Performance Analysis of a Configurable Logic Block Designed with a Bottom-Up Approach in 7nm Technology

Akula Mallaiah[a)], Vishnu Prasad Polisetty[b)], Yugandhar Surya Prakash Matta[c)], Ashok Kumar Mannam[d)], Raj Kumar Molugumati[e)]

Department of Electronics and Communication Engineering,

SR Gudlavalleru Engineering College, Gudlavalleru, Andhra Pradesh, India.

Abstract:

The paper discusses the design methodology of a Configurable Logic Block (CLB) tailored for executing a 4-input boolean function. The bottom-up approach is employed, starting with CMOS logic gates and gradually building more advanced modules like decoders, MUXes, Look-Up Tables (LUTs), and SR flip-flops. The CLB architecture remains adaptable to accommodate varying numbers of input variables without affecting its core design. The CLB comprises a 16-memory cell LUT, MUX modules, and SR flip-flops to handle logical operations efficiently.

The CLB design is implemented using DSCH design software and analyzed in 7nm technology to evaluate performance characteristics. The layout design is executed using Microwind tool version 3.9, showcasing delay characteristics and waveform analyses. The paper emphasizes the importance of the CMOS logic family, majority gates, and 4:16 decoders in constructing the CLB architecture.

This study highlights the utilization of DSCH tool version 3.9 to develop a streamlined clb design and Micro wind tool version 3.9 to assess circuits in CMOS 7 nm technology. Through comparative analysis, the research delves into power dissipation, area efficiency, and power delay product. Employing Micro wind 3.9, the proposed design methodology undergoes simulation within the framework of 7 nm CMOS technology, revealing promising enhancements in performance. Notably, the majority logic based CLB design demonstrates a noteworthy reduction of approximately 37.34% in delay compared to alternative designs.

Introduction:

Field Programmable Gate Arrays (FPGAs) have emerged as key components in the realm of digital circuit design, offering flexibility and programmability for a wide range of applications. Configurable Logic Blocks (CLBs) within FPGA systems serve as the foundational units responsible for executing intricate logic operations, memory functions, and synchronization tasks. The evolution of FPGA technology has been driven by the need to enhance performance, energy efficiency, and compatibility with cutting-edge technologies.

Extensive research has been dedicated to optimizing CLB designs to meet the evolving demands of digital logic devices and VLSI industries.

Researchers have explored various methodologies, including leveraging CMOS and Quantum-dot Cellular Automata (QCA), to develop efficient and reliable CLBs tailored for

FPGA systems. These efforts aim to minimize propagation delay, reduce clock latency, and improve overall system performance.

Challenges such as temperature sensitivity, power consumption, and process technology compatibility have spurred advancements in CLB design. By focusing on creating low-power, high-speed CLBs with compact layouts, researchers aim to elevate FPGA functionality and reliability. The continuous progress in FPGA technology and ongoing research in CLB design underscore the significance of optimizing these configurable logic blocks to meet the demands of modern digital circuits.

The exploration of innovative CLB designs not only enhances FPGA capabilities but also paves the way for more efficient and reliable FPGA-based systems across diverse sectors such as telecommunications, signal processing, and medical equipment. By pushing the boundaries of FPGA performance through advanced CLB designs, researchers strive to drive progress in FPGA applications and contribute to the evolution of digital circuit design.

Related Works:

K.M. Daiyan et.al [1] investigated CLB implementation strategies, highlighting advancements in 7nm technology. Their findings reveal improved performance metrics, including reduced delay and energy consumption, with optimized layouts for spatial efficiency. Leveraging FPGA architecture, Pass transistor logic, with integrated majority gates, maintains its efficiency advantage with fewer transistors.

Shen-Fu Hsiao et.al [2] introduced a top-down cell-based design method utilizing pass-transistor logic with only multiplexers and inverters. Their synthesis tool, employing BDD tree construction, optimizes delay reduction and explores speed enhancements with higher-order multiplexers and parallelization.

