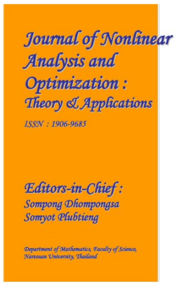
Journal of **N**onlinear **A**nalysis and **O**ptimization
Vol. 15, Issue. 1, No.10 : 2024
ISSN :**1906-9685**



Journal of Nonlinear Analysis and Optimization : Theory & Applications
ISSN : 1906-9685
Editors-in-Chief :
Sompong Dhompongsa
Somyot Plubtieng
Department of Mathematics, Faculty of Science, Naresuan University, Thailand

DESIGN OF HIGH EFFICIENT AND HIGH SPEED PARALLEL PREFIX MULTIPLIER

N.Nagaraju, M.Tech,**Miste[1],R. Sowmya Sree [2], I Sri Sai Sajana[3], V.Naga Vamsi [4], Ch.Hari Naga Babu[5].**

[1]Assistant Professor, Ece Department In Sri Vasavi Institute Of Engineering & Technology,
Nandamuru, Pedana, Andhra Pradesh 521369.

[2,3,4,5] Ug Students From Ece Department In Sri Vasavi Institute Of Engineering & Technology,
Nandamuru, Pedana, Andhra Pradesh 521369.

ABSTRACT:

Decimal computation is highly demanded in many human-centric applications such as banking, accounting, tax calculation, and currency conversion. Hence the design and implementation of radix-10 arithmetic units attract the attention of many researchers. Among the basic decimal arithmetic operations, multiplication is not only a frequent operation but also has high complexity and considerable power consumption. A multiplier is one of the key hardware blocks in most Digital Signal Processing (DSP) systems. The complexity of the circuit depends mainly on the multiplication count needed to develop the method. Therefore, by using parallel prefix multiplier, the system increases efficiency. By using this system accurate output is evaluated.

Keywords: **Digital Signal Processing (DSP), Multiplier, Parallel Prefix Multiplier.**

INTRODUCTION :

Ripple Carry Adder (RCA), carry save adder (CSA) , ***Carry Look Ahead*** (CLA) and ***Conventional Carry SkipAdder*** (CSKA) are the types of adders which provide effective configuration. Carry skip optimization calculation is familiar to map the issues that happened in the framework. Staggered tree structures are executed in the convey skip enhancement strategy. This will fix the length of modules in the framework. This will upgrade the quantity of levels, number of sizes and number of blocks. Anolog signals are used to speed up the communication. Complementary Metal-Oxide-Semiconductor (CMOS) will carry out the consistent designs in the tight fields. Static and dynamic doors are carried out to restrict the tasks of double adders.   In modern microprocessors, the number of adders is optimized to get effective output.   Adder generation path is introduced to generate the integers for execution. On the off chance that a skimming point unit is available to show up in the critical adder, at the base of multiplier exhibit, and in the divider. Low power adders show up in the type control hardware for increase and partition. Incrementers and comparators are additional types of adders, and they show up in different spots.Subsequently, the recognizable proof of a fitting adder generator is a high influence apparatus for making a productive plan.   For example, it is attractive again and again for the execution unit to be really convenient to operate, leaving speed and zone as optional requirements. The Intel 80486 execution unit circuit is typically designed to generate 8, 16, or 32 bit limits, as these are the native data types for that engineering. The top of the line Alpha has enormous word width, fine grain pipe and high clock speed, plan to make speed a necessary advance. There are not many adders in a Reduced Instruction Set Computer (RISC) processor outside of the execution units of the total number and costing point centers. Among them the advantages of RISC design is the exclusion of adders from basic ways, for example, very fast and efficient. However, not all processors are RISC architectures. The multiplication is an important central function in arithmetic logic operation in several application such as digital filtering, digital communication. The faster device with low power consumption is the demand of every consumer. High speed components of the device and reduce the power consumption leads to enhanced performance. These days, the demand of portable electronics modules is on rise in which various digital signal processing (DSP) are used, Therefore, the compact and the faster multiplier plays a vital role in designing such modules. The computational performance of a DSP system is limited by its multiplier performance and multiplicand dominates the execution time of most of the DSP algorithms, and therefore high-speed multiplier is desired with an ever -increasing quest for greater computing power on battery operated mobile devices. The design emphasis has shifted from optimizing conventional delay time, area, size to minimize power dissipation. Therefore, for maintaining the high performance, the speed and area of multiplier are a major design issues and they need to be optimized for enhancing performance. The architecture of multiplier can be split into three stages namely, partial product generation stage, partial product reduction stage and the final addition of the reduced partial product stage. Traditionally, the shift and add algorithm have been used to perform the multiplication However, it is not suitable for faster multiplication in VLSI because it requires a greater number of adder units which leads to higher delay in performing multiplication operation. Some of traditional approaches for speeding up multiplication operation is reducing the number of partial products by using multibit compressor.

