International Journal of Electronics Communication and Computer Engineering
Volume 7, Issue 3, ISSN (Online): 2249–071X

Faster Convolution using FFT by Polynomial Matrix Multiplication

Priyanka Bhalawi
priyankabhalawi41@gmail.com

Ashish Raghuwanshi
Asst. Prof. IES College of Technology Bhopal, M.P.

***Abstract* – Matrix multiplication is a primary computation for several scientific computing, graphics processing units and engineering applications. Matrix multiplication is also use in convolution operation of two discrete signals in DFT (Discrete Fourier Transform) and FFT (Fast Fourier Transform) application. For the timing simulation of, Xilinx tool is used. Xilinx is a digital front end design tool which supports the simulation and synthesis of VHDL and Verilog HDL codes. In this work, we considered two different examples (circular convolution and linear convolution) of matrix multiplier architecture where speed is the main constraint.**

***Keywords* – DFT, FFT, VHDL, MAC, PE.**

I. INTRODUCTION

The speed and accuracy of matrix multiplication is crucial for the performance of such applications. Its computational complexity is large than communication complexity. Because it takes large amount of multiplication and addition operations, so its performance can be improved significantly through parallel architecture design. The internal architecture of processing elements includes addressable register, a multiplier unit by byte multiplication, an adder for multiplier output, FIFO block to add and shift the multiplier output, and a controlling unit for controlling the operations within the specific PE. The counters are use to check the required matrix size. PEs also design using a pipeline registers, simple logic gates, and other miscellaneous blocks. The addressable data which is to be processing components of the transformed sequences for the multiplier and multiplicand matrices are send to the processing element.

II. MATRIX METHOD FOR CONVOLUTION

Due to the importance of Discrete Fourier Transform (DFT) in signal processing application, it is critical to have an efficient method to compute this algorithm. DFT operates on a N -point sequence of numbers, referred to as x(n) . The value x(n) is presented in time domain data and usually can be taught as a uniformly sampled version of a finite period of a continuous function f (x) . The DFT of x(n) sequence is transformed to X(k) in frequency domain representation employing by using Discrete Fourier Transform. The functions x(n) and X(k) is generally represented in complex signal form, given by

$$\sum_{n=0}^{N-1} x(n)\, e - \frac{j2\pi kn}{N} = X(K)$$

where x(n) is the input time domain representation and N is the number of input to the DFT. The value n represents the discrete time-domain index and k is the normalized frequency domain index. The description of efficient computation is discussed on DFT methods since the IDFT and DFT consumes the same type of computational algorithm. From the computation of each value of k , it is observed that direct computation of X(k) involves N complex multiplications (4N real multiplications) and N −1 complex additions (4N − 2 real additions). Eventually, to compute all N values of the DFT requires N^2 complex multiplications and N^2 − N complex additions [2,5,6].

The multiplication of two discrete time signal in discrete fourier transform is equivalent to the circular convolution of there sequences in time domain. For x(n) and h(n) signal convolution is express as:

$$\sum_{n=0}^{N-1} x(n)h((m-n))N = y(m)$$

Here the term $h(m-n)_N$ indicates the circular convolution.

The convolution in time domain of two signal x and h is perform by multiplying its discrete fourier transform and the converting it in time domain by inverse discrete fourier transform. The equation of DFT is the summation of discrete signal multiplied by twiddle factor given as:

$X(K)= \sum_{n=0}^{N-1} x(n)\, e^{-j2\pi kn/N}$

Where, $e^{-j2\pi kn/N}$ is called as twiddle factor.

For long convolution the FFT is faster method as compare to DFT.

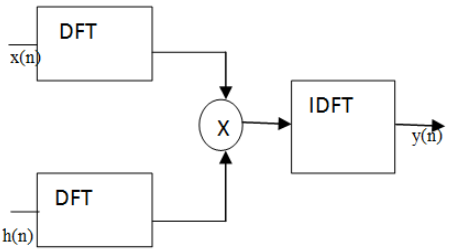

Fig. 1. Convolution by DFT-IDFT Method

The same convolution process is done by matrix method as:

$$y(m) = \begin{bmatrix} 1 & 1 & 0 & 2 \\ 2 & 1 & 1 & 0 \\ 0 & 2 & 1 & 1 \\ 1 & 0 & 2 & 1 \end{bmatrix} \begin{bmatrix} 2 \\ 2 \\ 1 \\ 1 \end{bmatrix} = \begin{bmatrix} 6 \\ 7 \\ 6 \\ 5 \end{bmatrix}$$

