Hardware Implementation of MAC using MATLAB Simulink and FPGA

Pritee Singh
M. Tech Student
Department of Electronics & Communication Engineering
Goel Institute of Technology & Management, Lucknow, India

Mr. Faseeh Ahmad
Assistant Professor
Department of Electronics & Communication Engineering
Goel Institute of Technology & Management, Lucknow, India

Abstract

The MAC unit is considered as one of the fundamental operations in DSP and it becomes a basic component in Application-Specific-Integrated-Circuits (ASIC).The MAC unit determines the speed of the overall system; it always lies in the critical path. Developing a high speed MAC is crucial for real time DSP applications. In other words the MAC operation is the main computational kernel in Digital Signal Processing (DSP) architectures. This paper shows the real time hardware and software implementation of MAC unit. This proposed MAC Unit is able to perform different arithmetic operations at high speed. Combinatorial form has been utilized to design all sub-modules being used in the MAC unit. And integrated in the final unit, reset and clock functionality has been provided in this final unit to have better control on the circuitry. The complete design has been developed by using MATLAB Simulink and then simulated and synthesized using XILINX ISE TOOL for FPGA Implementation... For this design the target FPGA device belongs to Spartan-6 (family), XC6lS45 (device), CSG324 (package) with speed grade of -4. Xilinx synthesis tool (XST) of Xilinx ISE-14.x has been used for synthesis purpose and for design MATLAB has been used. For the behavioral simulation purpose ISE simulator has been used.

Keywords- MAC, System Generator, DSP, FPGA, MATLAB, Simulink

I. INTRODUCTION

In every Digital Signal Processing applications the multiply accumulate unit is the most essential building blocks that is frequently used. Applications such as Digital filtering, speech processing, video coding and CDMA require high processing speed while maintaining a low power consumption that allows embedding powerful processors in applications that are using portable power supplies. In order to improve the speed of this unit, there are two major bottlenecks that need to be considered. The first one is the partial products reduction network that is used in the multiplication block and the second one is the accumulator and both stages requires the addition of large operands that involved in long paths for carry propagation. The classical straightforward approach to build multiply accumulate units implements each part separately using individual functional blocks and then cascade them to realize the complete operation. There is one another approach to faster the operation by implementing both the multiplication and the accumulation operations within the same functional block. And then merge the accumulator with the computational platform for hardware and software components in order to interact with the environment and other nodes of the multiplication circuit. In order to speed up the multiplication process there is another solution by using tree architectures for the partial products reduction network. Our contribution in this paper is introducing a merging technique for further speeding up the accumulation operation by merging it within the partial products reduction tree used for multiplication. Each sensor node has a processing unit. Many different types of processing units can be integrated into a sensor node. There are a large number of commercially available microcontrollers, DSPs, and field programmable gate arrays (FPGAs), which allows a big flexibility for processing unit implementations. Most of the sensor nodes available in the market depend on an 8-bit or 16-bit microcontroller. FPGA is not used in the current sensor nodes because its power consumption is not as low as sensor nodes should be. Moreover the FPGA is not compatible with traditional programming methodologies. Currently, DSP is being a challenge for node demanding, such as a gateway or a robust sensor node, which can be the head of hierarchical cluster in a wireless sensor network. The future WSN needs DSP for more computational capabilities in order to engage in signal processing operations of the complex applications. Signal processing in wireless sensor network has a huge range of applications. Infinite Impulse Response filtering (IIR), Finite Impulse Response filtering (FIR), and Kalman Filter (KF) find applications in object tracking, environmental monitoring, surveillance, and many other applications. These tasks require thoroughly simple to complex computational calculations and thus they could easily overextend the power/energy resources of any single computational node in a wireless sensor network. In other words, most sensor nodes do not have the computational resources to complete many of these signal- processing tasks repeatedly. Since MAC unit is the core unit of DSP architectures. Therefore, saving power at the MAC unit level will have a significant impact on the energy consumption of each node.

