NCRISET-2017 e-ISSN: 2456-3463
International Journal of Innovations in Engineering and Science, Vol. 2, No.6, 2017
www.ijies.net

Systolic Arrays for Reconfigurable DSP Systems

Rajashree Talatule
Department of Electronics and Telecommunication
G.H .Raisoni Institute of Engineering & Technology
Nagpur, India
Contact no.-7709731725
E-mail: rajashreetalatule20@gmail.com

Prof.K.S.Mankar
Department of Electronics and Telecommunication
G.H .Raisoni Institute of Engineering & Technology
Nagpur, India
contact no-7588749114
E-mail:Kanchan.mankar@raisoni.net

***Abstract*—Systolic arrays provide an alternative view of reconfigurable computing systems where real-time reconfiguration may be achieved by changing the sequence of events among the PEs forming the array. A rich selection of algorithms can be mapped into a systolic array structure .This paper is based on matrix multiplication using two methods i.e conventional and systolic architecture. . The simulation results have given that, the implementation of Systolic architecture requires less number of clock cycles then Conventional method.**

Keywords-systolic array, multiplier, digital signal processing, parallel architecture, pipelining, throughput, latency

I. INTRODUCTION

A reconfigurable computing system is a computing platform whose architecture can be modified by the software. In ASIC hardware, the logical functions of gates are fixed and cannot be modified. In FPGAs however, both the logic functions performed within the logic blocks and the connections between the blocks can be altered by sending signals to the chip. Traditionally FPGAs have been used as field programmable devices not as run-time reconfigurable devices. Several seconds or more have been required for a change in device configuration. New advances in technology are making it possible for FPGAs to be reconfigured in the microsecond range, and researchers are aiming at FPGA-based computing systems that would be able to adapt their hardware almost continuously in response to changes in the input data or processing environment.

II. THEORETICAL BACKGROUND

A. Matrix-Matrix Multiplication on Hardware

Computing matrix products is both a central operation in many numerical algorithms and potentially time consuming, making it one of the most well-studied problems in numerical computing. Various algorithms have been devised for computing C = AB, especially for large matrices. Mapping such algorithms to custom or general purpose hardware architecture is always a challenging task. By having a custom or ASIC hardware, the matrix-matrix multiplication can be implemented using least resources and can be accelerated to a large extent. Mapping the same algorithms on general purpose hardware, for example, implementing on general purpose Xilinx FPGA board always has inherent trade-offs such as area (power), time (maximum operating clock frequency), latency, hardware utilization efficiency and so on. The realistic way to compare two solutions would be to assign weights to each of these factors and choose a solution among multiple possible pareto optimal solutions.

B. Systolic Array Architecture

Systolic architectures (also referred to as systolic arrays) represent a network of processing elements (PEs) that rhythmically compute and pass data through the system. These PEs regularly pump data in and out such that a regular flow of data is maintained [1], [2]. As a result, systolic systems feature two important properties for VLSI design: • Modularity: Various functional blocks which make up the larger system have well-defined functions and interfaces. Hence, the concept of modularity enables the parallelisation of the design process. • Regularity: Hierarchical decomposition of a large system results in not only simple, but also sim- ilar blocks, as much as possible. The systolic array may be used as a coprocessor in combination with a host computer where the data samples received from the host computer pass through the PEs and the final result is returned to the host computer (see Fig. 1). This operation is analogous to the flow of blood through the heart, thus the name “systolic”. Typically, all the PEs in a systolic array are uniform and fully pipelined, i.e., all communicating edges among the PEs contain delay elements, and the whole system usually contains only local interconnections [3]. However, some relaxations have been introduced to increase the utility of systolic arrays. These relaxations include use of not only local but also neighbor (near, but not nearest) interconnections, use of data broadcast operations, and use of different PEs in the system, especially at the boundaries. With these relaxations, a family of modular, regular, and efficient data-driven array architectures can be designed for DSP applications, one of which is matrix-matrix multiplication.