Multiplier can be classified into two categories namely, serial and parallel multipliers. In a serial multiplier, each bit of multiplier is used for evaluating the partial product whereas in parallel multiplier, partial products from each bit of multiplier are computed in parallel. The main parameter that determines the performance of the parallel multiplier is the number of partial products, that is to be added. In a parallel multiplier, the speed is compromised to achieve better performance in terms of area and consumption.

LITERATURE SURVEY :

M. Mehri, M. H. M. Kouhani, N. Masoumi and R. Sarvari, et.al [1] More dependability is thought to be provided by the technique of the 16-bit Wallace Tree approximation multiplier with the 15-4 compressor mentioned. Xilinx Integrated Software Environment (ISE) 14.7 is used to simulate and synthesize the 16x16 Wallace tree multiplier. The computation time for a multiplier is greater than it produces and integrates incomplete products. Prior to addition, this Wallace tree technique utilizes compression to reduce the amount of partial products. This focuses on 16x16 Multiplier Design and Analysis to improve factors like Area and Delay.

F. Frustaci, M. Lanuzza, P. Zicari, S. Perri and P. Corsonello, et.al [2] suggests a novel, about 4–2 compressor architecture. Through the efficient use of the suggested compressor and to lower the output error, a redesigned architecture of the Dadda Multiplier is described. The suggested compressor and multiplier are assessed for efficiency in a 45 nm standard CMOS technology and their characteristics are compared to those of the most advanced approximation multipliers through exacting experimental testing. The results demonstrate that the suggested compressor significantly reduces error rates when compared to other approximate compressors that are documented in the literature. Additionally, compared to that of an exact multiplier, the suggested multiplier displays reductions in power consumption, delay, and shows the 35%, 36%, and 17%, respectively. Some of the image processing applications evaluate the multiplier's performance. According on the precise output image, the suggested multiplier typically analyses images that have 85% structural similarity.

M. Chinbat et al., [3] explained the cryptography system depends heavily on the Multiplication. For the SM2 (an elliptic curve based algorithm) algorithm, six modified multiplier techniques are provided. On a Xilinx Virtex-7 FPGA, each technique is implemented in 192 bits. The TriSection Pezaris Array Multiplier (TPAM), Carry Propagate Array Multiplier (CPAM), BaughWooley Array Multiplier (BWAM), Carry Save Array Multiplier (CSAM), and the mod m reducer module of the SM2 algorithm improved Modified Booth Multiplier (MBM) methods, which were included into multipliers for 192-bit architecture. According to data on the PublicKey Encryption (PKE) system's multipliers are total power use, timing speed, and die area compared to parallel array multipliers, the Montgomery multiplier performs better.

U. Farooq, I. Baig and B. A. Alzahrani, et.al [4] explained all digital processing systems depend heavily on multipliers, however there is still a research issue connected to the characteristics of area, delay, power, speed, and accuracy. Multiplication is basically done by doing repeated additions, multiplyers hence have more adders than adders.. Adders should be handled with extra care. The Partial Products (PP) step sits in the centre of the multiplier, multiplicand, and addition processes. Modified fast designs are suggested that provide the product with the lowest power loss and power delay for 32-bit, 4-bit, 16-bit, and 64-bit systems. When compared to the Brent Kung adders, which has roughly the same amount of computation nodes and logic depth as the proposed structure results in the power reductions of 3% to 7% and speed improvements of 15% to 35%.