A. General Construction of MAC

The multiplication and accumulation operation is a basic step in computing, especially for digital signal processing applications. This step computes the product of two numbers and adds that product to an accumulator. The hardware unit of DSP's that performs this operation is known as a multiplier–accumulator (MAC, or MAC unit) and the operation itself is usually called a MAC or a MAC operation. The MAC operation modifies an accumulator 'A' unlike in traditional processors/controllers. The MAC unit is considered as one of the fundamental operations in DSP and it becomes a basic component in Application-Specific-Integrated-Circuits (ASIC).The MAC unit determines the speed of the overall system; it always lies in the critical path. Developing a high speed MAC is crucial for real time DSP applications. Due to increasing demand for the wireless sensor networks, the MAC unit(s) with low power consumption will definitely lead the market. In order to improve the speed of the MAC unit, there are two major bottlenecks that need to be considered. The first one is the partial products reduction network that is used in the multiplication block and the second one is the accumulator. Multiply-Accumulate is a common operation that computes the product of two numbers and adds that product to an accumulator. The multiplier A and multiplicand B are assumed to have n bits each and the addend Z has (2n+1) bits. $Z \leftarrow (AxB) + Z$. The MAC Unit is made up of a multiplier and an accumulator as shown in Fig. 1.

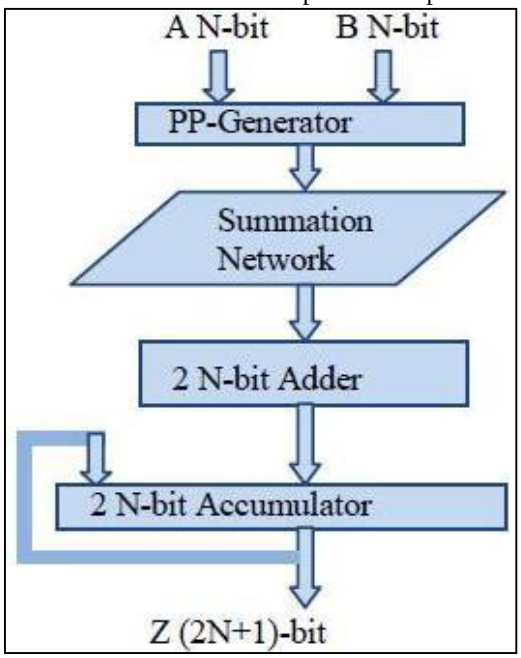

Fig. 1: Basic Mac Unit

The multiplier is divided into the partial products generator, summation network, and final adder. The summation network represents the core of the MAC unit; it reduces the number of partial products into two operands representing a sum and a carry. The summation network occupies most of the area and consumes most of the circuit area and delay. Several algorithms and architectures are proposed in an attempt to optimize the implementation of the summation network. The final adder is then used to generate the multiplication result out of these two operands. The accumulator is used to perform a double precision addition operation between the accumulated operand and multiplication result. Due to the large operand size, the accumulator required a very large adder.

II. RELATED WORK

Wide range of applications such as graphics designing, DSP applications, multimedia, image processing etc. use multiply-accumulate (MAC) functions for its efficient arithmetic operations. MAC arithmetic operations can be used multiply-addition operations i.e. multiplication followed by addition/subtraction. Alexandru Amaricai, et al [3], proposed low-precision FP multiply-addition fused units which depends upon FPGA characteristics so that the MAC (multiply-accumulate) can support the DSPs. In order to use DSPs efficiently and to achieve high performance thereby saving the system cost, aligning and shifting is done prior to the multiply-add stage. This includes right shifting of addend and multiplicands. This proposed method suits well for low-precision formats like half precision where only one digital signal processing block is used for multiplication. This method gives high performance and 25% cost effectiveness when compared with other approaches [3].

Chandrakasan, Sheg et al [2], a high speed 2-cycle MAC architecture is proposed which also energy efficient. MAC architecture includes saturation circuitry and guard bits and it also supports 2's complement numbers. 1st pipeline MAC stage is comprised of reduction tree and partial-product circuit only whereas the 2nd pipeline MAC stage is comprised of all other extra functions. The observations state that this new architecture is 31% faster in computation and decrease energy/operation by 32%.

This new architecture is being used in order to develop a versatile MAC block i.e. double-throughput MAC (DTMAC) that supports various modes of operations i.e. 3 for multiply-accumulate mode and 3 for multiply operations.

Wireless sensor networks have a lot many risks when it comes to energy efficiency and security concerns. Ching-Tsung Hsueh et al [4] proposed an energy efficient approach for securing the network against the attacks that can lessen network's lifetime are also known as power exhausting attacks like Denial-of-Sleep attack.

There is a huge research done on the implementation of MAC unit in different operating blocks such as filters, adders, multipliers, squares etc. Deepika & Nidhi Goel [10] Designed the FIR filter using Carry Look Ahead adder and Booth multiplier helps in making FIR filter operate faster. Booth multiplier works on the principle of "state machine". Proposed architecture can operate for 'n' number of bits but here n is taken to 16 (n=16). Proposed MAC architecture is very high speed architecture which makes FIR filter operates faster [10].

