

A VHDL Based Floating Point ALU for DSP Processing

¹K. Niranjan Reddy, ²D. Sri Harsha, ³S. Sri Vani, ⁴S. Tarun Prasad

^{1,2,3,4}Malla Reddy Engineering College for Women, AP, India

Abstract

Many DSP techniques depend upon exclusive floating point arithmetic for the dynamic range of operations which require millions of calculations per second. Digital signal processing of 1d and 2d signals need different mathematical calculations which are mainly of complex data. Operations like FFT, Z Transform constantly requires various Exponential Arithmetic. Such computationally challenging techniques need computational acceleration using Custom Computing Machines (CCMs) being designed exclusively for the application. Floating point operations require excessive memory and time for conventional implementations. Therefore, custom formats for individual applications like DSP are more feasible on CCMs, and can be easily designed and implemented on a fraction of a single FPGA. VHDL, facilitates the design and development of custom operations without effecting the performance. Most of the processor architectures are built to handle real and non exponential data. Even though there are several architectures proposed over the years to explicitly handle exponential arithmetic, none are targeted for a specific DSP operation based on exponential arithmetic. In this paper we propose design of a unique exponential arithmetic unit for exponential addition.

Keywords

Exponential Arithmetic, Custom Computing Machine, Digital Signal Processing, VHDL

1. Introduction

Exponentiation is a number representation, represented as b^n , which includes two numbers, the base b and the exponent n . When n is a positive integer, exponentiation corresponds to recursive multiplication. A product of n factors, each of which is equal to b can be represented in equation 1

$$b^n = \underbrace{b \times \dots \times b}_n \quad (1)$$

The multiplication itself can be represented as repeated addition like (2):

$$b \times n = \underbrace{b + \dots + b}_n \quad (2)$$

Therefore it is obvious that if a suitable ALU is designed to handle precision exponential number addition, than it can in the same way be used for exponential multiplication. Addition and subtraction follows the same structure. And division operation can also be developed with addition and multiplication. Therefore a suitable adder design is essentially the first step in designing a DSP unit that can handle exponential operations. In order to understand the design fundamental of exponential operations, we will further

try to understand the exponential notations and overview. The exponentiation b^n is read as: b raised to the power of n , or simply b raised by the exponent of n ,

Therefore exponential addition of two numbers

$$b3^{n3} = b1^{n1} + b2^{n2} \quad (3)$$

Here the objective is to determine $b3$ and $n3$, given $\{b1, n1\}$ and $\{b2, n2\}$. The representation clearly shows that any exponential numbers can be represented as a pair of number where the second number is the exponent. Further all the numbers can be represented with a fixed based exponent. In such a representation (3) becomes:

$$b3.e^{n3} = b1.e^{n1} + b2.e^{n2} \quad (4)$$

It is clear from (4) that if exponent parts can be made same at the right hand side of equation 4, that exponent of left hand side will also have the same value and result of addition can be simply obtained by addition of non exponent parts along with supplying the common exponent. The important mathematical constant e , sometimes called Euler's number, is approximately equal to 2.718 and is the base of the natural logarithm. Although exponentiation of e could, in principle, be treated the same as exponentiation of any other real number, such exponentials turn out to have particularly elegant and useful properties. Among other things, these properties allow exponentials of e to be generalized in a natural way to other types of exponents, such as complex numbers or even matrices, while coinciding with the familiar meaning of exponentiation with rational exponents. As a consequence, the notation ex usually denotes a generalized exponentiation definition called the exponential function, $\exp(x)$, which can be defined in many equivalent ways, for example by:

$$\exp(x) = \lim_{n \rightarrow \infty} \left(1 + \frac{x}{n} \right)^n \quad (5)$$

Among other properties, $\exp$ satisfies the exponential identity:

$$\exp(x + y) = \exp(x) \cdot \exp(y) \quad (6)$$

The exponential function is defined for all integer, fractional, real, and complex values of x . Since is equal to e and satisfies the exponential identity, it immediately follows that $\exp(x)$ coincides with the repeated-multiplication definition of ex for integer x , and it also follows that rational powers denote (positive) roots as usual, so $\exp(x)$ coincides with the ex definitions in the previous section for all real x by continuity. It is clear from equation (4)-(6) that exponential arithmetic is important aspect even for floating point representation where exponent part is real negative. In order to generalize the operations, IEEE proposed a format for exponential/ floating point arithmetic numbers which is drafted as IEEE 754-1985 [5]. Common representation of the numbers is given below.

