

Designing of ALU Block for RISC-V-Based Processor Core with Low Power

K. Madhava Rao^a, T. Vasudeava Reddy and Uttej
B. V. Raju institute of Technology, BVRIT, Narsapur, Marsapur, Medak, India

Keywords: RISC-V Architecture, Parallel-Prefix Adders, Kogge-Stone Adder, Galois-Field Multiplier, Approximate Computing, Instruction Sets.


Abstract: We are in an era of technology where the speed, power, and area of the VLSI chip are the three extremely crucial factors. Adders and multipliers form key components in a variety of arithmetic and logical processes. For binary addition, Parallel Prefix Adder (PPA) is an effective and useful circuit that has evolved. Fast Kogge-Stone Adder (KSA) is Parallel Prefix Adder that is used for making fast addition operations. A Galois Filed Multiplier (GFM) is also developed for making fast operations that work on shift operation technique. The objective of the research is to integrate FKS and GFM that accommodate superior systems to improvise a performance. Both of these modules are integrated into an ALU block of the RISC V processor. The design has been implemented for a 32-Bit sequence. and the entire design was implemented in Xilinx ISE and Open Lane toolset.

1 INTRODUCTION

In an emerging digital world, systems like microprocessors and data processing jobs like filtering, object recognition, convolution, image and video compression, modulation, and so on, the addition and multiplication of the binary values (say two) is amongst the most fundamental and important aspects of the arithmetic logic unit (ALU). The binary adders and multipliers in these systems are crucial aspects of an ALU. The requirements referring to an adder in VLSI systems need to be quick and reliable in terms of speed, area, and power. The adder design architecture is primarily responsible for performing operations. Because RISC V is an open-sourced definition of an Instruction Set Architecture, the architecture considered here is RISC architecture (ISA). In contrast to most other ISA designs.

Nowadays, we do not want to waste our crucial time doing a single task the same goes for the processors. Nobody would want to have an electronic device with a slow processor, but how are adders and multipliers related to processors? The answer is simple – the better the adder and multiplier design techniques the faster the processor works. We

measure the speed of processors by the time it takes them to complete a task; the time it takes the processors to complete a task is normally equal to the overall propagation delay in the circuit. The circuit/processor becomes slower as the propagation delay increases, and the opposite is also true. With the preceding argument, we may also conclude that 'the fewer the propagation delays, the faster the circuit/processor, and vice versa. The data transfer rate of the carry chain is a key concern for binary addition. As the input bit or operand size rises with the length of the carry chain. The Modern PPA architecture consists of Pre-Processing (Felzmann et al 2020), Carry Look-Ahead, and Post-Processing parts utilized to prevent the problem of Carry chain propagation. In binary addition of digital systems, these adders are used to build the most efficient circuit (Shilpa et al 2016) (here we used an ALU block based on RISC architecture). The performance and latency of these adders are mostly determined by total levels in carry generation. To reduce propagation latency, we added two modules to the CPU's ALU block: the Kogge-stone adder and the Galois field multiplier.

^a  <https://orcid.org/0000-0003-3115-6385>

2 RISC-V PROCESSOR UNIT

The Reduced Instruction Set Computer [RISC] processor is a computer arithmetic-logical unit that has a small set of instructions that optimizes the entire logic in most frequently used instructions which can be quickly accessible for faster execution. Compared to CISC and RISC processors software must handle a greater number of operations per cycle. The RISC processors have a trump card in entreaty that will be helpful to perform faster instruction execution. The process of designing, testing, and manufacturing them is also relatively inexpensive. The instruction register became very complex to design and the control unit occupies most of the chip area the instructions that access memory results in decreasing the speed execution. Thus, the RISC processor is developed, and a small set of instructions results in simplifies the design and improves the overall performance of the processor.

In this paper, we present a RISC processor equipped with three major components, which are summarized in Figure 1, the three most crucial components of the processor unit are data Path, control, and memory Unit.

2.1 Data Path Unit

The hardware responsible for making all the vital operations like ALU operations, internal bus communication, and registers is known as Data Path Unit.

2.2 Control Unit

The hardware decides on which data path needs to be selected in the call of selection of operation and the movement of data between the ALU components is known as Control Unit.