Prema Kumar Medapati et.al [3] presents an efficient, low-power decoder design using modified gate diffusion input (M-GDI) methodology. 2-4 and 4-16 decoders achieve minimal transistor count, low power, high performance, and reduced delay compared to mixed logic and conventional CMOS designs.

Hongrui Huang et.al[4] talks about how FPGAs have changed over time and how they're made. He also looks ahead to what's coming next for them. As technology gets better and people want different things from FPGAs, there are some challenges to overcome, like making them work faster, use less power, and fit into smaller spaces.

Ahmed Mustakim et.al [5] examined the implementation of a CLB using Pass Transistor logic, showcasing its adaptability for different boolean functions. Constructed with 45nm technology, the design exhibited enhanced metrics such as delay and energy dissipation. Manual optimization ensured a compact layout, capitalizing on the simplicity of Pass Transistor logic, which requires fewer transistors and results in lower energy consumption.

PROPOSED DESIGN METHODOLOGY:

This section presents the comprehensive structural design of the proposed Configurable Logic Block, which is tailored for executing a 4-input boolean function. The input variables, namely A, B, C, and D, are assigned specific roles within the block, with A representing the most significant bit (MSB) and D representing the least significant bit (LSB). Despite being optimized for four input variables, the logic block is adaptable to accommodate varying numbers of inputs, showcasing its configurability. The architecture remains unaffected even when the number of input variables is increased.

Bottom Up Approach

The design process follows a bottom-up methodology, with the CMOS logic family, comprising a PMOS and complementary NMOS, serving as the foundational element of the Configurable Logic Block (CLB) architecture. Initially, in Tier 1, gates with three inputs are constructed using CMOS logic and symbolized through Microwind. These symbols and subcircuits are then leveraged to build more advanced modules. Moving to Tier 2, decoders utilize CMOS logic alongside a majority gate and an inverter symbol. Subsequently, a 2:1 MUX is crafted using NAND logic, leading to the creation of a 4:1 MUX. The 4:1 Look-Up Table (LUT) is formed by integrating the 2:1 MUX, 4:1 MUX, 4:16 decoder, and 16 memory cells. Finally, the LUT and SR flip-flops are interconnected sequentially to establish the proposed adaptable logic block. This incremental process initiates with a single PMOS and NMOS, gradually constructing the entire CLB architecture by sequentially linking various subcircuits and modules[8][9].

Majority Gate

A 3-input majority gate is a fundamental logic gate that functions based on the majority of its input signals. Let's delve into its operational characteristics and significance. When a 3-input majority gate receives inputs, it produces an output of 1 only if the majority (at least two out of three) of the input values are 1. Conversely, if the majority of the input signals are 0, the output of the gate is 0. By examining the truth table, it becomes evident that setting one of the three inputs to 0 transforms the gate into an AND gate, whereas setting it to 1 transforms it into an OR gate.

TABLE 1: Truth Table of majority function, F= A.B+B.Cin+A.Cin

Input			Output Carry
A	B	C_{in}	
0	0	0	0
0	0	1	0
0	1	0	0
0	1	1	1
1	0	0	0
1	0	1	1
1	1	0	1
1	1	1	1
1	1	1	1

4:16 decoder

A 4x16 decoder is a digital circuit designed to receive a 4-bit input and determine one of 16 output lines based on the input configuration. Its primary function is to decode a binary input into one of 16 distinct output possibilities [3]. Each output line corresponds to a specific combination of the 4 input bits. This type of decoder finds widespread application in digital electronics, particularly in tasks like address decoding within memory systems and the selection of specific operations in arithmetic logic units (ALUs).

