International Journal of Ethics in Engineering & Management Education
Website: www.ijeee.in (ISSN: 2348-4748, Volume 2, Issue 7, July 2015)

The Efficient Implementation of Numerical Integration for FPGA Platforms

Hemavathi H
Department of Electronics and Communication Engineering
East West Institute of Technology
Bangalore, India
hemavathihu@gmail.com

Ravichandra V,
Asst.Professor ,Department of Electronics and Communication Engineering
East West Institute of Technology
Bangalore, India
Ravi33811@gmail.com

***Abstract*—The paper deals with the study of numerical techniques which are used for computing the approximate solutions of infinite integrals. These techniques are software oriented. It is necessary to develop a hardware oriented solution which asses the performance in terms of speed and area. Implementation of integrals have very lengthy critical path delays which restricts the throughput rates, this can be reduced by architectural modifications in which the structure can be operated at higher throughput rates and less area consumptions. Here implementation of pipelining structure, fine grain pipelined structure and parallel structure is carried out using trapezoidal rule. The same is provides a high performance ability. In the present study FPGA's are used as a implementation platform.**

Index Terms—FIR structure, Trapezoidal rule, Pipelining, Parallel structure, Fine-grain Pipelining etc.

I.INTRODUCTION

In many applications such as mathematics and engineering there sometimes rise the circumstances where it is tricky to find an anti-derivative of an integrand. An often used approach for obtaining approximate solutions for such integrals is the Numerical Integration (NI) technique. It is the approximate computation of an integral using numerical technique.

Its basic form, the definite integral is approximated as:

$$\int_a^b y(x)dx \approx nh\left[y(0) + \frac{n}{2}\Delta y(0) + \frac{n(2n-3)}{12}\Delta^2 y(0) + \cdots\right] \quad (1)$$

Substituting n = 1 in equation (1) results in the Trapezoidal Rule for NI as:

$$\int_a^b \mathbf{y(x)dx} \approx \frac{\mathbf{h}}{\mathbf{2}}[\mathbf{y(0)} + \mathbf{2}\{\mathbf{y(1)} + \cdots + \mathbf{y(n-1)} + \mathbf{y(n)}\}]$$

$$= \frac{\mathbf{h}}{\mathbf{2}}\mathbf{y(n)} + \mathbf{hy(n-1)} + \mathbf{hy(n-2)} + \cdots + \mathbf{hy(1)} + \mathbf{hy(0)} \quad (\mathbf{2})$$

Where, h=b-a

The generalized Finite Impulse Response (FIR) equation is Given by:

$$Z(n) = \sum_{k=0}^{m-1} a_k y(n-k) = a_0 y(n) + a_1 y(n-1) + a_2 y(n-2) + \quad (3)$$

Where $a_k(0 \leq k \leq M-1)$ are the coefficients needed to produce the necessary filtering response Equations (2) and (3) advice that there is a one-to-one communication between the Trapezoidal rule and the FIR equation. Then a hardware structure for NI can be obtained by mapping the Trapezoidal rule on to a worldwide FIR structure.These have been some hardware applications of NI by means of radix-2 encoding techniques. Although, these solutions have stretched computational delays and have huge non-recurring engineering (NRE) costs. With FIR jobs the complex arithmetic operations are applied in terms of multiply and add operations only. The multiply operation is the important computational bottleneck in the FIR structure; it needs a high computation time .The different variety of approaches have been followed to speed up the FIR filter structures. The Distributed arithmetic (DA) has been used as an alternative over the expectable multiply and Accumulate (MAC) operations. This, however, leads to slow structures as in effect, bandwidth is being traded off to save resources. Constant coefficient multiplication is yet another method that has been used to design efficient structures. These are based on using look-up tables and addition operations. However, the use of look-up tables has limited their usage in FIR process.

