High Speed and Area Efficient Matrix Multiplier based on Canonic Signed Digit Technique

Manish Kumar[1] and Prof. Satyarth Tiwari[2]

M. Tech. Scholar, Department of Electronics and Communication, Bhabha Engineering Research Institute, Bhopal[1]

Guide, Department of Electronics and Communication, Bhabha Engineering Research Institute, Bhopal[2]

Abstract—Due to advancement of new technology in the field of VLSI and Embedded system, there is an increasing demand of high speed and low power consumption processor. Speed of processor greatly depends on its multiplier as well as adder performance. Matrix multiplication is the kernel operation used in many transform, image and discrete signal processing application. We develop new algorithms and new techniques for matrix multiplication on configurable devices. In this paper, we have proposed designs for matrix-matrix multiplication. These design reduced hardware complexity, throughput rate and different input/output data format to match different application needs. In spite of complexity involved in canonic signed digit (CSD), its implementation is increasing day by day. Due to which high speed adder architecture become important. Several architecture designs have been developed to increase the efficiency of the multiplier-less technique. In this paper, we introduce an architecture that performs high speed matrix multiplier using CSD technique. These designs are implementation Xilinx Vertex device family.

Keywords: - Look Up Table (LUT), Read Only Memory (ROM), CSD Technique, Xilinx Simulation

I. INTRODUCTION

The merit of the new technology is to be evaluated by its ability to efficiently implement the computational algorithms. In the other words, the technology is developed with the aim to efficiently serve the computation. The reverse path; evaluating the merit of the algorithms should also be taken. Therefore, it is important to develop computational structures that fit well into the execution model of the processor an are optimized and for the current technology. In such a case, optimization of the algorithms is performed globally across the critical path of its implementation. In this research article, we will present fast 16 bit multiplier with some approximation technique which is used in arithmetic application. This project is design on Xilinx-14.1 and simulated on Modalism. Application analysis will be done on Mat lab for the application of 2D Gaussian smooth filter. Image quality analysis will be done by PSNR, SSIM.

Matrix multiplication is frequently used operation in a wide variety of graphics, image processing, robotics, and signal processing applications. The increases in the density and speed of field-programmable gate arrays (FPGAs) make them attractive as flexible and high-speed alternatives to DSPs and ASICs. It is a highly procedure oriented computation , there is only one way to multiply two matrices and it involves lots of multiplications and additions. But the simple part of matrix multiplication is that the evaluation of elements of the resultant elements can be done independent of the other, this point to distributed memory approach. In this paper, we propose an architecture that is capable of handling matrices of variable sizes Our designs minimize the gate count, area, improvements in latency, computational time, throughput for performing matrix multiplication and reduces the number of multiplication and additions hardware required to get the matrices multiplied on commercially available FPGA devices. Matrix multiplication is a frequently used kernel operation in a wide variety of graphics, image processing, robotics, and signal processing applications. Several signal and image processing operations can be reduced to matrix multiplication. Most of the previous work on matrix multiplication on FPGAs focuses on latency optimization. However, since mobile devices typically operate under various computational requirements and energy constrained environments, energy is a key performance metric in addition to latency and throughput. Hence, in this paper, we develop designs that minimize the energy dissipation. Our designs offer tradeoffs between energy, area, and latency for performing matrix multiplication. The architecture design in our work to multiply two numbers is use the multiplier unit used for multiplying two numbers in a single clock cycle. This increases the speed of the computation. The proposed architecture realized on FPGA which is based on Vedic Multiplication sutra(algorithm) ―Urdhava Trigyagbhyam‖ .The implementation architecture is also use of IP CORE(Intellectual Property) for adders which allows us to be optimized for speed and space. The objective of this paper is to propose a low area, speed, energy efficient, maximum running frequency, low power, matrix multiplier.

II. CSD TECHNIQUE

CSD is so named in light of the fact that the number juggling operations that show up in sign preparing (e.g., expansion, duplication) are not “lumped” as a solid useful element but rather are conveyed in a regularly unrecognizable manner. The frequently experienced type of calculation in advanced sign

preparing is a total of items (or in vector investigation speech or internal item era). This is additionally the calculation that is executed most proficiently by CSD. The inspiration for utilizing CSD is its compelling computational productivity. The points of interest are best misused in circuit plan, however off-the-rack equipment frequently can be designed adequately to perform DA. Via cautious outline, one may decrease the aggregate door tally in a sign preparing number-crunching unit by a number sometimes littler than 50 percent and regularly as extensive as 80 percent.

