The Seybold Report
ISSN 1533-9211
DOI: 10.5281/zenodo.8300481

DESIGN OF 3-BIT CMOS WALLACE MULTIPLIER

Dr. MANGALA GOWRI S G

Associate Professor, Department of Electronics and Communication Engineering, Atria Institute of Technology, Bengaluru. Email: Mangalag21@gmail.com

Dr. SHASHIDHAR T M

Professor, Department of Computer Science and Engineering, Acharya Institute of Technology, Bengaluru. Email: shashilara@gmail.com

Dr. SUNITHA R

Associate Professor, Department of Electronics and Communication Engineering, Rajarajeswari College of Engineering, Bengaluru. Email: ecesunitha2022@gmail.com

SHYLAJA V

Assistant Professor, Department of Electronics and Communication Engineering, Bangalore Institute of Technology, Bengaluru. Email: shylajav@bit-bangalore.edu.in

Dr. GIRISH H

Associate Professor, Department of Electronics and Communication Engineering, Cambridge Institute of Technology, Bengaluru. Email: hgirishphd@gmail.com, girish.ece@cambridge.edu.in

Abstract

A multiplier is one of the key hardware blocks in most digital and high performance systems such as FIR filters, digital signal processors and microprocessors etc. With advances in technology, many researchers have tried and are trying to design multipliers which offer either of the following- high speed, low power consumption , regularity of layout and hence less area or even combination of them in multiplier. Thus making them suitable for various high speed, low power, and compact VLSI implementations. However area and speed are two conflicting constraints. So improving speed results always in larger areas. So here we try to find out the best trade off solution among the both of them. Generally as we know multiplication goes in three basic steps. Partial product generation, reduction and final stage is addition. Hence in this paper we have first tried to design different adders and compare their speed and complexity of circuit i.e. the area occupied. And then we have designed Wallace tree multiplier then followed by Conventional, proposed Wallace multipliers and have compared the speed and Power consumption in both of them. While comparing the adders we found out that Ripple Carry Adder had a smaller area while having lesser speed, in contrast to which sklansky Adders are high speed but possess a larger area. After designing and comparing the adders we turned to multipliers. Initially we went for Parallel Multiplier and then Wallace Tree Multiplier. In the mean time we learned that delay amount was considerably reduced when sklansky adder were used in Wallace Tree applications.

INTRODUCTION

A **Wallace multiplier** is a hardware implementation of a binary multiplier, a digital circuit that multiplies two integers. It uses a selection of full and half adders (the **Wallace tree** or **Wallace reduction**) to sum partial products in stages until two numbers are left. Wallace multipliers reduce as much as possible on each layer, whereas Dadda multipliers try to minimize the required number of gates by postponing the reduction to the upper layers.

ISSN 1533-9211

DOI: 10.5281/zenodo.8300481

Wallace multipliers were devised by the Australian computer scientist Chris Wallace in 1964. The Wallace tree has three steps:

1. Multiply each bit of one of the arguments, by each bit of the other.
2. Reduce the number of partial products to two by layers of full and half adders.
3. Group the wires in two numbers, and add them with a conventional adder.

Compared to naively adding partial products with regular adders, the benefit of the Wallace tree is its faster speed. It has reduction layers, but each layer has only propagation delay. A naive addition of partial products would require time. As making the partial products is and the final addition is, the total multiplication is, not much slower than addition. From a complexity theoretic perspective, the Wallace tree algorithm puts multiplication in the class NC^1. The downside of the Wallace tree, compared to naive addition of partial products, is its much higher gate count.

These computations only consider gate delays and don't deal with wire delays, which can also be very substantial. The Wallace tree can be also represented by a tree of 3/2 or 4/2 adders.

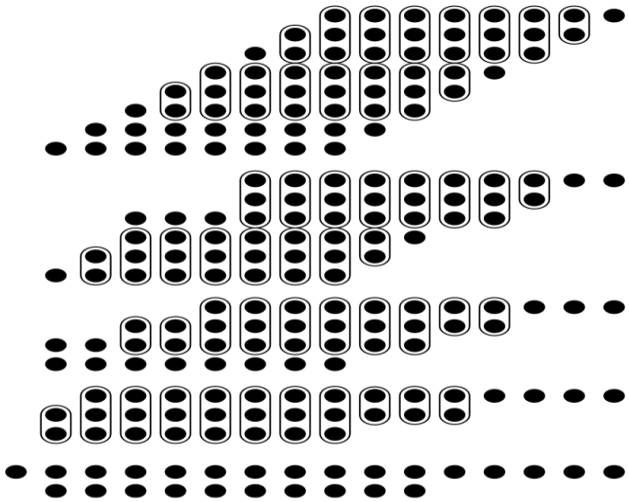


Figure 1

Ngspice

Ngspice is the open source spice simulator for electric and electronic circuits. Ngspice implements various circuits elements, like resistors, capacitors, inductors (single or mutual), transmission lines and a number of semiconductor devices like diodes, bipolar transistors, MOSFETs, MESFETs, JFETs and HFETs.