These expectable techniques have mainly focused on getting a speed up by adjusting the discrete individual components of the FIR structure. This can be revealed that by introducing architectural changes at the system level a subsequent speed-up in the performance can be achieved. This involves operating the hidden concurrencies within the algorithm to be realized. Pipelining and parallel processing are the methods that have been used at system level to operate these concurrencies. The key issue here is finding out the amount of parallelism that can be exploited within an algorithm.

The Computational complexity and data dependencies are communication bounds, filter lengths, finite arithmetic effects and so on. These are the factors that are needed to be taken as consideration before using the modifications.

Other major issue is the availability of a suitable platform that can support the hardware intensive processing with current FPGAs, the technology which provide bit-parallel multipliers that can be used to develop different sorts of architecture. FPGA's are therefore, being increasingly used for computationally intensive applications.

II. OBJECTIVE OF THE PROJECT

The main objective of this project is to perform the approximate solution for the NI by using Trapezoidal rule. The objective is supported by testing the proposed method that is different types of structure which are obtained by the modification of basic architecture. The aim of the project is to study the different structure and their performance which results in the better throughput.

The proposed method is using the Architectural modification by comparing the performance such as throughput, Number of Latency in the individual architecture, and to improve this evaluation parameter using the proposed method.

III. IMPLEMENTATION OF THE PROJECT

1. The basic architecture

The basic architecture consists of delay unit, Multiplier, and the summer block. The basic architecture is designed based on the one-to-one correspondence. This one-to-one correspondence exist in between equation 2 and 3 of a general FIR structure for the trapezoidal rule. The trapezoidal rule is given below in the figure 3.1.

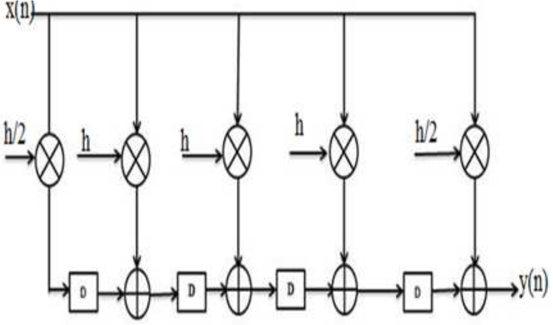

Fig 3.1: Architecture of FIR filter for Trapezoidal rule

The above architecture describes the FIR structure for Trapezoidal rule. In this structure the critical path is limited to one multiplication and N addition operations. The critical path computation time Tc is given by

$$T_C = T_M + NT_A \qquad (4)$$

Where, Tm is the computation time for one multiplication operation

TA is the computation time for addition operation

N is the numbers of taps in the filter.

Finally the sampling frequency or the throughput of this basic architecture is approximately given by

The sampling frequency and the critical path frequency is a function of filter length N, with increase in filter length the throughput or the sampling frequency decreases. This critical pathway of the basic structure can be minimized by transposing the structure. Where the transposed structure is obtained by the interchanging the input and output nodes and by changing the direction of data flow.

2. Transposed and pipelining structure

The next architecture is transposed structure. By interchanging the input and output nodes and by changing the direction of data flow on each link, the transposed structure is obtained. The transposed structure is shown in the fig3.2

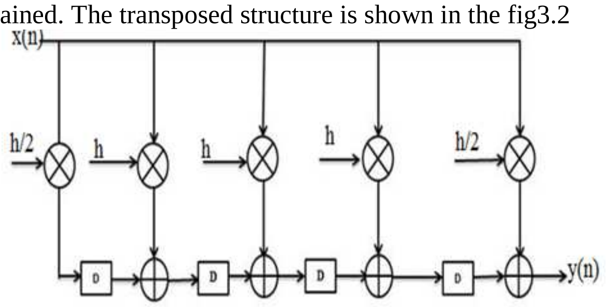

Fig 3.2: Architecture of Transposed FIR filter for Trapezoidal rule

The sampling frequency and the critical path of the transposed structure are restricted computation time of the one addition and the one Multiplication and also they are independent of the filter length.