CSD is fundamentally (however not so much) somewhat serial computational operation that structures an inward (spot) result of a couple of vectors in a solitary direct stride. The benefit of CSD is its productivity of motorization. Conveyed Arithmetic is regularly utilized, where figuring the internal result of two vectors involves the majority of the computational workload.

This kind of figuring profile depicts a huge segment of sign handling calculations; thus the potential utilization of Distributed Arithmetic is gigantic. The inward point is regularly registered utilizing multipliers and adders. At the point when registered successively, the duplication of two B-bit numbers requires B/2 to B increases, and is time concentrated. Then again, the augmentation can be registered in parallel utilizing B/2 to B adders, however is territory serious (K. Hwang 1979, D. L. Jones 1993: 1077–1086). Whether a K-tap channel is processed serially or in parallel, it requires in any event B/2 increases for every duplication in addition to K – 1 expansion for summing the items together. In the most ideal situation, K.(B +2)/(2 – 1) increases are required for a K-tap channel utilizing multipliers and adders. A aggressive contrasting option for utilizing a multiplier is Distributed Arithmetic. It packs the calculation of a K-tap channel from K augmentations and K - 1 expansion into a memory table and creates result in B-bit time, utilizing B - 1 expansion.

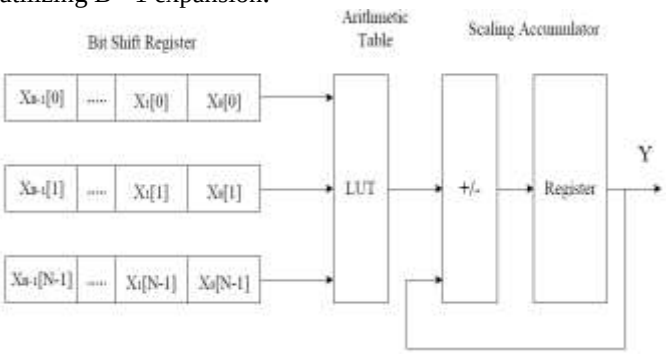

Figure 1: CSD Block Diagram

III. PROPOSED METHODOLOGY

In this design, we opted for faster operating speed by increasing the number of multipliers and registers performing the matrix multiplication operation. From equation 2 we have derived for parallel computation of 3 × 3 matrix-matrix multiplication and the structure is shown in figure 1.

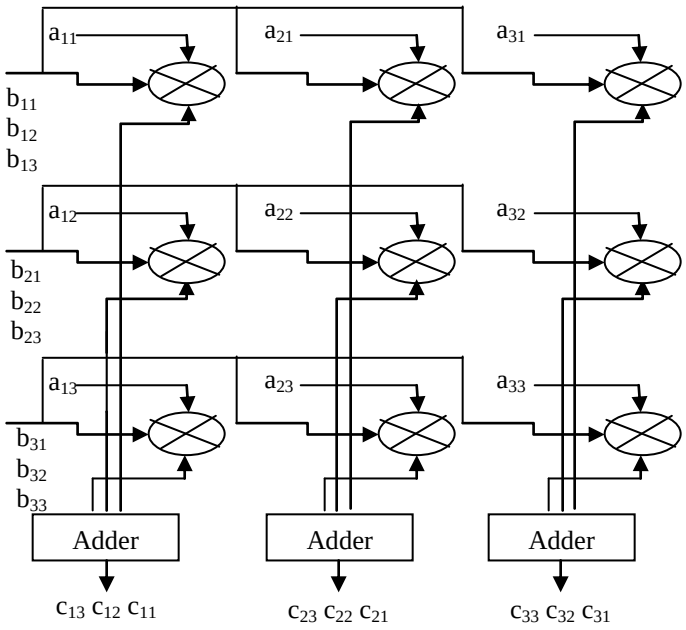

Figure 2: Proposed PPI – MO Design for n = 3

For an n×n matrix – matrix multiplication, the operation is performed using n^2 number of multipliers, n^2 number of registers and $n^2 - n$ number of adders. The registers are used to store the partial product results. Each of the n^2 number of multipliers has one input from matrix B and the other input is obtained from a particular element of matrix A.

