Implementation of FIR Filter on VHDL Using Bit-Parallel Arithmetic

Dimna Dinesh[1] , Prof. Sanjeet Kumar[2]
[1, 2] Dept of Electrical & Electronics Engineering
[1, 2] National Institute of Technical Teachers' Training & Research,
Bhopal (M.P.), India

Abstract- *The Bit-Parallel Arithmetic is extensively used for FIR filter implementation based on VHDL. Bit- Parallel Arithmetic has high computational speed and it also provide an efficient architecture in terms of power consumption and size of the system. In this brief, an architecture using bit-parallel arithmetic is proposed for implementation of FIR filters on VHDL. In the proposed architecture, the pipelining is used to increase the maximum frequency of FIR filter. The proposed architecture consumes less area and offers higher speed operation because the multiplier is omitted. XST (VHDL/Verilog) is used as a synthesis tool, ISim (VHDL/Verilog) is used as a simulator.*

Thus, by using bit-parallel arithmetic the FIR filter will have high processing speed, small area and superior performance for suppressing out-of-band signals and increasing the signal to noise ratio.

Keywords- VHDL, FIR filter, Bit-Parallel Arithmetic, Xilinx 14.7 ISE desin suit.

I. INTRODUCTION

Digital filters are typically used to modify the attributes of a
signal in the time or frequency domain and play important roles in Digital Signal Processing (DSP). Digital filters are preferred over analog filters mainly because their performance does not change with environmental changes and these can be designed for very low frequencies. In the FIR filter, the impulse response sequence is of finite duration i.e. it has a finite number of non-zero terms. The general difference equation for an FIR filter is:

$$y(n) = \sum h(k)x(n-k) \quad k = 0,1,2,\ldots\ldots N-1 \qquad (1)$$

where h(k) is the set of filter coefficients, x(n-k) is the filter input delayed by 'k' samples, y(n) is the filter output and N is the order of the filter. FIR filters are widely used because they provide linear phase, high stability and are easy to implement. Over the years, along with the fast development in very large scale integration (VLSI) field, FIR filters with good performance parameters has become very important. With the increasing length of filter, complexity of implementation increases. Many methods have been proposed to develop efficient architectures for the realization of FIR filters in Very high speed integrated circuit Hardware Descriptive Language (VHDL). One of the algorithm is bit-parallel arithmetic. The main component of bit-parallel arithmetic based computation is lookup table (LUT) that stores pre-computed partial values and can be accessed easily. Inputs to a bit-parallel arithmetic operation are stored in registers. In bit-parallel arithmetic, all bits are conceptually processed at once, i.e. all bits in the inputs are applied in parallel and all of the bits in the output occur simultaneously and the obtained output is stored in the registers.

An advantage of bit-parallel arithmetic is that the amount of work performed by a processing element during one clock cycle is relatively large, and the clock frequency can be kept low. It means it has high computational speed.

In this brief, a low pass FIR filter is designed and implemented on VHDL using an architecture based on bit-parallel arithmetic. The implemented filter meets the design objective for low power, small area and high speed.

This brief is organized as follows: section II presents Bit-Parallel Arithmetic, section III architecture , section IV conclusion and section V reference.

II. BIT - PARALLEL ARITHMETIC

Typically, inputs and outputs to a bit-parallel arithmetic operation are stored in registers. In bit-parallel arithmetic, all bits are conceptually processed at once, i.e., all bits in the inputs are applied in parallel and all of the bits in the output occur simultaneously. However, in practice it is necessary to process them sequentially. An advantage of bit-parallel arithmetic compared to bit-serial arithmetic is that the amount of work performed by a processing element during one clock cycle is relatively large, and the clock frequency can therefore be kept low.