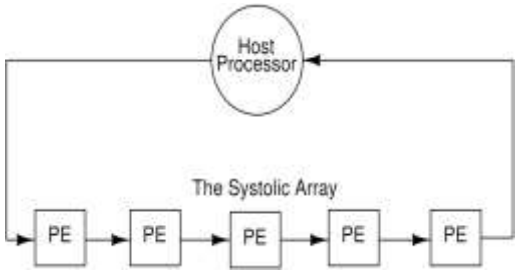

Figure 1. Systolic array concept

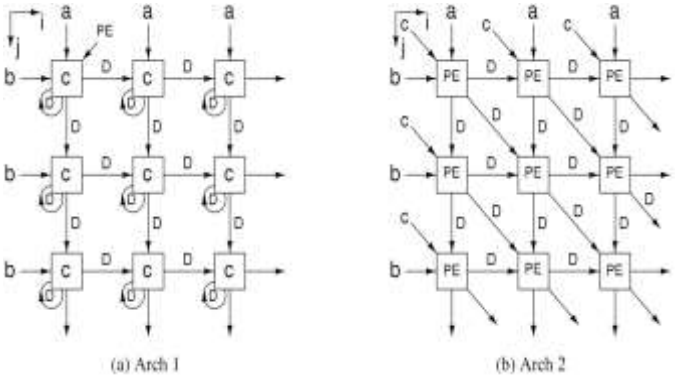

Figure 2. Two dimensional systolic array for matrix multiplication

Architecture Types

The computing elements in a systolic array have a number of serial and/or parallel I/O ports. For processors with 32-bit word lengths parallel I/O ports are impractical therefore serial I/O ports are used. For example, Intel's 8051 microcontroller has four 4-bit parallel I/O ports for a total of thirty two pins dedicated to data I/O. Whereas Texas Instruments' C6x and C54x families have one full-duplex serial I/O port called McBSP (Multichannel Buffered Serial Port) with a 128 TDM-channel capacity. The I/O ports can be used for communication with external devices such as A/D and D/A converters and interprocessor communication in multiprocessor systems.

In serial I/O ports data is input to the receive shift register one bit at a time, until an entire word is loaded. The word is then shifted in parallel and loaded into the main processor, which uses one instruction to move the data from the receive buffer into main memory. The receive buffer allows the processor to move the data asynchronously upon reception of the word through the serial port. A reverse sequence of operations is used to output parallel data words through the serial port. Multiple serial ports allow us to connect multiple PEs in an array. As we saw in the example above, the number of serial ports available within each PE varies and it is typically between one and six.

Figure 3 shows a block diagram for a serial interface. The transmit and receive data buffers are connected to the internal parallel data bus. The transmit and receive shift registers prepare the data for output and synchronization is achieved by means of interface control signals given to the serial control unit. The three different architectures are bus/pipeline, parallel and massively par- allel. Figure 4 shows the block diagram for the bus/pipeline architecture. Note that only two serial ports are needed and one is configured as a receive port while the other is configured as a transmit port. Figure 5 shows the port connections for the parallel array.