The dataflow for matrix B is in row major order and is fed simultaneously to the particular row of multipliers such that the i^{th} row of matrix B is simultaneously input to the i^{th} row of multipliers, where 1 < i < n . The elements of matrix are input to the multipliers such that, $(j,i)^{th}$ element of matrix A is input to The $(i, j)^{th}$ multiplier, where1 < i,j < n. The resultant products from each column of multipliers are then added to give the elements of output matrix C. In one cycle, n elements of matrix C are calculated, so the entire matrix the elements of matrix C are obtained in column major order with n elements multiplication operation requires n cycles to complete.

Let us consider the example of a 3×3 matrix – matrix multiplication operation, for a better analysis of the design (as shown in figure 1). The hardware complexities involved for this design are 9 multipliers, 9 registers and 6 adders. Elements from the first row of matrix B (b_{11} b_{12} b_{13}) are input simultaneously to the first row of multipliers (M_{11} M_{12} M_{13}) in 3 cycles. Similarly, elements from other two rows of matrix B are input to the rest

two rows of multipliers. A single element from matrix A is input to each of the multipliers such that, $(j,i)^{th}$ element of matrix A is input to the multiplier M_{ij}, where 1 < i,j < 3. The resultant partial products from each column of multipliers (M_{1k} M_{2k} M_{3k} where 1 < k 3) are added up in the adder to output the elements of matrix C. In each cycle, one column of elements from matrix C is obtained (C_{1k} C_{2k} C_{3k} where1 < k < 3) and so the entire matrix multiplication operation is completed in 3 cycles.

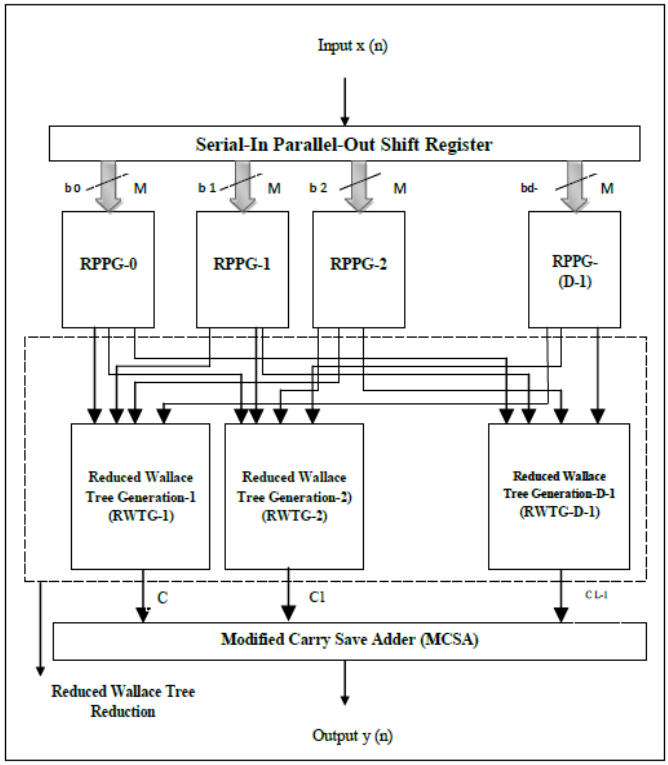

Figure 3: Proposed structure of Reconfigurable CSD Technique

In PPG unit, partial products of multiplication results are generated. Traditionally N2 AND gates are used to produce the partial product results. To provide the partial products, different architectures are developed which have been described in the literature part of this thesis. One of the important PPG generation multiplications is "Russian Peasant Multiplication (RPM)". In RPM based PPG unit, Multiplexors are used to generate the partial product results.

In PPR unit, the outputs coming from PPG units are reduced to two “2*n” bit rows by using Wallace or Reduced Wallace Tree Reduction methodologies. Traditionally Full Adders (FAs) and Half Adders (HAs) are considered to reduce the results of PPG outputs.

In the final stage of PPA unit, there is one important addition structure will be required to add two “2*n” bit data.

IV. RESULT AND SIMULATION

From the above graphical representation it can be inferred that the MM using CSD technique gives the best performance as compared with previous algorithm. Implementation Ankit Gupta and proposed matrix multiplication based on modified serial in serial out resister. The proposed design has been captured by VHDL and the functionality is verified by RTL and gate level simulation.

VHDL is an acronym for VHSIC (Very high Speed Integrated Circuit) Hardware Description Language. It is a hardware description language that can be used to describe the structure and/or behavior of hardware designs and to model digital systems.