Table 1: Common Representation of the Numbers as per IEEE 754 [5]

Name	Base	Digits	E min	E max	Decimal Digits	Decimal E max
decimal32	10	7	-95	+96	7	96
decimal64	10	16	-383	+384	16	384
decimal128	10	34	-6143	+6144	34	6144

Decimal digits is $\text{digits} \times \log_{10} \text{base}$, this gives an approximate precision in decimal.

II. Related Work

Multiple executions can be done using pipelining. Performance of ALU can be increased by pipelining which is realized using VHDL [1]. An approach for the design of FPM is developed and presented synthesis on FPGA circuit of the same with 4-bit for fraction and exponent using VHDL successfully [2]. Feasibility of the use of floating point arithmetic in back propagation algorithms on single FPGA platforms was examined. Quantization errors in calculations occur due to 16-bit fixed point limited precision. [3]. LNS offers faster multiplication with the expense of addition and CLNS is better than using LNS [4]. IEEE754 can be implemented and can be realized fully in software or hardware or using both. Conversions shall be monotonic, which means increase in the value of binary floating point number will not get decreased after conversion to decimal string and vice versa [5]. Utilization of less chip area means the number of slices used in the reconfigurable hardware is less [6]. Designs for decimal floating point addition and subtraction for single precision in IEEE - 754 were presented [7]. Different techniques which results in direct decimal sums have been discussed [8]. Circuit based implementation and ALU design, algorithms and speeding up of algorithms using parallel processing is discussed [9].

III. Proposed System

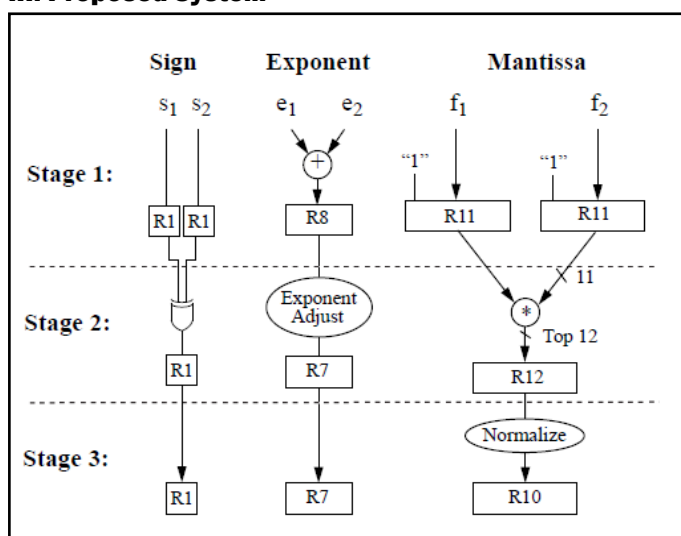


Fig. 1: Simple Architecture of the Proposed System

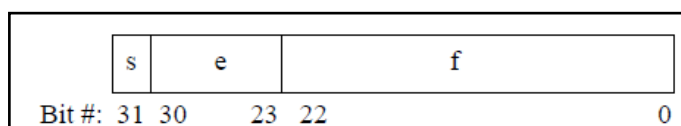


Fig. 2: Representation of a Single Precision Floating Point Number

In this work we will use decimal32 single precision floating point numbers.

The number value is interpreted as

$$V = -1^{s2} (e-127) \times 1.f \quad (7)$$

Let us now consider the representation of a number -3.625.

$$v = -11 \ 2^{(128-127)} 1.1101$$

where:

$s = 1$, $e = 128$ (dec) 80 (hex), and $f = 680000$ (hex). Therefore -3.625 is stored as: C0680000 (hex).

A. Algorithm

If sign bits of both the numbers are same.

- Let us assume that two numbers x_a, x_b be represented as $x_a = d_a + E_a$, $x_b = d_b + E_b$.
- If $x_a < x_b$, swap both the numbers such that $x_a > x_b$.
- Compare the exponents only. If $E_a \neq E_b$. Compute Exponent difference.
- Shift the mantissa of the number whose exponent is smaller by exp-diff along with hidden bit.
- Increment Exponent at each shift such that both the exponents are same.
- Perform mantissa addition including hidden bit.
- If sign bits of both the numbers are different
- Take 2's complement of the Mantissa of the number Whose sign bit is +ve such that both numbers have same -ve sign.
- Repeat the process from 3 to 5 and perform subtraction instead of addition in step 6 If sign bits are same and $E_a = E_b$ then
- Add hidden bit to mantissa and compute the addition of floating point mantissa along hidden bit. If carry is generated, right shift mantissa along with carry for 1 bit.
- If no carry, check hidden bit. If hidden bit is 1, then result is normalized
- If hidden bit is 0, then shift mantissa left till hidden bit is 1 till hidden bit becomes 0. count number of shifts.
- Adjust the exponent by decrementing it to the count number of times

The main contribution of the proposed work is developing a 20 bit number system that represents the floating point numbers in Exponent-Mantissa format and introduction of a hidden bit as the MSB of Mantissa. This hidden bit significantly reduces the number of explicit instructions to be performed by CPU for any floating point arithmetic.

