Design An ALU Capable of Calculating Definite Integrals of Simple Polynomials Based on Digital Circuit Simulation Software

Enzhong Zeng *

School of Electronic Information, Shanghai Dianji University, Shanghai, China

* Corresponding Author Email: zez13701843349@outlook.com

Abstract. Considering the chip area and energy consumption, the arithmetic operation unit only performs the most basic operations, while complex operations such as definite integrals are carried out at the software level. This article will step by step demonstrate the design process of a 8-bit ALU circuit with a simple polynomial definite integral operation function based on digital circuit simulation software, fully leveraging the characteristics of hardware computing such as parallelism and pipelines to achieve ultra-high throughput and low latency. After comparison and selecting, logisim was selected as the simulation software in this study, and the Newtonian - Leibniz formula was chosen as the operation path. Through the construction of polynomial coefficient input, upper and lower limit input, preprocessing of polynomial coefficients, polynomial evaluation, and subtraction of upper and lower limit polynomials, the main circuit of this ALU was finally formed.This design has indeed been tested to be capable of performing simple polynomial definite integral operations, but it also faces issues such as wasting space,existing room for efficiency optimization and complex processes that require further research to address. It is also hoped that in the future, a truly practical hardware-level definite integral and other multi-functional operation unit can be developed.

Keywords: 8-bit ALU; Logisim; Polynomial Definite Integrals; Simulation Software.

1. Introduction

Central Processing Unit (CPU) is a key support for the development of modern information technology and plays a crucial role in computer systems, the information technology industry, and modern social life.

Arithmetic logic unit (ALU) is regarded as the primary functional block of the processor that handles almost all arithmetic operations.General ALUs, considering the chip area and power consumption, only perform the most basic operations such as addition, subtraction, multiplication and division. This means that for a computer to perform complex operations such as definite integration, it needs to rely on software-level tools like algorithm libraries to transform continuous calculus problems into discrete numerical operations.

The production of a ALU with simple definite integral operation functions using digital circuit simulation software will be demonstrated in this article. Instead of using software tools, it will compute polynomial calculus using its own hardware and Newton-Leibniz formulae. Furthermore, since the formula used in the calculation is the Newton-Leibniz formula rather than a discrete numerical calculation method like the Taylor formula, the result calculated by ALU will be more accurate.

Mochammad Hannats Hanafi Ichsan proposes a design and implementation of an 8-bit CPU (Central Processing Unit) architecture using the simplest design, Mic-1, which is built and implemented step-by-step in Logisim [1]. The undergraduate student can then learn how to design and build a computer with a specific goal.M.Saravanan demonstrates a flexible and efficient calculator design that can perform operations in 8, 16, and 32-bit versions [2]. Verilog and System Verilog are used to implement it. Along with more sophisticated features like exponentiation, square root, and factorial calculations, the design includes the basic arithmetic operations of addition, subtraction, multiplication, and division. It also features bitwise operations and shift capabilities for effective binary data processing. Jitesh R. Shinde proposes a step-by-step optimization approach for

ALUs at the logic circuit level [3]. The concept of resource sharing and optimized arithmetic expressions were utilized to optimize combinational blocks in an ALU. This paper demonstrates how simple tools like Deeds Digital Circuit Simulator or Aldec's Active HDL, when combined with synthesis tools, can effectively teach the concept of digital circuit design to beginners and guide them through successful VLSI projects in the digital domain.V.Priyanka Brahmaiah demonstrated that the tree structure is the traditional approach to ALU design.All the input operations and input functions are fed into a single multiplexer in Figure's tree structure one [4].The output provides the selected function according to the mode assignment, while the input lines of the mode act as multiplex choices.The number of status lines in the ALU is determined by the number of operations carried out.M Ben-Ari demonstrated a finite state machine (FSM) comprises a collection of states (denoted as si) and a set of transitions that connect pairs of these states (si,sj) [5]. Each transition is marked with a "condition/action" label: the condition refers to the trigger that initiates the transition, while the action represents the operation executed upon the transition being activated. This function of FSM enables it to act as an administrator to drive the circuit operation and become one of the important sub-circuits of the ALU built in this paper. Shuguo Wang designed a large-scale integrated circuit chip for a scientific calculator, forming a calculator system with more comprehensive computing functions than the C9318 [6]. Enable the operations of trigonometric functions, exponential functions and logarithmic functions to all be implemented in this design.

This article will then conduct an analysis and comparison of the simulation software and the definite integral calculation path. Ultimately, based on the selected platform and methodology, the construction process of the target circuit will be presented in three parts: A, B, and C. The research objective is to build an 8-bit ALU that can correctly perform definite integrals of simple polynomials

Both M Ben-Ari and Shuguo Wang use FSM in their circuit design. FSM can indeed accurately and effectively convert circuit states. However, considering complexity and area, this design does not separately design an FSM module. Instead, the function of the FSM module in commanding the operation of other modules is integrated into the main circuit, and the operation is carried out in a pipeline manner through methods such as delay.

