Journal of Engineering Science and Technology Review 17 (5) (2024) 104-109

Research Article

JOURNAL OF
Engineering Science and Technology Review

www.jestr.org

A Modified 2:1 Multiplexer-Based Low Power Ternary ALU for IoT Applications

S. Allwin Devaraj[1,*], D. Magdalin Mary[2], P. Kannan[1], S. Esakki Rajavel[3], Cynthia Anbuselvi Thangaraj[4], K. B. Gurumoorthy[5] and Blanie Scrimshaw William[6]

[1]*Department of Electronics and Communication Engineering, Francis Xavier Engineering College, Tirunelveli-627003, Tamilnadu, India.*
[2]*Department of Electrical and Electronics Engineering, Sri Krishna College of Technology, Coimbatore-641042, Tamilnadu, India.*
[3]*Department of Electronics and Communication Engineering, Karpagam Academy of Higher Education, Coimbatore- 641021, Tamilnadu, India.*
[4]*Department of Electronics and Communication Engineering, SEA College of Engineering and Technology, Bengaluru-560049, Karnataka, India.*
[5]*Department of Electronics and Communication Engineering, KPR Institute of Engineering and Technology, Coimbatore-641407, Tamilnadu, India.*
[6]*Department of Computer Science and Engineering, Rohini College of Engineering and Technology, Kanyakumari-629401, Tamilnadu, India.*

Received 27 March 2024; Accepted 13 October 2024

Abstract

The ternary logic has a benefit over the binary logic which provides a secured solution to achieve a trade-off between the area and power of the design. However, from the structure of the ternary Aritmetic Logic Unit (ALU), it is clear that its architecture increases the area, propagation delay, and power consumption. To overcome this drawback, a loopback algorithm is proposed to achieve low power and high throughput Internet of Things (IoT) processors. The loopback algorithm reduces the number of processing stages in multipliers and adders which can significantly reduce area and power dissipation. The proposed 2:1 multiplexer-based approach reduces the need for a decoder and results in low power consumption. The proposed design will be implemented in Xilinx ISE 13.0 and simulation will be done in Modelsim. The modified Ternary ALU (TALU) performs finer than the previous TALU method. The number of registers used in this architecture is reduced by up to 25% than the existing system therefore there is a reduction in power dissipation.

Keywords: TALU, OR, EXOR, Multiplexer, Delay, Power Consumption.

1. Introduction

Digital signal processor plays a significant role in electronic devices, biomedical applications, communication protocols, LTE devices, etc [1-3]. Efficient IC design is a key factor to achieve low power and high throughput IP core development for portable and LPD [4]. Internet of Things plays a significant role in real-time computing and processing [5-7]. Now that every object can be connected to the internet. These devices range from ordinary household objects to industrial tools but area overhead and power consumption are major drawbacks to achieving efficient design constraints. In modern society, the most important component used in the electronic system is Integrated Circuits (IC) [8-10]. The binary logic is widely used due to its accuracy and user friendly but in recent years its performance become slow and there is a reduction in scalability to overcome this multi-valued logic (MVL) is being researched due to the reduction in area and power [11]. The digital operation is performed better in ternary logic rather than the binary logic. The value (0,1,2) are the three logic levels of ternary logic that refer to the voltage of 0, Vdd/2, and Vdd. Due to the usage of 3 valued logic tons of data can be effortlessly shifted with a less amount of devices. The complexity of the ternary logic is the main reason for the lesser usage of the design to make it use widely lots of analysis is being done [12-14].

In an earlier paper, based on the pipelined technique the TALU designs are built. It makes the design more complex because of the usage of area and the processing stages. To overcome the drawbacks in the previous works the proposed design uses the loopback algorithm so that there will be a reduction in area and consumption of power and also reduces the interconnection and the computational costs [15-16]. The loopback technique stores the data in the memory and gives it as output when it is needed so there is a reduction in the processing stages than the previous method. It uses a 2:1 multiplexer-based technique so there will be a reduction in the decoder and to make the arithmetic circuit implement effectively [17-18].