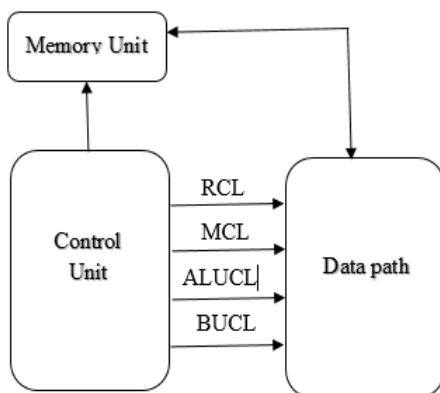


Figure 1: System Architecture.

2.3 Memory Unit

The data can be stored in a Memory unit also known as a Storage unit. The data stored can be expressed in terms of Bytes.

2.4 Operation

The overall processor architecture diagram is shown in figure 2. The architecture consists of five stages of operations such as fetch, decode, execute, memory, and write Back.

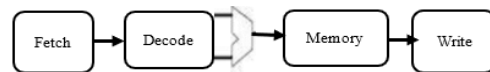


Figure 2: Data Flow model.

Since we are integrating the modules into the ALU block, our design mainly focuses on Execute stage of Operation.

3 INTEGRATING ADDITIONAL LOGICS INTO THE EXISTING ALU BLOCK

The existing RISC V-based ALU block consider here is the Risk-5 module. In this module, Kogge-Stone Adder is the fast adder adopted in the industry, and Galois Field Multiplier is best proven used for cryptographic applications integrated into the ALU block.

The two modules which are integrated into the Risk-5 ALU block were Kogge-Stone Adder and Galois Field Multiplier.

3.1 Kogge-Stone Adder

KS Adder is a CLA is a Parallel-Prefix form of logic attached to it. It generates carry in no time 0 (Log2 N) (can say it as ZERO time) and is commonly regarded as the quickest adder in the industry upon its great work in terms of performance in arithmetic and logic circuits. The Kogge-stone adder performs faster computations by having parallel computing with a little overhead of area which is acceptable considering the area-to-performance ratio. The Carry outputs generated from this adder were very quick in that the computations were done in parallel and thus the pace of operation is fairly inflated as the depth of the node is low & the result is relying on initial inputs. The whole design of architecture and its functioning

can be effortlessly realized by looking over it concerning three evident sections:

- a. Pre-Processing Block
- b. Carry Look Ahead Generation Block
- c. Post-Processing block

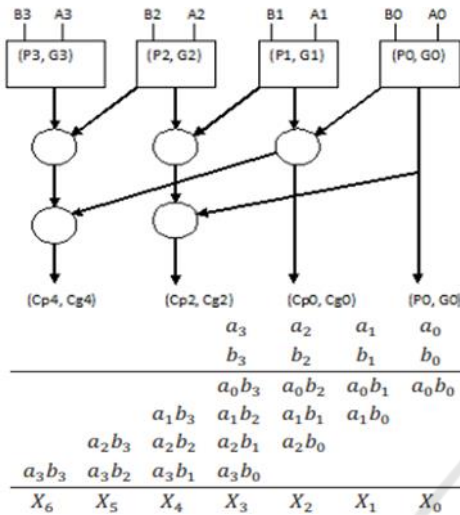
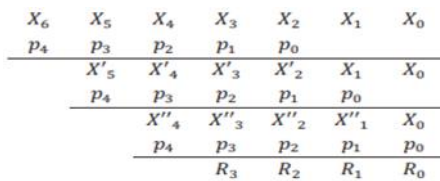


Figure 3: Stages of KS parallel prefix adder.

After product output X is obtained, results are divided by p(x) to reduce represents by LPR as shown below (Chen et al 2014). The basic building block of Kogge-Stone Adder is consisting of three blocks as discussed above. Below is the figure of basic 4bit length Kogge-Stone Adder.

1. Pre-Processing. This stage is responsible for creating a pair of create and cultivate signals for A and B inputs. Below two are the logic equations for create and cultivate signals.



$$ppi = Ai \text{ xor } Bi$$

$$gi = Ai \text{ and } Bi$$

3.2 CLA Network

KSA has an advantage over other adders due to this block and is the source of its high performance. The calculation of carries for each bit takes place in this stage. And utilizes the obtained intermediate cultivate and create signals. The logic equations for the intermediate create and cultivate group of signals is given.

$$Pi,j = Pi:k+1 \text{ and } Pk:j$$

$$Gi,j = Gi:k+1 \text{ or } (Pi:k+1 \text{ and } Gk:j)$$

3.3 Post Processing

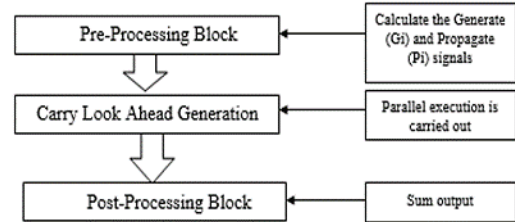


Figure 4: KS Parallel Prefix Adder Architecture.