The critical path computation time T_{CT} of the Transposed structure is given by

$$T_{CT} = T_M + T_A \qquad (6)$$

Where $T_M$is the computation time for one multiplication.

T_A is the computation time for one addition.

The throughput or the sampling frequency of the transposed structure is given by

Now the data is being appear to all the multipliers at a time or simultaneously. This structure is also known as the" Data Broadcast" structure.

The critical path which is appearing in this structure can be further reduced by pipelining the structure. This can be achieved by placing the latches with the feed-forward cut sets of the transposed structure.

3. Pipelining Structures

The pipelining can be done by placing the latches with the feed-forward cut sets of the transposed structures. The pipelining structure for trapezoidal rule is shown in figure3.3

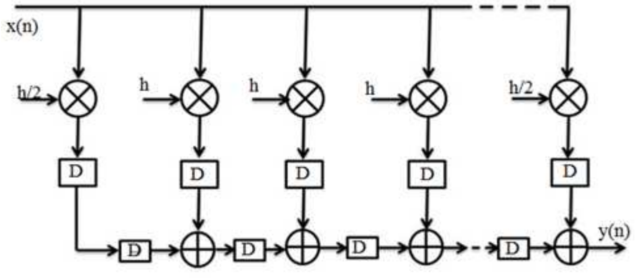

Fig 3.3: Structured pipeline for Trapezoidal rule

The pipelining structure minimizes the critical path to T_M by making the slight increase in latency. The critical Fora pipelined structure is given by

Where,

T_M is the computation time for one multiplication.
Then the throughput or sampling frequency of a pipelined structure is given by

—

The computation time of the multiplier unit limits the critical path of the pipelined structure. For large input words lengths; the computation time taken by the multiplier is significantly large. This factor limits the throughput of the overall structure. For large input words lengths, it is advantageous to break the multiplication unit into smaller multiplication units. Then a pipeline register is introduced in between the two smaller units which will increase the sampling rate, at the expense of an increased latency. This kind of structure is known as fine- grain pipelined structure.

4. Fine-Grain Pipelined Structure

By introducing the pipeline register is introduced between the two smaller units to increase the sampling rates, by increasing the latency. This type of structure is known as Fine-grain structure. The fine-grain structure is shown in figure3.4.

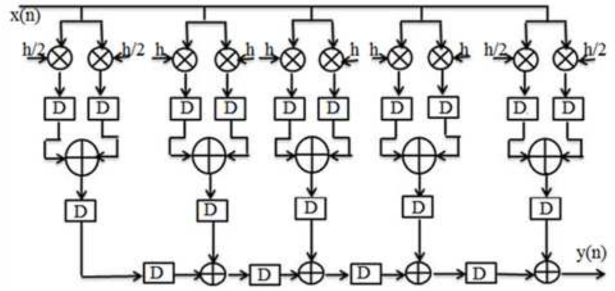

Fig 3.4: Structure of fine grain pipeline for trapezoidal rule

5. Parallel Structures

Pipelining and parallel processing are duals each other, if the structure is pipelined it can also be processed in parallel. The representation of a trapezoidal rule for a Single- input Single-output (SISO) system is given by the equation:

Z(n) =–

The SISO system wants to be converted to a multiple-input multiple- output (MIMO) to get a parallel structure.

For example, the MIMO equations are representing the 4-parallel structure of above equation are

z(4k) =–y(4k)+hy(4k-1)+hy(4k+2)+… (11)

–

–

z(4h+3)=–

The each delay in the 4-parallel structure is a block delay of four clock cycles. The critical path of the structure does not change by parallel processing, since the four samples are processed in a single clock cycle. This increases the overall throughput rate is four times the origin.

