Design of 16-bit Floating Point Multiply and Accumulate Unit

Saravanan R[1], Balaji P[2], Prabu R[3]
[1](Electronics & Communication Engg., Aksheyaa College of Engg., Madhuranthagam, India, saravananr.me@gmail.com)
[2](Electronics & Communication Engg., Aksheyaa College of Engg., Madhuranthagam, India, balaae12@gmail.com)
[3](Electronics & Communication Engg., Aksheyaa College of Engg., Madhuranthagam, India, prabu.6037@gmail.com)

***Abstract*—*In most systems using digital signal processing Multiply-Accumulate (MAC) is one of the main functions. The performance of the whole system depends on the performance of the MAC units. This paper presents the design and implementation of 16-bit floating point Multiply and Accumulate (MAC) unit. Generally MAC unit consists of three units - Floating-point multiplier, Adder and an Accumulator. The input takes form of half-precision format where there is 1-bit for sign, 8-bits for exponent and 7-bits for mantissa thereby making it a high performance unit. The design is coded in Verilog HDL and simulation is done using ModelSim. The proposed 16-bit floating point MAC unit is implemented on Xilinx Spartan 3E field programmable gate array (FPGA) device and synthesized with standard cell library.*

Keywords*—*MAC, Floating point, critical time delay, FPGA.

1. INTRODUCTION

Because of the high-accuracy, impressing dynamic range and use of easily operable rules, floating-point operations have found applications in communication and multimedia systems, the signal processing techniques. The main components used in Digital signal processor (DSP) are multiplier, adder and multiplier and accumulator (MAC) unit. MAC is used extensively in many applications like convolution and filtering. A basic MAC architecture consists of a multiplier and accumulator. The products generated by the multiplier are added and stored in accumulator. The performance of MAC unit plays an important role in the design of filters which are used in DSP applications. The performance of MAC can be increased by the optimized design of multiplier and adder. A high speed MAC capable for handling large range numbers with better precision will be required for many of the DSP application. A floating point number is represented as M × BE where M is mantissa, B is base and E is exponent. The floating point representation used here is a half precision. In this design the input is in 16 bit floating point representation (half precision) and the output is in 16 bit floating point representation (half precision). We have coded the proposed 16-bit floating point MAC in Verilog HDL, and it is synthesized with Xilinx Spartan 3E XC3S250E-5-PQ208 device.

The rest of the paper is structured as follows. In Section II, introduction to floating point, MAC and the factors influencing its performance improvements and computational complexity are compared with the existing methods. In Section III, digital hardware architectures are supplied for 16-bit floating point MAC analysis. Hardware resource consumptions using field programmable gate array (FPGA). The hardware implementation for the proposed 16-bit floating point MAC are detailed in Section IV. Conclusion and final remarks are given in Section V.

2. PROPOSED 16-BIT FLOATING POINT NUMBER

The proposed half precision radix-2 binary floating-point is shown in fig.1.

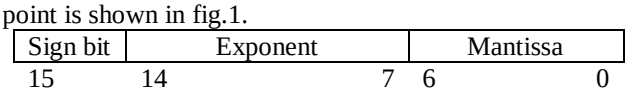

Fig.1. Proposed half precision floating point number

The sign bit indicates whether the number is positive (sign bit=0) or negative (sign bit=1). Exponent is biased exponent and the mantissa is in normalized form.