The paper is arranged as given below. The existing Pipelined TALU Design is explained in section 2. Section 3 presents the Modified TALU design and implementation and functions in detail. In section, 4 results are discussed and the conclusion is discussed in Section 5.

2. Existing Pipelined TALU Design

In the existing method, the TALU design is built based on the pipelined technique with a 2:1 multiplexer-based approach. Figure (1) shows the block diagram of the existing pipelined method. The component in the pipelined TALU design is the function processing module which consists of 9 operations. And function selection lines S1 and S0 are used to select the operation which has to be given as an output

*E-mail address: babu.allwin@gmail.com
ISSN: 1791-2377
doi:10.25103/jestr.175.14

and then the main unit is the pipelined stage which is used to reduce the delay at last the component is a multiplexer module. In this architecture, the 9:1 multiplexer uses 9 inputs for different operations. It receives 9 inputs and sends one output. Here 2-digit multiplexer is used to design the architecture of the TALU. A and B are considered as input with 2-digit values and 2-digit Y act as an output. The usage of the pipelined technique reduces delay but it uses a large number of processing stages and lots of registers for processing so that it consumes lots of areas and it consumes tons of power due to that the computational cost will be high. It consists of lots of interconnections.

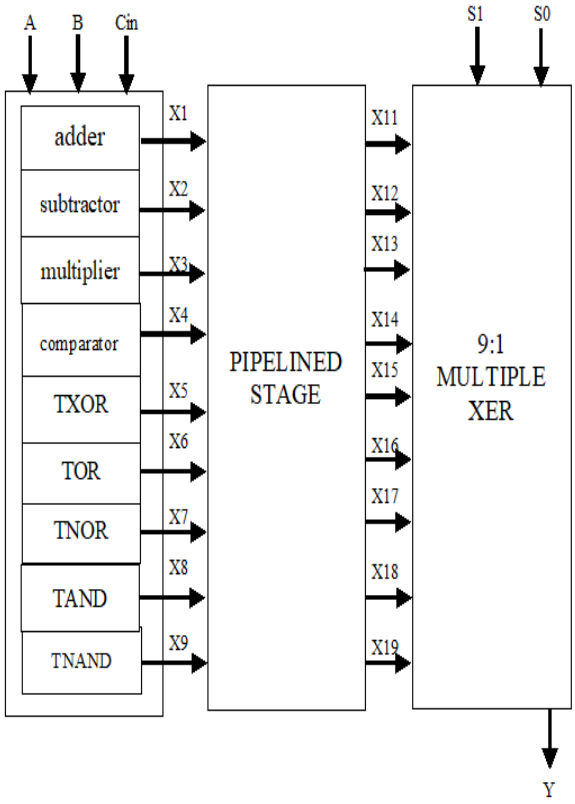


Fig. 1. Block diagram of existing pipelined ALU

3. Modified TALU Design (2-digit)

3.1 Modified 2-digit TALU Design and implementation

A modified 2-digit TALU is established in this section. Each module is designed using the 2-digit multiplexer. The inputs are 2-digit A and B and also have 2-digit output Y. Figure (2) is the block diagram of the modified TALU. The selection lines are S1 and S0 which select the operation within the 9 different operations. The processing module processes the operation such as logic and arithmetic and gets the output. As shown in Figure (2) the three main blocks in the modified TALU are a functional processing module, a functional selection module, and a 9:1 multiplexer. The drawbacks of the previous method can be overcome by the loopback algorithm. The loopback algorithm is used to reduce the number of processing stages in the architecture so there will be a reduction in the number of chips. It reduces the area of the TALU design and the power consumption of the design will be low. It stores the outputs in the memory and loopback the output when the input needs the correct output.

Here the module designs are designed by the 2:1 multiplexer therefore the processor in this proposed will be effective and also it reduces the area of the architecture. The functional processing module consists of the arithmetic and logical operations and processes them to get the output related to the given inputs. The output Y will be selected by the selection block. The truth table of the TALU is shown in Table 1 where all the operations are described and used to verify which operation has been done. It consists of less number of interconnections so there is a lesser number of area. The processing module is directly connected to the 9:1 multiplexer to get the final output.

