Advanced Research in Electrical and Electronic Engineering
Print ISSN: 2349-5804; Online ISSN: 2349-5812 Volume 2, Number 9 April-June (2015) pp. 76-79
© Krishi Sanskriti Publications
http://www.krishisanskriti.org/areee.html

Basic Trigonometric Function Core in Duble FPU

Raman Kumar[1] and Sumit Kumar[2]

[1,2]M.Tech (VLSI Design) CDAC Noida (GGSIPU)
E-mail: [1]raman16491@gmail.com, [2]sksaxena05112@gmail.com

Abstract—*A floating-point unit(FPU) is a math Co-processor, which is a part of a computer system specially designed to carry out operations on floating point numbers . Double-precision floating-point is a commonly used format on PC's due to its wider range over single-precision floating point.. The proposed work is to build an efficient basic trigonometric function core using double precision floating point unit that performs basic trigonometric functions with reduced complexity of the logic used and reduce the memory requirement . The functions performed are handling of Floating Point data, perform any one of the following trigonometric operations like angle of sine , cosine , tan, cot, sec and cosec. All the above modules have been clocked and evaluated under Spartan 3E Synthesis environment. All the functions are built by possible efficient algorithms with several changes incorporated at our end as far as the scope permitted. The coding is done in verilog.*

1. INTRODUCTION

Many people consider floating-point arithmetic unit an esoteric subject. This is rather surprising because floating-point is ubiquitous in computer systems. Almost every language has a floating-point data type; computers from PC's to supercomputers have floating-point accelerators; most compilers will be called upon to compile floating-point algorithms from time to time; and virtually every operating system must respond to floating-point exceptions such as overflow. There are some aspects of floating point that have a direct impact on designers of computer systems.

Every computer has a floating point processor or a dedicated accelerator that fulfills the requirements of precision using detailed floating point unit. The main applications of floating points today are in the field of medical imaging, motion capture, geography to measure distances between landmarks, motion capture and audio applications, including broadcast, conferencing, musical instruments and professional audio & in satellite navigation systems, and hence in Global positioning system(GPS). The performances of computers that handle such applications are measured in terms of the number of floating point operations they perform per second. Double Precision floating point format is a format which occupies 8 bytes (64 bits) in memory and represents a wide dynamic range of values by using a float point.

IEEE 754 specifies four formats for representing floating-point values:

1. Single-precision (32-bit)
2. Double-precision (64-bit)
3. Single-extended precision (≥ 43-bit, not commonly used)
4. Double-extended precision (≥ 79-bit, usually implemented with 80 bits

But single and double precision is preferred. For our project we decided to verify trigonometric functions in double precision Floating point unit for higher precision and with compatibility to machine system where three basic components are sign , exponent and mantissa field. Sign field is 1 bit long and in sign field '0' denotes a positive number and a '1' denotes a negative number. Exponent field is 11 bit long, occupying bits 62-52. The value in this 11 bit field is offset by 1023 so actual exponent used to calculate the value of the number is 2^(e^{-1023}) and mantissa bit is 52 bit long for Double precision FPU. The main reason of employing double precision over single precision is the high significand precision of about 16 decimal digits and range of approx.10^{-308} to 10^{+308}.

As single precision FPU consist of 1 bit long sign field, 8 bit long exponent field and 23 bit long mantissa field giving 32 bit output. Sign bit is used to denote whether output number will be positive or negative.

2. ARCHITECTURE

The overall Architecture is shown in Fig. 1. It consist of ACTV Unit, and basic trigonometric function unit,dividor unit and N-Bit input and 64-bit output .

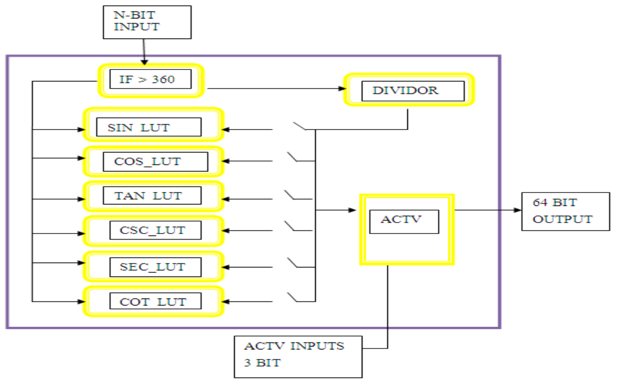

Fig. 1

Some Registers used are :

ACTV which decides which function to activate

QUAD which specifies value in quadrants.

DATA1 which gives final result.

3. HOW IT WORKS

In this version all the trigonometric modules are created as look up table (LUT).To all the input values there is an equivalent double precision floating point unit value, to the input an un-signed value is given.It also supports all quadrants.