EXSISTING SYSTEM :

GENERAL ARCHITECTURE OF PARALLEL DECIMAL MULTIPLIER

Decimal multipliers, like their binary counterparts, have three main steps, which are called partial product generation(PPG), partial product reduction (PPR), and the final addition (or redundant to not-redundant conversion). However, decimal multiplication is more complicated than binary multiplication in the entire aforementioned steps. The PPG in binary multiplication can be done by a simple AND-gate matrix. However, due to the wider range of decimal digits ,in decimal multiplication, various techniques like lookuptables, decimal digit-multipliers, or pre-computed multiples must be used to provide various multiples of the multiplic and.Moreover, due to binary logic and BCD encoding of decimal numbers in the decimal implementations, for compensating he difference carry value in decimal and binary, all the decimal add operations in the PPR and final addition needsa correction step.All partial products are generated simultaneously in the PPG step. As mentionedabove, various methods exist for generating each partial product; using pre-computed multiples is dominant. In the naïvei mplementation, all the possible multiples of multiplicandX(i.e., {0X, 1X, . . . , 9X)} are needed. However, in practice,just a limited subset of multiples of the multiplicand, called primary multiples, are generated (e.g., {1X, 2X, 4X, 5X}or {±1X, ±2X, 5X, 10X}). The primary multiples can be generated in constant time. The other multiples are computed by using the primary multiples (e.g., 9X = 4X +5X or 9X =10X − 1X). For constructing the PPG matrix (i.e., aligned all the partial products), the multiplier digits are recoded and these recoded values are used for selecting proper multiples.

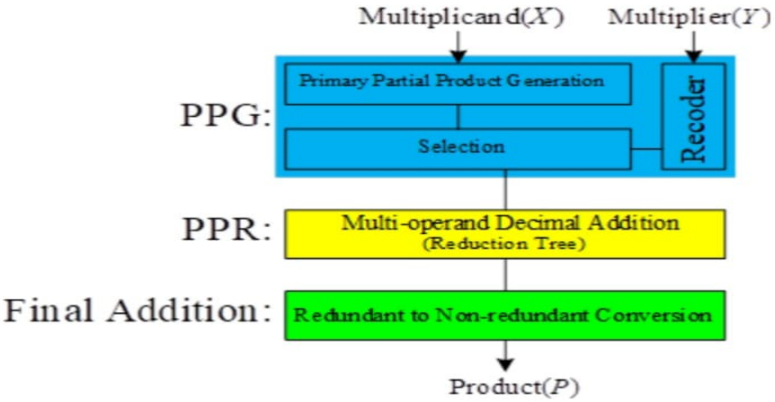

Multiplicand(X)
Multiplier(Y)
PPG:
Primary Partial Product Generation
Selection
Recoder
PPR:
Multi-operand Decimal Addition
(Reduction Tree)
Final Addition:
Redundant to Non-redundant Conversion
Product(P)

Fig: General Architecture of Parallel Decimal Multiplier.

PROPOSED SYSTEM :

Modular arithmetic is a system of arithmetic for integers, which considers the remainder arithmetic, numbers "wrap around" upon reaching a given fixed quantity (this given quantity is known as the modulus) to leave a remainder. Modular arithmetic is often tied to prime numbers, for instance, in Wilson's theorem, Lucas's theorem, and lemma, and generally appears in fields cryptography, computer science, and computer algebra. An intuitive usage of modular arithmetic is with a 12 hour clock. If it is 10:00 now, then in 5 hours the clock will show 3:00 instead of 15:00. 3 is the remainder of 15 with a modulus of 12. A number x mod N is the equivalent of asking for the remainder of when divided by . Two integers and are said to be congruent (or in the same equivalence class) modulo if they have the same remainder upon division by N. In such a case, we say that a=b (mod N). In modular arithmetic computation, modular multiplication, more commonly referred to as Montgomery multiplication, is a method for performing fast modular multiplication. Given two integers *a* and *b* classical modular multiplication algorithm computesthe double-width product *ab* performs a division, subtracting multiples ofcancel out the unwanted high bits until the remainderis once again less than *N*.instead *adds* multiples of *N* to cancel out theuntil the result is a multiple of a convenient (i.e.power of two) constant *R >N* discarded, producing a result less thanconditional subtract reduces this to less tprocedure avoids the complexity of quotient digitestimation and correction found in standardalgorithms. In this proposed diagram, Multiplier and Multiplicand will generate partial product generation. In this partial product generation, partial products are generated. These products are aligned and generate the propagator and generator. The parallel prefix addition will be applied to the propagator and generator. Then the final output is evaluated. Multiplier is one of the key component in arithmetic and logic unit. Higher throughput arithmetic operations are important to achieve the desired performance in many real time signal and image processing applications. In multiplier power dissipation and speed are the most important parameter. The main disadvantage of the multiplier is the worst case delay. Which leads to reducing the time delay as well as the path delay. Digital signal that travel from input of logic gate to that of the output gate will cause delay due to the minimum switching activity ie., the total number of signal transition of the system. The low power will reduce the complexity, execution time and power which can overcome the drawback of other multiplier.