III. DESIGN IMPLEMENTATION

This section shows the various steps involved in designing our MAC unit. We are designing the MAC unit for DSP applications especially for the ADC based applications. Most of the ADCs available in the market are prefabricated with 12-bit resolution. So this MAC unit is very much adequate for such type of ADCs interfacing. Also a ADCs are the circuit which used in any kind of real time sensing and monitoring system e.g. for sensing of temperature(s), sending the voice signals (which are analog in nature) over the far distance or on any specified channel.

Here we had taken 12-bits for multiplicand and 8-bit for the multiplier for the multiplication operation. The 12-bits is for the any signal coming from the ADC or any equivalent unit and 8-bits for the additional amplification or modulated signal parameter (the number of bits for this operand could be vary). The output from the multiplier unit would be of the 20-bit , as per the nature of the multiplier circuit i.e. for multiplication of two n-bit operands the 2n- bits would be appear at the output side.

We are creating this MAC circuit by Xilinx System Generator. And then implemented the entire circuit in Xilinx FPGA. This section will show the complete Implementation procedure for the proposed architecture. Although there are lots of MAC circuits architectures are available in various research papers but hardly anyone shows the implementation of MAC by using MATLAB Simulink and Xilinx FPGA. In other words, if we have any suitable FPGA board we can also provide the software and real time hardware simulation and implementation as well.

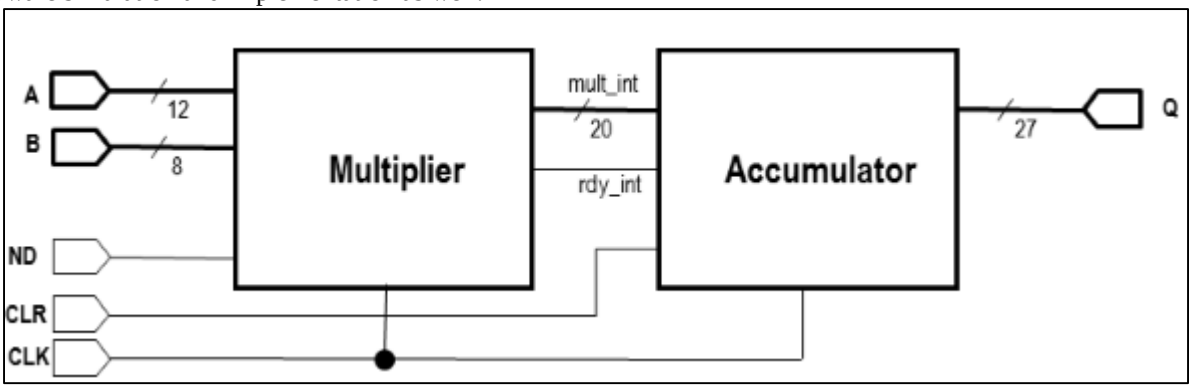

Fig.2: Proposed MAC unit Block Diagram

The XILINX ISE tool is comes with various options such as Web pack version for beginner and simplex design implementation, Logic edition with properties of on chip verification module called chipscopepro, EDK for the lovers of the embedded operations using either Hard IP or SOFT IP core processors and at last is the System Generator. The System Generator provides the digital signaling processors block set such as adders, multipliers, registers, filters and memories for application specific design. These blocks leverage the Xilinx IP core generators to deliver optimized results for the selected device. For using the system generator it doesn't requires the previous experience with Xilinx FPGAs or RTL design methodologies. Designs are captured in the DSP friendly Simulink modeling environment using a Xilinx specific Block set. And for implementing the created design on to the FPGA, the steps including synthesis and place and route are automatically performed. The major advantage of using Xilinx system generator for hardware implementation is that Xilinx Block set provides close integration with MATLAB Simulink that helps in co-simulating the FPGA module with pixel vector provided by MATLAB Simulink Blocks.

The System Generator block has the feature to define which type of FPGA board can be used, as well as provide several additional options for clock speed, compilation type and analysis. There is a library of more than 90 DSP building blocks, allows System Generator for fast prototyping and design from a high-level programming stand point. There is also an option for other tool user(s)/designer(s) to use the M-code and Black box for direct programming in MATLAB M-code, C code, and Verilog or VHDL to simplify integration with existing projects or customized block behavior. System Generator projects can also easily be placed directly onto the FPGA as an executable bit stream file as well as generating Verilog code or VHDL for additional optimizations or integration with existing projects within the Xilinx ISE environment..