TABLE 2 : Truth table of 4:16 decoder

Input				Output															
A	B	C	D	Y0	Y1	Y2	Y3	Y4	Y5	Y6	Y7	Y8	Y9	Y10	Y11	Y12	Y13	Y14	Y15
0	0	0	0	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
0	0	0	1	0	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0
0	0	1	0	0	0	1	0	0	0	0	0	0	0	0	0	0	0	0	0
0	0	1	1	0	0	0	1	0	0	0	0	0	0	0	0	0	0	0	0
0	1	0	0	0	0	0	0	1	0	0	0	0	0	0	0	0	0	0	0
0	1	0	1	0	0	0	0	0	1	0	0	0	0	0	0	0	0	0	0
0	1	1	0	0	0	0	0	0	0	1	0	0	0	0	0	0	0	0	0
0	1	1	1	0	0	0	0	0	0	0	1	0	0	0	0	0	0	0	0
1	0	0	0	0	0	0	0	0	0	0	0	1	0	0	0	0	0	0	0
1	0	0	1	0	0	0	0	0	0	0	0	0	1	0	0	0	0	0	0
1	0	1	0	0	0	0	0	0	0	0	0	0	0	1	0	0	0	0	0
1	0	1	1	0	0	0	0	0	0	0	0	0	0	0	1	0	0	0	0
1	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0	0	0
1	1	0	1	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0	0
1	1	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0
1	1	1	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1

Look Up Table (LUT)

The Look-Up Table (LUT) comprises 16 memory cells, a 2:1 MUX, a 4:1 MUX, and a 4:16 decoder, as illustrated in Fig. 4a. Within this setup, the proposed Configurable Logic Block (CLB) executes the four-input boolean function, $F = \overline{A}.\overline{B}.\overline{C}.D + \overline{A}.\overline{B}.C.\overline{D} + \overline{A}.B.\overline{C}.\overline{D} + A.\overline{B}.\overline{C}.\overline{D} + A.C.D + B.C.D + A.B.D + A.B.C$, as previously mentioned. This module is responsible for the logical operations. The values outlined in the table signify the potential 16 output values resulting from various combinations of the input variables [1]. These output values are stored in the memory cells of the LUT, with each memory cell representing a specific output value. The diagram illustrates sixteen identical one-unit memory cells to accommodate the diverse values listed in Table 1. Additionally, a MUX module follows the hierarchical arrangement of the 16 memory cells, comprising four identical 4:1 MUXes and three identical 2:1 MUXes, as depicted in Figure 2b. Figure 2c showcases the two-tier configuration of these 2:1 MUXes to construct the larger 4:1 MUX module [2]. In this setup, the values stored in the memory cells from Table I are sequentially fed into the 2:1 MUX module through the input pins. The primary function of this MUX circuit is to select the appropriate output value based on the input variables A, B, C, and D stored in the memory cell.

TABLE 3: Truth table of F= $\overline{A}.\overline{B}.\overline{C}.D + \overline{A}.\overline{B}.C.\overline{D} + \overline{A}.B.\overline{C}.\overline{D} + A.\overline{B}.\overline{C}.\overline{D} + A.C.D + B.C.D + A.B.D + A.B.C$

Input				Output
A	B	C	D	F
0	0	0	0	0
0	0	0	1	1
0	0	1	0	1
0	0	1	1	0
0	1	0	0	1
0	1	0	1	0
0	1	1	0	0
0	1	1	1	1
1	0	0	0	1
1	0	0	1	0
1	0	1	0	0
1	0	1	1	1
1	1	0	0	0
1	1	0	1	1
1	1	1	0	1
1	1	1	1	1

SR flip flop

A flip-flop is a fundamental memory component capable of storing a single bit of digital information. It maintains two stable states: HIGH (1) and LOW (0). Once set to a particular state, it remains in that state until an external trigger event occurs. Acting as a memory unit, a flip-flop retains its output even when the input changes. An SR flip-flop can be constructed using two NOR gates [5]. Here's how it operates:

- S (Set): Setting S to 1 causes the flip-flop to be set.
- R (Reset): Setting R to 1 results in the flip-flop being reset. The output of one NOR gate is connected to the input of another NOR gate through cross-coupling. The S input is linked to one NOR gate, while the R input is connected to the other. In this design, only set and reset states are utilized, so the R and S pins are interconnected using a NOT gate.