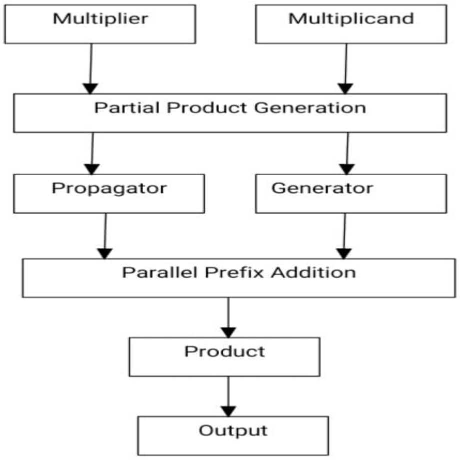


Fig: Block Diagram

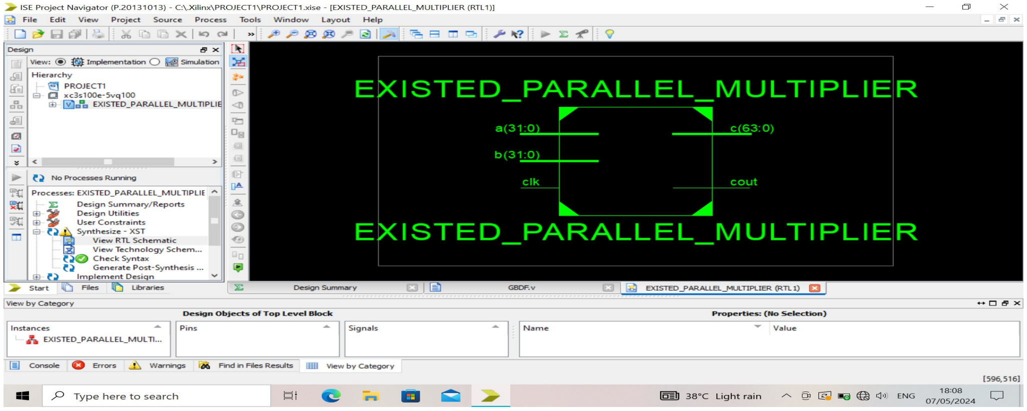
MULTIPLICAND AND MULTIPLIER:

Multiplicand and Multiplier are considered as contribution to the framework. As multiplicand and multiplier 0's and 1's are distinguished. Figuring strategy is used to limit exchanging speed up and energy activity. Zeros finding rationale can recognize zeros from acquired item. The item can be added utilizing equal prefix adder to limit region.