(A) Addition and Subtractions:-

A sum Z of two numbers X and Y in two's complement representation is computed by adding
the bits of two and two, as shown in Eq. (2). Carry values are propagating from least significant bit (LSB) up to the most significant bit (MSB).

$$S = X + Y = -x_0 + \sum_{k=1}^{W_d-1} x_k 2^{-k} + -y_0 + \sum_{k=1}^{W_d-1} y_k 2^{-k} = -(x_0 + y_0) + \sum_{k=1}^{W_d-1} (x_k + y_k) 2^{-k}$$

(2)

This can be implemented in parallel using a set of full - adders, which adds the bits on the same significance level including a carry bit from the lower significance level. A straight forward implementation is shown in Fig (1). The carry input at the LSB is set to zero, and the carry output from each significance level is connected to the next significance level.

The result of bit depends on every input bit of equal or lower significance level. There will therefore be a combinatorial path from LSB through all full-adders to the MSB resulting in a long propagation delay.

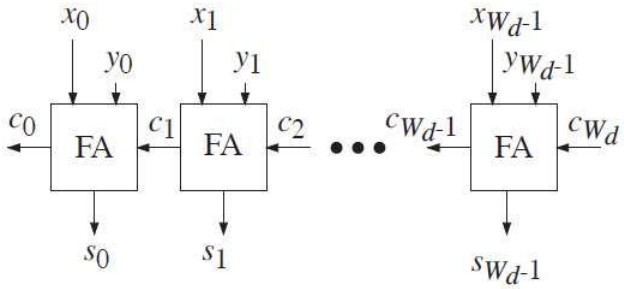

Fig: 1: Bit-parallel ripple carry adder

The computation of the result will be sequential in the worst case, starting at LSB and generating carry values up to MSB.

Many techniques have been proposed to avoid this problem of long carry propagation paths, e.g. carry look-ahead, carry-save, and carry-select. One common property of these solutions is the increase of resources that are required to speed up the computation compared to the ripple-carry implementation.

Unwanted switching in the logic circuits is generated by implementations using the simple full adder based structure in Fig. ,as intermediate incorrect results are Computed before the correct carry has arrived to a bit level stage. The number of full-adders, and therefore also the carry propagation path limiting the addition time, is proportional to the data word length Wd. Subtraction is carried out using the same structure as for addition. By using the property that the sign of a two's complement number is changed by inverting all bits and adding one to the LSB position, the addition is converted into subtraction by inverting the value to be subtracted, and setting the input carry bit at the LSB.

(B) Multiplication:-

Binary multiplication may be carried out using a scheme similar to common hand calculation. An array of partial-product terms is generated and then added as shown in Fig. 4.2. Each dot in the summation array corresponds to two digits multiplied, and this is in the binary case equivalent with a logic AND function of two bits.

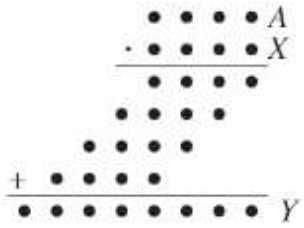

Fig.2: Matrix of partial product generated in multiplications

Summation of the partial-products can be performed in various ways [8]. The straightforward method of using a full-adder for the addition of each dot will result in the array multiplier shown in Fig.3, with a multiplication time proportional to the sum of the data word length and coefficient

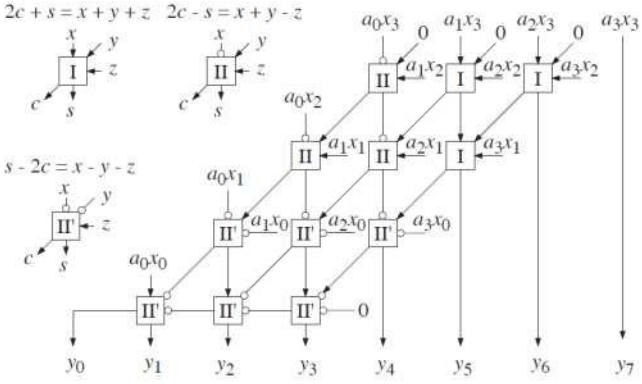

Fig.3: Array multiplier for two's compliment numbers

The required area will be proportional to the data word length and the coefficient word length (Wd × Wc). Other methods of adding the partial product terms include Wallace trees and similar structures, where the carry

propagation is reduced by changing the addition order of the input data.