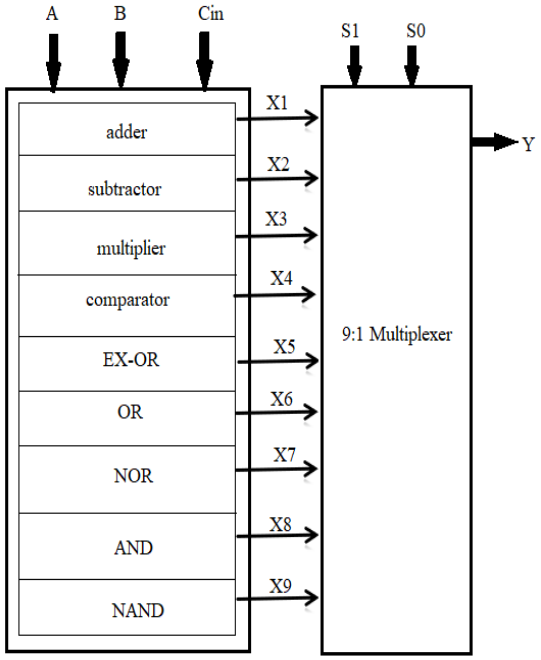


Fig. 2. Block diagram of modified TALU

Table 1. TALU Truth Table

S_0	S_1	Operations
2	2	NAND
1	2	AND
0	2	NOR
2	1	OR
1	1	EXOR
0	1	Comparison
2	0	Multiplication
1	0	Subtraction
0	0	Addition

The block diagram of the function selection line (FSL) is shown in Figure (3). The block diagram of the FSL is the same as the previous paper which is referred to in [2]. Here the decoder is used only in the selection line The inputs of the selection lines S1 and S2 have 3 input values 0,1,2 for each. It has nine different operations consisting of logic and arithmetic units. Based on the selection line the operations which has be done will be selected. Based on Table 1 the operation which has to be performed can be verified. The multiplexer module is connected to the processing modules. The processing module has 9 different operations. It receives input A, B, and C_{in}. Based on the given input each operation gives its outputs. The outputs are connected to the multiplexer where the selection line is inserted. Based on the selection lines given in the multiplexer the specific operation in which output is needed is given as an output Y. Here loopback algorithm is being used so the performance of the applications will be effective.

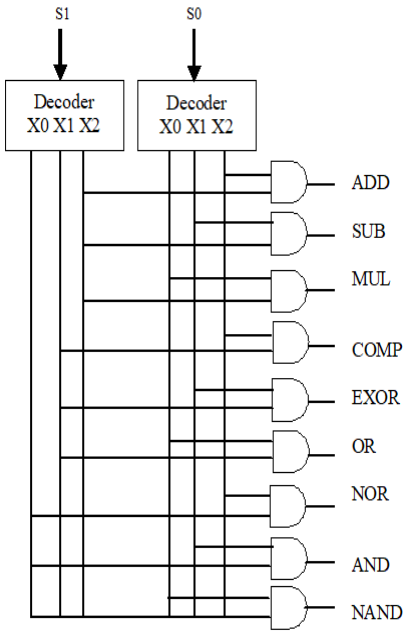


Fig. 3. Block diagram of FSL

The previous pipelined design uses the pipelining stages so there will be a lot of processing stages. These stages occupy a large amount of area and there will be an increase in power consumption to reduce the area loopback algorithm is used. Here 2:1 multiplexer is used to design each module so the applications will be done effectively and with less complexity. The unary operators are the basis for the multiplexer-based approach. Table 2 shows the values of the unary operators. The operators in Table 2 are the main operators in the modified TALU design.