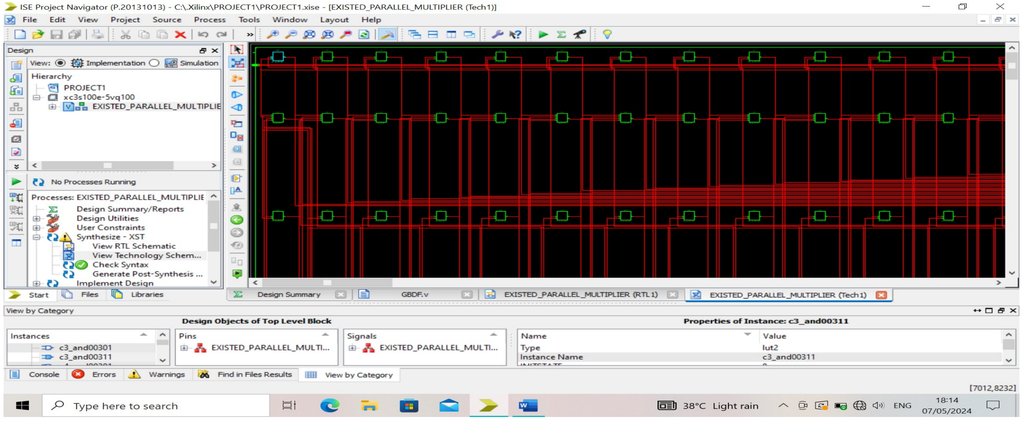
RESULT ANALYSIS

EXISTING SYSTEM

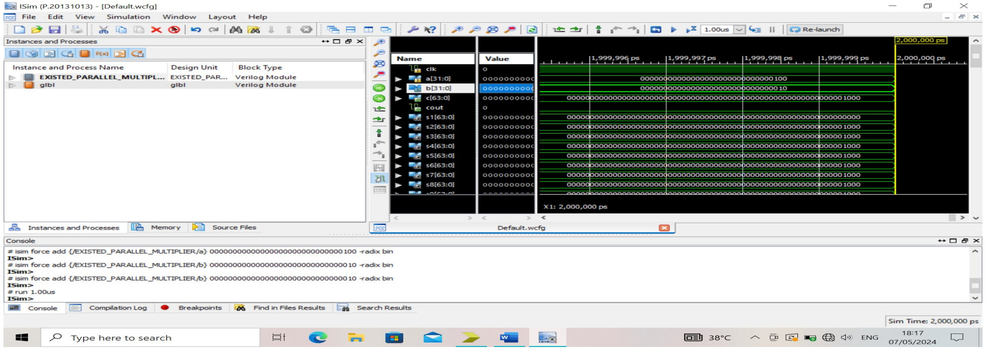
RTL SCHEMATIC:


ISE Project Navigator (P.20131013) - C:\Xilinx\PROJECT1\PROJECT1.xise - [EXISTED_PARALLEL_MULTIPLIER (RTL1)]
File Edit View Project Source Process Tools Window Layout Help
Design
View: ● Implementation ○ Simulation
Hierarchy
PROJECT1
xc3s100e-5vq100
EXISTED_PARALLEL_MULTIPLIE
No Processes Running
Processes: EXISTED_PARALLEL_MULTIPLIE
Design Summary/Reports
Design Utilities
User Constraints
Synthesize - XST
View RTL Schematic
View Technology Schem...
Check Syntax
Generate Post-Synthesis ...
Implement Design
EXISTED_PARALLEL_MULTIPLIER
a(31:0)
b(31:0)
clk
c(63:0)
cout
EXISTED_PARALLEL_MULTIPLIER
Start Files Libraries
Design Summary
GBDF.v
EXISTED_PARALLEL_MULTIPLIER (RTL1)
View by Category
Design Objects of Top Level Block
Properties: (No Selection)
Instances
EXISTED_PARALLEL_MULTI...
Pins
Signals
Name
Value
Console Errors Warnings Find in Files Results View by Category
[596,516]
Type here to search
38°C Light rain
ENG
18:08
07/05/2024

TECHNOLOGICAL SCHEMATIC:

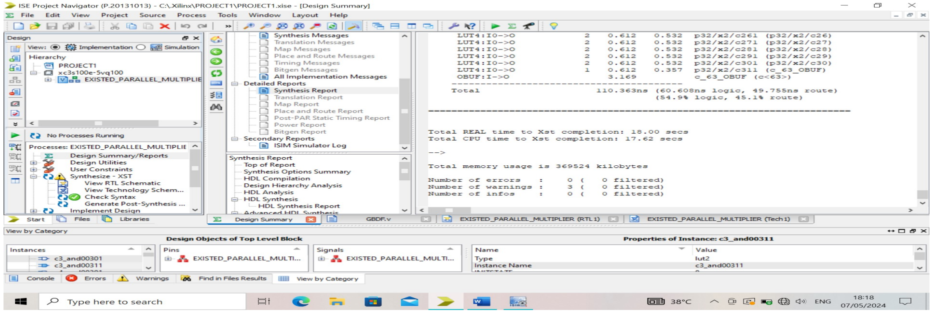

ISE Project Navigator (P.20131013) - C:\Xilinx\PROJECT1\PROJECT1.xise - [EXISTED_PARALLEL_MULTIPLIER (Tech1)]
File Edit View Project Source Process Tools Window Layout Help
Design
View: ● Implementation ○ Simulation
Hierarchy
PROJECT1
xc3s100e-5vq100
EXISTED_PARALLEL_MULTIPLIE
No Processes Running
Processes: EXISTED_PARALLEL_MULTIPLIE
Design Summary/Reports
Design Utilities
User Constraints
Synthesize - XST
View RTL Schematic
View Technology Schem...
Check Syntax
Generate Post-Synthesis ...
Implement Design
Start Files Libraries
Design Summary
GBDF.v
EXISTED_PARALLEL_MULTIPLIER (RTL1)
EXISTED_PARALLEL_MULTIPLIER (Tech1)
View by Category
Design Objects of Top Level Block
Properties of Instance: c3_and00311
Instances
c3_and00301
c3_and00311
Pins
EXISTED_PARALLEL_MULTI...
Signals
EXISTED_PARALLEL_MULTI...
Name
Type
Instance Name
Value
lut2
c3_and00311
Console Errors Warnings Find in Files Results View by Category
[7012,8232]
Type here to search
38°C Light rain
ENG
18:14
07/05/2024

SIMUALTION RESULT:



```
Console
# isim force add {/EXISTED_PARALLEL_MULTIPLIER/a} 00000000000000000000000000000100 -radix bin
ISim>
# isim force add {/EXISTED_PARALLEL_MULTIPLIER/b} 00000000000000000000000000000010 -radix bin
ISim>
# isim force add {/EXISTED_PARALLEL_MULTIPLIER/b} 00000000000000000000000000000010 -radix bin
ISim>
# run 1.00us
ISim>
```

DELAY:



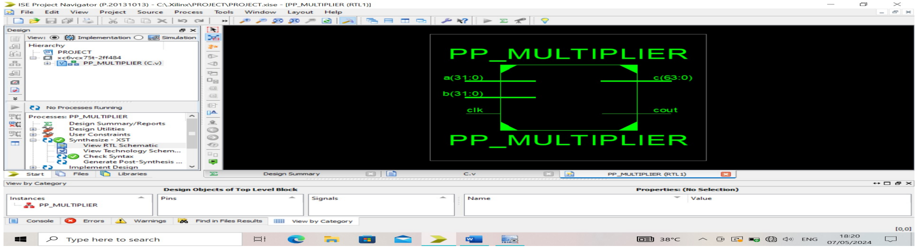
```
        LUT4:I0->O            2   0.612   0.532  p32/x2/c261 (p32/x2/c26)
        LUT4:I0->O            2   0.612   0.532  p32/x2/c271 (p32/x2/c27)
        LUT4:I0->O            2   0.612   0.532  p32/x2/c281 (p32/x2/c28)
        LUT4:I0->O            2   0.612   0.532  p32/x2/c291 (p32/x2/c29)
        LUT4:I0->O            2   0.612   0.532  p32/x2/c301 (p32/x2/c30)
        LUT4:I0->O            1   0.612   0.357  p32/x2/c311 (c_63_OBUF)
        OBUF:I->O                 3.169          c_63_OBUF (c<63>)
      ----------------------------------------
      Total                     110.363ns (60.608ns logic, 49.755ns route)
                                          (54.9% logic, 45.1% route)

=========================================================================


Total REAL time to Xst completion: 18.00 secs
Total CPU time to Xst completion: 17.62 secs

-->

Total memory usage is 369524 kilobytes

Number of errors   :    0 (   0 filtered)
Number of warnings :    3 (   0 filtered)
Number of infos    :    0 (   0 filtered)
```

Properties of Instance: c3_and00311

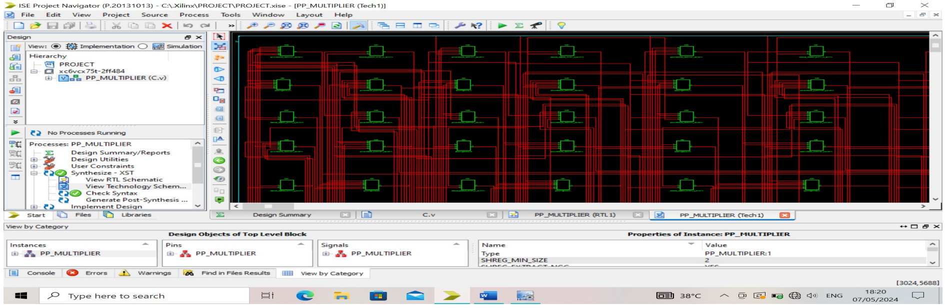
Name	Value
Type	lut2
Instance Name	c3_and00311