The NxM matrix multiplication equation is written as:

y(0) = h(0)x(0) + h(N-1)x(1) + h(N-2)x(2) + ---------- h(2)x(N-2) + h(1)x(N-1)

y(1) = h(1)x(0) + h(0)x(1) + h(N-1)x(2) + ---------- h(2)x(N-2) + h(1)x(N-1)

y(2) = h(2)x(0) + h(1)x(1) + h(0)x(2) + ----------h(4)x(N-2) + h(3)x(N-1)
y(N-2) = h(N-2)x(0) + h(N-3)x(1) + h(N-4)x(2) + ----------h(0)x(N-2) + h(N-1)x(N-1)
y(N-1) = h(N-1)x(0) + h(N-2)x(1) + h(N-3)x(2) + ----------h(1)x(N-2) + h(0)x(N-1)

Thus circular convolution is obtaining quickly using matrix multiplication approach.

III. Related Work

Soydan Redif and Server Kasap [1] design matrix multiplication is a computationally- intensive fundamental matrix operation in several algorithm used in scientific computation for presents 4X4 matrix architecture. The architecture is based on word-width decomposition for flexible but high-speed operation..This paper present an FPGA-based hardware realization of matrix multiplication based on parallel architecture, a delay and resource efficient implementation of the matrix multiplication algorithm for fix point and floating point design. For image processing tasks where operational tight packing lead to increased packing in comparison to previously – operational packing is increased ,the processing throughput is increased by up to 25%. The utilization of resources for the proposed architecture implemented on a Xilinx Virtex-5 XC5VLX110T FPGA chip, for the cases of N = 64 and N = 128. The simulation results shows that resource utilization is less than 55%, except for the DSP48E slices that are fully utilized to implement the multipliers in each PE of the systolic array within the design [1].

Bahram Hamraz, Nicholas HM Caldwell, and P. John Clarkson [3] work on a matrix computation algorithm which can be use for numerical product model to analyze change propagation and support change prediction. The implemented algorithm for vector matrix multiplication base on is based on a path-by-path analysis of all possible paths within the network. This work employs matrix multiplications on changes of a given design structure matrix accounting for the exclusion of self-dependences and cyclic propagation paths and delivers the same results as the exhaustive search-based Trail Counting algorithm.

Nan Zhang [4] in their work a novel parallel algorithm for computing the scan operations on x86 multi core processors is design. The computational matrix form with m (row) X n (column) sparse matrix M with w non zero elements is stored in memory in the compressed sparse row (CSR) format, where a w-dimensional vector A stores all the w nonzero elements, a w dimensional vector E stores the column indexes of each of the nonzero elements, and a (m+1) dimensional array (T) keeps track of where each row starts in A. ‘A’ is a diagonal matrix elements. The last element of T is the number of non zeros in M. proposed a method for decomposing a perfect binary multiplication into smaller size using Nikhilam sutra and hence reducing the computation time and power consumption. The algorithm was broadly divided in three parts namely the initialization, pre-processing and processing. The algorithm evidently reduces a given 4 bit multiplication to a 2-bit multiplication by making use of basic shifting and addition operations, as a result of which the carry propagation in any standard 4 x 4 - bit multiplier is reduced to a great extent.

Syed M. Qasim, Shuja A. Abbasi and Bandar A. Almashary [7] consider the problem of implementing fixed point matrix multiplication algorithm as deeply-nested loops and then mapping the algorithm to parallel architecture on FPGA. They design methodology is based on a parallel array design that maps a nested loop algorithm onto the parallel architecture. The contributions of this paper are:
1) Proposed design provides better speed as compared to previously reported FPGA implementation of matrix multiplication and also it requires less area.
2) High throughput architecture for matrix multiplier is developed using advanced design techniques. Their paper presents an FPGA-based hardware realization of matrix multiplication based on a parallel architecture. The proposed parallel structural design employs advanced design techniques and exploits architectural features of FPGA. In their work they present a delay and resource efficient implementation of the matrix multiplication algorithm for fixed-point and floating-point designs. We take advantage of recent FPGA technology to significantly reduce the resource utilization and total computation time over previous methods. These advances have been made possible by our novel algorithm which removes the intercommunication between parallel processing elements (PEs), and allows each PE to operate in isolation.

IV. Methodology

To optimize the performance of the matrix multiplier various design techniques and architectures were considered. For hardware realization, distributed memory approach is used. The proposed design array is similar to previous method except with slight modification. The proposed architecture consists of identical processing elements (PEs) and the number of PE depends on the size of the matrices. Each PE performs the necessary multiply accumulate (MAC) operation. Each PE operates independently with connection only to the input and output ports. This greatly helps in reducing the interconnection between the PEs and as a result the hardware resource utilization is minimized. Both distributed memory processing techniques are used to improve the performance of the matrix multiplier. In high speed applications, massive throughput is a requirement and this is achieved by using pipelined architecture that computes the product of two matrices at every clock cycle. Pipelining is achieved by inserting registers at appropriate places. Propose processing is exploited in the design, where the matrix multiplication operation is divided into several smaller operations that are executed in parallel and finally the output is obtained by accumulating all the partial results generated from these parallel processes.