Sky130 PDKs: Open source PDK

The Sky Water Open Source PDK is a collaboration between Google and Sky Water Technology Foundry to provide a fully open source Process Design Kit and related resources, which can be used to create manufacturable designs at Sky Water's facility

The Seybold Report
ISSN 1533-9211
DOI: 10.5281/zenodo.8300481

LITERATURE SURVEY

Various types of multipliers are discussed in the literature to achieve power and performance optimization. Column compression architecture for fast multiplication proposed by Wallace offers a total delay which is proportional to the logarithm of the operand word length of the multiplier. These column compression multipliers are faster than array multipliers, because delay in an array multiplier varies linearly with the operand word length.

A unique placement strategy for reducing the stage counters in column compression architecture was proposed by Dadda. To achieve a compact layout, Goto et al have proposed a regularly structured tree multiplier with recurring blocks. Different methods for compressing the bits in a Wallace tree to achieve improved column compression was proposed by Oklobdzija and Villeger .Itoh et al have proposed a rectangular styled tree multiplier by folding to achieve a compact layout at the expense of more complicated interconnects. . Andrea Bickerstaff et al, have reported that Wallace multipliers have slightly more area and approximately the same worst case delay as that of Dadda multipliers in deep submicron technologies.

Wallace tree multipliers based on adiabatic 4-2 compressors proposed by Xien Ye et al achieve considerable amount of energy savings but with increased latency. Karthick et al have proposed XOR-XNOR based 3:2, 4:2 and 5:2 compressors for partial product reduction in Wallace tree multiplier. Naveen Kr. Gahlan et al have constructed a Wallace tree multiplier using 3:2, 4:2, 5:2, 6:2 and 7:2 compressors. They have reported that the propagation delay is reduced but with slight over head in power and area. Dakupati. Ravi Sankar et al have proposed a Wallace tree multiplier with Sklansky Adder in final stage of addition.

The fans out of some signals are high in Sklansky Adder which may account for delay penalty. B. Ramkumar et al proposed a new design technique for Dadda multiplier by partitioning partial products which has slight improvement in the speed, area and power for wider operand word-lengths. Palaniappan Ramanathan et al have proposed high speed multiplier using decomposition logic and reported that decomposition logic improves speed and parallelism with little increase in power dissipation. The multipliers presented in the most of the existing literatures do not offer sufficient parallelism to minimize the glitch power.

METHODOLOGY

An AND gate is a logic gate having two or more inputs and a single output. An AND gate operates on logical multiplication rules. In this gate, if either of the inputs is low (0), then the output is also low. If all of the inputs are high (1), then the output will also be high. An AND gate can have any number of inputs, although 2 input and 3 input AND gates are the most common. An AND gate may have any number of input probes but only one output probe. A high digital signal means logically 1 and a low digital signal means logically 0.

The Seybold Report
ISSN 1533-9211
DOI: 10.5281/zenodo.8300481

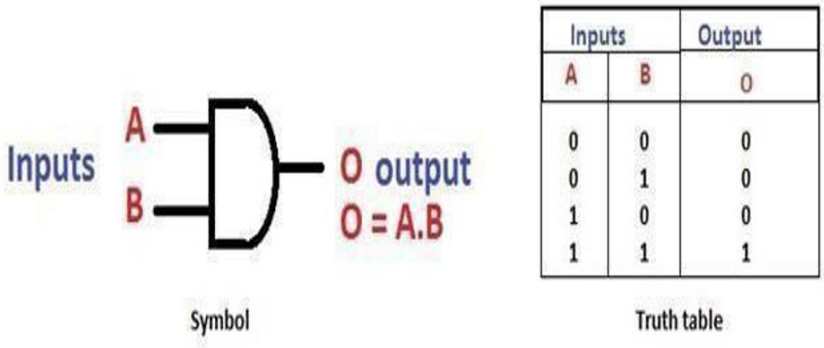
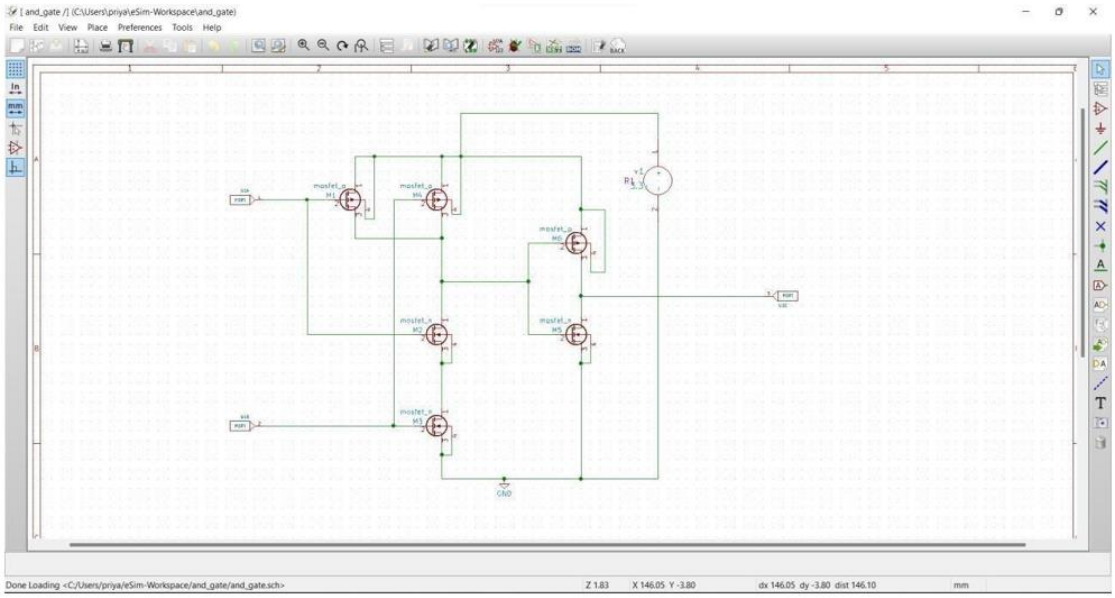