2. Method

2.1. Circuit Design Based on simulation software

An 8-bit ALU simulation circuit with the function of simple polynomial definite integral calculation and its construction process will be presented in this research.

Compared with building circuits with hardware, simulation software is less costly, less likely to be affected by component issues and wire connection problems in the normal operation of the circuit, and can more clearly and intuitively observe the operating principle of the ALU.

2.2. Introduction of Logisim and reasons for selection

Mehmet Kayaalp demonstrated the application of logisim and built many data paths and control tutorials based on logicism in his research [7]. Wijaya Kurniawan compared different policy software and summarized the advantages and disadvantages of Logisim in his research [8]. Mochammad Hannats Hanafi Ichsan suggests utilizing the Logisim simulator alone to design, develop, and assess an 8-bit CPU architecture for an undergraduate computer engineering course on computer organization and architecture (COA) [9]. Eliminating inefficiencies is the primary benefit of utilizing a single simulator. In order for students to focus more on the material covered in the course without wasting their time simply learning how to operate the many kinds of simulators, a basic CPU architecture is already created and be carried out in the Logisim simulator. Their Study helped this research select logisim as the platform for building this ALU and summarize the following features and advantages of Logisim.

1.Logisim is equipped with a large number of digital logic components, including basic gate circuits, flip-flops, counters, etc., covering the main aspects of digital logic design and can meet the

basic requirements for building an ALU. Meanwhile, it supports custom component libraries and hierarchical circuit design. For the complex polynomial definite integral operation function ALU design, it can be decomposed into several sub-circuits, each responsible for a part of the function, improving design efficiency and the readability of the circuit.

2.Logisim is an educational simulation tool. This software features a simple and intuitive graphical interface, requiring no programming background. Circuits can be built by simply dragging components, lowering the design threshold. It is highly suitable for the research purpose of this study, which focuses on demonstrating the construction process.

3.Logisim is a cross-platform software that can run on operating systems such as Windows, Mac, and Linux, facilitating the replication of this circuit in different environments and enhancing the universality of this study.

2.3. Theoretical basis: Newton-Leibniz formula

$$\int_b^a f(x)dx = F(b) - F(a) \tag{1}$$

$$F(x)=a_0x+a_1x^2+a_2x^3+a_3x^4 \tag{2}$$

There are the following advantages in building circuits using the Newton-Leibniz formula as the calculation path:

1.It is simple, convenient, and easy to be implemented at the hardware level.

2.Compared with other definite integral calculation methods such as the Riemann sum, the Newton-Leibniz formula does not rely on approximation strategies. It can obtain theoretical true values without approximation through the formula, thereby improving the accuracy of the operation results

2.4. Demonstration of the construction process

Due to the inherent limitations of the logisim software or version issues, the attributes or functions of some of the basic circuit components that come with logisim cannot meet the requirements of the target circuit.

3. The Process of Circuit Construction

3.1. 3.1 Component Preparation

This prat demonstrates the basic component circuit design applicable to the target circuit, such as a multiplier that receives two 8-bit inputs and outputs a 16-bit multiplication result. If the version or simulation software you are using already has the appropriate basic components, this part will be unnecessary.

3.1.1 SignedRegister(8-bits)

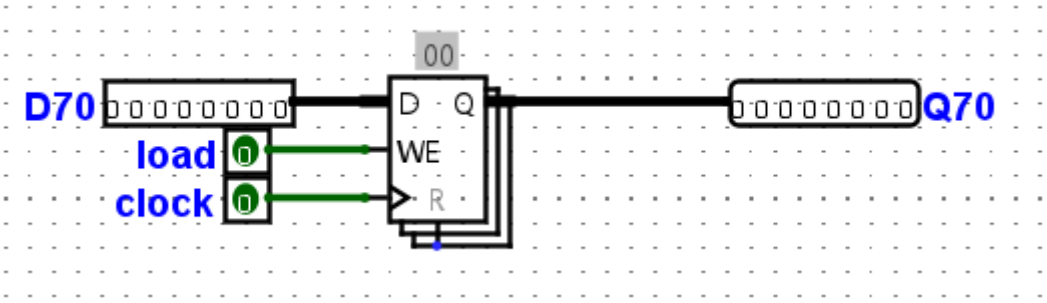

Fig. 1 circuit of a signed 8-bit register

Figure 1 shows a circuit diagram of an 8-bit register with symbols, which is used to store inputs such as coefficients in the subsequent operator circuit.