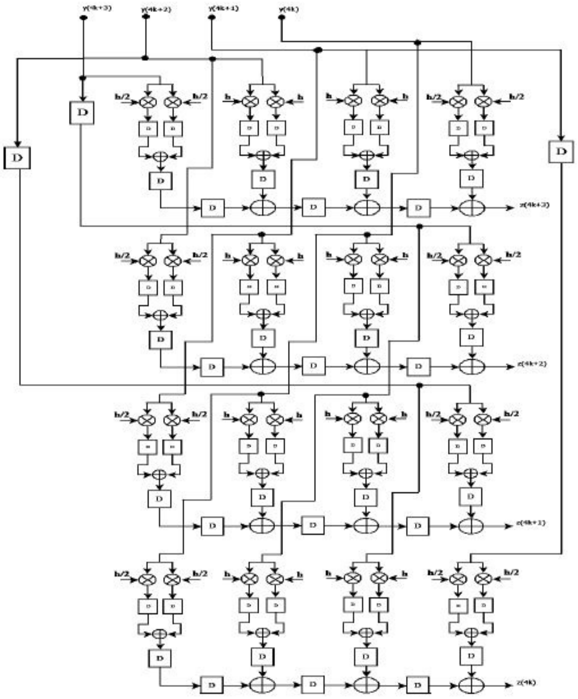

Fig 3.5: Paralleled structure for Trapezoidal rule

The major disadvantage of the parallel structure is that a lot of on-chip resources are required due to the duplication of the hardware. However, the process is targeted for FPGA devices; the resources used for hardware are quite high. This high amount of fundamental logic can be used efficiently, so that area is not a major concern.

For a 4-parallel the critical path is given by

The sampling frequency or throughput is given by
f_{par} sample= 4xfpip sample

IV. RESULTS AND DISCUSSION

A. METHODOLOGY

The Trapezoidal rule for FIR structures are implemented by using FPGA platform. The implementation is done for an 8-tap filter which is having input word length of 8 bits. VHDL is used as an Initial design entry. XILINX ISE is used for Design synthesis, mapping and translation are carried out and through-put and area is analyzed.

B. RESULTS

In the present work, implementation is done on different architecture that is Basic architecture, Transposed architecture, pipeline and Fine-grain architecture; these are implemented by doing an architectural modification.

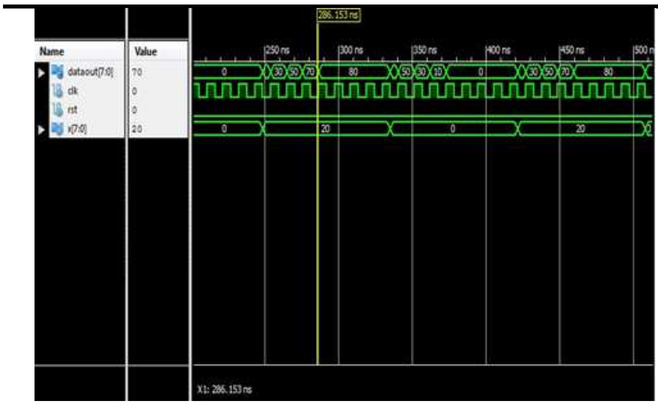

Fig 4.1: Basic architecture output

The above figure 4.1 shows that Basic architecture implementation. The basic architecture requires more computation time because of the multiplication process requires more time to compute when compare to Addition. As the number of taps increases it results in the more delay, as the filter length increases the through put will decreases. For every 4 clock cycle the results will be obtained.

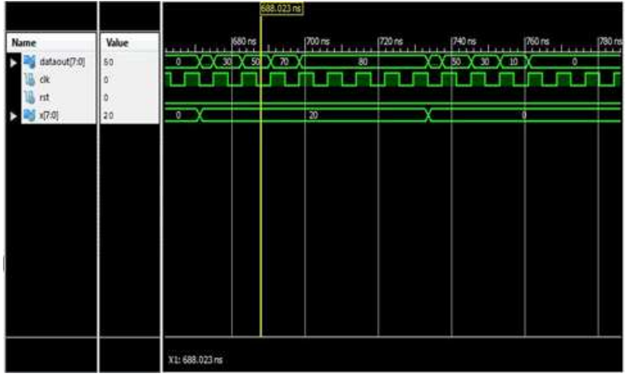

Fig 4.2: Transposed architecture output

The above figure 4.2 shows the output of a Transposed structure. This is obtained by interchanging the input and output nodes. In this the filter length N is removed. So that compare to basic architecture the through put is little high for the same latency of basic architecture. In this also for every four clock cycles results are obtained, thus the computational time is little faster.