Table 2. Unary Operators

A	0	1	2
A^0	2	0	0
A^2	0	0	2
$1.A^2$	0	0	1
A^{+2}	2	0	1
A^{+1}	1	2	0
$\overline{A^0}$	0	2	2
$\overline{A^2}$	2	2	0
$\overline{1.A^0}$	0	1	1

1) Adder and subtractor Module: Based on the 2:1 multiplexer approach the adder and subtractor module is designed using the loopback algorithm. Figure 4 shows the block diagram of the adder subtractor module. Here 2 digits A and B are given as input for both the HAS and FAS. The output will be a 2-digit sum, carry and difference, borrow B_0, B_1. Then the output in the HAS will be given to FAS as an input. M in this module acts as a controller because it chooses whether the module should execute an adder or subtractor. If the adder module is to be performed then the M value should be 0, if it has to perform subtraction the M value should be 2. For addition operation the controller M=0.

At first, the HAS output of the addition operation is

$$\text{SUM: } S = B^0.A + B^1.A^{+1} + B^2.A^{+2} \quad (1)$$

$$\text{CARRY: } C = B^0 + B^1.(1.A^2) + B^2(1.A^0) \quad (2)$$

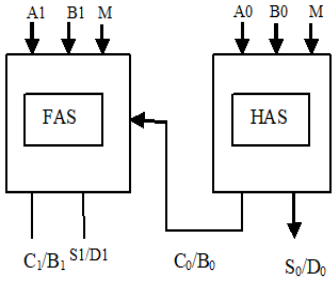


Fig. 4. Block diagram- Adder-subtractor module

From the above equation (1), (2) B which also includes B^0, B^1, and B^2 are the selection lines and $A^{+1,}$ A^{+2}, $(1.A^2)$, $(1.A^0)$ are the unary operators its value are expressed in Table 2. The addition operation equations are further transformed to get the accurate 2-digit multiplexer-based design. The transformed equations are (3), and (4) These Sum and Carry from equations (3), and (4).

$$S = B^0.A + \vec{B^0}(\vec{B^2}.A^{+1} + B^2.A^{+2}) \quad (3)$$

$$C = B^0.0 + \vec{B^0}(\vec{B^2}.(1.A^2) + B^2(1.\vec{A^0})) \quad (4)$$

Same as the adder operation the subtraction operation will also be done. For subtraction, the value of the controller M will be 2. The HAS output for subtraction is

$$D = B^0.A + B^1.A^{+2} + B^2. \quad (5)$$

$$B = B^0.0 + B^1.(1.A^0) + B^2(1.\vec{A^2}) \quad (6)$$

From the above equation (5), (6) B which also includes B^0, B^1, and B^2 are the selection lines and $A^{+1,}$ A, $(1.A^2)$, $(1.A^0)$ are the unary operators its value are expressed in Table 2. The above equation (5), and (6) is transformed into (7), and (8) equation to get an accurate design. The output will be of difference and borrow.

$$D = B^0.A + \vec{B^0}(\vec{B^2}.A^{+2} + B^2.A^{+1}) \quad (7)$$

$$B = B^0.0 + \vec{B^0}(\vec{B^2}.(1.A^0) + B^2(1.\vec{A^2})) \quad (8)$$

The execution of FAS is similar to the pattern of HAS. The output of the HAS goes to the FAS. Here selection line is B and the values related to B and $A^{+1,}$ A^{+2}, $(1.A^2)$, and $(1.A^0)$ are the unary operators. Table 3 presents the truth table of the FAS. Table 3 shows the values for adder operation and the values for subtraction. Based on the truth table the output of addition and subtraction operations can be verified.

Table 3. FAS Truth Table

	Sum			**Carry**		
A/B	**0**	**1**	**2**	**0**	**1**	**2**
0	1	2	0	0	0	1
1	2	0	1	0	1	1
2	0	1	2	1	1	1
	Difference			**Borrow**		
A/B	**0**	**1**	**2**	**0**	**1**	**2**
0	0	2	1	0	1	1
1	1	0	2	0	0	1
2	2	1	0	0	0	0

When the controller M value is 0 then the addition operation of FAS will be done. Equations (9), and (10) show the sum and carry of FAS:

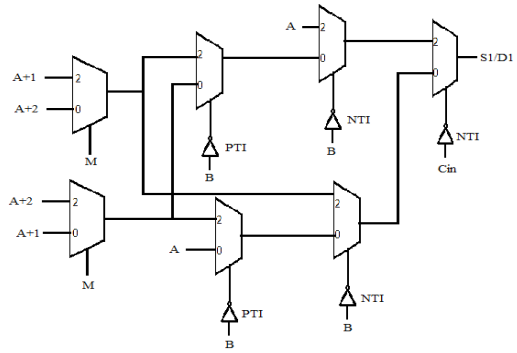
$$S = C_{in}^{0}.f1 + \overrightarrow{C_{in}^{0}}.f2 \quad (9)$$

$$C = C_{in}^{0}.f3 + \overrightarrow{C_{in}^{0}}.f4 \quad (10)$$

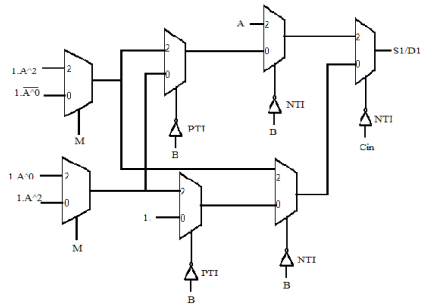
When the controller M value is 2 then the subtraction operation of FAS is done. Equations (11), and (12) show the difference and borrow of FAS:

$$D = B_{in}^{0}.f1 + \overrightarrow{B_{in}^{0}}.f2 \quad (11)$$

$$B = B_{in}^{0}.f3 + \overrightarrow{B_{in}^{0}}.f4 \quad (12)$$



(a) Design -Sum or difference



(b) Design - carry or borrow

Fig. 5. 2 digit - adder subtractor design

The above figure 5 shows the design of the adder and subtractor module. The design of the module is referred from the reference paper [1]. The module is designed with the 2-digit multiplexer which makes the design less complex and makes the modules work effectively.

2) Multiplier Module: In this module multiplication operation will be done. The design of the module is referred from the reference paper [1]. Here the value A and B act as input. The module is designed with the 2-digit multiplexer. The equations (13), and (14) show the output of the product and carry.

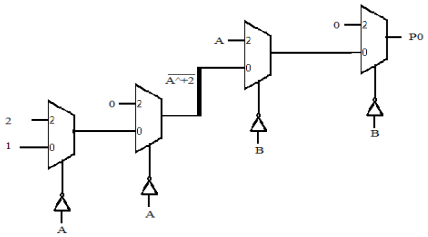


Fig. 6. (2-digit) Multiplier design

$$P = B^{0}.0 + B^{1}(A) + B^{2}.(A^{\overrightarrow{+2}}) \quad (13)$$

$$C = B^{0}.0 + B^{2}.(1.A^{2}) \quad (14)$$

Here the value ($1.A^2$), and A^{+2} are the unary operators of this module. Figure (6) shows the design of the multiplier module which is done based on the equation (15), and (16). The 2-digit multiplier design is done using the loopback algorithm. By using this the module can work effectively.

$$P = B^{0}.0 + B^{\vec{0}}(B^{\vec{2}}.A) + B^{2}.(A^{\overrightarrow{+2}}) \quad (15)$$

$$C = B^{0}.0 + B^{2}.(1.A^{2}) \quad (16)$$

3) Comparator Module: This module shows the design of the comparator which is designed using the 2-digit multiplexer. It usually compares the two input values A and B whether it is greater or lesser or equal. The value of g and l are shown in the equation (19) and (20) using a 2:1 multiplexer. Based on these two equations Figure (7) is designed referred by paper [1]. Equation (17) and (18) are the equation that shows whether the input is greater or lesser.

