International Journal of Engineering and Advanced Technology (IJEAT)
ISSN: 2249-8958 (Online), Volume-8 Issue-6S3, September 2019

# A Modified Double-Precision Floating-Point Multiplier to Achieve Reduced Hardware Utilization

**Y. Srinivasa Rao, A. Bhanu Chandar**

***Abstract—The DP(Double-Precision) floating-point multiplier require a bulky 52x52 mantissa multiplications. The enactment of the DP floating number multiplication predominantlybe influenced by on the area and speed. This paper presents a improveduniquemethod to diminution this huge multiplicationpracticeof mantissa. The UTmethodpermitsusinga reducedquantityof multiplication hardware equaled to the conventional method. In old-fashionedschemeaccumulation of the partial products are independentlycompleted and it may perhapsyieldextraperiod of time in contrast to thesuggested method. In the suggestedprocess the partial products are parallel added with the multiplication actions and it canreduce the time delay. The method wasinstigated using VerilogHDL with Xilinx 14.2 ISE tools on Virtex-5 FPGA.***

***Keywords-DP (Double-Precision), Floating point, Multiplication, Vedic, Urdhva Tiryagbhyam (UT), IEEE-754, Virtex-5 FPGA.***

## I. INTRODUCTION

The Fig.1 symbolizes single-precision(32-bit) floating point digitpreparation. In single-precision floating point number has 32-bits ofsize. The whole 32-bits are distributed into three arenas, 31st bit of the digitsign (1-bit)arena, 30th bit to 23rd bit of the digit is exponent (8-bits) arenaand the 22nd bit to 0thbit of the digit is mantissa (23-bits)arena.

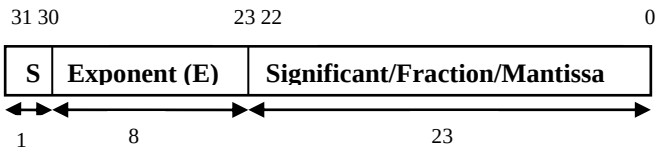



Fig. 1: Single Precision(32-bit) floating point digitpresentationdepiction.

The Fig. 2 symbolizes the DP (64-bit) floating point numberpresentation. The DP floating point number has 64-bits ofspan. The entire 64-bits are distributed into three arenas, 63rd bit of the numeral is a sign (1-bit), 62nd to a 52nd bit of the numeral is an exponent (11-bits) and 51st to the 0th bit of the numeral is mantissa (52-bits).

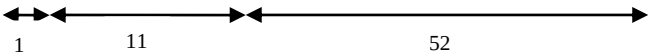




Fig. 2: Double Precision(64-bit) floating point number format representation.

The IEEE-754 standard of hardware enactmentsintended for floating point arithmetic tasks areidenticalin all CPUs. The multiplication set-up is fundamental and identical among all of them. The area of the application is the foremostrestraint which had better to be smallestand affords a superiorspeediness. Utmost of the high-performance computations can be instigated with the FPGAs (Field Programmable Gate Arrays)[7]. The FPGAs ropes the extra ordinary rapidity, the huge amount of logic resources i.e. great slice count and vast LUTsand in adequate accessible on – board intellectual property (IP)[4] cores forms a hefty set of applications.The suggested work is predominantlyaiming on the DP floating point multiplication carrying outon FPGA boards. The mantissa multiplication is utmostvitaltaskformultiplication of the DP floating point numbers. This is the focallogjam of the performance. TheDP floating point number's mantissa is in 53-bits span, and this ecessitatesa hardware enactment of 53x53 multipliers, /hich is precise cost effective and expensive. In this exertion,one algorithm haswished-for for the DP floating numbers multiplication and it can be legitimate to using a compactsum of multiplication accomplisha high speed atcomparatively low hardware resources cost. The prosed scheme of multiplication consequences theParalleling with Karatsuba algorithm[1]results. The enactment is mainly concerned only normalized numbers. The enactmentcan be through by using Xilinx ISE synthesis tool, ISIM simulator andVirtex-5[4](xc5vlx110t-1ff1136) speed grade-1 FPGAplatform. The paper contains the Introduction in section 1, section 2 has Proposed Approach of Design, section 3 has implemented the design, section 4 has design results and final conclusion of the paper is presenting in section 5.