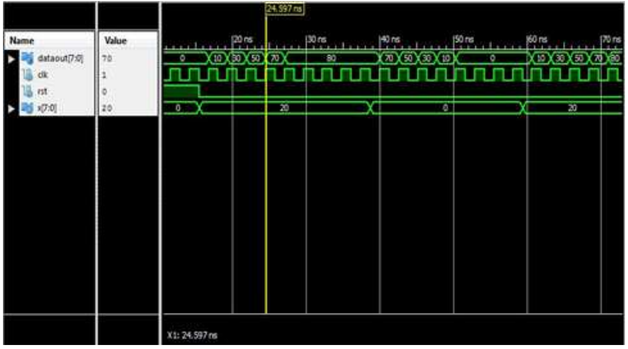

Fig 4.3: pipelined architecture output

The figure 4.3 shows the output of a pipelined structure.This will results in the reduction in the critical path. It takes the more latency to produce the output but it's much faster compared to Transposed structure. Here the latencies are used to activate the latches. It uses 5-latency but the throughput will be more compare to Transposed.

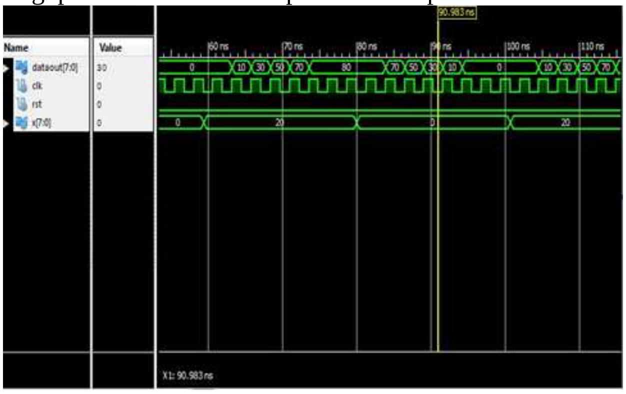

Fig 4.4: Fine-grain pipelined architecture output

The figure 4.4 shows the output of Fine-grain Pipelined structure. This structure results in the reduction of critical path. It takes the 6 latency to activate the latches. Because of the architectural modification it will do the computation fast and throughput will be more because it performs the operation by parallel processing.

Table 1: Throughput and Latency Comparison for Different Architecture

Structure	Throughput (MHz)	Latency (No.of clock cycles)	Clock period (ns)
Basic FIR structure	122.69	4	10.89
Transposed FIR structure	169.49	4	7.804
Pipelined FIR structure	484.96	5	5.277
Fine-grain pipelined FIR structure	598.800	6	2.581

The table 1 shows that area comparison between the different architecture. The table 2 shows the utilization of area for 8-tap filter which is having 8 word input length.

Table 2: The area comparisons for different Architecture

Structure	No. of LUTs	No. of occupied slices	No. of IOBs
Basic FIR structure	31	18	18
Transposed FIR structure	32	17	18
Pipelined FIR structure	32	19	18
Fine-grain pipelined FIR structure	32	24	18

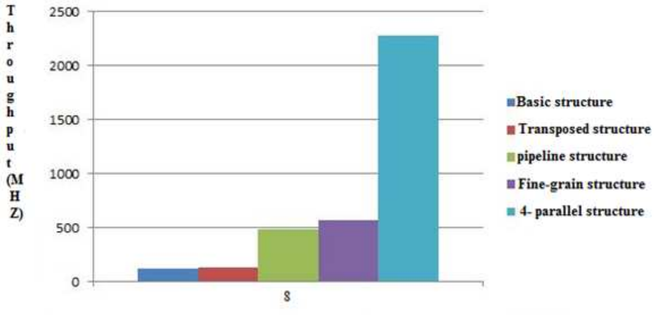

Fig4.5: Through-put variations for 8 order filter

The fig4.5 shows the through-put variation in the different architecture. By doing the architectural modification the through-put will increase, 4-Parallel Structure will give better through-put