3.1.2 SignedRegister(16-bits)

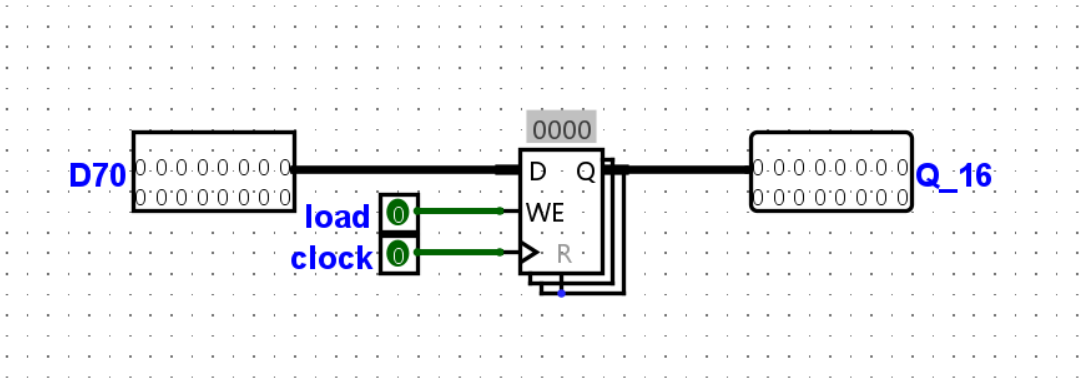

Fig. 2 circuit of a signed 16-bit register

Figure 2 shows a circuit diagram of a signed 16-bit register, which is used to store the calculation results of polynomials and the final results of definite integrals in the main circuit.

3.1.3 Pulse

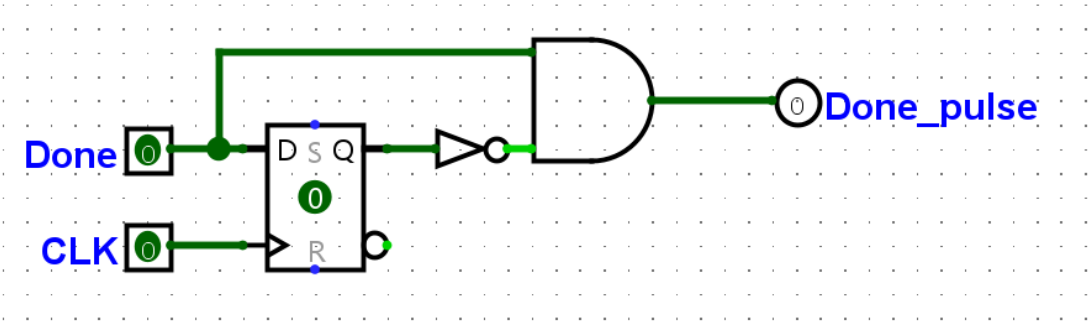

Fig. 3 circuit of a signed 16-bit register

Figure 3 shows a circuit diagram of a pulse generator triggered by the rising edge, which converts the input signal into a fixed-width pulse, is a key component for the normal operation of other operator circuits.

3.1.4 ConstROM

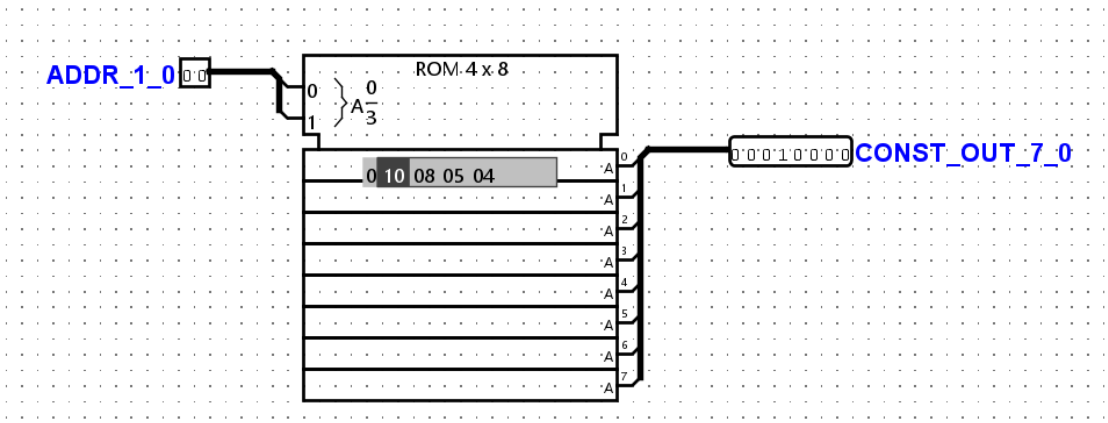

Fig. 4 circuit of a signed 16-bit register

Figure 4 shows the circuit diagram of the Changshu lookup table unit. It is a fixed constant memory, whose function is to provide a fixed constant factor for the integration coefficient calculation (CoeffProc) to control which coefficient is currently being processed