The proposed design is implemented using the VHDL hardware description language. The implementation supports a range of parameters to facilitate the experimental evaluation of design choices. A functional, parameterize implementation is then produced, based on the design specification. The analysis of design decisions suggests that some design options provide no advantages over their alternatives. Such options are not supported in the implementation. The design is implemented using VHDL, a standard hardware description language. The implementation serves two main purposes. First, it verifies the correctness of the design. Second, it allows for verifying the analysis experimentally. The VHDL implementation is tested and verified both through simulation and on hardware. Modelsim from Mentor Graphics is used for logic simulation.

A simulation test bench is used to verify the correctness of the implementation, by checking the produced results. By analyzing the described method for parallel matrix multiplication, we suggest two strategies that can improve the performance of the algorithm. Since the size of the numbers that are added in the PE unit is large, we could use a high-performance adder. This modification will decrease the value of a_a, respectively decreasing the system latency . Another improvement could be using pipelining in multiplication operation of the PE unit. By using a pipelined multiplier, we will decrease the a_m value, which will also improve the system latency[2,5,6]..

V. TOOLS USED

For the timing simulation of, Xilinx tool is used. Xilinx is a digital front end design tool which supports the simulation and synthesis of VHDL and Verilog HDL codes. VHDL is an acronym for VHSlC Hardware Description Language (VHSIC is an acronym for Very High Speed Integrated Circuits). It is a hardware description language that can be used to model a digital system at many levels of abstraction ranging from the algorithmic level to the gate level. The complexity of the digital system being modelled could vary from that of a simple gate to a complete digital electronic system, or anything in between. The digital system can also be described hierarchically. Timing can also be explicitly modelled in the same description. The language not only defines the syntax but also defines very clear simulation semantics for each language construct. Therefore, models written in this language can be verified using a VHDL simulator. It is a strongly typed language and is often verbose to write. VHDL provides an extensive range of modelling capabilities, it is often difficult to understand.

VI. CONCLUSION

In this work, we considered two different examples (circular convolution and linear convolution) of matrix multiplier architecture where speed is the main constraint. The architecture of matrix multiplication operate concurrently, and then the additions performed simultaneously. This parallelism can improved the computational time. Our designs minimize the number gate count require for multiplier, adder, FIFO, counter and control logic modules. It improvements in latency, computational-time, throughput for performing matrix multiplication

REFERENCES

[1] Soydan Redif, and Server Kasap "Novel Reconfigurable Hardware Architecture for Polynomial Matrix Multiplications" IEEE Transactions On Very Large Scale Integration (Vlsi) Systems" March 10, 2014.

[2] Tai-Chi Lee, Mark White, and Michael Gubody "Matrix Multiplication on FPGA-Based Platform" Proceedings of the World Congress on Engineering and Computer Science 2013 Vol I.

[3] Bahram Hamraz, Nicholas HM Caldwell, and P. John Clarkson "A Matrix-Calculation-Based Algorithm for Numerical Change Propagation Analysis" IEEE Transactions On Engineering Management, Vol. 60, No. 1, February 2013 pp. no. 186.

[4] Nan Zhang "A Novel Parallel Scan for Multicore Processors and Its Application in Sparse Matrix-Vector Multiplication" IEEE Transactions On Parallel And Distributed Systems, Vol. 23, No. 3, March 2012 pp. no. 397.

[5] Mr. Rounak R. Gupta, 2Prof. Atul S. Joshi "Matrix Manipulation Using High Computing Field Programmable Gate Arrays" International Journal of Enterprise Computing and Business System Vol 2 issue 2 Jully 2012.

[6] Syed M. Qasim, Ahmed A. Telba and Abdulhameed Y. AlMazroo "Syed M. Qasim, Ahmed A. Telba and Abdulhameed Y. AlMazroo" IJCSNS International Journal of Computer Science and Network Security, VOL.10 No.2, February 2010 pp no. 168.

[7] Syed M. Qasim, Ahmed A. Telba and Abdulhameed Y. AlMazroo "FPGA Design and Implementation of Matrix Multiplier Architectures for Image and Signal Processing Applications" IJCSNS International Journal of Computer Science and Network Security, VOL.10 No.2, February 2010 pp. no. 168.