All adders of this family go through this closing step (carry look ahead). This stage eventually necessitates in computation of sum bits. The logic equation below computes the sum bits:

$$Si = pi \text{ xor } Ci-1$$

KSA and other adders have a greater distance when more bits are required.

3.4 Galois Field Multiplier

The efficient Galois Field arithmetic logic (GF(2^m)) are likely used in error decoding, error controlling and cryptosystems. The efficient execution and implementation of GF(2^m) multiplication can immensely improve system intricacy and timing performance.

As GF(q) is expounded by primitive polynomial to form GF(q^m), where GF(q) is a field with finite elements. Exor operation is represents GF addition and multiplication is seen as multiplication as modulo p(x) (Chen et al 2014). The multiplication of a and b symbols were given with a mod primitive polynomial. The primitive AND gate is used to multiply bits and the product output X obtained are connected to each other nearly to the actual as given below.

After product output X is obtained, results are divided by p(x) to reduce represents by LPR as shown below (Chen et al 2014. Torres et al. 2024).

This circuit affects regular two-bit multiplication elements of Galois Field (2^m). Then each bit of the field is intimated as m-bit binary number. The 2m-1bit partial products are generated when two elements are given to binary multiplier and final m-bit binary products are generated, that is divided by Galois field generator of polynomial.

When a(x) & b(x) are two field elements also s(x) is their product.

$$s(x) = a(x)*b(x) \text{ p}(x)I \tag{1}$$

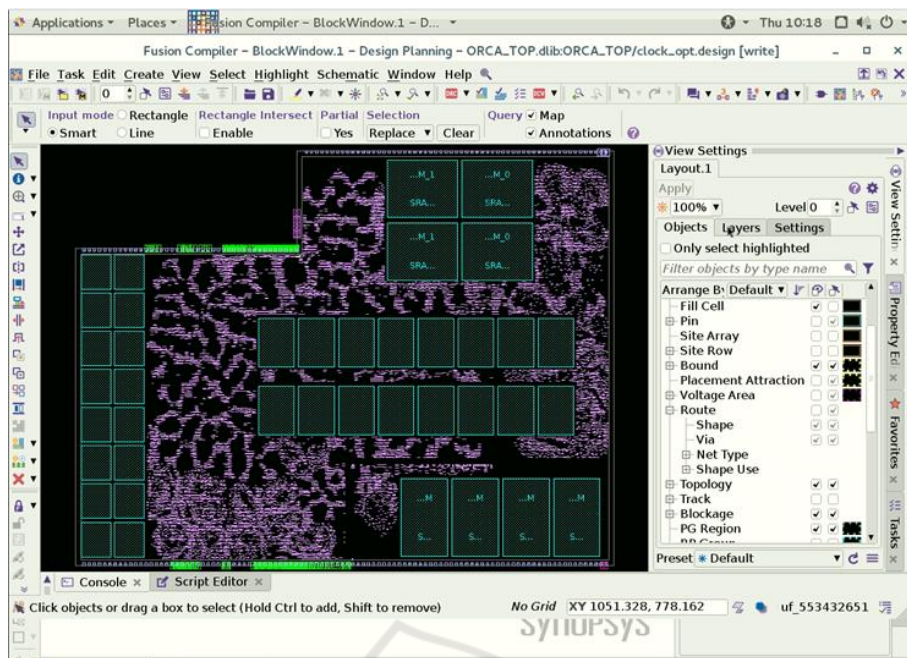


Figure 5: Cell-View of RISC-V design in Synopsys ICC2.