Table I: Hardware Complexity of Matrix Multiplication

Structure	Dimension	Register	Multiplier	Adder
Previous Design [1]	3×3	-	27	18
MM using PPI-SO		72	3	2
MM using PPI-MO		36	9	6
MM using PFI-MO		36	5	6
Previous Design [1]	4×4	-	64	48
MM using PPI-SO		128	4	3
MM using PPI-MO		64	16	12
MM using PFI-MO		64	11	12
Previous Design [1]	8×8	-	512	448
MM using PPI-SO		512	8	7
MM using PPI-MO		256	64	56
MM using PFI-MO		256	54	56

Table II: Simulation result for 3×3 and 4×4 Matrix Multiplication

Structure	Dimension	Slice	LUTs	IOBs	Delay (ns)
Previous Design [1]	3×3	368	640	144	15.517
MM using PPI-SO		44	15	34	11.222
MM using PPI-MO		93	154	74	15.058
MM using PFI-MO		34	55	38	9.128

Previous Design [1]	4×4	966	1687	256	17.227
MM using PPI-SO		49	88	42	13.771
MM using PPI-MO		221	388	74	15.058
MM using PFI-MO		39	72	48	11.543

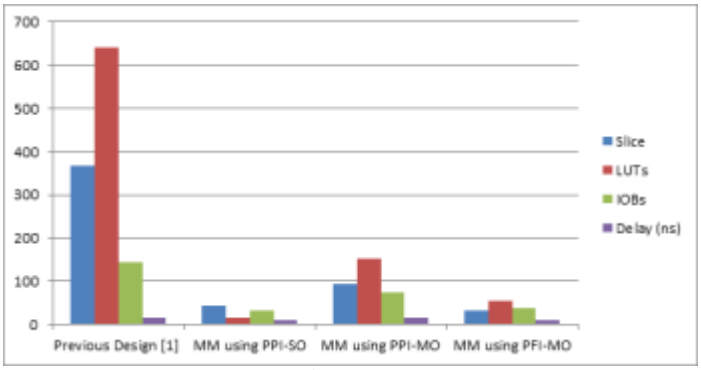

Figure 4: Bar graph of the 3×3 Matrix Multiplication

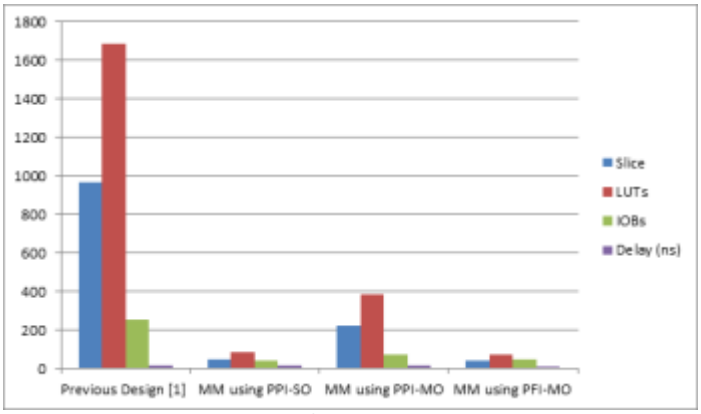

Figure 5: Bar graph of the 4×4 Matrix Multiplication

V. CONCLUSION

Most of the digital signal processing (DSP) algorithms is formulated as matrix-matrix multiplication, matrix-vector multiplication and vector-vector (Inner-product and outer-product) form. Few such algorithms are digital filtering, sinusoidal transforms, wavelet transform etc. The size of matrix multiplication or inner-product computation is usually large for various practical applications. On the other hand, most of these algorithms are currently implemented in hardware to meet the temporal requirement of real-time application. When large size matrix multiplication or inner product computation is implemented in hardware, the design is resource intensive. It consumes large amount of chip area and power. With such a vast application domain, new designs are required to cater to the constraints of chip area and power and high speed.

REFERENCES

[1] Ankit Gupta and Ankit Gupta, "Hardware Design of Approximate Matrix Multiplier based on FPGA in Verilog", International Conference on Intelligent Computing and Control Systems (ICICCS 2020).

[2] M Shanmugakumar, Vegesna S. M. Srinivasavarma and Sk Noor Mahammad, "Energy Efficient Hardware Architecture for Matrix Multiplication", IEEE 4th Conference on Information & Communication Technology (CICT), IEEE 2018.