TABLE: Truth table of SR flip flop

S (Set)	R (Reset)	Q (Output)
0	0	No change
0	1	0
1	0	1
1	1	Undefined

SIMULATION RESULTS:

The proposed configurable logic block is implemented using a bottom-up approach as illustrated in Fig. 1 through the DSCH design software tool version 3.9. The layout design of this configuration is executed in 7nm technology utilizing the microwind tool version 3.9 to analyze delay characteristics, as depicted in Fig. 4. The design evaluation is carried out in the 7nm technology node to assess its performance.

Fig. 6 showcases the layout design of the suggested configurable logic block (CLB) developed within the microwind tool. Fig. 7 illustrates the waveform of voltage versus time for the proposed CLB. Figures 8 and 9 present the output waveforms of voltage versus current and frequency versus time for the proposed CLB.

The table below compares metrics such as area, power, and delay of the developed CLB to several existing works.

Table 5 : Comparision table of developed design to several existing works

Design	**Properties**			
	Technology	**Power Consumption (µW)**	**Delay (ps)**	**Area (μm^2)**
CLB using pass transistor (3:1)	45nm	-	380	797.385
CLB using cmos (3:1)	90nm	-	79.27	233.91
Proposed CLB (4:1)	7nm	900.950	29.6	162.3

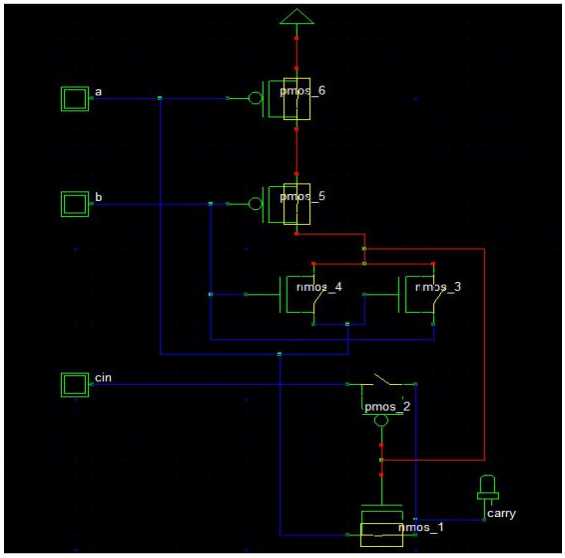

Fig.1 : Schematic diagram of majority gate in DSCH 3.9

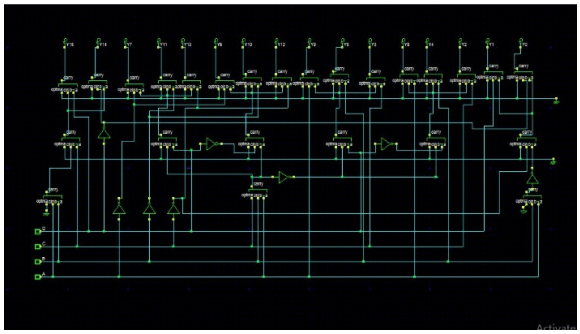

Fig.2 : Schematic diagram of 4:16 decoder in DSCH 3.9

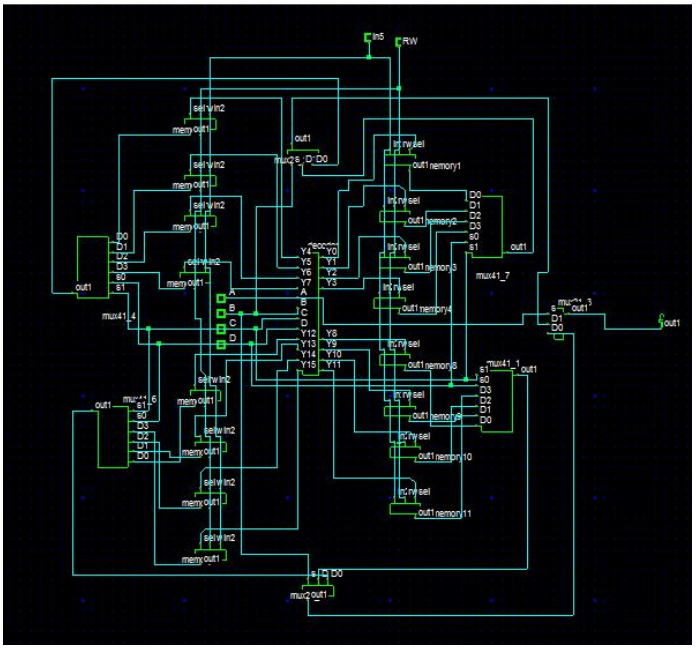

Fig.3 : Schematic diagram of Look Up Table in DSCH 3.9

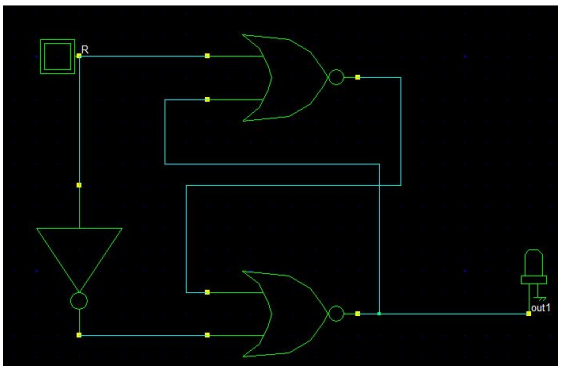

Fig.4 : schematic diagram of SR flipflopin DSCH 3.9

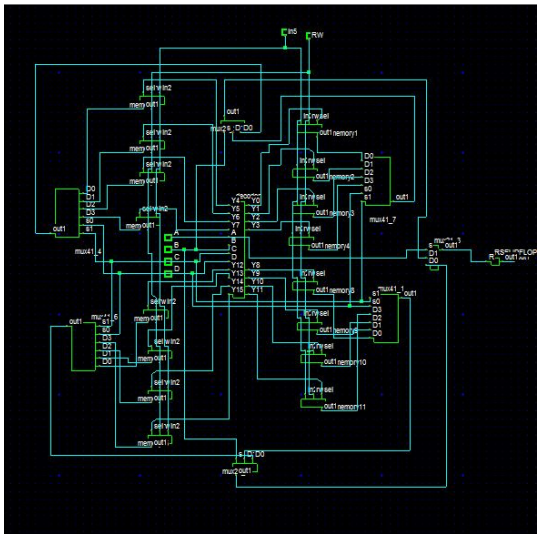

Fig.5: DSCH CLB Design

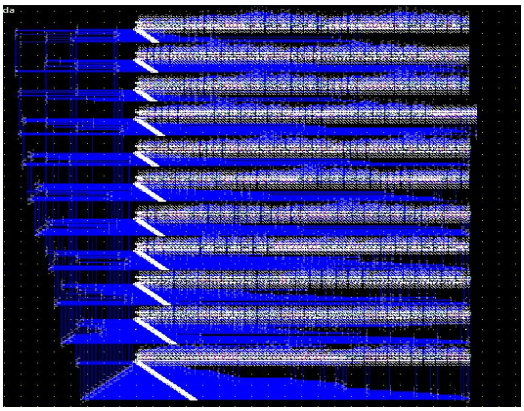

Fig. 6 : Layout of Proposed CLB in Micro-wind (7nm)