A. FPGA Design Flow

In the HDL flow, two sets of codes must be written: HDL design and Verification code (test bench) for Behavioral and Functional simulation. The designer is responsible for testing and should create the environment for verification. The figure 3.2 shows the complete design flow for the FPGA implementation. The design process typically begins with a functional description of the desired functionality by means of a high level description language. Several languages have been developed to this purpose (VHDL, Verilog HDL, and Hardware C).

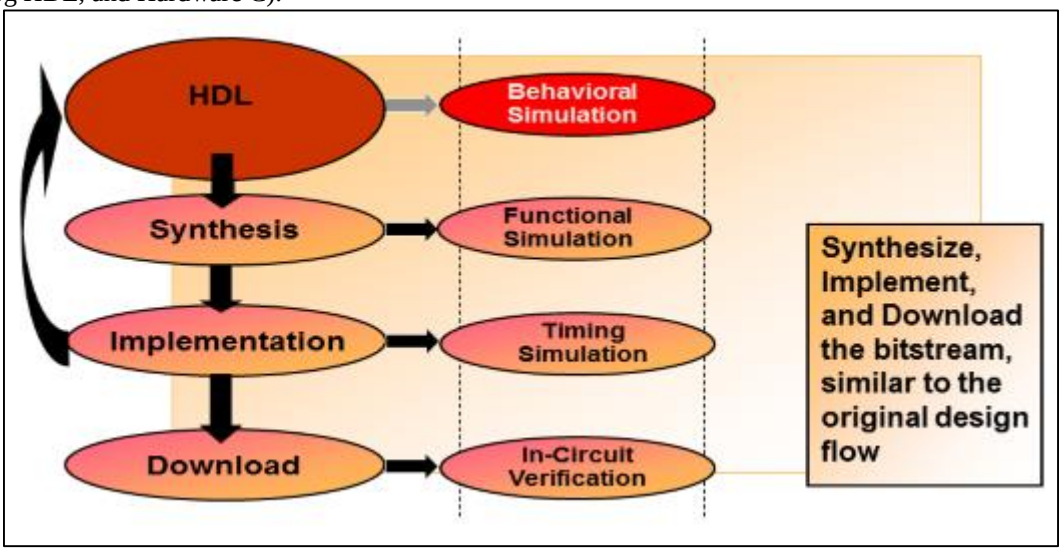

Fig. 3: Traditional FPGA design Flow

The CAD Tools have been developed for High Level Synthesis as the first design step, which consist of functional description mapping along with a set of area & Performance constraints in terms of registers & combinational functional Units i.e. called Primitives e.g. ports, adders, comparators, multipliers, shifters. This stage is far differing from the format of data & Control Signals.

Register Transfer Level (RTL) is the output of High Level Synthesis. This representation is typically divided into control and data path portions. The control unit is responsible for activation of the portions of data path as per the given schedule, so the desired computation can be achieved. The major problem with High level synthesis is the selection of a scheduling in such a manner that a minimal number of resources of a device are being attracted. The high level synthesis does not allow accurate estimates of the figures of merit of a circuit. Therefore, a direct mapping of an RTL design into a logic circuit rarely meets area, speed or power requirements. Thus optimization at the logic level is always needful step. Now we can say that the FPGA design process is much similar to that for usual technologies such as standard cell or gate array. Which consists of the system-level design (high level synthesis), the logic-level design (logic synthesis), and the physical-level design (layout synthesis).In logic-level design for FPGAs consists of combinational logic synthesis, which optimizes the logic gate networks and sequential logic synthesis, which deals with the assignment and/or arrangement of storage elements. On the other hand system level synthesis consists of the process that automatically generates high level behavioral specifications into circuit implementations with the increasing design complexity of ICs.

