Galois Field Arithmetic Unit (GFAU)

Sabbir Ahmed, Jeffrey Osazuwa, Howard To, Brian Weber, Dr. E.F. Charles LaBerge
Department of Computer Science and Electrical Engineering, University of Maryland, Baltimore County

What are Galois Fields?

Galois fields (pronounced “Gal-o-AH”) are sets with finite field orders where addition and multiplication are well defined. They are a key part of number theory, abstract algebra, arithmetic algebraic geometry, and cryptography. In error detection and correction, Galois fields are utilized in cyclic redundancy check (CRC) which are used in digital networks and storage devices to detect accidental changes to raw data.

Table 1: Elements of $GF[x](2) = x^3 + x^2 + x^0$

Element	Symbol	Polynomial	Symbol
0	NULL	$0+0+0$	000
α^0	000	$0+0+\alpha^0$	001
α^1	001	$0+\alpha^1+0$	010
α^2	010	α^2+0+0	100
α^3	011	$\alpha^2+0+\alpha^0$	101
α^4	100	$\alpha^2+\alpha^1+\alpha^0$	111
α^5	101	$0+\alpha^1+\alpha^0$	011
α^6	110	$\alpha^2+\alpha^1+0$	110

$$\alpha^5 + \alpha^3 = \alpha^0$$

$$\alpha^5 \times \alpha^3 = \alpha^1$$

$$\alpha^5 \div \alpha^3 = \alpha^2$$

$$\log(\alpha^5) = 5$$

Figure 1: Example Operations in $GF[x](2)$

Objective

To design a scalable arithmetic logic unit (ALU) capable of generating elements in the Galois field of an irreducible polynomial and perform addition, subtraction, multiplication, division and logarithm for low powered devices.

Design Approach

- Scalable, parameterized and efficient design prioritized over specific platform hardware requirements
- Designed entirely in VHSIC Hardware Description Language (VHDL) modules and packages
- Capability of design limited only by external memory capacity
- Simple scalable interface for speed and flexibility.

Design Overview

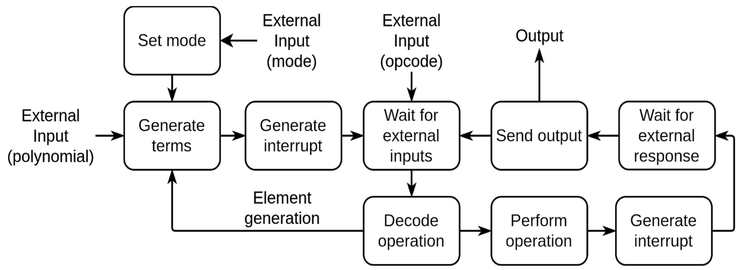

Figure 3: Functional Flow Diagram

Modules

Global Registers
- Generated by priority encoders
- Size index, most significant bit index, and mask

Generator
- Generates elements in their element and polynomial forms

Operators
- Performs addition, subtraction, multiplication, division and logarithm of Galois operands
- Checks null errors

Control unit
- Determines operations requested through 6-bit opcode
- Converts operands into their counterpart forms if necessary
- Checks operand memberships ($x \in GF[x](2)$) and null operands ($x = \emptyset$)

IO Handler
- Handles all communication between GFAU and external device
- Simple parallel protocol and scalable IO bus make communication fast and flexible

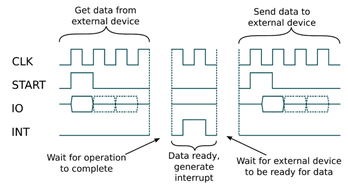

Figure 3: Timing Diagram

Memory wrapper
- Handle memory read and write requests from the generator, operators and control unit

Results

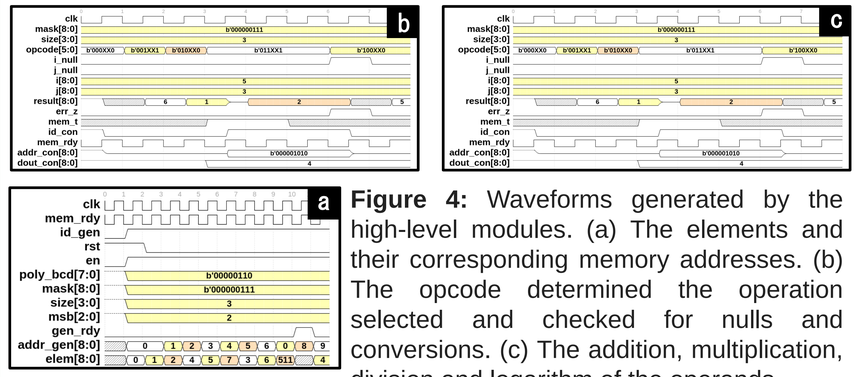

Figure 4: Waveforms generated by the high-level modules. (a) The elements and their corresponding memory addresses. (b) The opcode determined the operation selected and checked for nulls and conversions. (c) The addition, multiplication, division and logarithm of the operands.