Such addition schemes use a treelike adder structure to speed up the additions, thereby reducing the propagation delay. Carry is only propagated from one level to another, resulting in short combinatorial paths. Only the final step, where the two last intermediate results are to be added, requires a carry-propagate adder.

III. ARCHITECTURE

The function is implemented by designing a correct LUT for it. The LUT will have 2^N(i.e. 2^4=16) entries and it is preprogrammed to accept the 4-bit input vector $b_{n,1}$ and its corresponding output $b_{n,1}$ h(n) . Depending on the input bit combinations, the corresponding values of LUT data will be accessed and it will be appropriately scaled by the 'power of two'. This scaled value will be accumulated. After following this process for N (i.e. 4) look-up accesses, the sum of products y(n) is computed. Table I shows the detail of 4 – input LUT of size $2^4 \times$ M (where M is the word length of coefficients).

TABLE I. BASIC 4- INPUT LUT

$b_{3,3}$	$b_{2,3}$	$b_{1,3}$	$b_{0,3}$	LUT Data $\{b_{n,1}h(n)\}$
0	0	0	0	0
0	0	0	1	h(0)
0	0	1	0	h(1)
0	0	1	1	h(1)+h(0)
0	1	0	0	h(2)
0	1	0	1	h(2)+h(0)
0	1	1	0	h(2)+h(1)
0	1	1	1	h(2)+h(1)+h(0)
1	0	0	0	h(3)
1	0	0	1	h(3)+h(0)
1	0	1	0	h(3)+h(1)
1	0	1	1	h(3)+h(1)+h(0)
1	1	0	0	h(3)+h(2)
1	1	0	1	h(3)+h(2)+h(0)
1	1	1	0	h(3)+h(2)+h(1)
1	1	1	1	h(3)+h(2)+h(1)+h(0)

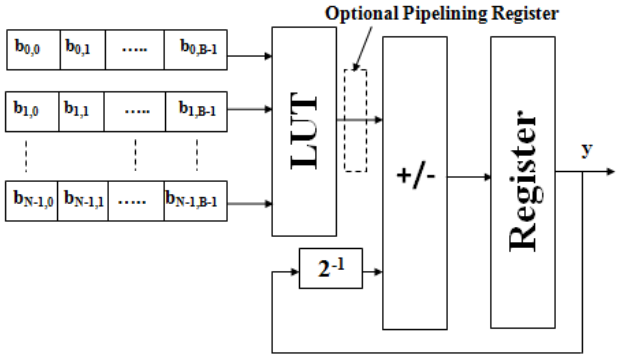

Fig.4: Basic LUT based parallel architecture

Fig4 shows the basic implementation of parallel arithmetic architecture discussed above.

In case the number of coefficients N is a large number,
then the size of the LUT (which will increase exponentially in the power of two) will be quite large. Therefore, it will be tough to implement it with the use of one LUT. In this case, the partial tables can be used and their results are added to get the final output.

FIR filter is designed in VHDL and will take its synthesis and simulation results in ISE Design suit 14.7.

IV. RESULTS ANALYSIS

Registers are used for simply storing a binary word temporarily. Registers are needed to perform the storing the bits during the multiplication. We have also used multipliers, converters, full adder in designed circuit.