## II. PROPOSED APPROACH OF DESIGN

The Karatsuba MultiplicationPractice is expending to deceitful the wished-fortechnique of multiplication. This multiplication technique isdisruption upthe two mantissas into three parts in Fig. 3.

**Manuscript published on 30 September 2019.**
* Correspondence Author (s)
**Y. Srinivasa Rao**, Asst. Professor, Dept. of ECE, Malla Reddy Engineering College for Women (Autonomous), Secunderabad-TS
srinivass.y25@gmail.com
**A. Bhanu Chandar,** Asst. Professor, Dept. of ECE, Malla Reddy Engineering College for Women (Autonomous), Secunderabad-TS
abcreddy.gp@gmail.com

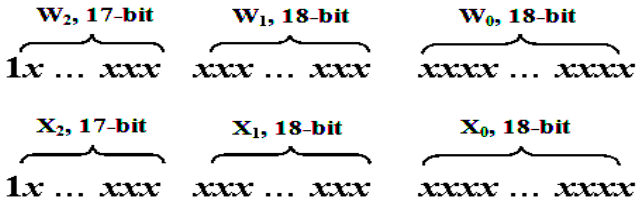



Fig. 3 53-bit mantissa is split into 17-bit and 18-bit fractions

The 53-bit mantissa[1] is fragmented into 17-bit and 18-bit signed portions. The terms $w_0$ and $w_1$ represent the 18-bit fraction and $w_2$ represents the 17-bit fraction of the one of the input operand. In the same way, $x_0$ and $x_1$represents the 18-bit fractions and $x_2$ represent a 17-bit fraction.

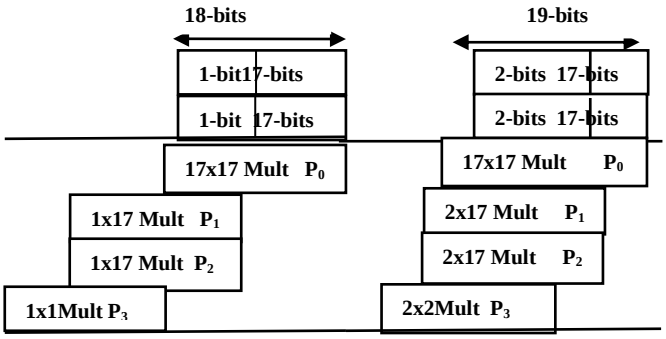



Fig.418-bit and 19-bit Multipliers

The designing of 53x53-bit mantissa multiplication is realized by using the 18-bit multiplier and 19-bit multipliers. The Fig. 4 represents the 18-bit multiplier and 19-bit multipliers. The multiplier of 18-bit has 1-bit sign bit and 17-bits fractional bits, the first step is to multiply two 17-bits of the input operands and generate a partial product $p_0$ then generate a $p_1$,$p_2$, and $p_3$. In the same way, 19-bit multiplier also designs.

The deceitful of Vedic multiplier is reinforced Vedic multiplication [5] ideologies (sutras). These ideologies are largely used for the multiplication of two hefty decimal numbers. This equivalent sutra can be cast-off to accomplish multiplication[5] on two hefty binary numbers and is presentedin Fig. 5.. This wished-for method is well-matched to project digital hardware system. The UT technique[4] can take (2n-1) steps for designing the n-bit multiplier. In Fig. 5 4-bit multiplier can take 7-steps for designing of the multiplier.

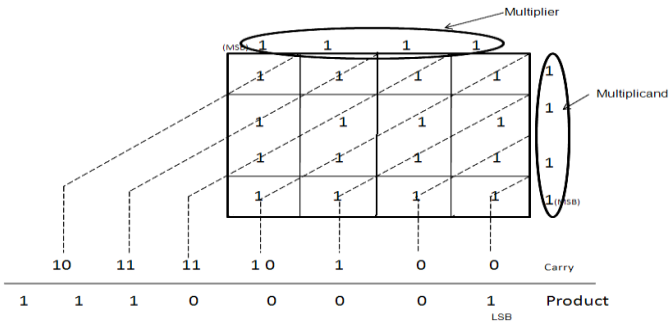



Fig.5 Steps involved for 4-bit binary numbers multiplication using UTTechnique.