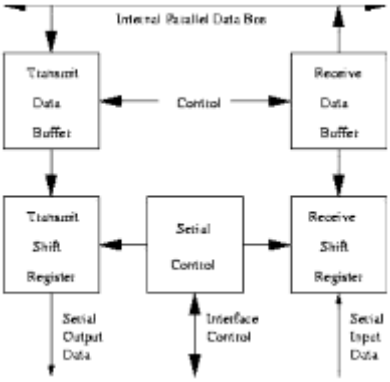

Figure 3. Serial interface structure

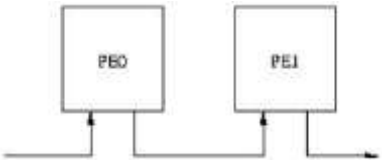

Figure 4. Bus pipelined architecture

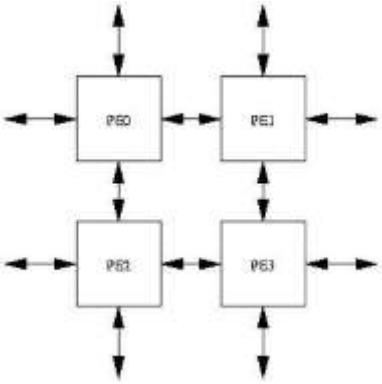

Figure 5. Parallel pipelined architecture

III. PROPOSED ARCHITECTURE

The Parallel Matrix Multiplication [7] has many different identifications, but all with the similar implementation. That is, they immediately multiplex a pair of matrix elements in special. Parallel Matrix Multiplication on Systolic Array (PMMSA) uses this approach. In [5], PMMSA is characterized by processing data input in pipeline and comprised of regularly arrayed PE. Where neighbor PEs are connected with each other by shortest line and therefore mass data has no need to be stored before processing. Decrease of distance between the PEs in an array greatly reduces the internal communication delay and improves the utility of processing units. It also removes time consumption for controlling the establishment

of data stream. In, this research, the PE is replaced with Multiplication and Accumulation (MAC) to enhance the speed and reduce the complexity of Systolic Architecture.

Figures 6 show matrix multiplication being performed on a systolic architecture. Figure 7 shows the final result of the multiplication. Figure 8 shows an alternative data injection mechanism. The computations involved in obtaining a single element of the output matrix are shown in figure 9.

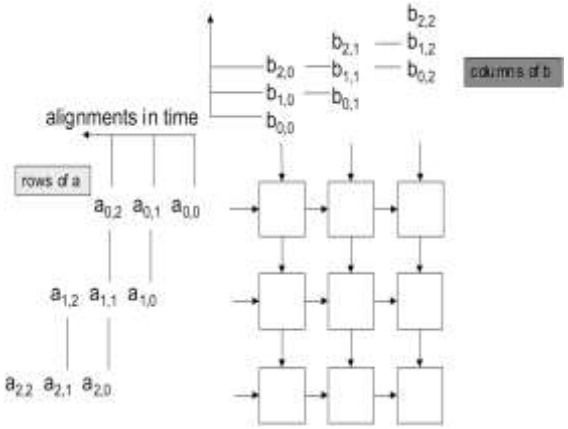

Figure 6. Systolic Matrix Multiplication

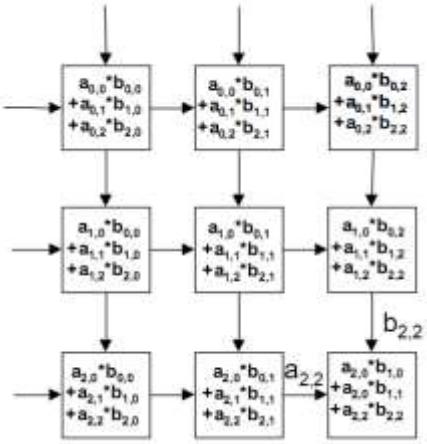

Figure 7. Fully populated configuration

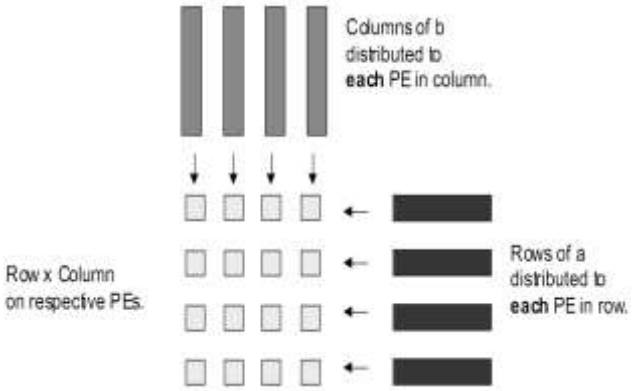

Figure 8. Data injection

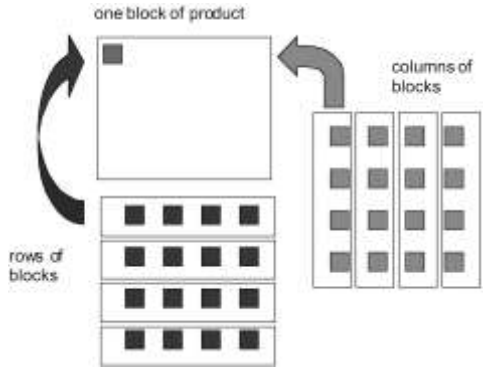

Figure 9. Computation of single element

The algorithm for the matrix multiplication of order N×N is shown bellow.