$$A > B = g_1 + g_0 \,.l_1 \quad (17)$$

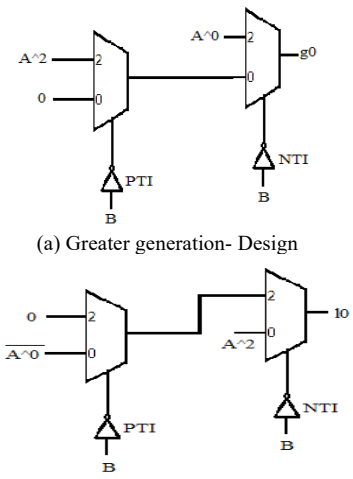
$$A < B = l_0 + g_1 \,.l_0 \quad (18)$$

The value of g_1,g_0, l_1, and l_0 are given in the below equation (19) and (20) where g_1, and g_0 is referred to as g, and l_1 and l_0 are referred to as l.

$$g = B^{0}.A^{0} + B^{\vec{0}}(B^{\vec{2}}.B^{2} + B^{2}.0) \quad (19)$$

$$l = B^{2}.A^{\vec{2}} + B^{\vec{2}}(B^{\vec{2}}.A^{\vec{0}}) + B^{0}.0) \quad (20)$$

The selection line of the comparator module is B. The values A^0, A^2, A^0, and A^2 are the unary operators. The 2-digit multiplier in this design is done using the loopback algorithm. By using this the module can work effectively.



(a) Greater generation- Design

(b) Lesser generation- Design

Fig. 7. 2-digit- Comparator design

4) Multiplexer Module: Here in this module design of the multiplexer is described. Figure (8) is the design of the multiplexer. It mainly uses a 9:1 multiplexer because 9 different operations are done. It also shows the working of the selection line which is used to select the operations. The

multiplexer module is designed by a 3:1 multiplexer. It is of two stages. In the first stage S_0 act as a selection line which gives their output to the next stage where S_1 acts as a selection line according to the operation which has to be done. By the given inputs in the S_0 and S_1, the output Y will be given according to the chosen operation

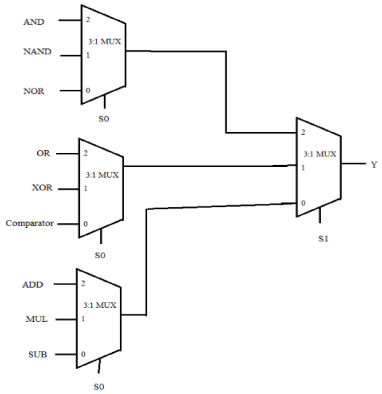


Fig. 8. Multiplexer- Design

4. Result and Discussion

4.1 Software Used

In the proposed modified TALU design the software used is Xilinx ISE 13.2 because it is easy to handle and gives the result accurately. This algorithm makes the design run fast more than 30% which makes the project cost-effective and can be done in less time. Here the coding used for the project is Verilog. Verilog code is user-friendly and can be used by many because it is understandable. The simulation result in this TALU consumes less amount of voltage and also it consumes lesser power. The power dissipation will also be lesser compared to the other.

4.2 Comparison and Simulation of modified TALU

The simulation waveform of the arithmetic and logical operation is presented in this section and the comparison of the modified and the existing pipelined TALU is given. The comparison table in Table 4 shows the difference in the area, delay, and power consumption of both TALU designs. The final simulation waveform of the modified TALU design is shown in Figure (11). The output of the TALU can be verified in Table 1.

The waveform of the subtractor which is one of the arithmetic operations is shown in Figure (9) and one of the logical operations NAND is shown in Figure (10) is given for reference. Based on the truth table of their ternary operation the output can be determined. The input is of ternary value (0,1,2) and the output will also be in ternary value and the interconnections between the two modules are considered as wire X. There are lesser interconnections compared to the previous work. There is also an improvement in the area and power consumption. The computational costs will also be reduced because there is a reduction in the consumption of the area. Due to low power consumption, it can perform effectively in IoT applications. In figure (9) the subtraction operation is shown based on the ternary input value the output will be obtained. This can be verified by Table 2 likewise the operation NAND is shown in Figure (10) is done. The modified TALU design simulated waveform is shown in Figure (11). The value a, b, S1, S0, Cin, and clk are the inputs of the TALU simulation the value Y acts as an output, The wire value X {X1, X2, X3,… X9} are the interconnections between the processing module and the 9:1 multiplexer. The S1 and S0 are the selection lines based on the values in the selection line the operation which has to be performed will process and get an output Y. Here loop-back algorithm is used so there is a lot of reduction in the processing module and makes the process done effectively. The loopback algorithm is used to reduce the number of chips, and the area is reduced in the modified TALU.