Conclusion

The GFAU achieved the objectives with minimal hardware usage. The design was parameterized to be both scalable and inexpensive enough to interface with microcontrollers. Future improvements may include: generate the multiplicative inverse of an element, check if input polynomials are primitive in linear time, and further optimize the overall hardware usage.

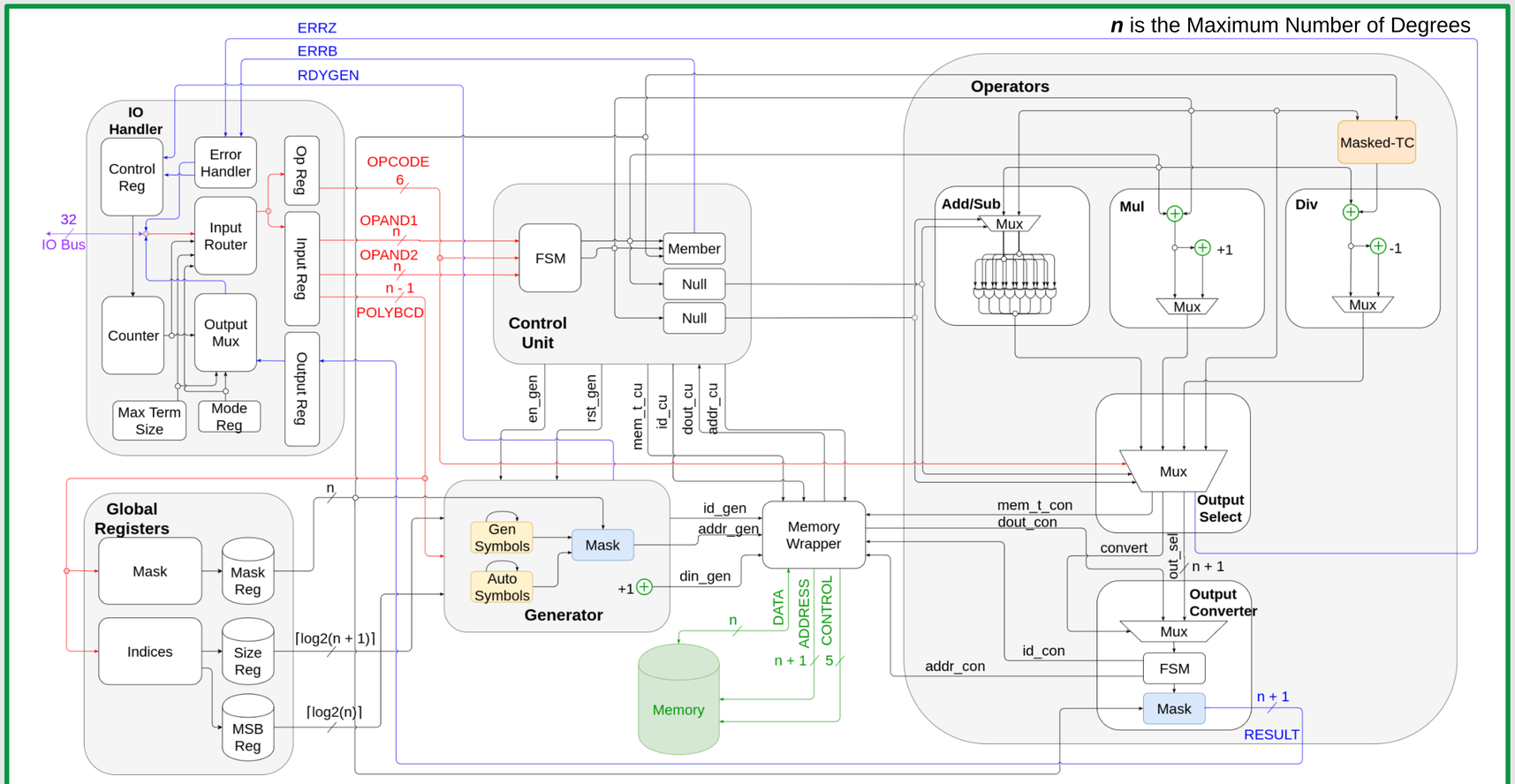

Figure 4: GFAU Block Schematic