1. For I = 1 to N Start of for loop 1
2. For J = 1 to N Start of for loop 2
3. For K = 1 to N Start of for loop 3
4. C[I,J] = C[I,J] + A[J,K] * B[K,J]
 Computation of Matrix Multiplication and it will be implemented by using systolic array
5. End End of for loop1
6. End End of for loop2
7. End End of for loop3

The above algorithm can be implemented in two methods (1) Conventional method (without Pipeline and Parallel Processing) (2) Systolic Architecture (Pipeline and Parallel Processing).

$$P = \begin{bmatrix} P_{11} & P_{12} & \cdots & P_{1N} \\ P_{21} & P_{22} & \cdots & P_{2N} \\ \vdots & \vdots & \ddots & \vdots \\ P_{N1} & P_{N2} & \cdots & P_{NN} \end{bmatrix} = A \times B$$

$$= \begin{bmatrix} A_{11} & A_{12} & \cdots & A_{1N} \\ A_{21} & A_{22} & \cdots & A_{2N} \\ \vdots & \vdots & \ddots & \vdots \\ A_{N1} & A_{N2} & \cdots & A_{NN} \end{bmatrix} \times \begin{bmatrix} B_{11} & B_{12} & \cdots & B_{1N} \\ B_{21} & B_{22} & \cdots & B_{2N} \\ \vdots & \vdots & \ddots & \vdots \\ B_{N1} & B_{N2} & \cdots & B_{NN} \end{bmatrix}$$

Where

$$P_{ij} = \sum_{k=1}^{N} A_{ik} \cdot B_{kj}$$

IV. IMPLEMENTATION SCHEME

In this paper, we aim to compute the equation (1) with a two dimensional systolic array:

C(m,n) = A(m,k) x B(k,n)

Where A, B and C are the matrices with order (m,k) and (k,n) respectively. Each PE of systolic array computes the multiplication of elements and accumulates to the corresponding element and then elements will be passed to neighbor PE in the systolic array. First elements in row i of matrix A are injected first into PE as pipeline with the sequence of and the input time to the element of is one time unit later than . Similarly, elements in column j of matrix B are injected first into PE as pipeline with the sequence of and

the input time to the element of the sequence of is one time unit later than . The architecture of PE in this approach is shown in figure 10 which performs the Multiplication and Accumulation on data.

A. Systolic Array Architecture for Matrix Multiplication

A systolic architecture is an arrangement of processors i.e. PEs in an array (AB2 Architecture in [3]) where data flows synchronously across the array between neighbors, usually with different data flowing in different directions. PE at each step takes input data from one or more neighbors (e.g. Left and Top), processes it and, in the next step, outputs results in the opposite direction (Right and Bottom). The Proposed two dimensional systolic Architecture is given in the Figure 11.

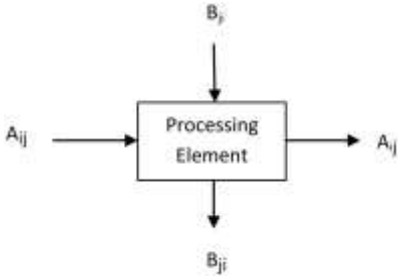

Figure 10. Schematic of processing element

The array architecture given above takes input data in parallel into first PEs in the array and processes the Multiplication and Accumulation on them and then outputs result to the next level PEs of array. Systolic arrays do not lost their speed due to their connection like any other parallelism. Where, each cell (PE) is an independent Processor (CPU) and has its own registers and Arithmetic and Logic Units (ALUs) i.e. Multiplication and Accumulation unit. The cells share the information with their neighbors, after performing the necessary operations on the data.