Figure 2

Construct the "and gate" as shown in the figure below using eSim:



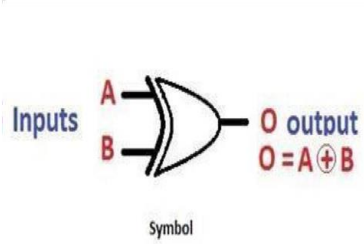
i. Once every step is followed perfectly open the Netlist that is generated and make the necessary changes to add the Sky130 models ii. The Netlist generated initially is as shown below :

Step2 Construct XOR Gate

XOR gate (sometimes EOR, or EXOR and pronounced as Exclusive OR) is a digital logic gate that gives a true (1 or HIGH) output when the number of true inputs is odd. An XOR gate implements an exclusive or ({\displaystyle \nleftrightarrow}\nleftrightarrow) from mathematical logic; that is, a true output results if one, and only one, of the inputs to the gate is true. If both inputs are false (0/LOW) or both are true, a false output results. XOR represents the inequality function, i.e., the output is true if the inputs are not alike otherwise the output is false. A way to remember XOR is "must have one or the other but not both". XOR gate XOR

The Seybold Report
ISSN 1533-9211
DOI: 10.5281/zenodo.8300481

can also be viewed as addition modulo 2. As a result, XOR gates are used to implement binary addition in computers. A half adder consists of an XOR gate and an AND gate. Other uses include subtractors, comparators, and controlled inverters.

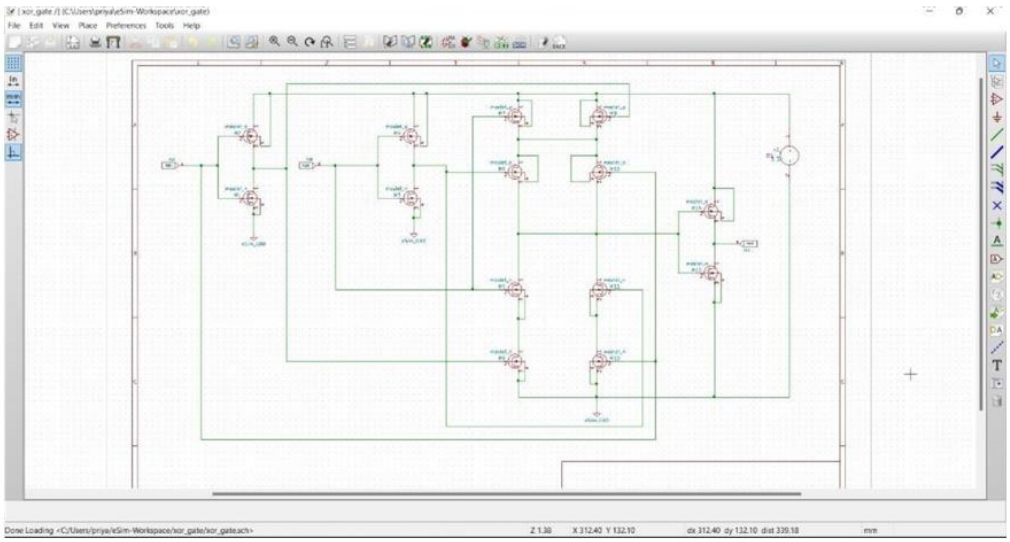


Inputs		Output
A	B	O
0	0	0
0	1	1
1	0	1
1	1	0

Truth table

Figure: 3

i. Construct the "and gate" as shown in the figure below using eSim :



- Follow the similar steps as shown for AND Gate to implement XOR Gate and generate subcircuit & Netlist for the same.
- Once every step is followed perfectly open the Netlist that is generated and make the necessary changes to add the Sky130 models.

Step3 Construct Half Adder

A half adder is used to add two single-digit binary numbers and results into a two-digit output. It is named as such because putting two half adders together with the use of an OR gate results in a full adder. In other words, it only does half the work of a full adder. He adder works by combining the operations of basic logic gates, with the simplest form using only a XOR and an AND gate. This can also be converted into a circuit that only has AND, OR and NOT gates. This is especially useful since these three simpler logic gate ICs (integrated circuits) are more

The Seybold Report
ISSN 1533-9211
DOI: 10.5281/zenodo.8300481

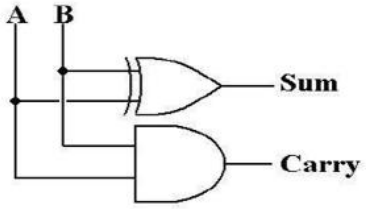
common and available than the XOR IC, though this might result in a bigger circuit since three different chips are used instead of just one.