B. Simulink FPGA Design Flow

For DSP designer(s) especially in the past, if wanted to target an FPGA, he/she would have no option but a "dual path" of development. The DSP designer writes an algorithm in pseudo-C, using filters, certain C code and certain precision. The basic problem with the DSP engineer(s) that he may know everything about DSP and Simulink models, but may not know anything about FPGAs or real time implementation for verification. Thus, due to this gap he not knows how to use an FPGA, he doesn't know how to take advantage of the FPGA architecture, or how to write a design to avoid a bad FPGA implementation. So after completing his DSP design or may have a working model in Simulink, he must have to design the same thing in HDLs, or he has to gives his design to an FPGA implementer (who may know nothing about DSP) who writes the VHDL for him. The implementer might end up using a core that doesn't do exactly what the designer wants, by not being a DSP expert, the FPGA implementer is just trying to translate the pseudo code that came to him into VHDL for an FPGA. There is also no way to co-simulate: one is simulating in C in MATLAB, the other simulating in VHDL in a behavioral simulation. It's only when they get into the lab and simulate the board, late in the process, that they find out something's wrong.

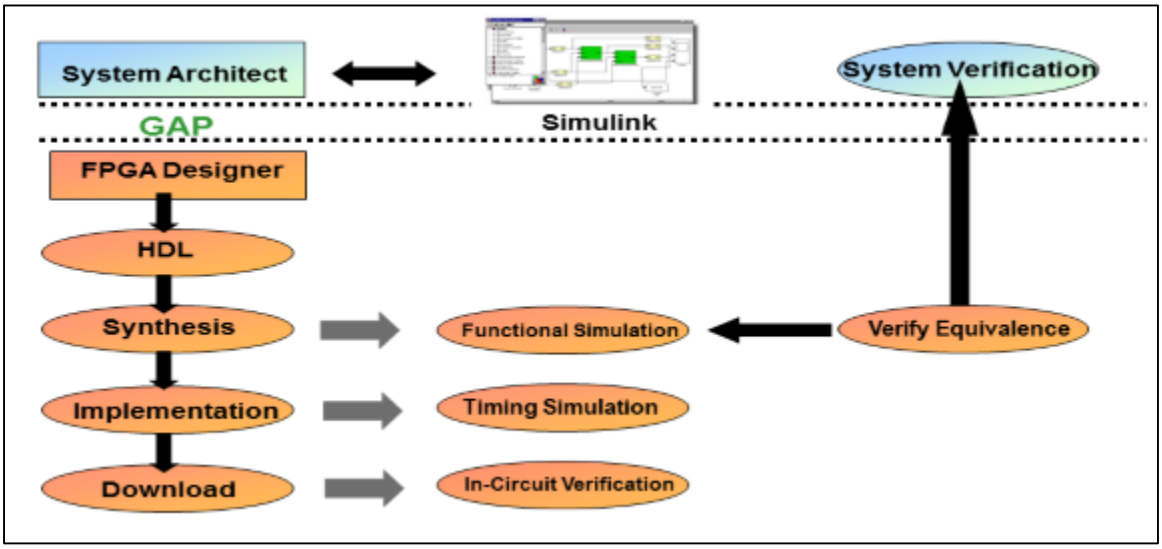

Fig. 4: Simulink FPGA Design Flow

IV. DESIGN IMPLEMENTATION

This part shows the circuit implementation using Simulink. In this we had taken three Basic Inputs (basically a RAMP Signal) there are some blocks for converting analog signal into digital format as our Xilinx Tool deals only with Digital format. So at every input source we had attached the Gateway In block. Then attached the Delay blocks for smoothing the results. There are two main blocks had been used one is Multiplier and another one is Accumulator. And for showing the output waveforms we had attached the Scope(s). There are three scopes has been placed with name of scope, scope1 and scope2. The Scope and scope1 is showing the input signal(s) while scope2 is for final output from our MAC unit. In the input side we are providing the RAMP signal(s). One can provide any type of signals i.e. Sinusoidal, Triangular waves etc.