Table 2: Example of Addition of Two Numbers

S	EXP	H	Mantissa
0	10001010	1	0000000000
0	10001010	1	0000100000
0	10001010	0	0000100000
0	10001010	1	0000010000

IV. Results

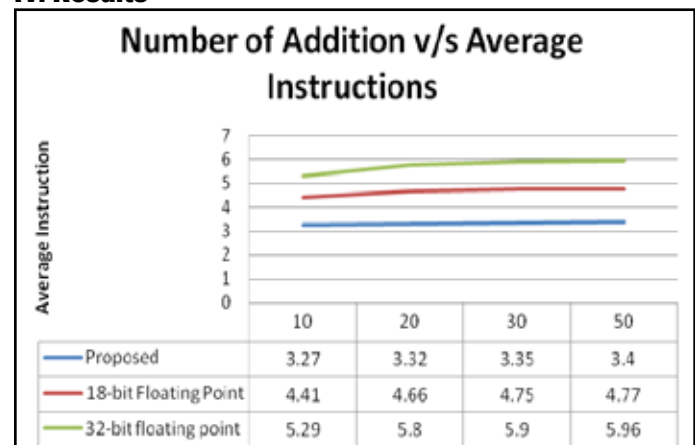


Fig. 3: Comparison of Proposed Technique with Other Floating Point Values for Different Number Additions

Fig. 4 shows the result of step by step calculation of each stage as described in the methodology section. It represents dividing the number into mantissa and floating part, processing them separately and finally integrating the result as its chip output.

Fig. 5 shows the simulation result of addition of two floating point numbers by following fixed point arithmetic. It clearly justifies that the calculation period for the proposed system is very nominal.

Performance is tested with a real time FPGA device with 100Mhz clock. Explicit instruction is defined as total number of instructions including comparison, shifting and addition. Numbers are selected randomly with random sign bits. It is clear that the average number of instructions to be performed by the CPU for the same purpose is substantially low for the case of proposed system.