HA takes in 2 input bits and gives Sum and Carry

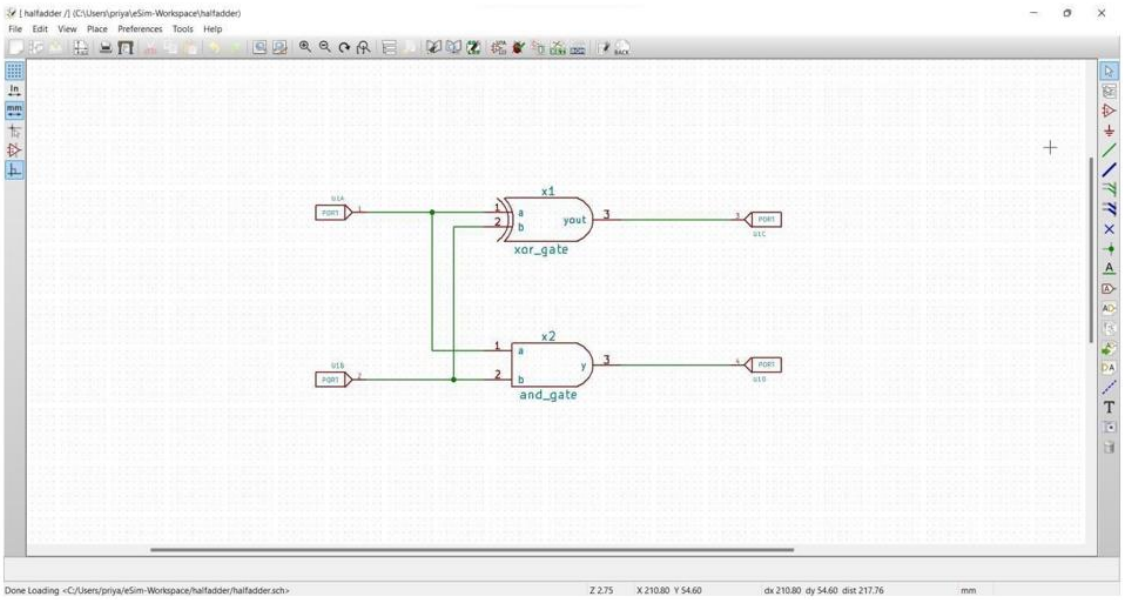
A	B	Sum	Carry
0	0	0	0
0	1	1	0
1	0	1	0
1	1	0	1

Through K Maps, we arrive at the function for Sum and Carry

Sum = A'.B + A.B' = A XOR B
Carry = A.B



1 Construct the "Half Adder" as shown in the figure below using eSim:



1. Follow the similar steps as shown for AND Gate and implement Half Adder using the subcircuits of And Gate, XOR Gate and then generate the subcircuit & Netlist for the same.
2. Once every step is followed perfectly open the Netlist that is generated and make the necessary changes to add the Sky130 models.

ISSN 1533-9211

DOI: 10.5281/zenodo.8300481

Step4 Construct Full Adder

The difference between a half-adder and a full-adder is that the full-adder has three inputs and two outputs, whereas half adder has only two inputs and two outputs. The first two inputs are A and B and the third input is an input carry as C-IN. When a full-adder logic is designed, you string eight of them together to create a byte-wide adder and cascade the carry bit from one adder to the next.

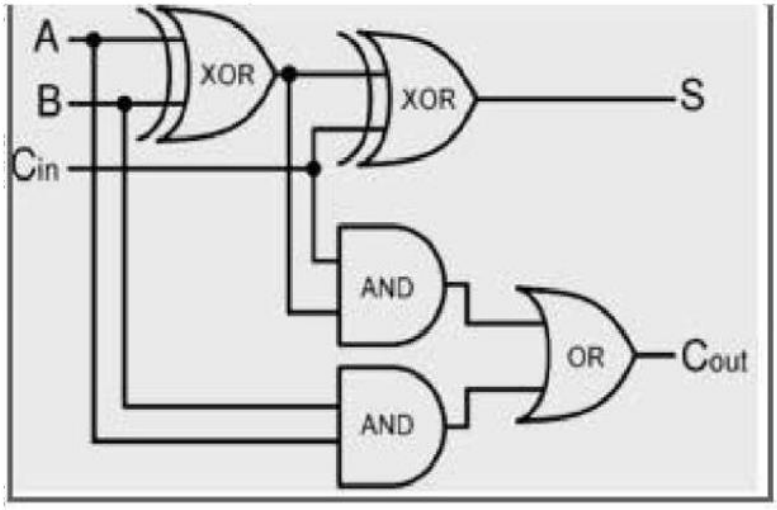


Figure 4

The output carry is designated as C-OUT and the normal output is represented as S which is 'SUM'.