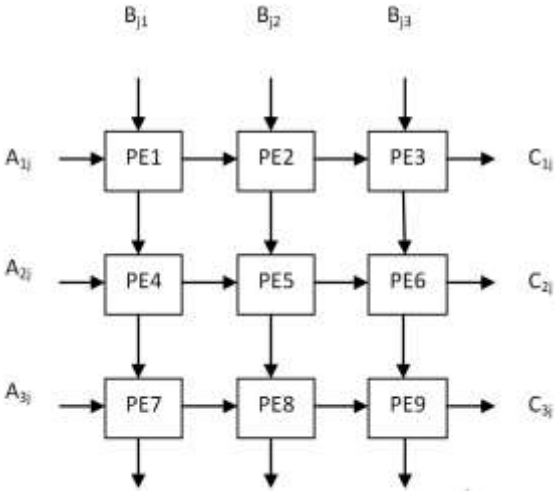

Figure 11. Systolic array for matrix multiplication

Systolic Array Architecture (SAA) for Matrix Multiplication is shown in the Figure 11. Where each cell takes inputs from left and top, multiplies them and accumulates in the local register which is inside the each PE. After N clock pulses the result would be stored in each PE. The proposed systolic array architecture needs N2 magnitude Multipliers, 2N magnitude Accumulators and 4N registers are needed to compute matrix multiplication where N is order of matrix.

V. RESULTS & DISCUSSION

The implementation of Matrix Multiplication is done in both methods i.e. Conventional and Systolic Architecture, as described above, on FPGA. The RTL code is written in Verilog HDL, verification of logic and simulation is done by ModelSim XE 6.4b. The simulation results have given that, the Systolic architecture implementation requires less number of clock cycles then Conventional method and is shown in Figure 12.

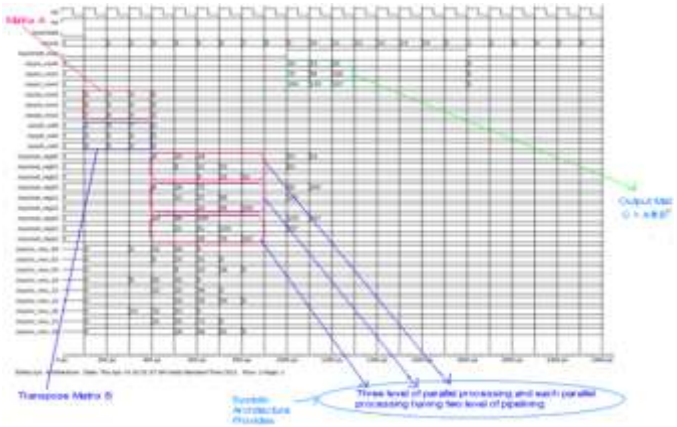

Figure 12. Simulation results

The simulation results in Figure 12, exposes the parallel processing and pipelining by the systolic array architecture and also the input and output matrices , and respectively where the matrix elements are of 4 bit each. After simulation, the design is passed for synthesis onto the platform XILINX ISE 9.2i to convert RTL logic into gate level netlist and also the schematic diagram is captured. The schematic diagrams are shown in Figure 13 and Figure 14. The Figure 13 represents the top level hierarchy of design and the Figure 14 shows internal hierarchy of top level schematic.