The 4-bit binary multiplier using UT technique is shown in Fig. 5. The 4-bit multiplier has 4-bit multiplier as one of the operands and 4-bit multiplicand is another operand. The carry resents the previous state generated a carry and it can be added to the current to get the final product.

## III. IMPLEMENTATION

Designing of the multiplier for floating-point numbers carries the operation separately by the sign bit, exponent bits and mantissa bits.

### *A. Sign and Exponent operations.*

The sign and exponent operations are performed in a straightforward manner. The operand 1 and operand 2 of sign-bits are carried outthe operation of logical XOR.

$$Sign\_out=Sign\_in1 \ominus Sign\_in2$$

The resultant exponent field of the number is the addition of operand 1's exponent and operand 2's exponent and subtracting the BIAS i.e.

$$Exp\_out=Exp\_in1+Exp\_in2-1023$$

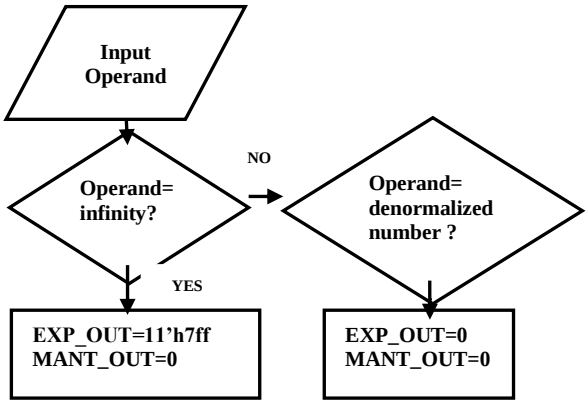
For DP floating number, the BIAS is equals to 1023 i.e. $(2^{11-1}-1)$.

The BIAS of any floating point number is determined by $(2^{\text{exponent bits}-1}-1)$.

### *B. Exceptional Case Handling*

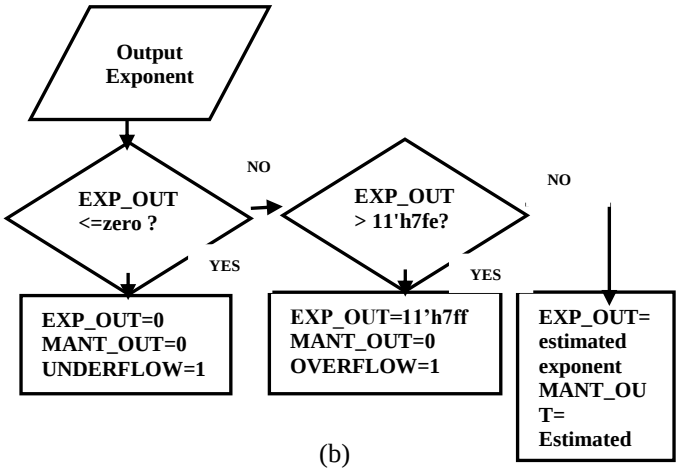
The IEEE standard as defined by the many exceptional cases like NaN (Not a Number), INFINITE, ZERO, UNDERFLOW, OVERFLOW. These are appearing for all the floating point arithmetic operations. The main operation has been also combined with the detection of all the exceptional cases, and determining the final output as per standard. All the exceptional cases executions are in parallel with the IEEE-754 standard.

Fig. 6 represents the flowchart of the handling of exceptional cases.




(a)

If one of the operands or both of the operands is infinity; thengeneratean INFINITY output (with calculating sign-bit). If the denormalized number is one of the input operands, then the output is zero (w.r.to sign-bit). If the zero or below zero as output exponent, the displayed output is UNDERFLOW, and the 11'h7fe (2046 in decimal)is the output exponent, OVERFLOW as displayed output.