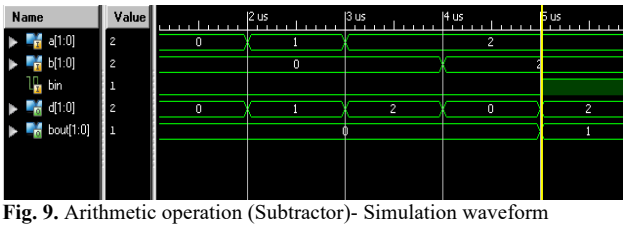


Fig. 9. Arithmetic operation (Subtractor)- Simulation waveform

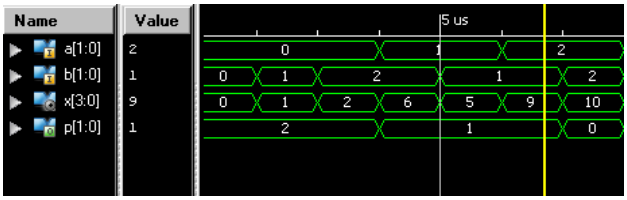


Fig. 10. Logical operation (NAND)- Simulation waveform

Table 4. Comparisons Table

Parameters	Number of slice registers used	Delay(ns)	Power(W)
Existing Pielined TALU	52	0.678	1.110
Modified TALU	21	1.046	1.109

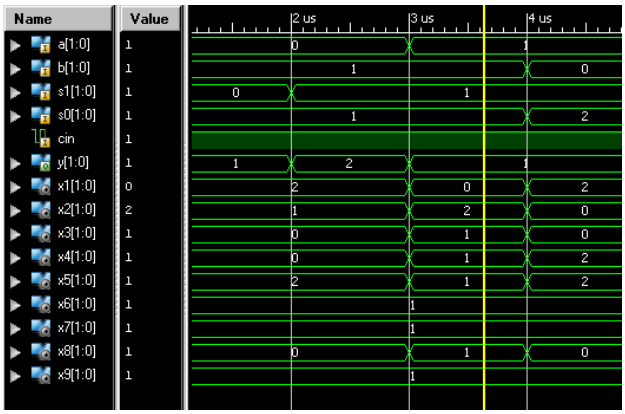


Fig. 11. Modified TALU design- Simulation waveform

5. Conclusion

The modified TALU is proposed by the loopback algorithm to make the design suitable for the applications to work fast and to make it cost-effective. The main components in the modified TALU are the function selection module, function processing modules, and 9:1 multiplexer. Due to this loopback algorithm, the processing stages reduce therefore the number of chips and registers is reduced. Due to this, there consumes a small amount of area. There is a reduction in power consumption because it stores the values in memory which makes the applications run effectively. Using

Xilinx ISE 13.2 the designs and simulation of modified TALU are obtained because it is user-friendly. The modified TALU performance will reduce the area and power consumption comparing the previous pipelined work. Here there is a reduction in area reduction in power consumption and power leakage and there is a reduction in computational costs. The proposed modified TALU design attain a delay up to 1.046 ns, the number of chips used in this TALU is 21 which is reduced up to 50% than existing, and the value of power consumption up to 0.109 W.

References

[1] S. Gadgil and C. Vudadha, "Design of CNTFET-Based Ternary ALU Using 2:1 Multiplexer Based Approach," *IEEE Transact. Nanotechn.*, vol. 19, no. 1, pp. 661–671, Aug. 2020.

[2] A. P. Dhande and V. T. Ingole, "Design and Implementation Of 2 Bit Ternary ALU Slice," in *3rd Int. Conf.: Sci. Electr., Technol. Inform. Telecommun.*, Tunisa, Jan. 2005, pp. 1-11.