Case1: If the given input value is less than 90 degrees from the top module "00" value is passed on to the "quad" register, this register helps us to know in which quadrant the value lies in.

Case2: If the value is between 91 and 180, then the values should be subtracted from the decimal value 180 so that value will be mirrored. For example the value of 89 and 91 would be the same. When the value is in between 90 and 180, 180 is subtracted from input value as the value is less than 180, the resultant will be positive. In the same way all the values can be mirrored using the existing 90 values, "01" value is passed on to "quad" reg.

Case3: If the value is between 181 and 360, then the value should be subtracted from 180. Again in this two things should be considered, after subtracting the input from 180 if the value is less than 90 then the case 1 is repeated "10" value is passed on to quad, else if the resultant is greater than 90 then the case 2 is repeated, "11" value is passed on to the "quad" register .

Case4: If the value is greater than 360 then the value is passed on to divider module. Modulo division is performed in this block until the remainder value is less than 360, remainder value is taken and all the above cases will be repeated.

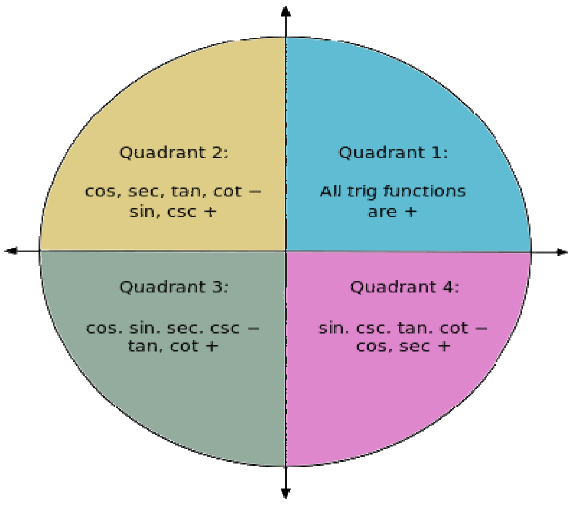

Fig. 2

4. SYNTHESIS RESULT

4.1 Top Module (CORE)

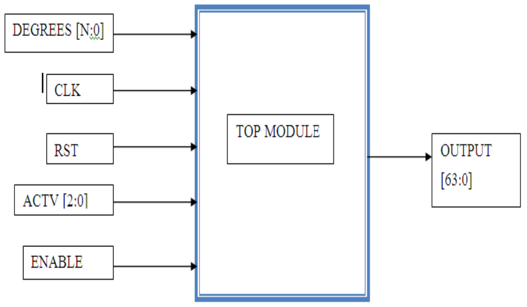

Fig. 3

The input signals to the top level module are the following:

- CLK (global)
- RST (global)
- ENABLE (set high to start operation)
- ACTV (activation, 3 bits, 000 = sin_enable, 001 = cos_enable, 010 = tan_enable, 011 = csc_enable, 100 = sec_enable, 101 = cot_enable)
- DEGREES (32 bits)
- DATA1 (output from operation, in 64 bits)