PROPOSED SYSTEM

RTL SCHEMATIC:



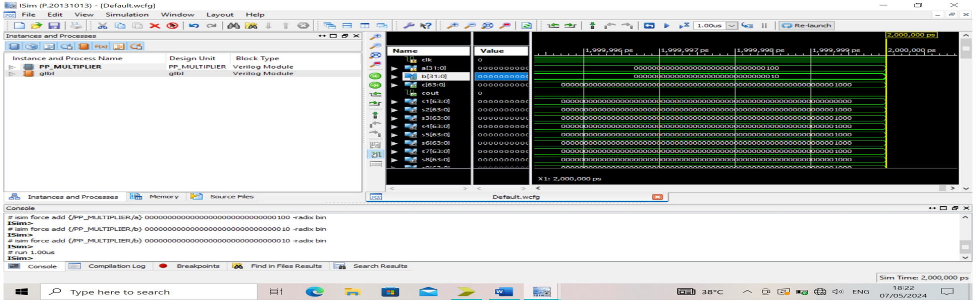
TECHNOLOGICAL SCHEMATIC:



Properties of Instance: PP_MULTIPLIER

Name	Value
Type	PP_MULTIPLIER:1
SHREG_MIN_SIZE	2

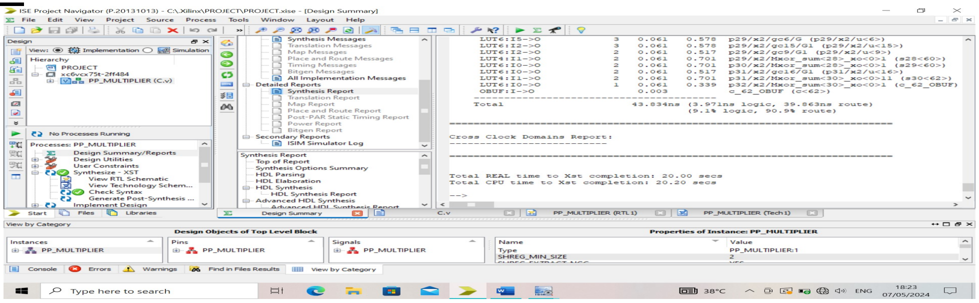
SIMULATION RESULTS:



```
# isim force add {/PP_MULTIPLIER/a} 00000000000000000000000000000100 -radix bin
ISim>
# isim force add {/PP_MULTIPLIER/b} 00000000000000000000000000000010 -radix bin
ISim>
# isim force add {/PP_MULTIPLIER/b} 00000000000000000000000000000010 -radix bin
ISim>
# run 1.00us
ISim>
```

Sim Time: 2,000,000 ps

DELAY:



```
        LUT6:I5->O            3   0.061   0.578  p29/x2/gc6/G (p29/x2/u<6>)
        LUT6:I2->O            3   0.061   0.578  p29/x2/gc15/G1 (p29/x2/u<15>)
        LUT6:I2->O            2   0.061   0.517  p29/x2/gc9/G1 (p29/x2/u<9>)
        LUT4:I1->O            2   0.061   0.701  p29/x2/Mxor_sum<28>_xo<0>1 (s28<60>)
        LUT6:I0->O            2   0.061   0.701  p30/x2/Mxor_sum<28>_xo<0>1 (s29<60>)
        LUT6:I0->O            2   0.061   0.517  p31/x2/gc16/G1 (p31/x2/u<16>)
        LUT4:I1->O            2   0.061   0.701  p31/x2/Mxor_sum<30>_xo<0>11 (s30<62>)
        LUT6:I0->O            1   0.061   0.339  p32/x2/Mxor_sum<30>_xo<0>1 (c_62_OBUF)
        OBUF:I->O                 0.003          c_62_OBUF (c<62>)
      ----------------------------------------
      Total                      43.834ns (3.971ns logic, 39.863ns route)
                                          (9.1% logic, 90.9% route)

=========================================================================

Cross Clock Domains Report:
--------------------------

=========================================================================


Total REAL time to Xst completion: 20.00 secs
Total CPU time to Xst completion: 20.20 secs

-->
```