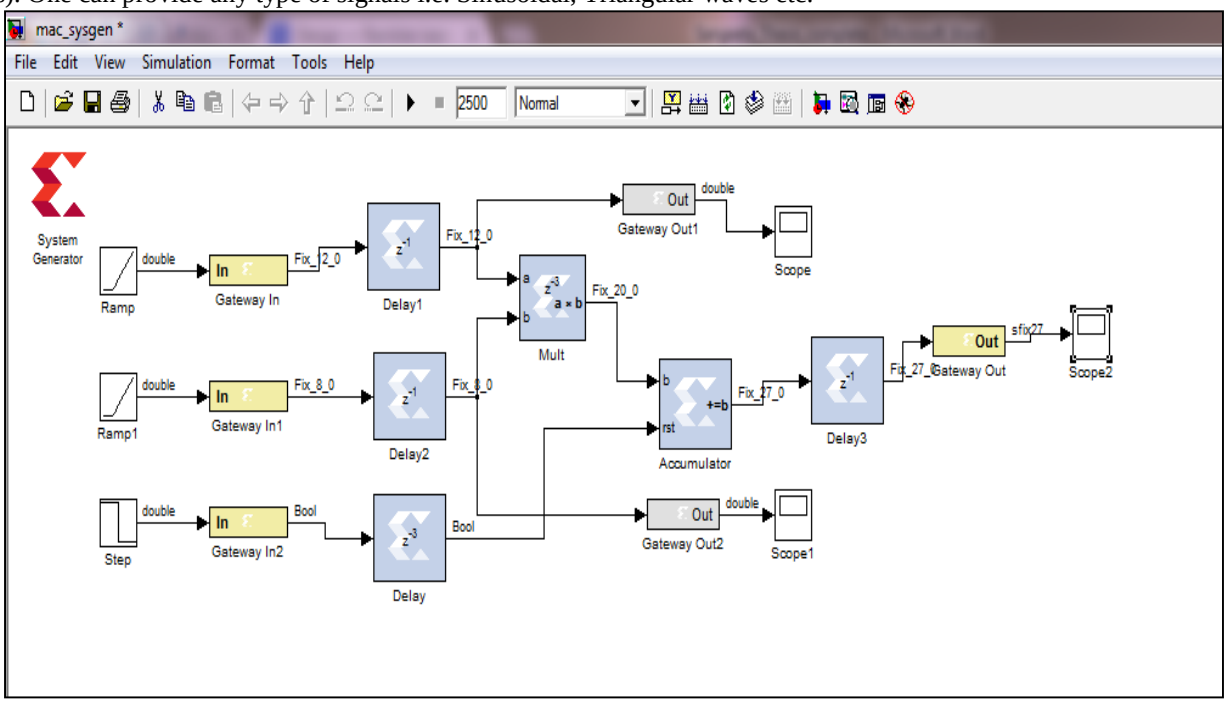

Fig. 5: MAC unit implemented in Simulink

V. RESULTS

After providing the two ramp inputs of different amplitude level as shown in figure 6 & 7 we got the simulation result from the MAC unit as per shown in figure 8. The first ramp input has the highest amplitude level of 2500 and for second ramp signal is provided with 140 and simulate the design for 2500ns time.