[3] Chetan Vudadha, S. Katragadda, and P. Sai Phaneendra, "2:1 Multiplexer based design for ternary logic circuits," *IEEE Asia Pacific Conf. Postgrad. Res. Microelectron. Electron.* (PrimeAsia), Visakhapatnam, Dec. 2013, pp. 46-51.

[4] Chetan Vudadha, Sai Phaneendra Parlapalli, and M. B. Srinivas, "Energy efficient design of CNFET-based multi-digit ternary adders," *Microelectr. J.,* vol. 75, no.5, pp. 75–86, May 2018.

[5] G. Hills et al., "Understanding Energy Efficiency Benefits of Carbon Nanotube Field-Effect Transistors for Digital VLSI," *IEEE Transact. Nanotechn.*, vol. 17, no. 6, pp. 1259–1269, Sep. 2018.

[6] G. Hills et al., "Modern microprocessor built from complementary carbon nanotube transistors," *Nature*, vol. 572, no. 7771, pp. 595–602, Aug. 2019.

[7] S. L. Murotiya and A. Gupta, "Design of CNTFET-based 2-bit ternary ALU for nanoelectronics," *Int. J. Electr.*, vol. 101, no. 9, pp. 1244–1257, Aug. 2013.

[8] G. Thakur, H. Sohal, and S. Jain, "Design and Analysis of Low Power Approximate Multiplier Using Novel Compressor," *SN Comp. Sci.*, vol. 5, no. 5, Apr. 2024.

[9] J. Deng and H.-S., P. Wong, "A Compact SPICE Model for Carbon-Nanotube Field-Effect Transistors Including Nonidealities and Its Application—Part I: Model of the Intrinsic Channel Region," *IEEE Trans. Electron Dev.*, vol. 54, no. 12, pp. 3186–3194, Dec. 2007.

[10] Subhendu Kumar Sahoo, K. Dhoot, and R. Sahoo, "High Performance Ternary Multiplier Using CNTFET," *IEEE Comp. Soc. Annual Sympos. VLSI*, Hong Kong, Jul. 2018, pp. 269-275.

[11] M. M. Ghanatghestani, B. Ghavami, and H. Pedram, "A Ternary Full Adder Cell Based on Carbon Nanotube FET for High-Speed Arithmetic Units," *J. Nanoelectr. Optoelectr.*, vol. 13, no. 3, pp. 368–377, Mar. 2018.

[12] S. L. Murotiya and A. Gupta, "Hardware-efficient low-power 2-bit ternary ALU design in CNTFET technology," *Int. J. Electr.*, vol. 103, no. 5, pp. 913–927, May. 2015.

[13] C. J. Williams et al., "MolProbity: More and better reference data for improved all-atom structure validation," *Protein Sci.*, vol. 27, no. 1, pp. 293–315, Nov. 2017.

[14] I. Takahashi and T. Noguchi, "A New Quick-Response and High-Efficiency Control Strategy of an Induction Motor," *IEEE Transact. Industry Applic.,* vol. IA-22, no. 5, pp. 820–827, Sep. 1986.

[15] R. Lorenzo and R. Pailly, "Single bit-line 11T SRAM cell for low power and improved stability," *IET Comp. Dig. Techniq.,* vol. 14, no. 3, pp. 114–121, Mar. 2020.

[16] S. Pal, V. Gupta, W. H. Ki, and A. Islam, "Transmission gate-based 9T SRAM cell for variation resilient low power and reliable internet of things applications," *IET Circ., Dev. & Sys.*, vol. 13, no. 5, pp. 584–595, Aug. 2019.

[17] Y. Lu, L. Li, H. Peng, and Y. Yang, "An Energy Efficient Mutual Authentication and Key Agreement Scheme Preserving Anonymity for Wireless Sensor Networks," *Sensors*, vol. 16, no. 6, p. 837, Jun. 2016.

[18] N. S. Bhat, "Design and Modelling of Different SRAM'S Based on CNTFET 32NM Technology," *Int. J. VLSI Des. Commun. Sys.*, vol. 3, no. 1, pp. 69–83, Feb. 2012.