3.1.5 oX_d

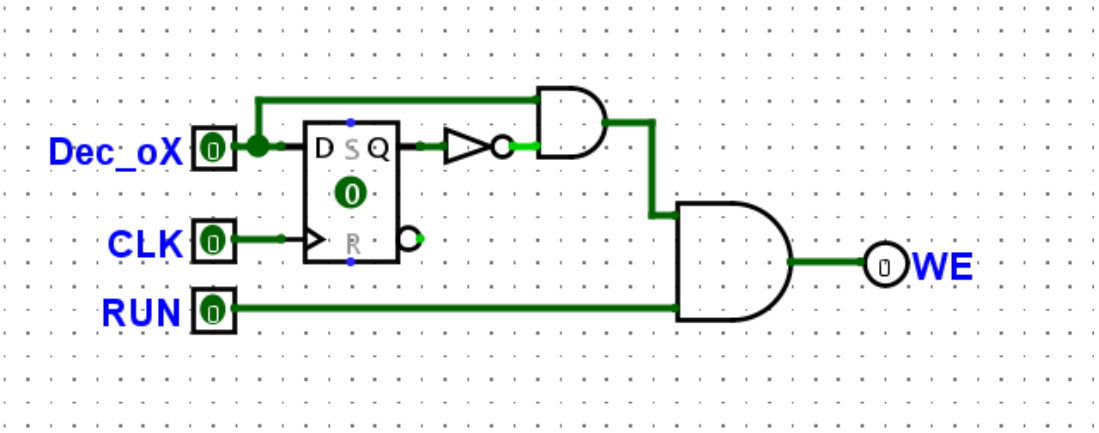

Fig. 5 circuit of WE Pulse Generator

Figure 5 shows the circuit diagram of WE Pulse Generator. The function of this component is to convert the decoder output signal into a single-beat write pulse, which is used to safely and synchronously trigger register writes

3.1.6 A Signed 8-bit Multiplier(with 16-bit output)

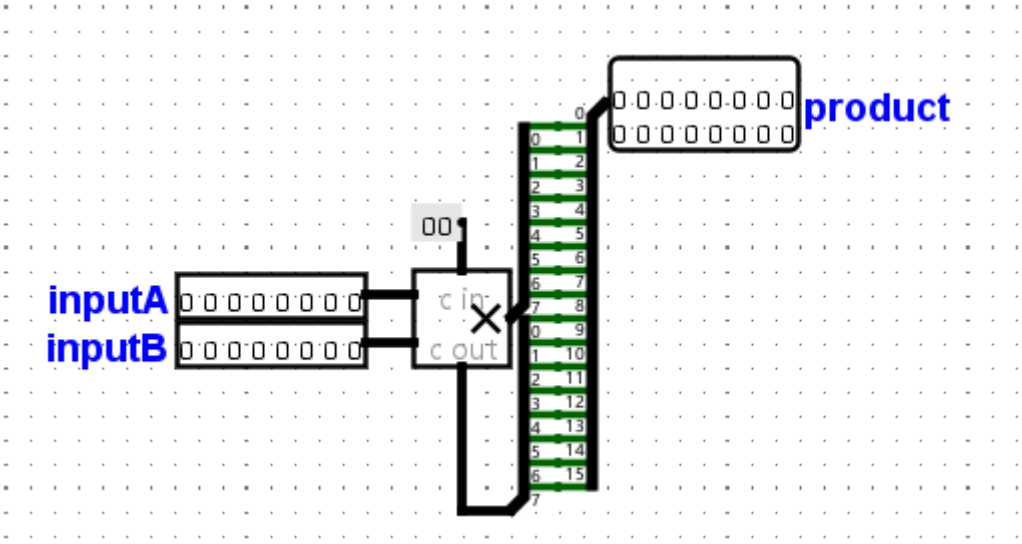

Fig. 6 circuit of a signed 8-bit multiplier

The multiplier in logisim consists of two eight-bit inputs and one eight-bit output. According to the circuit in Figure 6, a multiplier with eight-bit input and 16-bit output can be obtained

3.2. Sub-circuits

This part will present several key operational module sub-circuits in the ALU to be developed in this paper.