3. DIGITAL ARCHITECTURES AND REALIZATIONS

A. 16-bit Floating Point Multiplier

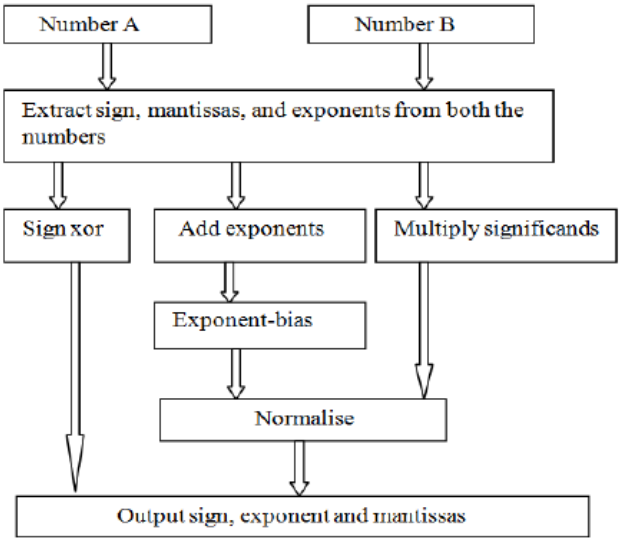

Fig.2. Block diagram of 16-bit floating point multiplier

In Fig.2, we present a 16-bit floating point multiplier block diagram. The sign, exponent and mantissas are extracted from both the numbers respectively. Pipelining has been used for designing multiplier. The sign bits of both the numbers are XORed. The 8 bit exponents are added and then bias (127) is subtracted from it. For multiplying 8 bit multiplier is used. The 8 bits are considered as the most significant bits out of which 16 bits are the mantissa bits. The result is then normalized for proper approximation to closest value. The approximation consists of a possible single bit right shift and corresponding exponent is incremented depending on b1 bit. The resultant sign, exponent and mantissas are obtained.

B. 16-bit Floating point adder

The circuit in the first stage compares the magnitudes and calculate the larger number and the smaller number. The comparison is done between exponent1 & mantissa1 and exponent2 & mantissa2. The circuit in the second stage performs alignment. It first calculates the difference between the two exponents, which is exponent2 – exponent2, and then shifts the mantissa2 to the right by this amount. The circuit in the third stage performs sign-magnitude addition. Note that the operands are extended by 1 bit to accommodate the carry-out bit. The circuit in the fourth stage performs normalization, which adjusts the result to make the final output conform to the normalized format.

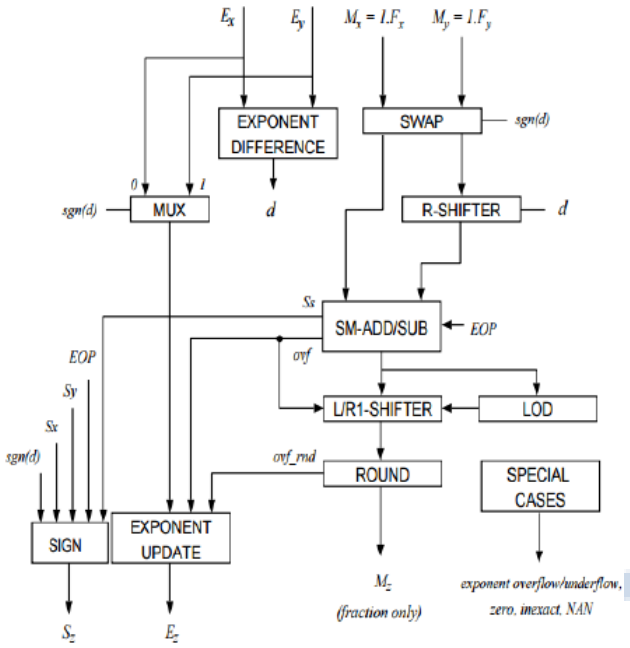

Fig.3. Block diagram of 16-bit floating point adder

C. 16-bit Floating point MAC unit

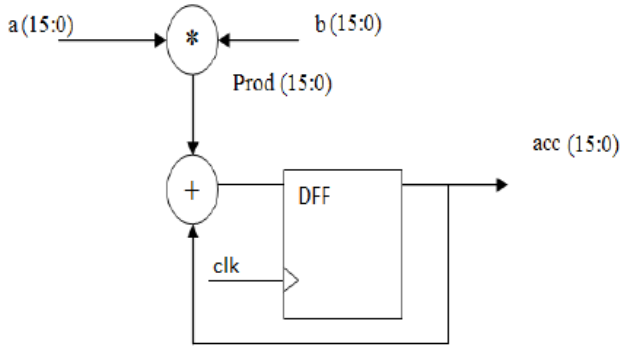

Fig.4. Block diagram of 16-bit floating point MAC unit

The digital architecture of 16-bit floating point MAC unit as shown in Fig.2, basically is composed of adders, multiplier and an accumulator. The inputs which are given to the MAC are fetched from memory location and fed to the multiplier of block of MAC which performs multiplication and gives the result back to adder which will accumulate the result and store it in a memory location. Complete process is achieved in a single cycle. The design consists of 16-bit floating point adder and 1 register for memory location. The most important feature that differentiates general processor from digital signal processor is it's multiply and accumulate unit. Each DSP algorithm would require some form of Multiplication and accumulation system. This one is the most important block in DSP systems. The hardware cost is measured by the number of adders, multipliers, and shifters used in the architecture, and the computing time is normalized as clock cycles.