Fig.5: Simulation output waveform of Filters

Filter Project Status (02/20/2018 - 11:22:00)			
Project File:	FIR_Filters_bit_parallel_arithmetic.xise	Parser Errors:	No Errors
Module Name:	filter	Implementation State:	Synthesized
Target Device:	xc3s1000-5cp224	• Errors:	No Errors
Product Version:	ISE 14.7	• Warnings:	3 Warnings (3 new)
Design Goal:	Balanced	• Routing Results:	
Design Strategy:	Xilinx Default (unlocked)	• Timing Constraints:	
Environment:	System Settings	• Final Timing Score:	

Device Utilization Summary (estimated values)			
Logic Utilization	Used	Available	Utilization
Number of Slice LUTs	4	17400	0%
Number of fully used LUT-FF pairs	0	4	0%
Number of bonded IOBs	28	216	13%

Detailed Reports					
Report Name	Status	Generated	Errors	Warnings	Infos
Synthesis Report	Current	वस्त. 20. फरवरी 11:21:14 2018	0	3 Warnings (3 new)	0
Translation Report					
Map Report					
Place and Route Report					
Power Report					
Post-PAR Static Timing Report					
Bitgen Report					

Secondary Reports		
Report Name	Status	Generated
ISIM Simulator Log	Current	मंगल. 20. फरवरी 11:48:18 2018
Post-Synthesis Simulation Model Report	Current	मंगल. 20. फरवरी 11:21:58 2018

Date Generated: 02/20/2018 - 11:59:26

Fig.6: Design summary

V. CONCLUSION

The bit parallel arithmetic is used to design FIR filter in VHDL. The Symmetry property of FIR system will be used to reduce the size while the bit-parallel approach and pipelining in structure is used to increase the speed of the proposed system for optimized cases. The FIR filter will have high processing speed, small area and superior performance for suppressing out-of-band signals and increasing the signal to noise ratio.

VI. APPENDIX

This research is to develop/improve the design of DSP algorithm and hardware co-design with aim of obtaining efficient architectures with respect to design effort, throughput, chip-area and power consumption.

VII. ACKNOLEDGMENT

I take this opportunity to express a deep sense of gratitude for the guide Prof. Sanjeet Kumar faculty of Department of Electrical and Electronics Engineering, National Institute of Technical Teachers' Training & Research, Bhopal (M.P.) for providing excellent guidance and unbelievable support. It is because of their constant and general interest and assistance that this dissertation has been successful. I would like to thank all the faculty members of Department of Electrical and Electronics Engineering, NITTTR, Bhopal for their valuable suggestions and helpful discussions.

I also wish to thank Prof. (Mrs.) Susan S. Mathew, HOD Electrical, and Electronics Engineering, NITTTR, Bhopal who helped me during the period of entire course including dissertation work. I am especially thankful to our Director Dr. C. Thangaraj, for his kind co-operation and rendering me all possible facilities to carry out this research work successfully. I would also like to thank my family members who have been a source of encouragement and inspiration throughout the duration of this dissertation.

REFERENCES

[1] Hanho Lee and Gerald E. Sobelman "FPGA based FIR filter using Digit-serial Arithmetic" **1063-0988/97/$10.00, 1997 IEEE.**

[2] Jitesh R Shinde & Suresh S Salankar " VLSI Implementation of Bit Serial Architecture based Multiplier in Floating Point Arithmetic" 978-1-4799-8792-4/15/$31.00 , 2015 IEEE.

[3] Sheikh Md. Rabiul Islam, Robin Sarker, Shumit Saha, A. F. M. Nokib Uddin " Design of a programmable digital IIR filter based on FPGA" 978-1-4673-1154-0112/$31.00 , 2012 IEEE.

[4] Michael Francis "Infinite Impulse Response Filter structures in Xilinx" WP330 (v1.2) August 10, 2009.

[5] Shaheen Khan & Zainul Abdin Jaffery "low power FIR filter implementation on FPGA using parallel distributed arithmetic" IEEE INDICON 2015 1570173627 .

[6] 'DSP Integrated circuit' book by Lars Wanhammar, 1999.

[7] 'Digital Signal Processing with Field Programmable Gate Arrays' book by U. Meyer-Baese, © Springer-Verlag Berlin Heidelberg 2007.

[8] A VHDL primer third edition by J. Bhasker. Copyright © 1999 Pearson Education Inc., upper saddle river, New Jersey 07458, U.S.A. ISBN-978-81-203-2366-7. 1999