3.2.1 CoeffReg

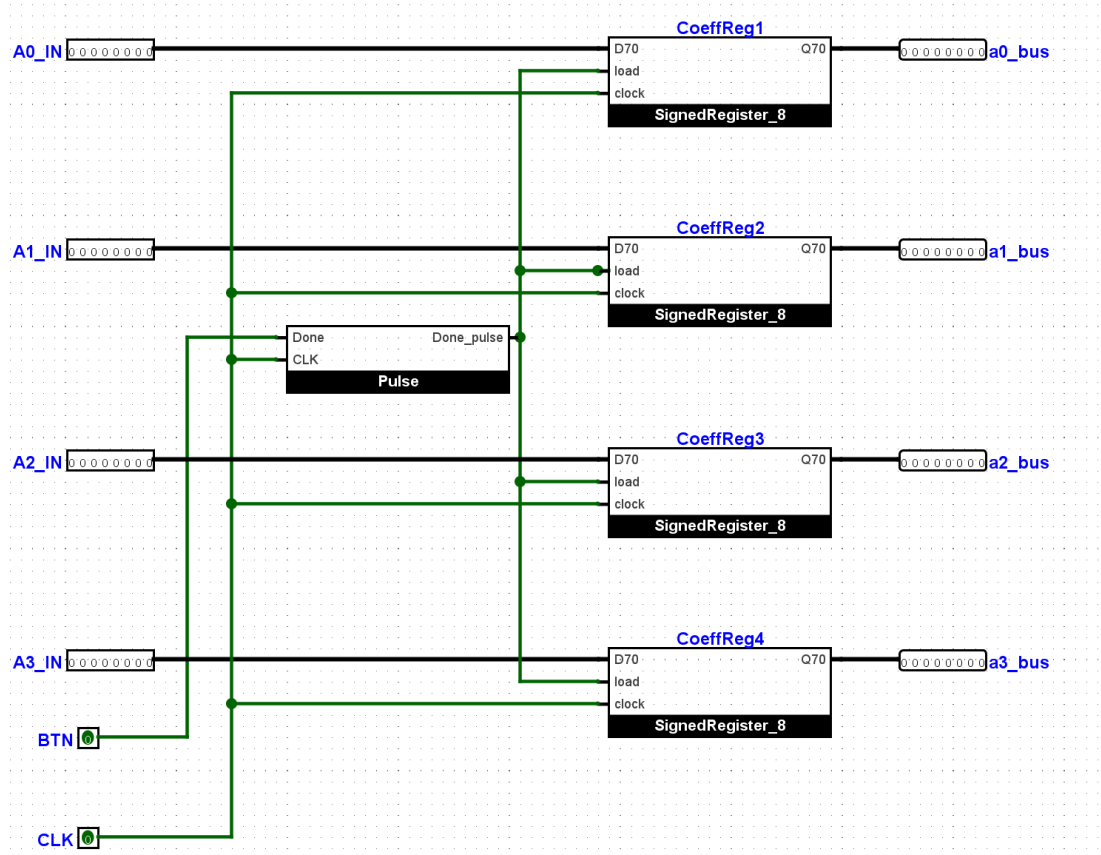

Fig. 7 circuit of CoeffReg

Figure 7 is a Coefficient Register Bank. This module stores polynomial coefficients in four 8-bit signed registers and updates them synchronously via a single pulse trigger.

3.2.2 Limits

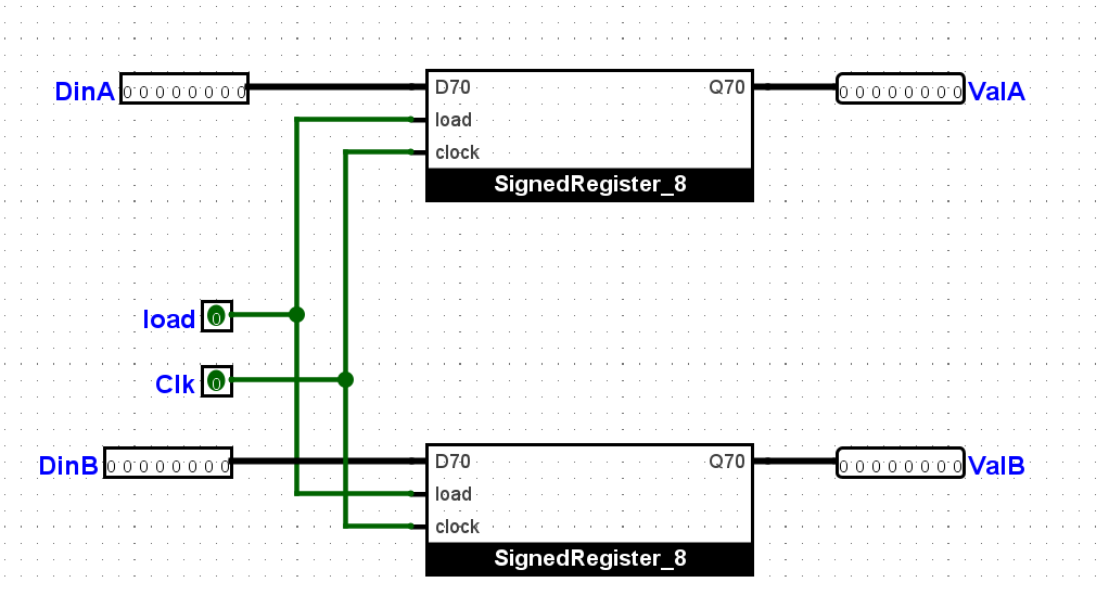

Fig. 8 circuit of Limits

Figure 8 shows the circuit diagram of a sub-circuit for storing the upper and lower limits of definite integrals, which will respectively send the upper and lower limits of definite integrals to the values of the two subsequent PolyEval to calculate polynomials.