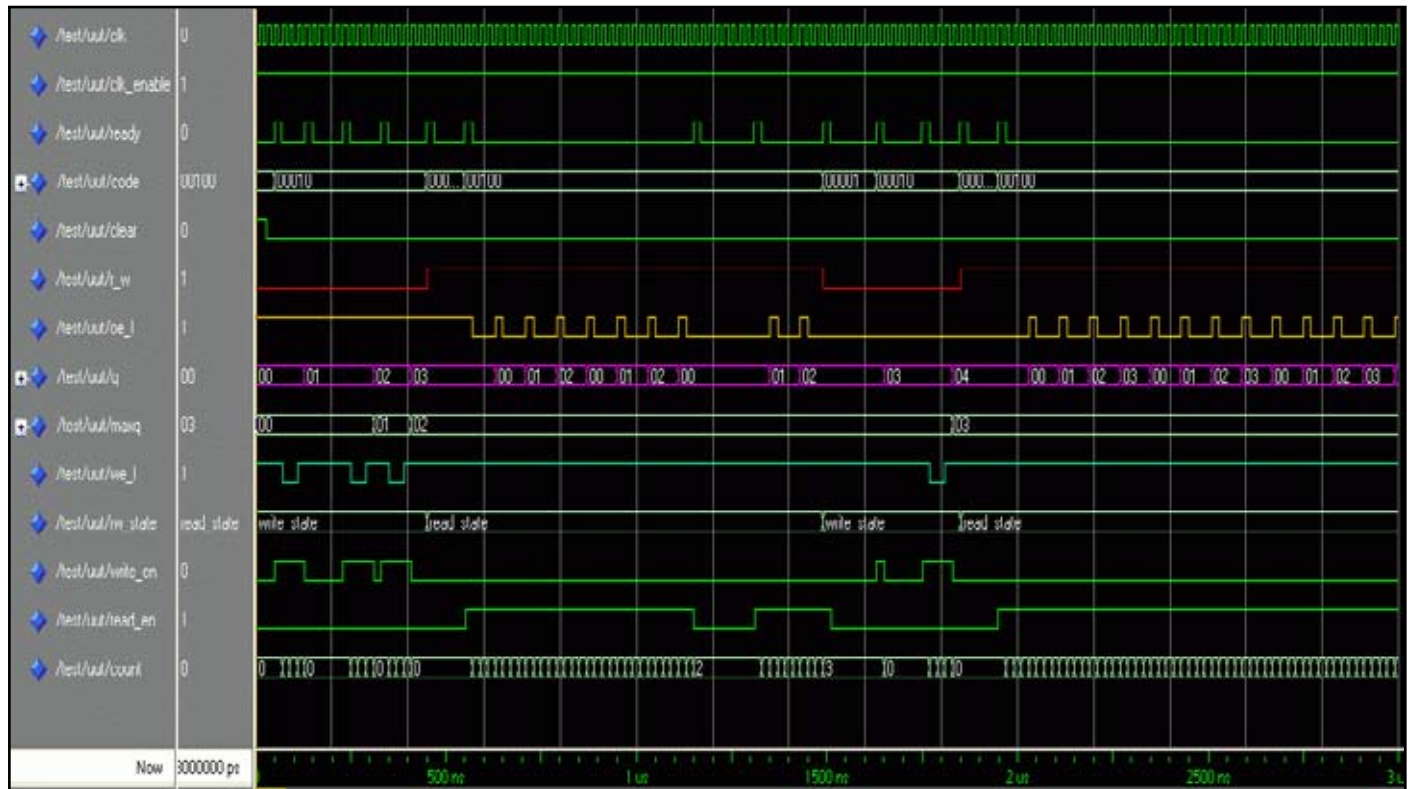


Fig. 4: Step by Step Calculation

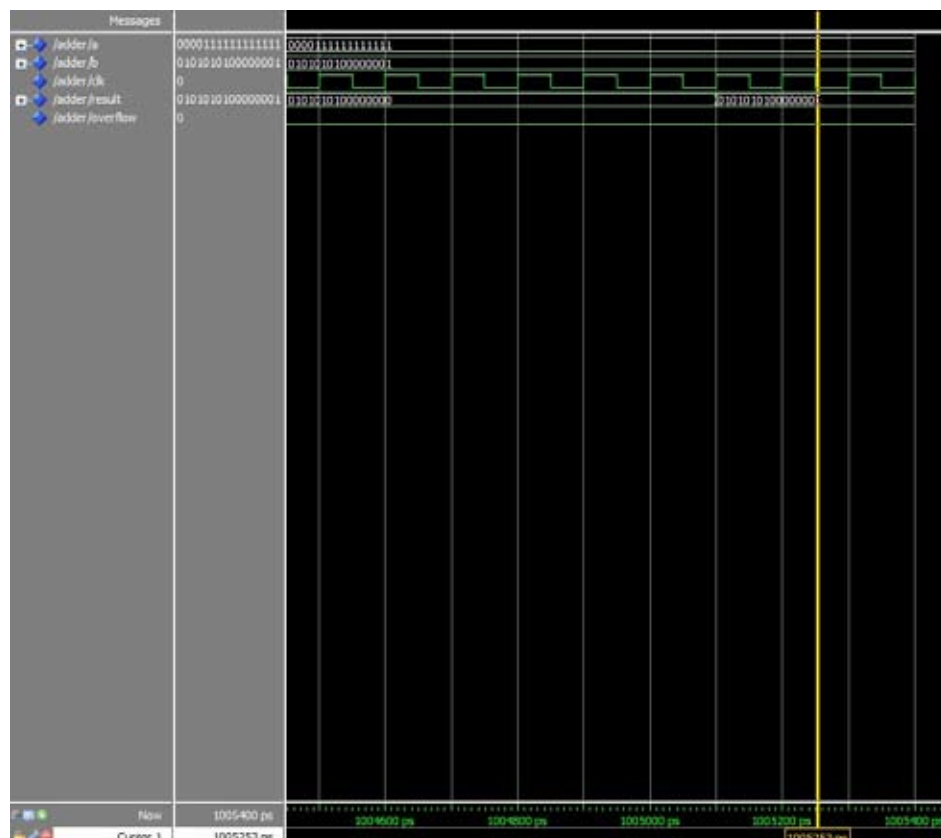


Fig. 5: Result of Addition of Two Floating Point Numbers

V. Conclusion

IEEE 754 provides a very good documentation about performing fixed point floating number arithmetic. However the number pattern is inefficient while taking into consideration of DSP applications. This is due to high memory and time cycle required by such operations. In this work we have developed a unique 20 bits number pattern for performing the same along with introducing a hidden bit in Mantissa for quick normalization of the result. The performance of the proposed system is way better than both standard 18-bit DSP suitable number pattern and 32 bit pattern proposed by IEEE 754. The number of instructions is reduced by checking the hidden bit similarity in both the numbers which reduces the complexity of the process performed by the processor. Further the most significant comparisons and shifting are 1 bit which reduces the over burden of excessive memory.

