.40sec	11.92sec	43.82sec
10	Memory Usage	249352KB	303048KB	252424KB	366664KB

Here we have few application for hashing techniques like verifying the integrity of files or messages, password verification, proof-of-work and also file or data identifier

VII. Conclusion

In this paper design of reconfigurable fabric of reconfigurable processor CGRA is presented. In particular, it is noted that reconfigurable computing is grown to be a vast discipline and therefore demands separate attention to the sub disciplines like reconfigurable processors. Hashing techniques in cryptography is used to analyze and tried to optimize proposed design with various parameters like area, speed, time, power and throughput. Anticipating the future challenges, several research directions are optimization of PE, RF with these parameters and many others. It is likely that our understanding of SoC architectures will evolve with time. The design will be apply for other application like communications, cryptography image/video decoding which give reconfiguration and scalability also.

References

[1]. “High-level Modelling and Exploration of Coarse-grained Re-configurable Architectures “, Amupam Chattopadhyay, Design, Automation and Test in Europe, DATE 2008, Munich, Germany, March 10-14, 2008,

[2]. Hyunchul Park, Kevin Fan ,“Modulo Graph Embedding: Mapping Applications onto CoarseGrained Reconfigurable Architectures”, Proceedings of the 2006 - dl.acm.org

[3]. “Low Power Reconfiguration Technique for Coarse-Grained Reconfigurable Architecture”, Yoonjin Kim, , Rabi N. Mahapatra, Ilhyun Park, IEEE TRANSACTIONS ON VERY LARGE SCALE INTEGRATION (VLSI) SYSTEMS, VOL. 17, NO. 5, MAY 2009

[4]. “Design Space Exploration for Efficient Resource Utilization in Coarse-Grained Reconfigurable Architecture”, Yoonjin Kim, , Rabi N. Mahapatra, Ilhyun Park, IEEE, IEEE TRANSACTIONS ON VERY LARGE SCALE INTEGRATION (VLSI) SYSTEMS, VOL. 18, NO. 10, OCTOBER 2010

[5]. “A Design Flow for Architecture Exploration and Implementation of Partially Reconfigurable Processors” Kingshuk Karuri, Anupam Chattopadhyay, Xiaolin Chen, David Kammler, Ling Hao, Rainer Leupers, IEEE TRANSACTIONS ON VERY LARGE SCALE INTEGRATION (VLSI) SYSTEMS, VOL. 16, NO. 10, OCTOBER 2008

[6]. “Reducing Control Power in CGRAs with Token Flow”, Hyunchul Park, Yongjun Park, and Scott Mahlke, DATE 2008

[7]. Allan Carroll, Stephen Friedman, “Designing a Coarse-grained Reconfigurable Architecture for Power Efficiency”, proceeding 08 Proceedings of the ACM/SIGDA international symposium on Field programmable gate arrays ,pages 161-170

[8]. Oliver C.S.Chay,Ray C.C. Cheung,”Reconfigurable computing :Architecture,tools and application”,by Sringer,8th international symposium ,ARC 2012,Hongkong,china,March 2012

[9]. “A Coarse Grained Reconfigurable Architecture Framework supporting Macro-Dataflow Execution”, thesis submitted Keshavan Varadarajan, december 2012 at IIS,Banglore.

[10]. ”SPR:an architecture –adaptive CGRA mapping tool”, Allan Carroll ,Stephen Fridmen, proceeding FPGA '09 Proceedings of the ACM/SIGDA international symposium on Field programmable gate arrays ,pages 191-200