3.2.3 Coeffproc

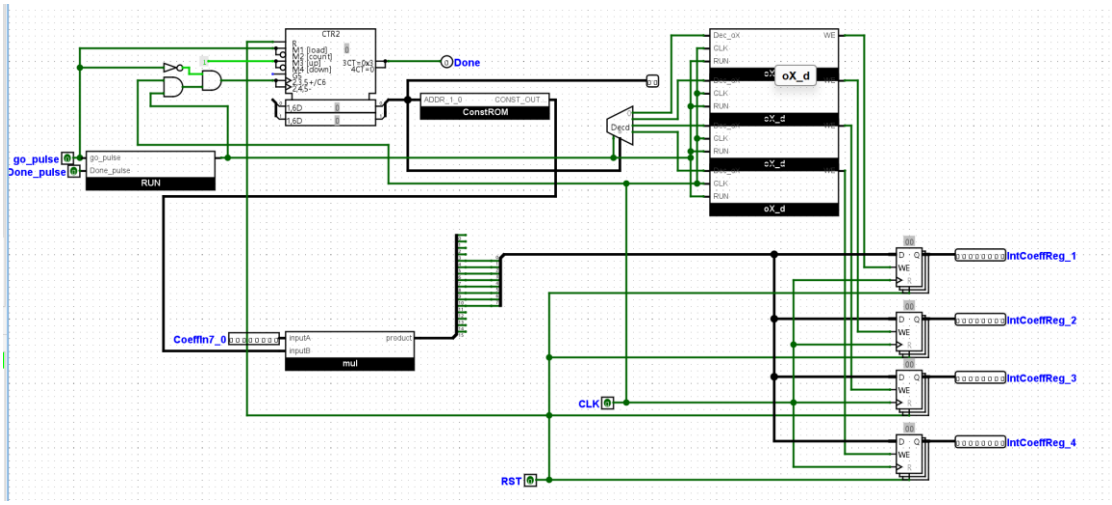

Fig. 9 circuit of Coeffproc

Figure 9 is the CoeffProc module. It automatically calculates the integral coefficients of a polynomial by multiplying each input coefficient a_n by its corresponding constant $1/(n+1)$ stored in a ROM table.

It sequentially processes each term using a counter-driven control mechanism, stores the results in dedicated registers, and outputs a completion signal when all coefficients have been integrated.

3.2.4 PolyEval

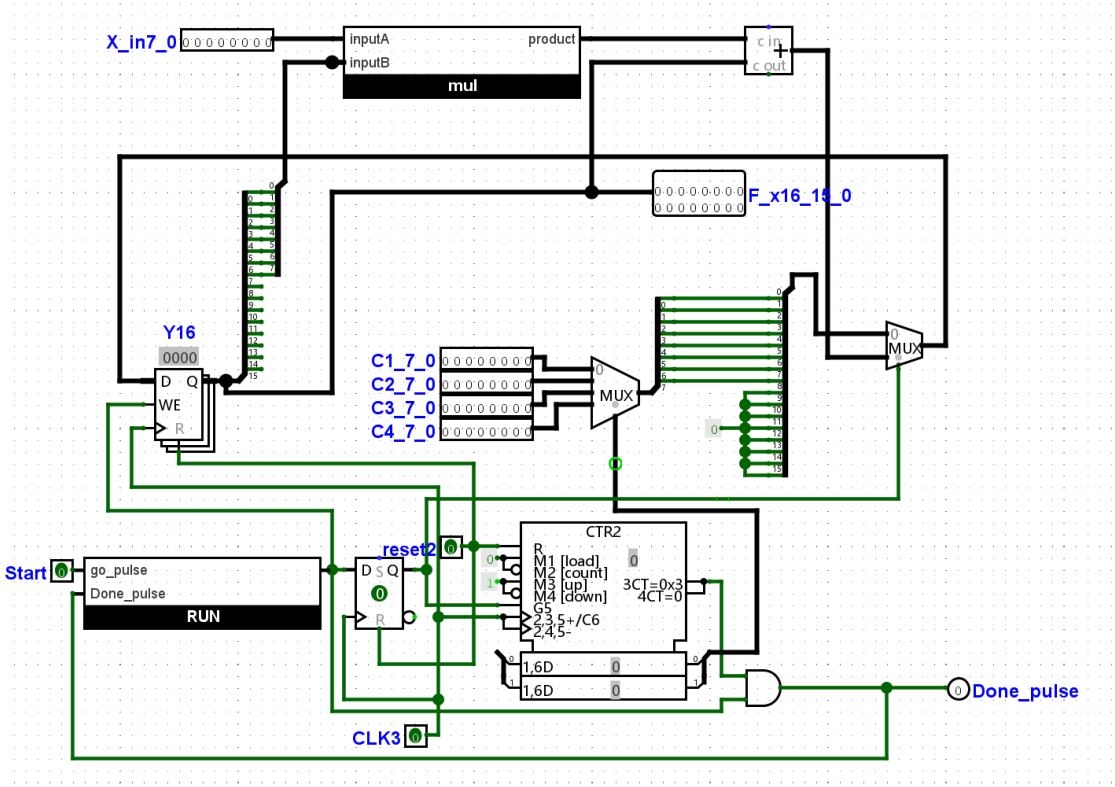

Fig. 10 circuit of PolyEval

Figure 10 is the CoeffProc module. It performs the polynomial evaluation process by calculating $F(x) = a_0 + a_1x + a_2x^2 + a_3x^3$.

It includes a power generator for producing successive powers of x, a multiplier array for term-wise computation, and an adder chain to accumulate the results.