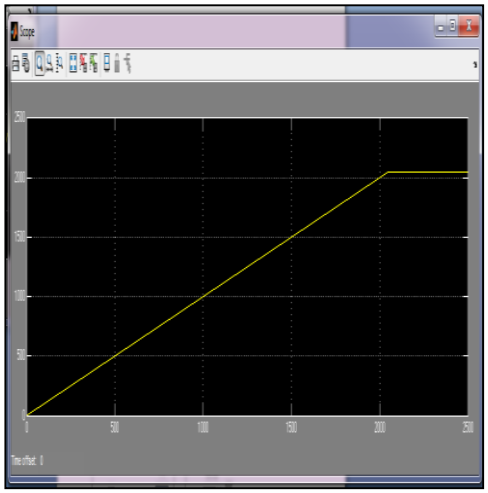

Fig. 6: First RAMP input signal

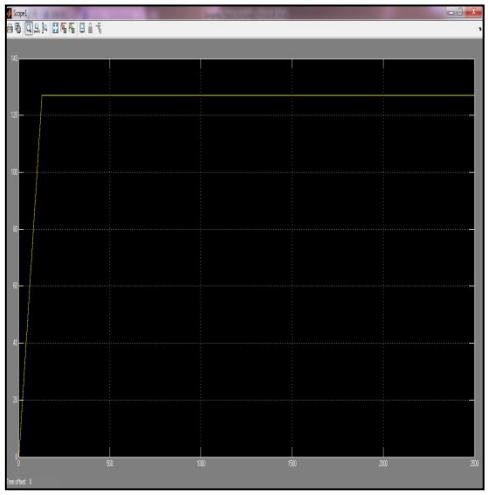

Fig. 7: Second RAMP input signal

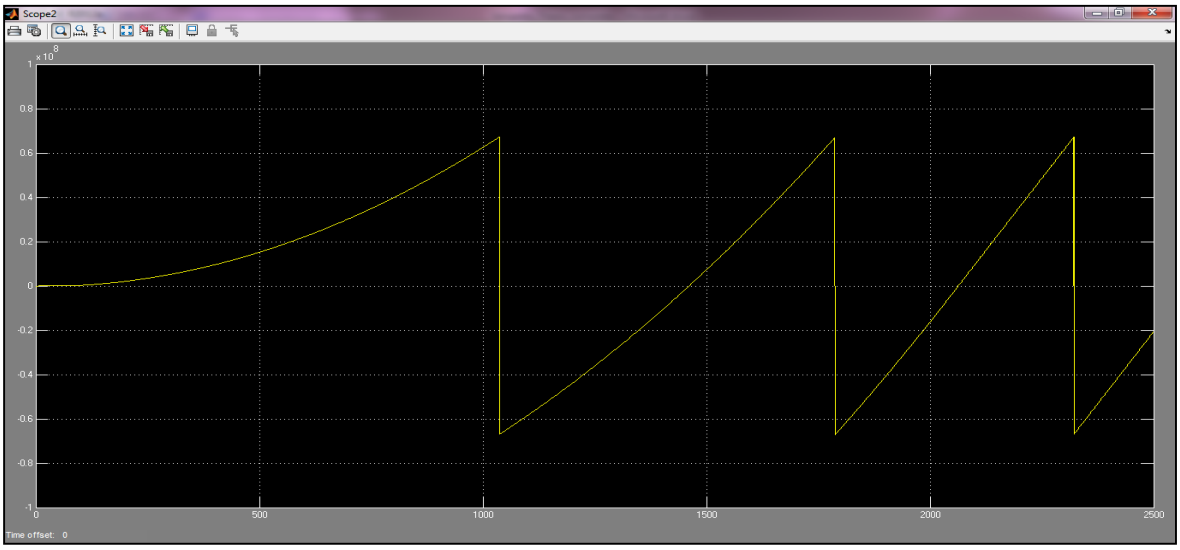

Fig. 8: MAC result from Simulink

Figure 9 will show the result from the XILINX ISIM after providing the input in digital format i.e. binary/decimal/hexadecimal. It shows the digital output for the MAC unit. This is done by writing the VHDL code in Xilinx ISE tool. In this we had taken two inputs for multiplication and accumulation process. CLK, reset are the signals for the synchronous and initialization process. After feeding the input values of '2' in decimal for 12-bit operand and '1' in decimal for 8-bit operand we are getting the result starts from '2' in decimal to '28' in decimal for different clock pulses after reset signal goes low.