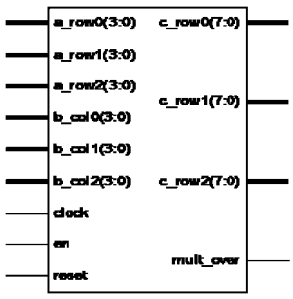

Figure 13. Top level schematic

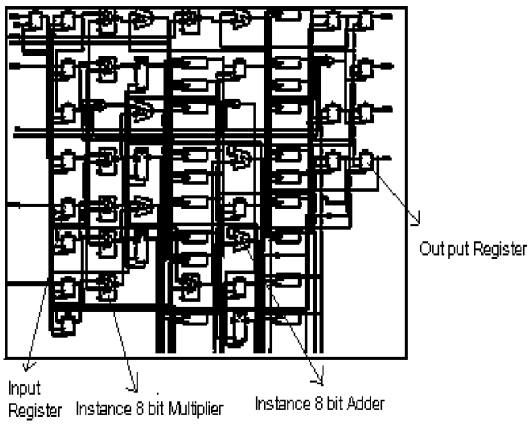

Figure 14. Internal hierarchy of top level design

The both designs Conventional method and Systolic Architecture for Matrix Multiplication are targeted to the device xc3s500e-5-ft256 and the synthesis report of the designs provides the gate level netlist with critical path delay between input and output. The critical path delay represents the core speed of the design. The brief summary of synthesis report is exposed in Table 1. From the Table 1, it is noticed that the core speed of Systolic Array Architecture for matrix multiplication is 210.2MHz which is more than two times of conventional method 101.7MHz.

Table 1. Performance evaluation of Systolic Array architecture for Matrix Multiplication

S.No	Name of Component	Number of components used	
		Conventional Method	Systolic Array Architecture
1	Critical path delay	9.831ns	4.757ns
2	4x4-bit registered multiplier	27	9
3	8-bit adder	18	6
4	4-bit up counter	0	1
5	8-bit up accumulator	0	3
6	8-bit register	72	34

VI. CONCLUSION

The Systolic Array Architecture is designed for Matrix Multiplication and it is targeted to the Field Programmable Gate Array device xc3s500e-5-ft256. The parallel processing and pipelining is introduced into the proposed systolic architecture to enhance the speed and reduce the complexity of the Matrix Multiplier. The proposed design is simulated, synthesized, implemented on FPGA device xc3s500e-5-ft256 and it has given the core speed 210.2MHz.

REFERENCES

[1] H. T. Kung "Why systolic architectures?," IEEE Computer, vol. 15, pp. 37, Jan. 1982.

[2] Sung Burn Pan, Seung Soo Chae and Rae-Hong Park, VLSI Architecture for Block Matching Algorithms using Systolic Array, IEEE Transactions on Circuits and Systems for Video Technology, Vol. 6, No. 1, February 1996.

[3] Kuan-i Lee, Algorith and VLSI architecture design for H.264/AVC Inter Frame Coding, A PhD Thesis at National Cheng Kung University, Tainan, Taiwan, in 2007.

[4] Doru Florin Chiper, M. N. S. Swamy, M. Ohmair Ahmad, and Thanos Stouraitis, A Systolic Array Architecture for the Discrete Cosine Transform, IEEE Transactions on Signal Processig, Vol. 50, no. 9, September, 2002.

[5] Ganapathi Hegde, Cyril Prasanna Raj P and P.R.Vaya, Implementation of Systolic Array Architecture for Full Search Block Matching Algorithm on FPGA, European Journal of Scientific Research, Vol.33 No.4 (2009), pp.606- 616.

[6] Chien-Min Ou, Chian-Feng Le and Wen-Jyi Hwang, An Efficient VLSI Architecture for H.264 Variable Block Size Motion Estimation, IEEE Transactions on Consumer Electronics, Vol. 51, No. 4, NOVEMBER 2005.

[7] Feifei Dong, Sihan Zhang and Cheng Chen, Improved Design and Analyse of Parallel Matrix Multiplication on Systolic Array Matrix, IEEE, 2009.

[8] Ziad Al-Qadi and and Musbah Aqel, erformance Analysis of Parallel Matrix Multiplication Algorithms Used in Image Processing, World Applied Sciences Journal 6 (1): 45-52, 2009.

[9] Mohammad Mahdi Azadfar, Implementation of A Optimized Systolic Array Architecture for FSBMA using FPGA for Real-time Applications, IJCSNS International Journal of Computer Science and Network Security, VOL.8 No.3, March 2008.