Properties of Instance: PP_MULTIPLIER

Name	Value
Type	PP_MULTIPLIER:1
SHREG_MIN_SIZE	2

CONCLUSION :

Decimal computation is highly demanded in many human-centric applications such as banking, accounting, tax calculation, and currency conversion. With advances in technology, many researchers have tried and are trying to design multipliers which offer either of the following design targets high speed, low power consumption, regularity of layout. Hence less area or even combination of them in one multiplier thus making them suitable for various high speed, low power and compact VLSI implementation. Multiplication is the basic building block for several DSP processors, Image processing and many other. The latency of existing multiplier has been reduced. The conditional sum technique used in the adder is a good technique for energy efficiency. Therefore, by using parallel prefix multiplier, the system increases efficiency. By using this system accurate output is evaluated.

REFERENCES

[1] M. Mehri, M. H. M. Kouhani, N. Masoumi and R. Sarvari, "New Approach to VLSI Buffer Modeling, Considering Overshooting Effect," in *IEEE Transactions on Very Large Scale Integration (VLSI) Systems*, vol. 21, no. 8, pp. 1568-1572, Aug. 2013, doi: 10.1109/TVLSI.2012.2211629.

[2] F. Frustaci, M. Lanuzza, P. Zicari, S. Perri and P. Corsonello, "Designing High-Speed Adders in Power-Constrained Environments," in *IEEE Transactions on Circuits and Systems II: Express Briefs*, vol. 56, no. 2, pp. 172-176, Feb. 2009, doi: 10.1109/TCSII.2008.2010187.

[3] M. Chinbat, "Performance Comparison of Finite Field Multipliers for SM2 Algorithm based on FPGA Implementation," *2020 IEEE 14th International Conference on Anticounterfeiting, Security, and Identification (ASID)*, Xiamen, China, 2020, pp. 69-72, doi: 10.1109/ASID50160.2020.9271714.

[4] U. Farooq, I. Baig and B. A. Alzahrani, "An Efficient Inter-FPGA Routing Exploration Environment for Multi-FPGA Systems," in *IEEE Access*, vol. 6, pp. 56301-56310, 2018, doi: 10.1109/ACCESS.2018.2873041.

[5] Kesava, R. Bala Sai, B. Lingeswara Rao, K. Bala Sindhuri, and N. Udaya Kumar. "Low power and area efficient Wallace tree multiplier using carry select adder with binary to excess1 converter." In 2016 Conference on Advances in Signal Processing (CASP), pp. 248253. IEEE, 2016

[6] Marimuthu, R., Elsie Rezinold, Y., &Malick, P.S., “Design and analysis of multiplier using approximate 15-4 compressor”, *IEEE Acess, 5*, 2017, pp. 1027-1036.

[7] Momeni, A., Han, J., Montuschi, P., & Lombardi, F., “Design and analysis of approximate compressors for multiplication”, *IEEE Transactions on Computers*, Vol.64, Issue 4, 2015.

[8] M. Swathi and B. Rudra, "Implementation of Reversible Logic Gates with Quantum Gates," *2021 IEEE 11th Annual Computing and Communication Workshop and Conference (CCWC)*, NV, USA, 2021, pp. 1557-1563, doi: 10.1109/CCWC51732.2021.9376060.

[9] Lin, C. H., & Lin, I. C., “High accuracy approximate multiplier with error correction”, *IEEE 31st InternationalConference on Computer Design*, 2013, pp. 33–38.

[10] N. Zhu, W. L. Goh, W. Zhang, K. S. Yeo and Z. H. Kong, "Design of Low-Power HighSpeed Truncation-Error-Tolerant Adder and Its Application in Digital Signal Processing," in *IEEE Transactions on Very Large Scale Integration (VLSI) Systems*, vol. 18, no. 8, pp. 1225-1229, Aug. 2010, doi: 10.1109/TVLSI.2009.2020591.