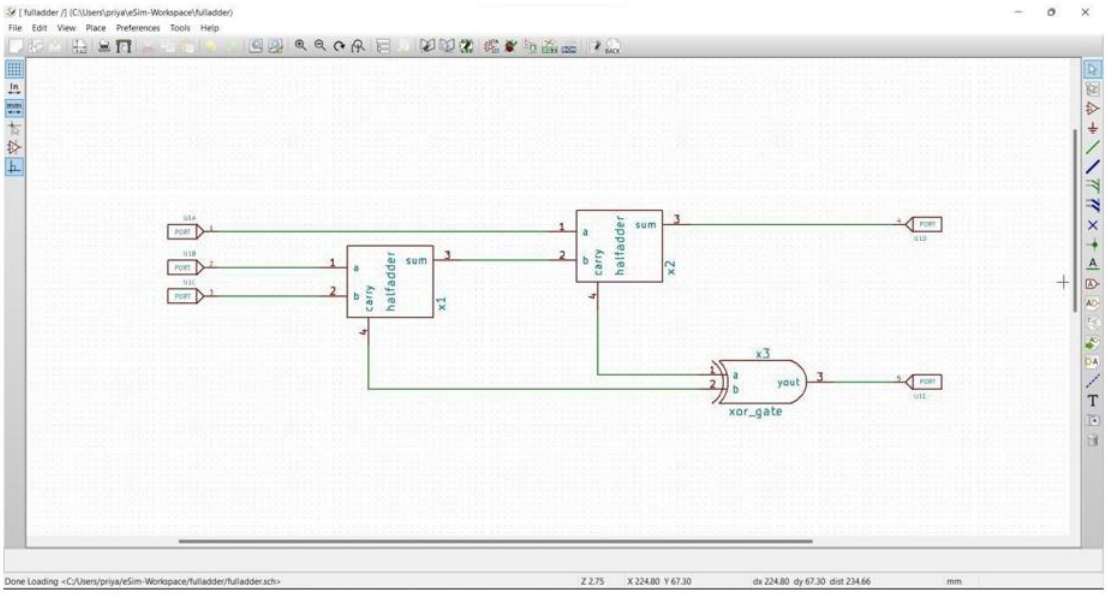
With the above **full adder truth-table**, the implementation of a full adder circuit can be understood easily. The SUM 'S' is produced in two steps:

1. By XORing the provided inputs 'A' and 'B'
2. The result of A XOR B is then XORed with the C-IN

This generates SUM and C-OUT is true only when either two of three inputs are HIGH, then the C-OUT will be HIGH. So, we can implement a full adder circuit with the help of two half adder circuits. Initially, the half adder will be used to add A and B to produce a partial Sum and a second-half adder logic can be used to add C-IN to the Sum produced by the first half adder to get the final S output.

- Construct the "Full Adder" as shown in the figure below using eSim :

The Seybold Report
ISSN 1533-9211
DOI: 10.5281/zenodo.8300481



- Follow the similar steps as shown for AND Gate and implement Half Adder using the subcircuits of Half Adder, XOR Gate and then generate subcircuit & Netlist for the same.
- Once every step is followed perfectly open the Netlist that is generated and make the necessary changes to add the Sky130 models.

3-Bit Wallace Multiplier

- Construct the "Wallace Multiplier" as shown in the figure below using eSim :

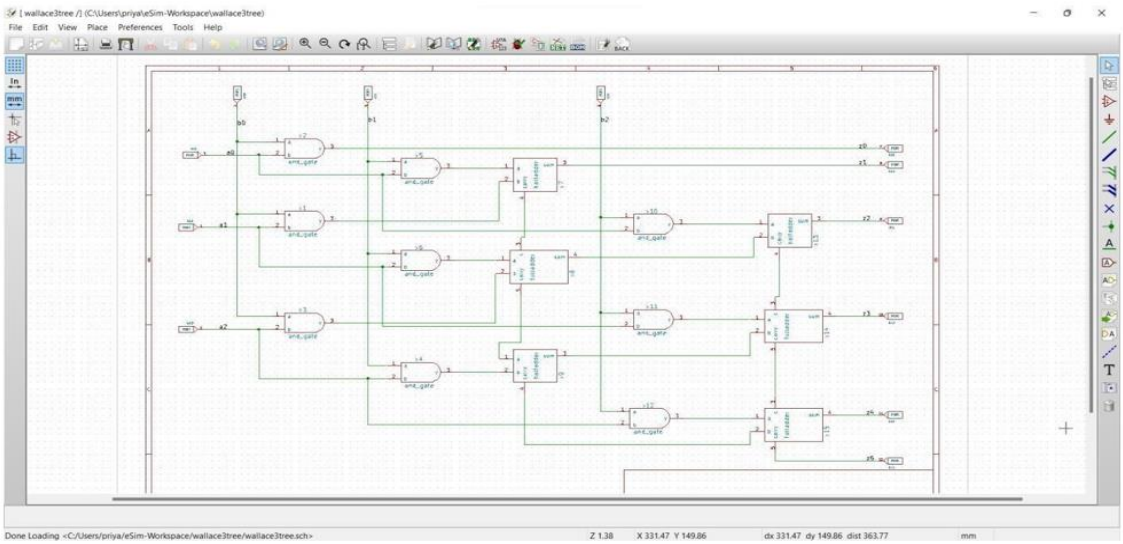


Figure 5

ISSN 1533-9211

DOI: 10.5281/zenodo.8300481

- Generate a Schematic and construct the Wallace Multiplier for a 3-bit date using the subcircuits of AND Gate, Half Adder and Full Adder & generate the Netlist for the same.
- Once every step is followed perfectly, open the Netlist that is generated and make the necessary changes to add the Sky130 models.