4. FPGA IMPLEMENTATIONS

Discussed methods were physically realized on a FPGA based rapid prototyping system for various register sizes and tested using on-chip hardware-in-the-loop co-simulation. The architectures were designed for digital realization within the Xilinx with synthesis options set to generic Verilog HDL generation. This was necessary because the auto-generated register transfer language (RTL) hardware descriptions are targeted on both FPGAs. The proposed architectures were physically realized on Xilinx 3E XC3S250E-5-PQ208 device at a maximum clock frequency of 820.197MHz. i.e Maximum critical time delay is 1.219ns

Selected Device : 3s250epq208-5

Number of Slices : 157 out of 2448 6%

Number of Slice Flip Flops: 55 out of 4896 1%

Number of 4 input LUTs : 279 out of 4896 5%

Number of IOs : 50

Number of bonded IOBs : 50 out of 158 31%

IOB Flip Flops : 16

Number of MULT18X18SIOs: 1 out of 12 8%

Number of GCLKs : 1 out of 4 4%

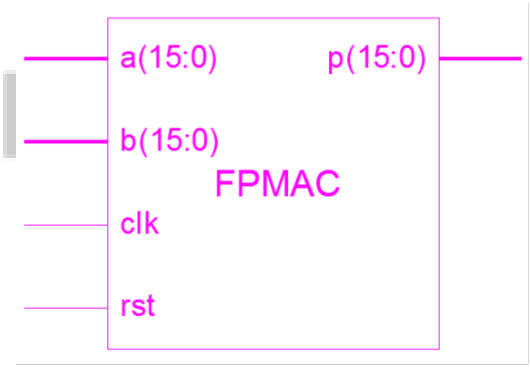

Fig.5. RTL Schematic

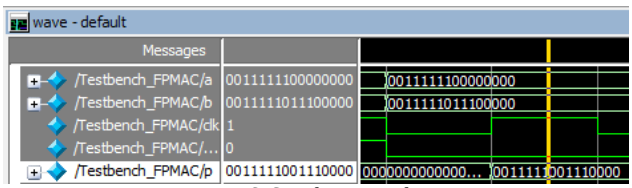

Fig.6. Simulation result

5. CONCLUSION

In this paper, we proposed a 16-bit floating point MAC unit that require only 1.219ns critical time delay. The proposed MAC unit possess lower computational complexity and are faster than all other MAC unit under the consideration. The proposed MAC unit could outperform in terms of digital signal processing. Hence, the new 16-bit floating point MAC unit is the best MAC unit for the DSP

applications in terms of computational complexity and speed. Introduced MAC unit was digitally implemented using Xilinx FPGA tools.

REFERENCES

[1] Strenski.D, Cappello.J.D, "A practical measure of FPGA floating point acceleration for High Performance Computing", Proceeding of 2013 IEEE 24th International Conference on Application-Specific Systems, Architectures and Processors (ASAP), pp.160-167, 5-7 June 2013

[2] Peter-Michael seidel and Guy Even, "On the Design of Floating-Point Adders", in proceedings of the 15th IEEE symposium of Computer Arithmetic.

[3] Mohamed Al-Ashrafy, Ashraf Salem and Wagdi Anis, "An Efficient Implementation of Floating Point Multiplier", proceeding of 2011 IEEE Saudi International Electronics , Communications and Photonics Conference (SIECPC), pp.1-5, April 2011.

[4] Behrooz Parhami, 'Computer Arithmetic: Algorithms and Hardware Designs', 1st ed. Oxford: Oxford University Press, 2000

[5] IEEE Standards Board, IEEE-754, IEEE Standard for Binary Floating-Point Arithmetic, New York: IEEE, 1985.

[6] Lamiaa S.A.Hamid, Khaled A.Sheata, Hassan El-Ghitani, Mohamed Elsaid, "Design of Generic Floating Point Multiplier and Adder/Subtractor Units", in proceedings of the 12th IEEE International Conference on Computer Modelling and Simulation, pp.615-618, March 2010.