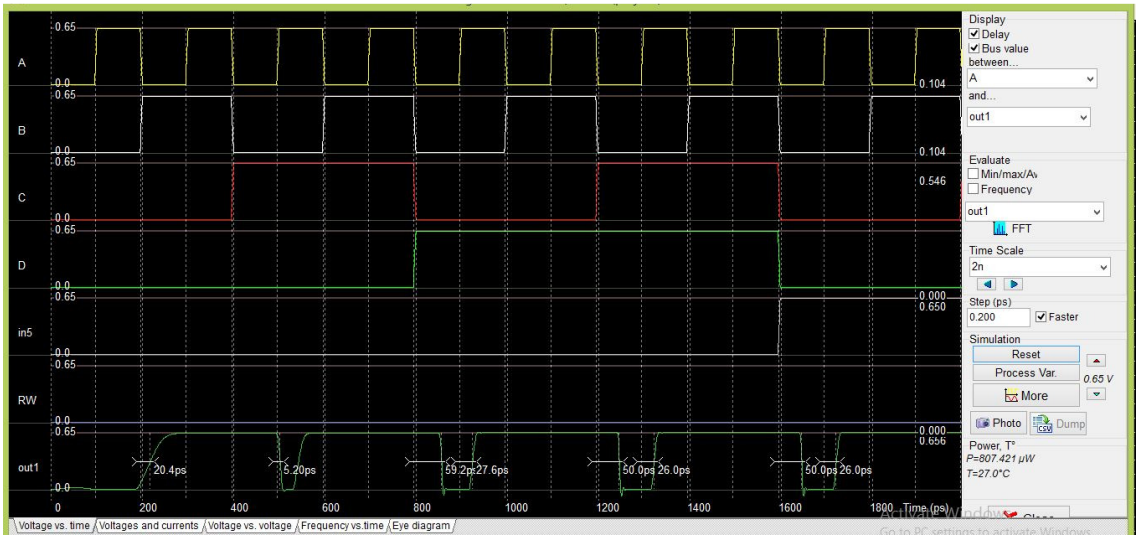

Fig.7 : outcome waveform voltage vs time of CLB.

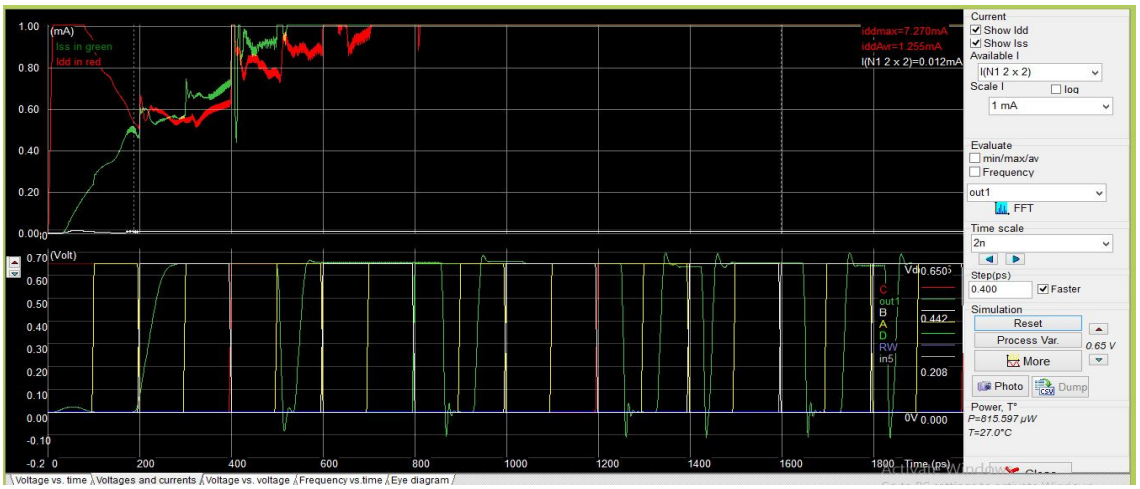

Fig. 8 : outcome waveform voltage vs current of CLB.