3.3. Main Circuit

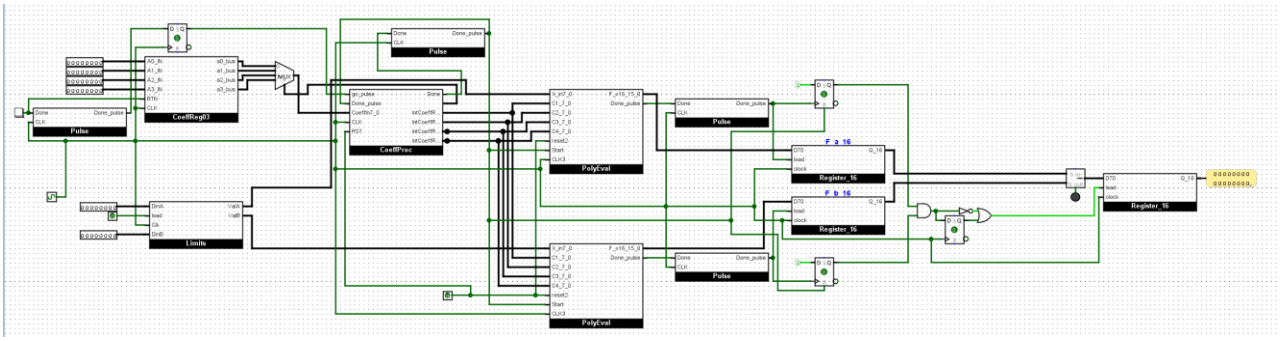

Fig. 11 main circuit

Figure 11 shows the main circuit of the ALU with the function of simple polynomial definite integral operation.

4. Discussion and Future Work

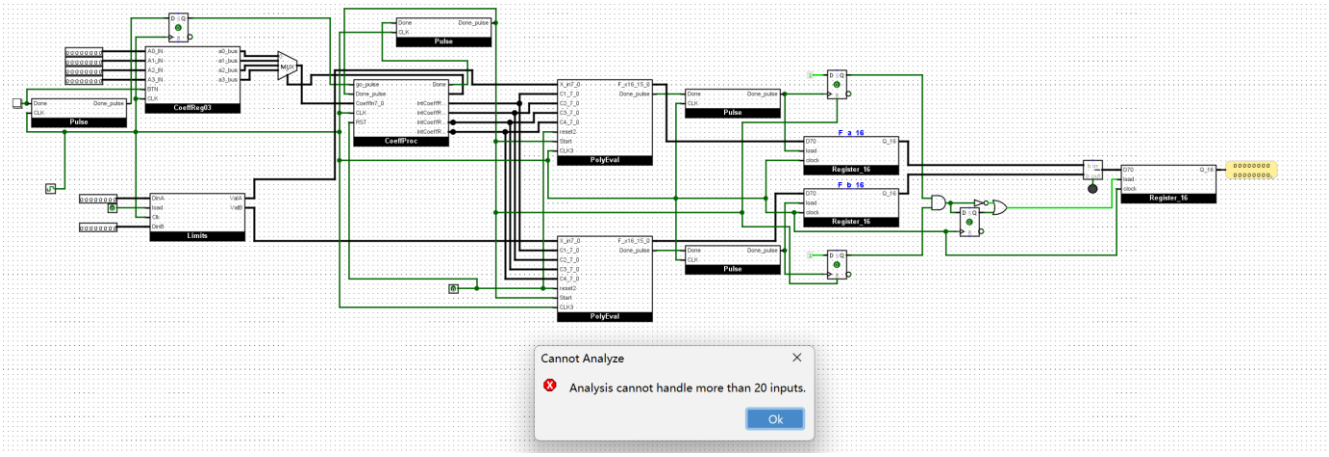

Fig. 12 main circuit

Logisim has the function of analyzing circuits and can automatically generate truth tables. However, due to the excessive input of this circuit, the analysis function reported an error as shown in Figure 12 and could not be used. Therefore, this paper will manually input data for functional verification

This design has indeed been tested to be capable of performing simple polynomial definite integral operations.

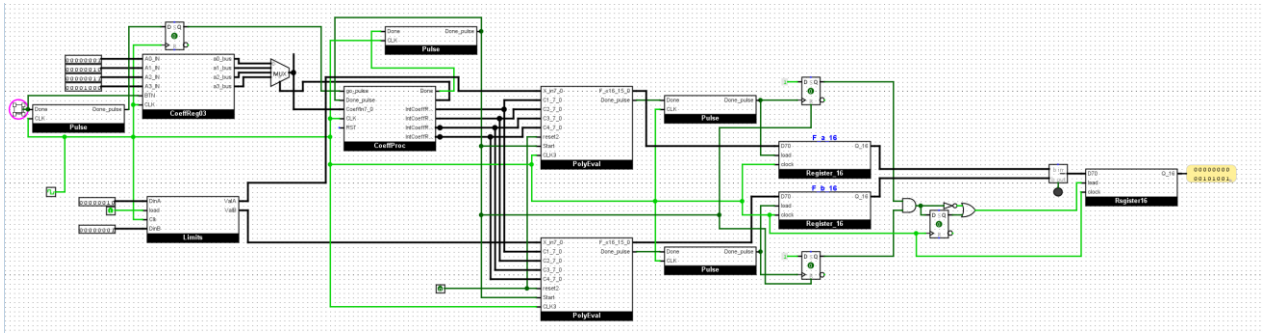

Fig. 13 Manually test the circuit function

Figure 13 shows the manual input of test data $8x^3+3x^2+2x+1$ into the circuit and the obtaining of result 41 to verify the correctness of the circuit.