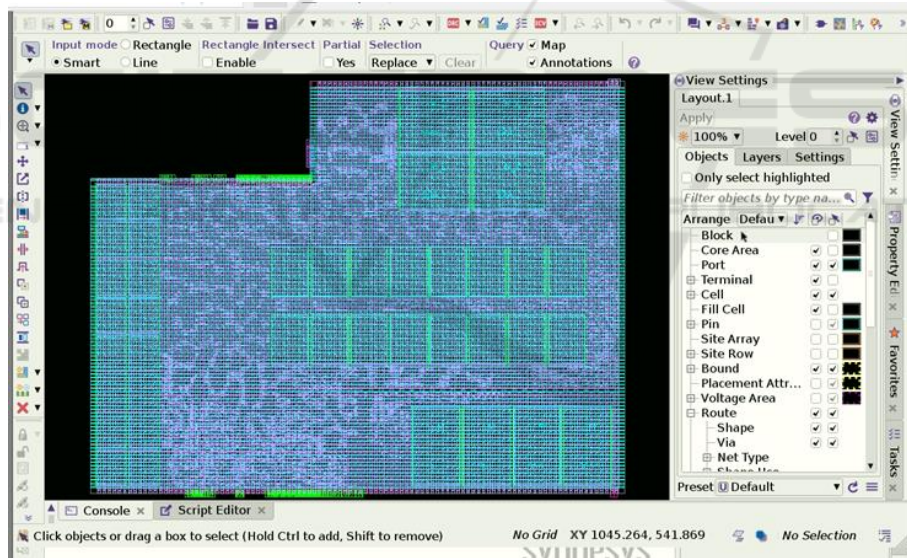


Figure 6: Routing-View of RISC-V design in Synopsys ICC2.

The product entails 2 steps that involves the polynomial multiplication and reduction to a non-reducible polynomial for a 4 bit multiplier as shown in figure 1. $2m-2$ polynomial of $d(x)$ indicates the product of polynomials indicates field elements $a(x)$ & $b(x)$, where $d(x)=a(x)b(x)$. The degree $2m-2$ polynomial $d(x)$ is decreased by degree of m , in modular reduction $s(x) = d(x)Ip(x)I$, by irreducible polynomial $p(x)$ iteratively” (Chiou et al 2007). Hence, modular reduction may be simplified by

choosing the irreducible polynomial $p(x)$. It is usually more efficient to use irreducible polynomials with fewer nonzero terms.

4 IMPLEMENTATION

The attached images below refer to the physical implementation of the RISC-V design with the in-

house developed Kogge-Stone Adder and Galois-Field Multiplier in cell-view and in routing-view.

5 RESULTS

The obtained results of Kogge-Stone Adder and Galois Field Multiplier shows the Power, Area and Timing parameters. The parametric figures obtained were good enough. The functionality seems to be working as the requirements with optimum power consumption, reasonable area consumed and have met the timing in worst case (Kaiyrakhmet 2024).

Table 1: Kogge_stone_32bit Project Statistics.

GALOIS-FIELD MULTIPLIER	AREA (mm ²)	POWER (mW)	DELAY (ns)
8-Bit	0.044	79.453	10.215
16-Bit	0.89	76.745	10.669
32-Bit	1.8	196.198	13.007

RISC-V Project Status (21/06/2024 - 03:19:16)			
Project File:	src_xive	Parser Errors:	No Errors
Model Name:	Galois_Field_32bit	Implementation State:	Synthesized
Target Device:	xc7s00c32csg324	• Errors:	0 Errors
Product Version:	ISE 14.7	• Warnings:	22 Warnings (5 give)
Design Goal:	Balanced	• Routing Results:	All Signals Completely Routed
Area (mm ²):	1.8	• Power (mW):	196
Environment:	System Settings	• Final Timing Score (ns):	13.007

Device Utilization Summary			
Logic Utilization	Used	Available	Utilization
Number of Slice LUTs	407	4096	10.18%
Number of fully used LUT4F pairs	0	800	0%
Number of bonded IOBs	43	79	1.29%

Figure 8: Galois_field_32bit Project Statistics.

6 COMPARISONS

Comparison of different types of Adders:

- Table 1 - Comparison of 8/16/32-Bit Kogge-Stone Adders.
- Table 2 - Comparison of Parallel Adders.
- Table 3 - Comparison of 32-Bit KSA with RCA, CLA, and Basic Adder.
- Table 4 - Comparison of 8/16/32-Bit Galois-Field Multipliers
- Table 5 - Comparison of 32-Bit Multipliers

Table 2: Comparison of 32-Bit Parallel Adders.

KOGGE-STONE ADDER	AREA (mm ²)	POWER (mW)	DELAY (ns)
8-Bit	0.013	1.213	20.015
16-Bit	0.065	0.240	20.423
32-Bit	0.141	0.223	20.892

Table 3: Comparison of 32-Bit Kogge-Stone Adder with other Adders.