(b)

Fig. 6 Flowchart for the handling of exceptional cases (a) infinite case (b) OVERFLOW and UNDERFLOW cases

### *C. Normalization and Rounding*

Forgetting of the final result after mantissa multiplication the normalization is mandatory. After the mantissa multiplication, the resultantnumber has one extra bit in the most significant bit before the decimal point. Similarly, in sometimes the same situation will arise after rounding. So, whenever the additional carry bit is produced after multiplication or rounding, the resultant product must be right-shifted for necessary shifts and the corresponding alterationsshould be made in exponent also to get the normalized result.

Rounding is necessary to get back the 106-bit mantissa multiplication result to 53-bit result only. In this paper only implementedRound to nearest rounding mode precise by the IEEE-754 standard and remainingmodes can beimplementing as per the application requirement.

## IV. RESULTS

The FPGA implementation of DP floating point multiplier using UT techniqueis divided into the 18-bit multiplier, 19-bit multiplier, and 53-bit mantissa multiplier. The 53-bit multiplier is designed by combining the 18-bit and 19-bit multipliers. The 18-bit multiplier, 19-bit multiplier, and 53-bit mantissa multipliers are in followed Figures 7-16.

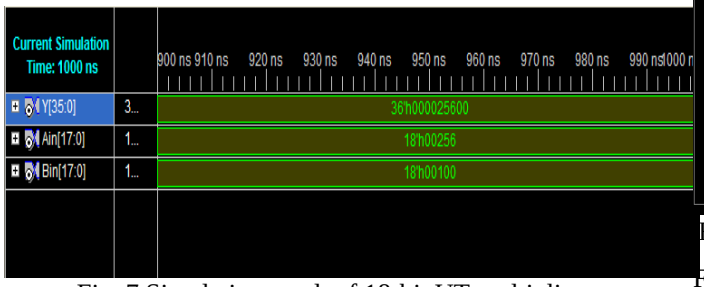



Fig. 7 Simulation result of 18-bit UT multiplier

The simulation result of 18-bit multiplier using UT multiplication technique is in Fig. 7. Here $A_{in}$and $B_{in}$are the two input signals. In this $A_{in}$ is one of the operands which is 18-bits in length and $B_{in}$ is another operand which is also an 18-bits in length. These two operands are performing a multiplication operation and produce a 36-bit of a result.

INPUTS: $A_{in}$[17:0]=18'h00256; $B_{in}$[17:0]=18'h00100;
OUTPUT: Y[35:0]=36'h000025600;

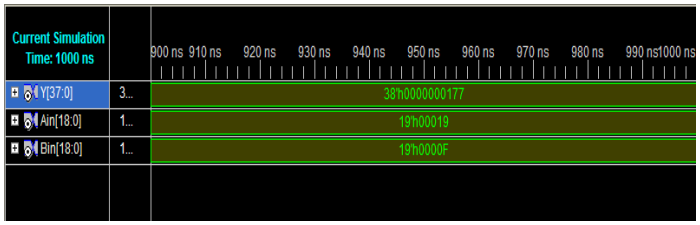



Fig. 8 Simulation result of 19-bit UT multiplier

The simulation results of 19-bit multiplier using UT multiplication technique is in Fig. 8. $A_{in}$ and $B_{in}$ are the two input operand of the 19-bit multiplier and Y are the output produced by the multiplier.

INPUTS: $A_{in}$[18:0]=19'h00019; $B_{in}$[18:0]=19'h0000F;
OUTPUT: Y[37:0]=38'h0000000177;

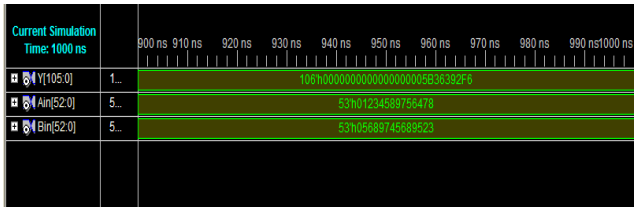



Fig. 9 Simulation result of 53-bits mantissa multiplier

Simulation result of the 53x53-bit multiplier is in Fig. 9. In this $A_{in}$ and $B_{in}$ are the inputs of the multiplier; each one has 53-bits in length and output Y has 106-bits in length.