References

- [1] L.F. Rahman, Md. Mamun, M.S. Amin, "VHDL Environment for Pipeline Floating Point Arithmetic Logic Unit Design and Simulation", Journal of Applied Sciences Research, 8(1), pp. 611-619, 2012.
- [2] F. Benrekia, M. Attari, "A Floating Point Multiplier based FPGA Synthesis for Neural Networks Enhancement", International Journal of Engineering Science and Technology, Vol. 2(5), pp. 1433-1440, 2010.
- [3] K. R. Nichols, M. A. Moussa, S. M. Areibi, "Feasibility of floating-point arithmetic in FPGA based artificial neural networks", In CAINE, pp. 8-13, 2002.
- [4] Arnold, M.G., S. Collange, "A Real/Complex Logarithmic Number System ALU", IEEE Transactions on Computers, 60(2), pp. 202-213, feb. 2011.
- [5] "IEEE Standard for Binary Floating-point Arithmetic", ANSUIEEE Std 754-1985, New York, The Institute of Electrical and Electronics Engineers, Inc., August 12, 1985.
- [6] Karan Gumber, Sharmelee Thangjam, "Performance Analysis of Floating Point Adder using VHDL on Reconfigurable Hardware", International Journal of Computer Applications, Vol. 46, No. 9, May 2012.
- [7] Subhash Kumar Shrama et. al., "Implementation of IEEE-754 Addition and Subtraction for Floating Point Arithmetic Logic Unit", International Transactions in Mathematical Sciences and Computer, Vol. 3, No. 1, 2010, pp. 131-140.
- [8] Schmookler M. S., Weinberger A. W., "High Speed Decimal Addition", IEEE Transaction on Computers, C(20), 862-867, August 1971.
- [9] Behrooz Parhami, "Computer Arithmetic: Algorithms and Hardware Designs", 1st ed. Oxford: Oxford University Press, 2000.



K. Niranjan Reddy received his B.E in Electronics & Communication Engineering (E.C.E) from Karnataka University, Dharwad in 1999 and M.Tech in VLSI DESIGN from Bharath Institute of Science & Technology, Chennai in 2005. At present he is pursuing his Ph.D from Goldan State University, Mississippi. At present he is working as an Associate Professor & HOD of ECE at MALLA REDDY ENGINEERING COLLEGE FOR WOMEN (MRECW). He has more than 8 years of experience in Teaching at various engineering colleges & having Industrial Experience in Linux System Administration in India & as well as Abroad. He contributed several research articles in International Journals & Conferences.



D. SRI HARSHA received his B.Tech in Electronics & Communication Engineering (E.C.E) from Indur Institute of Engineering & Technology, JNTUH in 2005 and M.Tech in VLSI SYSTEM DESIGN from Sree Dattha Institute of Engineering & Science, JNTUH 2011. He has more than 6 years of Teaching Experience at various engineering colleges. At present he is working as an Associate Professor, Department of ECE at MALLA REDDY ENGINEERING COLLEGE FOR WOMEN (MRECW). He contributed several research articles in International Journals & Conferences. His areas of interest are VLSI & Embedded Systems.



S. SRI VANI received her B.Tech in Electronics & Communication Engineering from JNTU, Anantapur in 2005 and M.Tech in EMBEDDED SYSTEMS from St. Mary Institute of Technology & Science, 2009. At present she is working as an Associate Professor, Department of ECE at MALLA REDDY ENGINEERING COLLEGE FOR WOMEN (MRECW). She has more than 6 years of Teaching Experience at various engineering colleges & having 6 years of Industry Experience in Yamuna Digital Electronics Pvt. Ltd. Her areas of interest are Embedded Systems & Communication.



S. Tarun Prasad received his B.Tech in Electronics & Communication Engineering from Viswanadha Institute of Technology & Management in 2010 and M.Tech in VLSI DESIGN from VIT University in 2012. At present he is working as an Assistant Professor, Department of ECE at MALLA REDDY ENGINEERING COLLEGE FOR WOMEN (MRECW) from July 2012. His areas of interest are Digital Design & ASIC.