RESULTS

Type the command as shown in the figure below: Syntax for the command : [Go to the Location of the ngspice folder] followed by typing "ngspice" (Location where the wallace3multiplier.cir file is present)

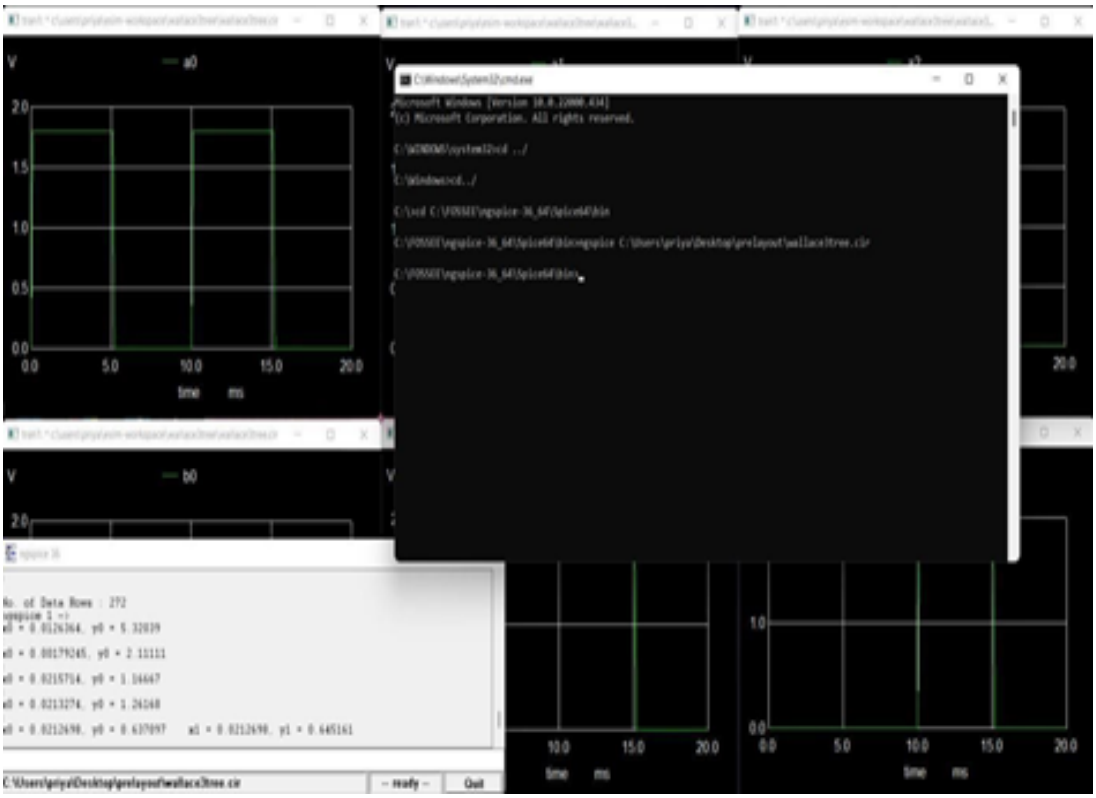


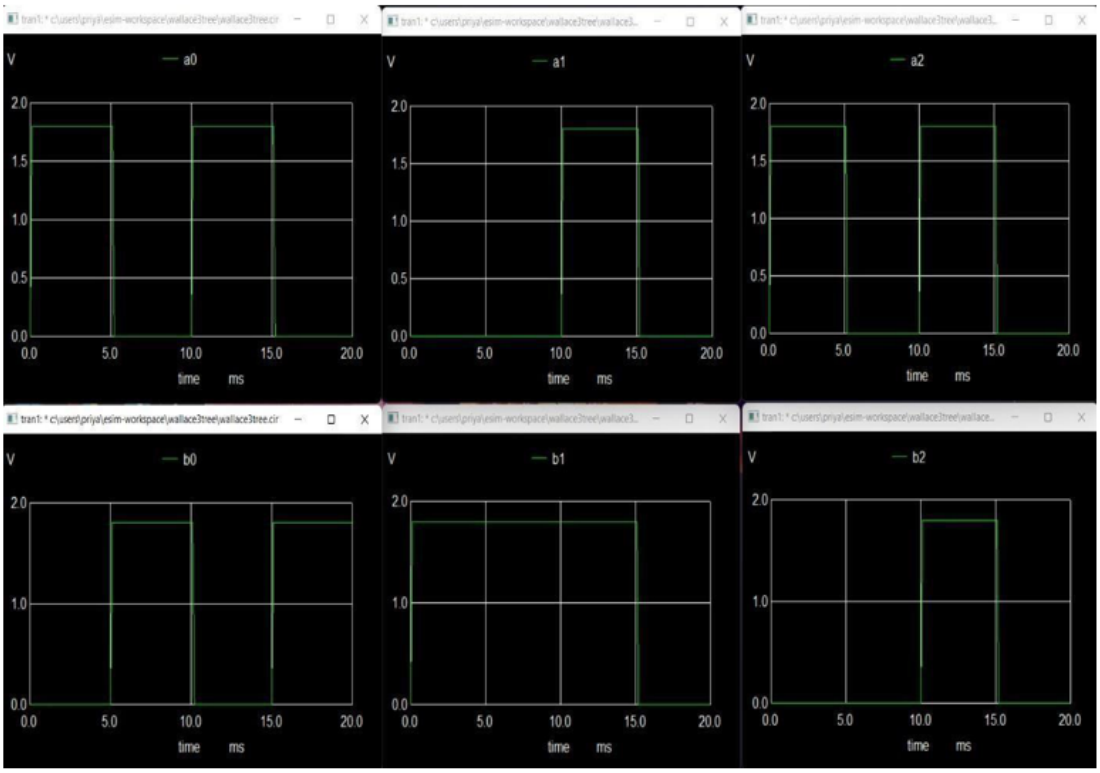
Figure 6

Obtained Output Waveforms (Fig 7)

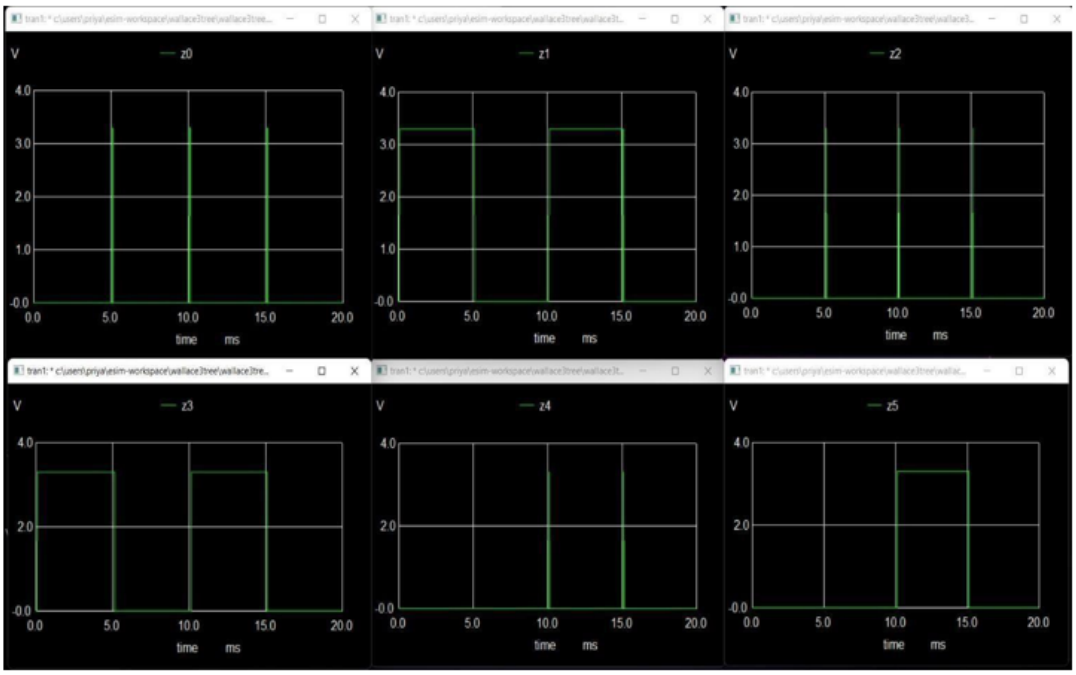
1 a0, a1, a2 and b0, b1, b2 waveforms obtained are :

ISSN 1533-9211

DOI: 10.5281/zenodo.8300481



2 z0, z1, z2, z3, z4, z5 waveforms obtained are :



The Seybold Report
ISSN 1533-9211
DOI: 10.5281/zenodo.8300481

CONCLUSION AND FUTURE WORK

It can be concluded that Wallace Multiplier is superior in all respect like speed, delay, area, complexity, power consumption. However Array Multiplier requires more power consumption and gives optimum number of components required, but delay for this multiplier is larger. Hence for low power requirement and for less delay requirement Wallace tree multiplier is suggested. This gives efficient algorithms or formulae for multiplication which increase the speed of devices. Wallace multiplier is an efficient parallel multiplier .If the multiplication is of NxN bits, then it produces N square partial product. Wallace tree multiplier or Wallace multiplier is the most popular multiplier among the existing multipliers. Wallace multiplier is also known for its fast speed and low power consumption. In multimedia and communication systems, FIR filters, digital signal processor, microprocessors etc. Many current DSP applications are targeted at portable, battery-operated systems, so that power dissipation becomes one of the primary design constraints. DSP, Image processing architectures and microprocessors. Fast Fourier Transform (FFT), Discrete and in Wavelet Transform (DWT) and autocorrelation.

References

1) G. C. Ram, D. S. Rani, R. Balasaikesava and K. B. Sindhuri, "Design of delay efficient modified 16 bit Wallace multiplier," 2016 IEEE International Conference on Recent Trends in Electronics, Information & Communication Technology (RTEICT), Bangalore, India, 2016, pp. 1887-1891, doi: 10.1109/RTEICT.2016.7808163.

2) B. C. Devnath, S. N. Biswas and M. R. Datta, "4-bit Wallace and Dadda Multiplier design using novel hybrid 3-2 Counter," 2020 2nd International Conference on Advanced Information and Communication Technology (ICAICT), Dhaka, Bangladesh, 2020, pp. 189-194, doi: 10.1109/ICAICT51780.2020.9333511.