INPUTS:$A_{in}$[52:0]=53'h01234589756478;
$B_{in}$[52:0]=53'h05689745689523;
OUTPUT:
Y[105:0]=106'h00000000000000000005B36392F6;
The mantissa bits of the DP floating point multiplier have 53-bits. The 53-bit x 53-bit multiplier design is very difficult, the 53-bits are divided into 18-bits and 19-bits and design the multipliers. These multipliers are added to get the 53-bit mantissa multiplier. The Fig.12shows the RTL Schematic of 53x53-bits multiplier using UT multiplication technique. Here Ain and Bin are the input operands of the 53-bit multiplier; Y is the output of the multiplier.

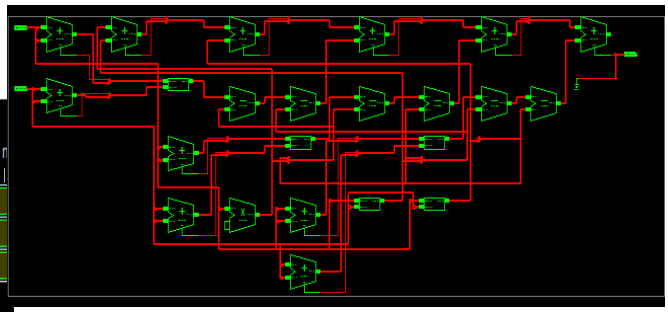


Fig. 10 Technology Schematic of 53-bit mantissa multiplier

Fig. 10 shows the technology schematic of the mantissa bits multiplier. In this, the red color wires are interconnecting wiring network of the circuit. The technology schematic has adders, subtractors, and multipliers. These components internally through wiring network to build the actual circuit.

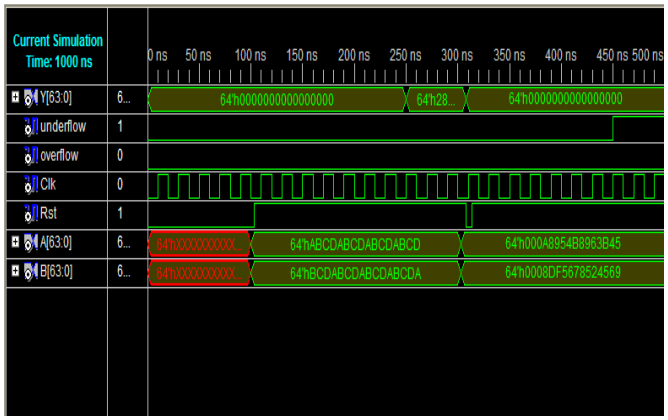



Fig. 11 Simulation result of DPUT multiplier

The resulting waveform of the DPUT multiplier has two inputs and each one has 64-bits wide. The two 64-bit input operands are performed multiplication operation; the resultant has 64-bits length. Fig. 11 shows the simulation resulting waveform of DPUT multiplier.In this the exceptional conditions like overflow and underflow are checked by applying the inputs A[63:0]=000a8954b8963b45 and B[63:0]=0008df5678524569. The output will display all zeros because of an exceptional case. The exceptional case is resultant exponent is less than or equal to zero.

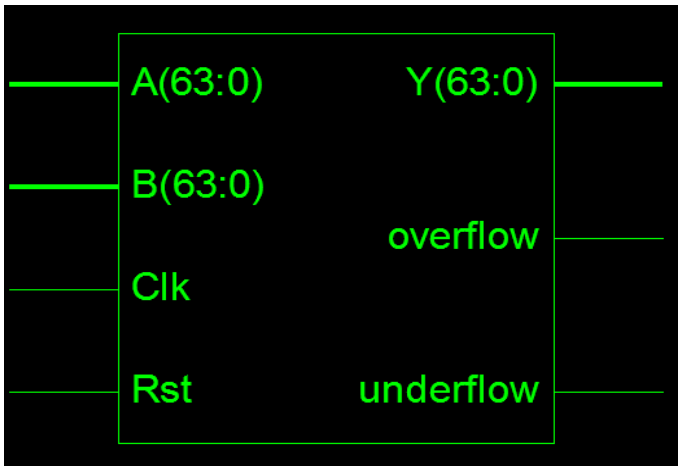



Fig. 12 RTL Schematic of DPUT multiplier

The RTL Schematic of DPUT multiplier is shown in Fig. 12. The inputs are A[63:0], B[63:0], clock and reset and outputs areY[63:0], underflow and overflow. A[63:0] has 64-bit of floating point operand and B[63:0] has 64-bit of another floating point operand. These two operands are applied to inputs of floating point multiplier. The Y[63:0] is 64-bit floating point multiplier output. Overflow and underflow are exceptional conditions occur during the floating point multiplication.