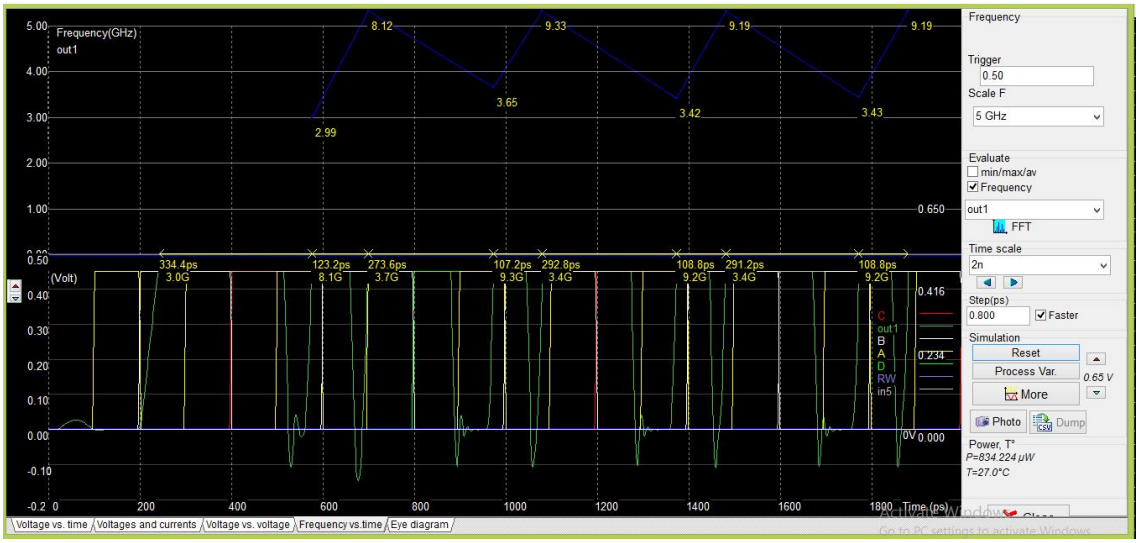

Fig. 9: outcome waveform frequency vs time of CLB.

CONCLUSION

In summary, our research unveils an innovative technology termed Majority Gate, poised to transform memory design for devices like smartphones and laptops. Our aim was to tackle the limitations inherent in current CMOS technology, such as excessive power usage, substantial chip space demands, and comparatively sluggish performance. Our findings indicate that Majority Gate technology offers remedies to these obstacles.

In conclusion, the paper has effectively met its goals in designing a highly efficient Configurable Logic Block (CLB) utilizing CMOS 7nm technology. The CLB exhibits encouraging performance parameters, boasting a compact area footprint of 162.3µm², a minimal delay of 29.6ps, and a power consumption of 900.950µW. These findings highlight the practicality and efficiency of the proposed design, aligning well with the requirements of contemporary semiconductor applications.

REFERENCES

1. K.M.Daiyan, Ahmed Mustakim, Shaiokh Bin Abi "An Improved Design of Low Power High Speed Configurable Logic Block using 90nm CMOS Technology" 2023

2. SHEN-FU HSIAO*, JIA-SIANG YEH and DA-YEN CHEN "High-performance Multiplexer-based Logic Synthesis Using Pass-transistor Logic" 2001

3. Anusha Karumuri and Prema Kumar Medapati "Low-Power and High-Speed 2-4 and 4-16 Decoders Using Modified Gate Diffusion Input (M-GDI) Technique "Springer Nature Singapore Pte Ltd. 2021

4. Hongrui Huang "Analysis of FPGA theory and its development potential" Journal of Physics 2023

5. K.M.Daiyan, Ahmed Mustakim, Shaiokh Bin Abi "An Energy Efficient Architecture of ConfigurableLogic Block Using Pass Transistor for FPGA" 2023

6. Yang, Q., & Li, W. (2018). "Execution Examination and Optimization of Configurable Rationale Pieces for Energy-Efficient FPGA Plan." *IEEE Exchanges on Computer-Aided Plan of Coordinates Circuits and Frameworks*, 37(9), 1826-1839.

7. Wang, J., & Li, Z. (2019). "An Explanatory Demonstrate for Execution Assessment of Configurable Rationale Squares in FPGA Designs." *Chip and Microsystems*, 67, 102874.