3) H. Girish, T. G. Manjunat and A. C. Vikramathithan, "Detection and Alerting Animals in Forest using Artificial Intelligence and IoT," 2022 IEEE Fourth International Conference on Advances in Electronics, Computers and Communications (ICAECC), 2022, pp. 1-5, doi: 10.1109/ICAECC54045.2022.9716679.

4) A. Devipriya, H. Girish, V. Srinivas, N. Adi Reddy and D. Rajkiran, "RTL Design and Logic Synthesis of Traffic Light Controller for 45nm Technology," 2022 3rd International Conference for Emerging Technology (INCET), 2022, pp. 1-5, doi: 10.1109/INCET54531.2022.9824833.

5) S. Vaddadi, V. Srinivas, N. A. Reddy, G. H, R. D and A. Devipriya, "Factory Inventory Automation using Industry 4.0 Technologies," 2022 IEEE IAS Global Conference on Emerging Technologies (GlobConET), 2022, pp. 734-738, doi: 10.1109/GlobConET53749.2022.9872416.

6) T G Manjunath , A C Vikramathithan , H Girish, “Analysis of Total Harmonic Distortion and implementation of Inverter Fault Diagnosis using Artificial Neural Network”, Journal of Physics: Conference Series, Volume 2161, 1st International Conference on Artificial Intelligence, Computational Electronics and Communication System (AICECS 2021) 28-30 October 2021, Manipal, India. https://iopscience.iop.org/issue/1742-6596/2161/1

7) GIRISH H , “Internet of Things Based Heart Beat Rate Monitoring System”, © September 2022 | IJIRT | Volume 6 Issue 4 | ISSN: 2349-6002 IJIRT 156592 INTERNATIONAL JOURNAL OF INNOVATIVE RESEARCH IN TECHNOLOGY 227

8) Dr Girish H, Dr. Mangala Gowri , Dr. Keshava Murthy , Chetan Naik J, Autonomous Car Using Deep Learning and Open CV, PAGE NO: 107-115, VOLUME 8 ISSUE 10 2022, Gradiva review journal, ISSN

The Seybold Report
ISSN 1533-9211
DOI: 10.5281/zenodo.8300481

NO : 0363-8057, DOI:10.37897.GRJ.2022.V8I8.22.50332

9) Girish H. "Intelligent Traffic Tracking System Using Wi-Fi ." International Journal for Scientific Research and Development 8.12 (2021): 86-90.

10) Girish H, Shashikumar D R, “Emission monitoring system using IOT”, Dogo Rangsang Research Journal, UGC Care Group I Journal, ISSN: 347-7180, Vol – 8 Issue-14, No.6, 2020.

11) Girish H, Shashikumar D R, “Display and misson computer software loading”, International journal of engineering research & Technology (IJERT), ISSN: 2278-0181, Vol – 10 Issue-08, No.13, August 2020.

12) Girish H, Shashikumar D R, “A Novel Optimization Framework for Controlling Stabilization Issue in Design Principle of FinFET based SRAM”, International Journal of Electrical and Computer Engineering (IJECE) Vol. 9, No. 5 October 2019, pp. 4027~4034. ISSN: 2088-8708, DOI: 10.11591/ijece.v9i5.pp.4027-4034

13) Girish H, Shashikumar D R, “PAOD: a predictive approach for optimization of design in FinFET/SRAM”, International Journal of Electrical and Computer Engineering (IJECE) Vol. 9, No. 2, April 2019, pp. 960~966. ISSN: 2088-8708, DOI: 10.11591/ijece.v9i2.pp.960-966

14) Girish H, Shashikumar D R, “SOPA: Search Optimization Based Predictive Approach for Design Optimization in FinFET/SRAM”, © Springer International Publishing AG, part of Springer Nature 2019 Silhavy (Ed.): CSOC 2018, AISC 764, pp. 21–29, 2019. https://doi.org/10.1007/978-3-319-91189-2_3.

15) Girish H, Shashikumar D R, “Cost-Effective Computational Modelling of Fault Tolerant Optimization of FinFET-based SRAM Cells”, © Springer International Publishing AG 2017 R. Silhavy et al. (eds.), Cybernetics and Mathematics Applications in Intelligent Systems, Advances in Intelligent Systems and Computing 574, DOI 10.1007/978-3-319-57264-2_1.

16) Girish H, Shashikumar D R, “A Survey on the Performance Analysis of FinFET SRAM Cells for Different Technologies”, International Journal of Engineering and Advanced Technology (IJEAT) ISSN: 2249 – 8958, Volume-4 Issue-6, 2016.

17) Girish H, Shashikumar D R, “Insights of Performance Enhancement Techniques on FinFET-based SRAM Cells”, Communications on Applied Electronics (CAE) – ISSN: 2394-4714, Foundation of Computer Science FCS, New York, USA .Volume 5 – No.6, July 2016 – www.caeaccess.org

18) Girish H, Shashikumar D R, “DESIGN OF FINFET”, International Journal of Engineering Research ISSN: 2319-6890) (online), 2347-5013(print) Volume No.5 Issue: Special 5, pp: 992-1128, doi: 10.17950/ijer/v5i5/013 2016.