If the polynomial is $8x^3+3x^2+2x+1$,which means the coefficient inputs from top to botton are 00000001,00000010,00000011,00001000 and the upper and lower limits of the integral are 00000010

and 00000001.The Coeffproc will turn the inputs into 00000010,00000001,00000001,00000001 according to the fourthe operation rules of indefinite integration of polynomials.Then the two PolyEvals will calculate the values of the polynomial are 46 and 5 respectively when X is the upper limit and the lower limit.Finally, it is handed over to the subtractor for subtraction to obtain the result 41.

Component	Library	Simple	Unique	Recursive
SignedRegister_8	ALU2	0	6	6
CoeffReg03	ALU2	1	1	1
ConstROM	ALU2	0	1	1
Limits	ALU2	1	1	1
CoeffProc	ALU2	1	1	1
mul	ALU2	0	2	3
PolyEval	ALU2	2	2	2
Pulse	ALU2	4	5	5
Register_16	ALU2	3	3	3
oX_d	ALU2	0	4	4
RUN	ALU2	0	2	3
Splitter	Wiring	0	9	19
Pin	Wiring	8	68	142
Probe	Wiring	1	5	8
Clock	Wiring	1	1	1
Constant	Wiring	2	8	15
NOT Gate	Gates	1	4	11
AND Gate	Gates	1	7	18
OR Gate	Gates	1	1	1
Multiplexer	Plexers	1	3	5
Decoder	Plexers	0	1	1
Adder	Arithmetic	0	1	2
Subtractor	Arithmetic	1	1	1
Multiplier	Arithmetic	0	1	3
D Flip-Flop	Memory	4	8	18
Register	Memory	0	7	15
Counter	Memory	0	2	3
ROM	Memory	0	1	1
Button	Input/Output	1	1	1
LED	Input/Output	1	1	1
TOTAL (without project's sub circ...		23	130	266
TOTAL (with sub circuits)		35	158	296

Fig. 14 Circuit statistics

Figure 14 shows the statistics used for this circuit.

Although this circuit basically realizes the operation function of simple polynomial definite integrals, it still faces issues such as wasting space, existing room for efficiency optimization and complex processes that require further research to address. In addition, the key parameters of measuring the performance of any CPU design are logical latency, power consumption and chip area according to N. Ravindran's paper [10]. This paper did not conduct measurements on delay, energy consumption and circuit area, which is waiting for further improved in subsequent researches.

In the future, it is planned to combine the ALU designed in this paper with common ALU with basic computing to create a multi-functional ALU. If technology permits, it is even hoped that more ALUs that can perform definite integral operations on polynomials, calculate double or even triple integrals, and be truly developed into hardware for practical application will be researched.

5. Conclusion

According to the experimental results, it is feasible to incorporate polynomial definite integral operations into the hardware level to create an ALU capable of calculating definite integrals of simple polynomials based on digital circuit simulation software. This design can be realized without relying on a professional or complex environment, but its shortcomings in terms of area and energy consumption can also be seen. This drawback will be magnified when the number of operation bits or the integration dimension increases.

References

[1] Hanafi M H, Kurniawan W, Design and implementation 8-bit CPU architecture on Logisim for undergraduate learning support, IEEE, 2017.

[2] Saravanan M, Saritha J, Vishnu Varthini S S, Sathya Priya R S, Siva Sakthi R S, Rahi S B, Performance Analysis of 8/16/32-bit Calculator Using Verilog and System Verilog, IEEE, 2024.

[3] Shinde J R, Shinde S J, An Optimization Design Strategy for Arithmetic Logic Unit, Universal Journal of Electrical and Electronic Engineering, 2019.

[4] Brahmaiah V P, Sirisinagandla Y, Mouktika B V, Sai Y P, Design and Implementation of Optimized Based 32-Bit Arithmetic and Logical Unit, IEEE, (year not given).

[5] Ben-Ari M, Mondad F, Finite State Machines, Springer, 2017.

[6] Wang S, Large-scale integrated circuit chip design for scientific calculators, CNKI, 2015.

[7] Kurniawan W, Hanafi M H, Teaching and learning support for computer architecture and organization courses design on computer engineering and computer science for undergraduate: A review, IEEE, 2017.

[8] Kayaalp M, Using Logisim-evolution for Teaching Datapath and Control, IEEE, 2021.

[9] Hanafi M H, Kurniawan W, CPU implementation using only Logisim simulator to achieve computer architecture learning outcome, BEEI, 2019.

[10] Ravindran N, Lourde R M, An optimum VLSI design of a 16-BIT ALU, IEEE, 2015.