V.CONCLUSIONS

In the present work the implementation of trapezoidal rule for NI is done by mapping the algorithm of integration on to the FIR structures which is having 8-tap filter with an input word length of 8.The solution for the hardware is entirely based on the modifications of architecture that will be carried out at a system level.

In the present work an experimental results are obtained which clearly shows that a notable improvement in the performance which is achieved by introducing the architectural modifications like pipelining and parallel processing.

REFERENCES

[1]. BurhanKhurshid,RoohieNaz Mir, “A Hardware Intensive Approach for Efficient Implementationof Numerical Integration for FPGA Platforms”,27th International Conference on VLSI Design and 13th International Conference on Embedded Systems PP 312-317, 2014.

[2]. Xiao-li Hu, Feng-ying Wang, Min Zhang, "Hardware Process of FIR Filter," proceedings of International Conference on Multimedia Technology 26-28 July 2011, p. no. 341-343 ISBN 978- 1-61284-771-9 (print.)

[3]. White, S. A. "Application of Distributed Arithmetic to Digital Signal Processing," IEEE ASSP magazine, Vol. 6 (3), pp. 4-19, July 1989.

[4]. Abedelgwad, “High speed and area efficient multiply Accumulate (MAC) Unit for Digital signal Processing applications,” IEEE International Symposium on Circuits and Systems, ISCAS 2007.

[5]. Ayaman.A.Fayed, “A merged Multiplier Accumulator for High Speed signal processing Applications,” IEEE International Conference on Acoustics, Speech, and Signal Processing (ICASSP), 2002.

[6]. ShahnamMirzaei, AnupHosangadi, Ryan Kastner," FPGA Process of High Speed FIR Filters Using Add and Shift Method," Proceedings International Conference on Calculater Design, 2006, pp 308-313.

[7]. K.D.Underwood and K.S.Hemmert, "Closing the Gap: CPU and FPGA Trends in Sustainable Floating-Point BLAS Performance," presented at International Symposium on Field-Programmable Custom Computing Machines, California, USA, 2004.

[8]. B. L. Hutchings and B. E. Nelson, "Gigaop DSP on FPGA," presented at Acoustics, Speech, and Signal Processing, 2001. Proceedings. (ICASSP '01). 2001 IEEE International Conference on, 2001.

[9]. A.Alsolaim, J.Becker, M.Glesner, and J.Starzyk, "Architecture and Application of a Dynamically Reconfigurable Hardware Array for Future Mobile Communication Systems," presented at International Symposium on Field Programmable Custom Computing Machines (FCCM), 2000.

[10]. T.Yokota, M.Nagafuchi, Y.Mekada, T.Yoshinaga, K.Ootsu, and T.Baba, "A Scalable FPGA-based Custom Computing Machine for Medical Image Processing," presented at International Symposium on Field-Programmable Custom Computing Machines (FCCM), 2002. 317-323.

[11]. WeikangQian, Chen Wang, Peng Li, David J. Lilja, Kia Bazargan, Marc D. Riedel, "An Efficient Process of NIUsing Logical Computation on Stochastic Bit Streams,"

About authors

Author 1:
Name: Hemavathi H
Email: hemavathihu@gmail.com
Designation: (Student) Master of Technology in Digital Electronics
Department: Electronics & communication Engineering
Name of College: East West institute of technology
Place of College: Bangalore-91

Author 2:
Name: Ravichandran V
Email: Ravi33811@gmail.com
Designation: Assistant Professor
Department: Electronics & communication engineering
Name of College: East West institute of technology
Place of College: Bangalore-91