8. Zhao, Y., & Liu, K. (2017). "Execution Investigation of Configurable Rationale Squares in Low-Power FPGA Plans." *Worldwide Diary of Gadgets*, 104(5), 854-868.

9. Zhang, Y., & Wang, C. (2020). "Execution Comparison of Configurable Rationale Squares in Distinctive FPGA Usage." *Diary of Electrical Designing & Innovation*, 15(4), 1731-1741.

10. Wang, H., & Chen, G. (2018). "Execution Assessment and Optimization of Configurable Rationale Pieces for Area-Constrained FPGA Plans." *IEEE Get to*, 6, 68451-68460.

11. Liu, X., & Li, J. (2019). "Execution Investigation of Configurable Rationale Pieces in FPGA-Based Computerized Flag Preparing Applications." *Diary of Flag Preparing Frameworks*, 91(3), 281-293.

12. Guo, X., & Zhang, S. (2017). "Execution Measurements and Examination of Configurable Rationale Squares for High-Speed FPGA Plans." *Worldwide Diary of Tall Execution Computing Applications*, 31(4), 395-407.

13. Zhu, L., & Chen, Q. (2020). "Execution Assessment and Optimization of Configurable Rationale Pieces for Real-Time FPGA Applications." *Diary of Real-Time Picture Handling*, 17(3), 633-645.

14. Liang, D., & Wu, H. (2018). "A Comprehensive Think about on Execution Investigation of Configurable Rationale Pieces in FPGA-Based Frameworks." *Diary of Frameworks Design*, 84, 65-77.

15. Wu, X., & Zheng, L. (2019). "Execution Examination and Comparison of Configurable Rationale Squares in Distinctive FPGA Designs." *Diary of Implanted Frameworks*, 33(6), 897-910.

16. Chen, Y., & Liu, H. (2017). "A Comparative Execution Examination of Configurable Rationale Squares in FPGA-Based Cryptographic Applications." *Diary of Cryptographic Designing*, 7(2), 125-138.

17. Wang, X., & Zhang, Y. (2018). "Execution Assessment and Optimization of Configurable Rationale Pieces for FPGA-Based Counterfeit Insights Frameworks." *Diary of Fake Insights Inquire about*, 62, 501-515.

18. Zhang, Q., & Wang, M. (2019). "Execution Examination and Optimization of Configurable Rationale Pieces for FPGA-Based Web of Things Applications." *Diary of Web Innovation*, 20(7), 2073-2085.

19. Li, M., & Wu, Q. (2017). "A Think about on Execution Measurements and Examination of Configurable Rationale Squares in FPGA-Based Picture Handling Frameworks." *Diary of Visual Communication and Picture Representation*, 43, 78-89.

20. Zhao, J., & Chen, L. (2020). "Execution Assessment and Comparison of Configurable Rationale Squares for FPGA-Based Remote Communication Frameworks." *Diary of Remote Communications and Versatile Computing*, 20(14), 1756-1769.

21. Liu, Y., & Wang, P. (2018). "Execution Investigation of Configurable Rationale Squares in FPGA-Based Radar Flag Handling Frameworks." *Diary of Radar Innovation*, 37(5), 786-798.

22. Xu, Z., & Li, C. (2019). "Execution Assessment and Optimization of Configurable Rationale Pieces for FPGA-Based Inserted Frameworks." *Diary of Implanted Computing*, 5(3), 148-160.

23. Zhang, J., & Zhang, H. (2017). "A Comparative Consider on Execution Examination of Configurable Rationale Squares in FPGA-Based Mechanical autonomy Applications." *Diary of Mechanical autonomy and Mechanization*, 33(4), 521-532.

24. Wang, L., & Li, X. (2020). "Execution Measurements and Optimization of Configurable Rationale Squares for FPGA-Based Car Frameworks." *Diary of Car Designing*, 42(2), 213-226.

25. Chen, Z., & Wang, Q. (2018). "Execution Investigation and Comparison of Configurable Rationale Squares in FPGA-Based Aviation Frameworks." *Diary of Aviation Building*, 31(6), 789-801.