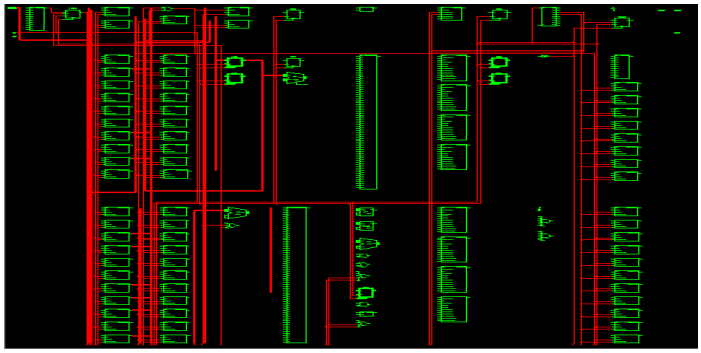


Fig.13 Technology Schematic of DPUT multiplier

The Technology schematic is in Fig. 13 represents the how the components are internally connected. It also represents how the circuits are actually arranged in the FPGA.

Table 1: Device utilization summary.

| Device | No. of. Slice Registers | | No. of. Slice LUTs | |
|---|---|---|---|---|
| | Karatsuba Multiplier | UT Multiplier | Karatsuba Multiplier | UT Multiplier |
| Spartan-2 (xc2s15-6cs144) | 1877 | 1970 | 3599 | 3607 |
| Spartan-2 (xc2s200-6fg256) | 1878 | 1970 | 3599 | 3607 |
| Spartan-2e (xc2s600e-7fg676) | 1933 | 1969 | 3599 | 3607 |
| Spartan-3 (xc3s4000l-4fg900) | 797 | 504 | 1384 | 900 |
| Spartan-3a (xc3s1400a-5fg676) | 798 | 526 | 1378 | 936 |
| Spartan-3e (xc3s1600e-5fg484) | 798 | 525 | 1378 | 936 |

| | | | | |
|---|---|---|---|---|
| **Virtex-2 (xc2v6000-6ff1517)** | **796** | **506** | **1384** | **902** |
| **Virtex-2p (xc2vpx70-7ff1704)** | **796** | **506** | **1384** | **902** |
| **Virtex-4 (xc4vlx200-11ff1513)** | **798** | **441** | **1384** | **775** |
| **Virtex-5(xc5vlx110t-1ff1136)** | **390** | **373** | **1456** | **617** |

Table 1 represents the Device utilization summary. In this, the comparison parameters are number slice registers and number of slice LUTs. These two parameters are a comparison with the Karatsuba multiplier and UT multiplier. The devices are changed from Spartan to Virtex the number slice registers used by the multipliers are reduced and also the number of slice LUTs is reduced. In this, the designing of the UTmultiplier requires a minimum amount of logic resources to perform DP floating point multiplication operations.

Fig. 14 represents the delay representation of Karatsuba and UT multipliers. The delay of both the multipliers is measured in nanoseconds. The delay of the Karatsuba multiplier is 18.139ns and delay of the UT multiplier is 15.034ns. The delay of the UT multiplier is less compared to the Karatsuaba multiplier so UT multiplier is fast in comparison with the Karatsuba multiplier.