[3] Chiranjit R Patel, Vivek Urankar, Vivek B A and Sampath Kumar R, "2x2 Matrix Multiplication with 4-Bit elements in 45nm CMOS Technology", 4th International Conference on Electronics, Communication and Aerospace Technology (ICECA), IEEE 2020.

[4] Y. Huang J. Shen Y. Qiao M. Wen and C. Zhang "Malmm: A multi-array architecture for large-scale matrix multiplication on fpga" IEICE Electronics Express vol. 15 no. 10 pp. 20 180 286-20 180 286 2018.

[5] I. Sayahi M. Machhout and R. Tourki "Fpga implementation of matrix-vector multiplication using xilinx system generator" 2018 International Conference on Advanced Systems and Electric Technologies (IC_ASET) pp. 290-295 March 2018.

[6] W. Liu and B. Vinter "Csr5: An efficient storage format for cross-platform sparse matrix-vector multiplication" Proceedings of the 29th ACM on International Conference on Supercomputing 2015.

[7] Usha Maddipati, Shaik Ahemedali, Maddipati Sri Sai Ramya, M D Praneeth Reddy and K N J Priya, "Comparative analysis of 16-tap FIR filter design using different adders", ICCCNT, IEEE 2020.

[8] Ankit Upadhyay and Prof. Uday Panwar, "High Performance VLSI Architecture for Transpose Form FIR Filter using Integrated Module", International Conference on Computer Communication and Informatics (ICCCI), IEEE 2018.

[9] Basant Kumar Mohanty, and Pramod Kumar Meher, High-Performance FIR Filter Architecture for Fixed and Reconfigurable Applications", IEEE Transactions on Very Large Scale Integration (VLSI) Systems, Vol. 78, No.06, April 2016.

[10] Indranil Hatai, Indrajit Chakrabarti, and Swapna Banerjee, "An Efficient VLSI Architecture of a Reconfigurable Pulse-Shaping FIR Interpolation Filter for Multi-standard DUC", IEEE Transactions on Very Large Scale Integration (VLSI) Systems, Vol. 23, No. 6, June 2015.

[11] Sang Yoon Park and Pramod Kumar Meher, "Efficient FPGA and ASIC Realizations of DA-Based Reconfigurable FIR Digital Filter", IEEE Transactions on Circuits And Systems-Ii: Express Briefs, 2014.

[12] B. K. Mohanty, P. K. Meher, S. Al-Maadeed, and A. Amira, "Memory footprint reduction for power-efficient realization of 2-D finite impulse response filters," *IEEE Trans. Circuits Syst. I, Reg. Papers*, vol. 61, no. 1, pp. 120–133, Jan. 2014.

[13] B. K. Mohanty and P. K. Meher, "A high-performance energy-efficient architecture for FIR adaptive filter based on new distributed arithmetic formulation of block LMS algorithm," *IEEE Trans. Signal Process.*, vol. 61, no. 4, pp. 921–932, Feb. 2013.

[14] G. Gokhale and P. D. Bahirgonde, "Design of Vedic Multiplier using Area-Efficient Carry Select Adder", 4th IEEE

International Conference on Advances in Computing, Communications and Informatics (ICACCI-2015), Kochi, August 10-13, 2015, India.

[15] G. Gokhale and Mr. S. R. Gokhale, "Design of Area and Delay Efficient Vedic Multiplier Using Carry Select Adder", 4th IEEE International Conference on Advances in Computing, Communications and Informatics (ICACCI-2015), Kochi, August 10-13, 2015, India.

[16] Pavan Kumar, Saiprasad Goud A, and A Radhika had published their research with the title "FPGA Implementation of high speed 8-bit Vedic multiplier using barrel shifter", 978-1-4673-6150-7/13 IEEE.

[17] B. Madhu Latha1, B. Nageswar Rao, published their research with title "Design and Implementation of High Speed 8-Bit Vedic Multiplier on FPGA" International Journal of Advanced Research in Electrical ,Electronics and Instrumentation Engineering, Vol. 3, Issue 8, August 2014.

[18] A Murali, G Vijaya Padma, T Saritha, published their research with title "An Optimized Implementation of Vedic Multiplier Using Barrel Shifter in FPGA Technology", Journal of Innovative Engineering 2014, 2(2).