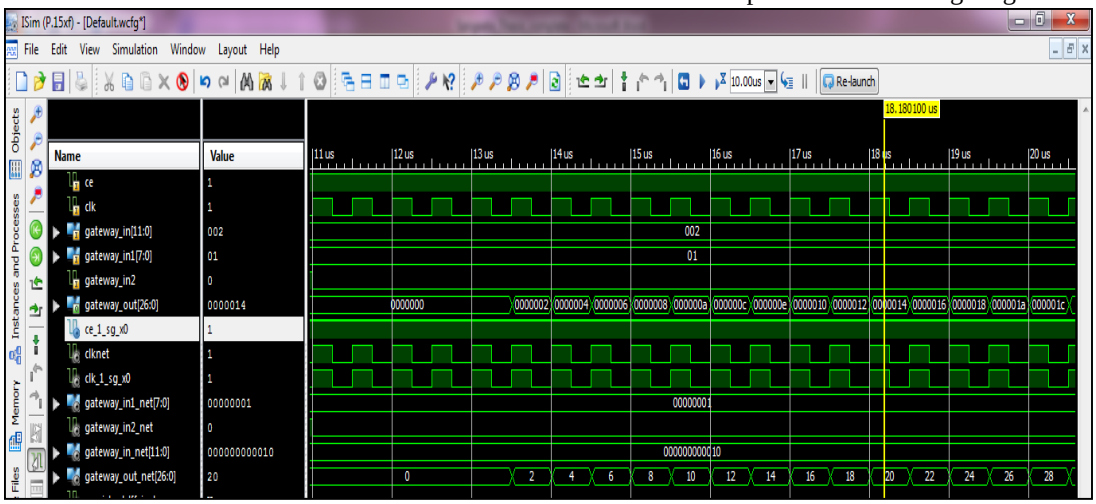

Fig. 9: XILINX ISIM simulation result for MAC

VI. CONCLUSION

Modern FPGAs are capable of implementing high performance DSP functions, but the relative unfamiliarity of many DSP engineers with hardware design has slowed their wider adoption. Tools like System Generator provide means for both high level modeling of DSP algorithms, and implementing custom high performance DSP data paths in FPGAs. We had successfully design and implemented the Multiplier Accumulator for 12 * 8 bit inputs. The complete design has been implemented by using MATLAB Simulink. Then written the VHDL code by the help of the Xilinx sysgen block to implement onto the FPGA. The Spartan 6(Family), XC6slx45 (Device) Csg324 (Package) has been used for the implementation. The complete system has been simulated in Xilinx ISIM and MATLAB Simulink as well. Thus we can say that our design can be used for Software and hardware validation as well. Our designed MAC unit is best suitable for Analog signal filtration. As maximum ADC ICs available in the market are fix with 12 bits. Thus we can easily put our design on any board which is having this type of ADCs. Also MAC is the basic unit for any type of signal processing. We had created this unit suitable for working up to 434.735MHz, with total delay 2.300 ns and power consumption is 0.068w.

Table 1: Comparison chart

Parameter	*Present*	*Previous*
Frequency(MHz)	*434.735*	*307.977*
Power(w)	*0.068*	*0.242*
Delay(ns)	*2.300*	*3.247*

REFERENCES

[1] Hardware Multiply/Accumulate (MAC) Unit, Motorola/
[2] Chandrakasan, Sheng, & Brodersen, 1992 and Weste & Harris, 3rd Ed.
[3] Alexandru Amaricai, Oana Boncalo, Constantina-Elena Gavriliu, "Low-precision DSP-based floating-point multiply-add fused for Field Programmable Gate Arrays", IET Computers & Digital Techniques, 2013.
[4] Ching-Tsung Hsueh, Chih-Yu Wen and Yen-Chieh Ouyang, "A Secure Scheme against Power Exhausting Attacks in Hierarchical Wireless Sensor Networks", IEEE Sensors Journal, 2014.
[5] Kouretas Member, IEEE, Ch. Basetas, and V. Paliouras Member, IEEE, "Low-power Logarithmic Number System Addition/Subtraction and their Impact on Digital Filters", IEEE TRANSACTIONS ON COMPUTERS, 2012.
[6] Lijuan Li and Shuguo Li, "High-Performance Pipelined Architecture of Elliptic Curve Scalar Multiplication Over GF(2m)", IEEE TRANSACTIONS ON VERY LARGE SCALE INTEGRATION (VLSI) SYSTEMS, 2015
[7] Patrick Maechler, Christoph Studer, David E. Bellasi, ArianMaleki, Andreas Burg, Norbert Felber, Hubert Kaeslin, and Richard G. Baraniuk, "VLSI Design of Approximate Message Passing for Signal Restoration and Compressive Sensing", IEEE JOURNAL ON EMERGING AND SELECTED TOPICS IN CIRCUITS AND SYSTEMS, VOL. 2, NO. 3, SEPTEMBER 2012
[8] Bo Marr, Brian Degnan, Paul Hasler, and David Anderson, "Scaling Energy per Operation via an Asynchronous Pipeline", IEEE TRANSACTIONS ON VERY LARGE SCALE INTEGRATION (VLSI) SYSTEMS, VOL. 21, NO. 1, JANUARY 2013.
[9] A. Sathya, S.Fathimabee, S. Divya "Parallel Multiplier-Accumulator based on Radix-2Modified Booth algorithm by using a VLSI Architecture", IEEE
[10] Deepika, Nidhi Goel "Design of FIR Filter Using Reconfigurable MAC Unit", 2016 3rd International Conference on Signal Processing and Integrated Networks (SPIN), 2016
[11] Ugur Cini, Olcay Kurt "A MAC Unit with Double Carry-Save Scheme Suitable for 6-Input LUT Based Reconfigurable Systems", International Conference on Design & Technology of Integrated Systems, 2015
[12] Gitika Bhatia, Karanbir Singh Bhatia, Osheen Chauhan, Soumya Chourasia and Pradeep Kumar "An Efficient MAC Unit with Low Area Consumption", IEEE INDICON 2015.
[13] Gitika Bhatia, Karanbir Singh Bhatia, Shashank Srivastava, and Pradeep Kumar "Design and Implementation of MAC Unit Based on Vedic Square, and It's Application", IEEE UP Section Conference on Electrical Computer and Electronics, 2015.
[14] Wikipedia.com
[15] www.xilinx.com/sysgen
[16] R. Malleshwari and E. Srinivas "FPGA Implementation of Low Power and High Speed 64-Bit Multiply Accumulate Unit for Wireless Applications" , Volume 5 Issue 4, April 2016.