4.2 Sine Module

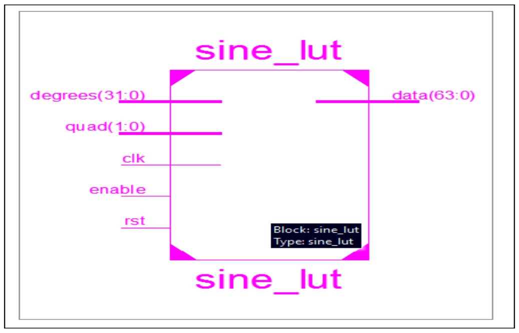

Fig. 4

4.3 Cosine Module

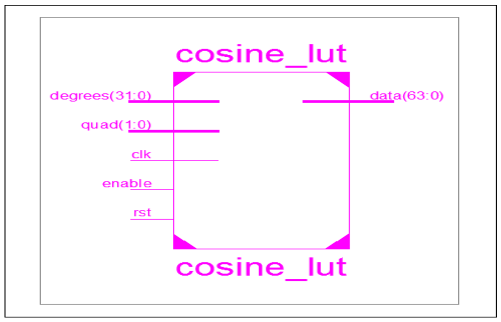

Fig. 5

4.4 Tangent Module

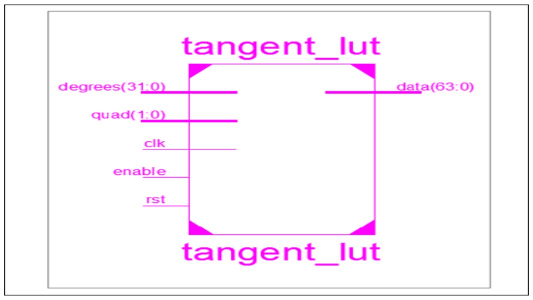

Fig. 6

4.5 Cotangent Module

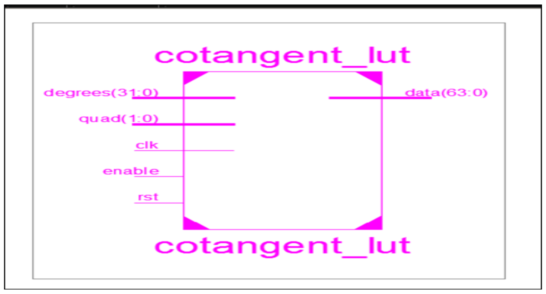

Fig. 7

4.6 Secant Module

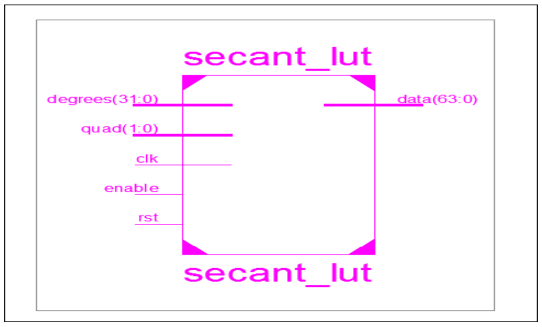

Fig. 8

4.7 Cosecant Module

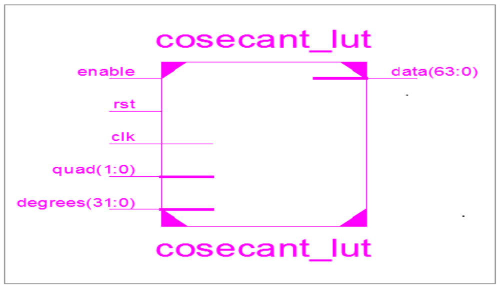

Fig. 9

5. SIMULATION RESULT

Case 1 : Positive output (Fig. 10)

- WHEN INPUT DEGREE IS 112, OUTPUT:- C003CCFA561175D3
- 1100000000000011110011001111101001010110000100 0101110101110100111 (binary form).
- In double fpu form = -1.M *2^E-1023 = -2.4750868534162958
- So Activating tangent function tan112 = -2.4750868534162958 (2nd quadrant) hence ouput is verified.

Case 2 : Negative output (Fig. 11)

- When INPUT is 40 degree, OUTPUT:- 3FF3116C3711527E
- 0011111111110011000100010110110000110111000100 0101010010011111110(binary form).
- In double fpu form =
- +1.M*2^E-1023= +1.1917535925942099
- By activating COTANGENT funcn cot 40=1.1917535925942099(1st quad.) so output is correct.

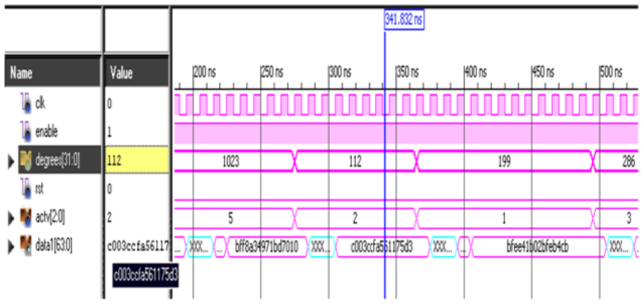

Fig. -10

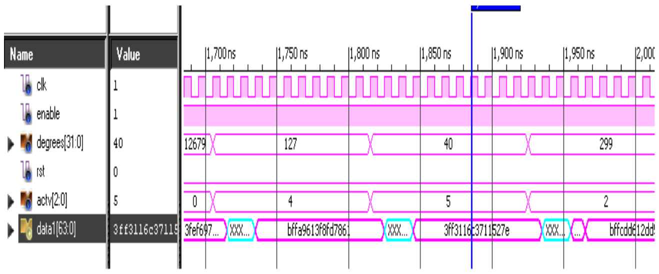

Fig. 11

6. CONCLUSION AND FUTURE WORK

We have to implement a trigonometric function core which is able to produce double-precision results with very high throughput. The logic can further be optimized, also to the input it needs to be changed from signed to unsigned or 64 bit fpu format so that trigonometric functions can be calculated in both degrees as well as radians. However, the current implementation can be further optimized through the design of a single cycle double-precision floating point multiplier. Inverse trigonometric functions, hyperbolic trigonometric functions will be implemented.

REFERENCES

[1] Altera Corp. Stratix II Device Overview. Data Sheet DSSTXGX-2.2, Altera Corp., San Jose, California, Dec 2004.

[2] Altera Corp. Floating-Point Multiplier. Functional Specifications A-FS-04-01, Altera Corp., San Jose, California, Jan1996.