PARALLEL ADDERS (32-Bit)	AREA (mm ²)	POWER (mW)	DELAY (ns)
Kogge-Stone Adder	0.141	0.223	20.892
Ladner-Fisher Adder	0.364	0.240	23.218
Brent-Kung Adder	0.956	0.232	22.967

Table 4: Comparison of 8-bit, 16-Bit, 32-Bit Galois-Field.

Different 32-Bit Adders	AREA (mm ²)	POWER (mW)	DELAY (ns)
RCA	0.093	0.221	66.845
CLA	0.097	0.225	63.428
Adder	0.092	0.240	80.874
KSA	0.141	0.223	20.892

Table 5: Comparison of different 32-Bit Multipliers.

Parameter	Array Multiplier	Booths Multiplier	Vedic Multiplier	Wallace Tree Multiplier
Delay (ns)	19.1	16.35	18.00	10.60
Area (mm ²)	3.71	3.24	5.66	6.87
Power (mW)	185.33	243.85	271.20	234.32

7 CONCLUSION

A fast Kogge-Stone adder and Galois-Field Multiplier are designed here, and simulated using Xilinx ISE 14.7 software. The obtained gate level netlist is integrated with ALU block which is based on RISC-V architecture. Technology based on 0.18um process is physically implemented. Validation of high-speed adder and multiplier are compared with other adder and multiplier architectures with RCA, CLA and Wallace Tree (Anjana et al. 2014)

All architectures were implemented using Verilog and synthesized using front-end tools of Xilinx ISE 14.7, back-end tools Open Lane, QFLow to generate the ASIC layouts (Lee Mei et al 2018). Significant improvement in delay area and power consumption is obtained by comparing KSA and GFM with other architectures to accelerate complex applications.

ACKNOWLEDGMENT

This research and implementation were supported by Mr. K. Madhava Rao and Mr. T Vasudeva Reddy for designing and integrating the fast adder and multiplier modules. Xilinx ISE 14.7 and Model Sim tools are used and implemented in the Verilog HDL and Python programming languages.

REFERENCES

- Felzmann, I., Filho, J. F., & Wanner, L. (November 2020). Risk-5: Controlled approximations for RISC-V.
- Shilpa C. N, Kunjan D. Shinde, Nithin H. V. (2016). Design, Implementation and Comparative Analysis of Kogge Stone Adder using CMOS and GDI design: A VLSI Based Approach. PESITM, Shivamogga.
- Lee Mei Xiang, Muhammad Mun'im Ahmad Zabidi, Ainy Haziyah Awab, Ab Al-Hadi Ab Rahman. (2018). VLSI Implementation of a Fast Kogge-Stone Parallel-Prefix Adder. Journal of Physics: Conference Series.
- Kim, N.S., & Karpuzcu, U.R. (2019). Approximate ultra-low voltage many-core processor design.
- Vimukth John, Shylu Sam, S. Radha, P. Sam Paul, Joel Samuel. (2020). Design of a power-efficient Kogge–Stone adder by exploring new OR gate in 45nm CMOS process. Circuit World.
- Sunil M, Ankith R D, Manjunath G D, & Premananda B S. (Jan 2014). Design and Implementation of Faster Parallel Prefix Kogge Stone Adder.
- Song, L., & Parhi, K.K. (1998). Low-Energy Digit-Serial/Parallel Finite Field Multipliers.
- Kaiyrakhmet, O. (2024). Next generation storage: Non-Volatile Memory. SPAST Reports, 2(2). <https://spast.org/ojspath/article/view/23>
- Chiou, C.W., Lee, C.Y., & Lin, J. M. (2007). Finite field polynomial multiplier with linear feedback shift register.
- Chen, R.J., Fan, J.W., & Liao, C.H. (2014). Reconfigurable Galois Field multiplier.
- Rosa Elia Martínez Torres, Thakur, S., Oruganti, S. K., & Nita Sukdeo. (2024). Ring Resonator Measurement: Characterizing High Dielectric Constant Materials for the Fourth Industrial Revolution. SPAST Reports, 2(2). <https://spast.org/ojspath/article/view/4952>
- R Anjana, B Abishna, M. S Harshitha, E Abhishek, V Ravichandra, M S Suma. (2014). Implementation of vedic multiplier using Kogge-stone adder. In 2014 International Conference on Embedded Systems (ICES).