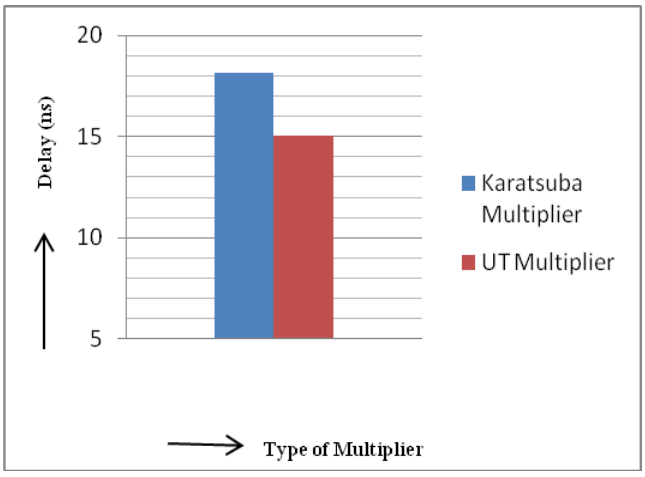



Fig. 14 Delay representation of Karatsuba and UT multipliers

Fig. 15 represents the Power consumption of the Karatsuba multiplier and UT multipliers. In this X-axis takes frequency in MHz and Y- axis takes power in terms of watts. The frequency is from 10MHz to 600MHz and notes down the power at the frequencies. In this Fig. 15 observes the Karatsuba multiplier consumes more power in comparison with the UT multiplier.

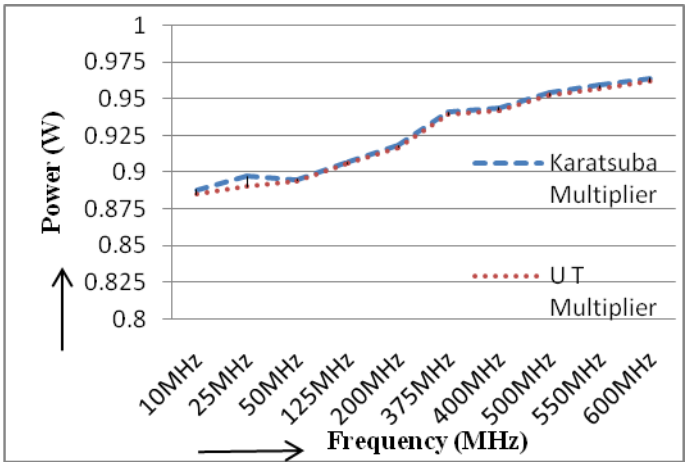



Fig. 15 Power consumption of Karatsuba and UT multipliers

## V. CONCLUSION

A Modified DP Floating-Point Multiplier architecture has been implemented using the knowledge of the existing system, which has resulted to achieve the reduction in delay and reduced hardware utilization. The modified system had a lesser delay and low power consumption. The design handles the various exceptional conditions like OVERFLOW and UNDERFLOW. concluded that after mitigation of PUI, the performance of the proposed cognitive relay network is significantly improved. for your paper size.

## REFERENCES

1. Manish Kumar Jaiswal, Ray C.C. Cheung "VLSI Implementation of DP Floating Multiplier Using Karatsuba Technique",Circuits Syst signal process, Springer science business media, vol. 32, pp. 15-27, July 2013.
2. Purna Ramesh addanki, Venkata Nagaratna Tilak Alapati and Mallikarjuna Prasad Avana "An FPGA Based High-Speed IEEE-754 DP Floating Point Adder/Subtractor and Multiplier Using Verilog" International Journal of Advanced Science and Technology, vol. 52, pp. 61-74, March 2013.
3. M. al-Ashrafy, A. Salem, W. Anis, " An Efficient Implementation of Floating Point Multiplier", Saudi International Electronics, Communications and Photonics Conference(SIECPC), pp. 1-5, April 2011.
4. Kandasamy, N., Ahmad, F., Reddy, S., Telagam, N., & Utlapalli, S. (2018). Performance evolution of 4-b bit MAC unit using hybrid GDI and transmission gate based adder and multiplier circuits in 180 and 90 nm technology. Microprocessors and Microsystems, 59, 15-28. https://doi.org/10.1016/j.micpro.2018.03.003
5. Devika Jaina, Kabiraj Sethi, and Rutupama Pamda, "Vedic Mathematics Based Multiply Accumulate Unit ", International conference on computational intelligence and communication system, pp 754-757, 2011.
6. Kandasamy, N., Telagam, N., & Devisupraja, C. (2018). Design of a Low-Power ALU and Synchronous Counter Using Clock Gating Technique. In Progress in Advanced Computing and Intelligent Engineering (pp. 511-518). Springer, Singapore.