

General Purpose FIR Filter Arrays Using Optimized Redundancy Over Direct Product Polynomial Rings

M. Shahkarami, G.A. Jullien, W.C. Miller, R. Muscedere
VLSI Research Group

University of Windsor, Ontario Canada N9B 3P4

mshahka@uwindsor.ca, jullien@uwindsor.ca, miller@uwindsor.ca musced3@uwindsor.ca

Abstract

This paper presents a floorplan for designing FIR filter arrays, using enhanced Fermat ALUs, complementing previous work done in the VLSI Group. The structure is based on polynomial mapping of the Modulus Replication RNS, with computational modulus 257. It exploits the redundancy in the input representation, thus reducing the coefficient growth due to the polynomial multiplication, leading to a smaller probability of overflow error. The silicon area overhead for the input/output mappings is less than 10%. This a great reduction compared to conventional and recent RNS designs of inner product processor array for DSP applications. A 0.35 μm process implementation of a 53 tap filter is detailed.

1. Introduction

The advantages of computing over cross product rings are well documented. From a VLSI view it includes reductions in clock skew, fault tolerance and ease of testability. Form a computation stand point it promises high speed arithmetic by removing the carry.

Conventional finite ring mapping are based on Residue Number System (RNS), where the input and coefficients are mapped to the residues via a modulo reduction operator for each moduli. The operation are then performed component wise for residues of like moduli. The final result is mapped back using the Chinese Remainder Theorem (CRT). The dynamic range is limited to the product of the moduli.

In the Modulus Replication Residue Number System (MRRNS), the computational moduli may be replicated, increasing the dynamic range. This is as a result of mapping data as polynomials, by selecting indeterminates equal to powers of two. The coefficients of the polynomial will be bound by the product of the moduli, but the actual dynamic range will be much greater.

2. Modulus Replication

Here we provide a brief technical introduction to the replication technique. We assume a signed binary representation of the integer data:

$$s = \sum_{i=0}^{B-1} s_i 2^i \quad (1)$$

where $s_i \in (-1, 0, 1)$. We may rewrite eqn. (1) using a polynomial of the form given in eqn. (2):

$$s = \sum_{i_1=0}^{d_1} \sum_{i_2=0}^{d_2} \dots \sum_{i_n=0}^{d_n} s_{i_1, i_2, \dots, i_n} 2^{(i_1 \beta_1 + i_2 \beta_2 + \dots + i_n \beta_n)} \quad (2)$$

where $B = \beta_0 > \beta_1 > \dots > \beta_n$. We thus represent the number as polynomials with indeterminates representing weights of powers of 2. There is a double redundancy present in this representation: the signed digit redundancy of eqn. (1) and the polynomial redundancy of eqn. (2).

The polynomial representation is now mapped to a finite polynomial ring (modulo M) and then to a direct product ring, $Z_M \times Z_M \times \dots \times Z_M$, using an evaluation map; this is implemented by multiplying the finite polynomial ring coefficient vector by a Vandermonde matrix of all the possible roots of the mapping ideal, [5]. The ideal should have greater degree than any resulting polynomial after computation, so that no actual reduction takes place when we invert the direct product map (the reduction is formal). We can now perform independent calculations over each of the copies of Z_M .

Conditions for reversing the mapping procedure are discussed in [5]. An important restriction is that the resulting finite polynomial ring coefficients do not exceed the modulus, M , during the computations, and we have discussed applications using both very small moduli [5] and large moduli [2]. In the latter case we introduced the Fermat ALU in which the ring modulus was $M = 257 \times 17$; both prime factors being Fermat primes. The ALU operates over the *half-index domain* using index calculus for the multiplication and Leibowitz's diminished-ones representation [6] for the accumulation. This was found to be a better solution than the Zech Logarithm [7] approach recently adopted by Zelniker and Taylor [8] for their *Gauss machine* [9].

For completeness, a block diagram of the original Fermat ALU is shown in Fig. 1 for the Mod 257 computational channel. Since we are using index calculus with a Fermat prime, the index addition uses a binary adder (8-bits for 257 and 4-bits for 17). The diminished ones accumulation also

uses a binary adder with zero mapped to the “overflow” value; this is easily handled with a small amount of extra logic.

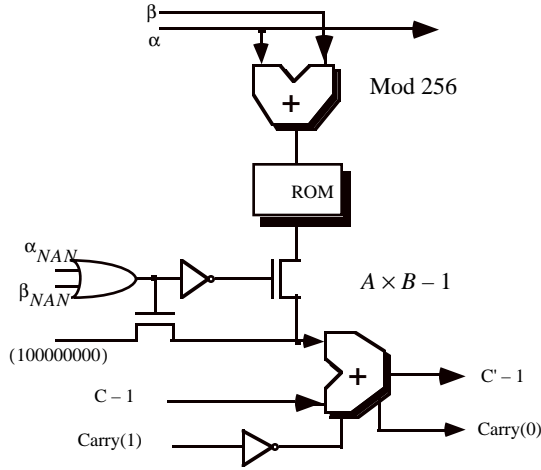


Fig. 1 Fermat ALU mod 257

In the following section, we will demonstrate exploiting the double redundancy in the polynomial map to reduce the computational modulus to 257. Furthermore we will present a floorplan for the design of FIR array based on Fermat ALU discussed in [2].

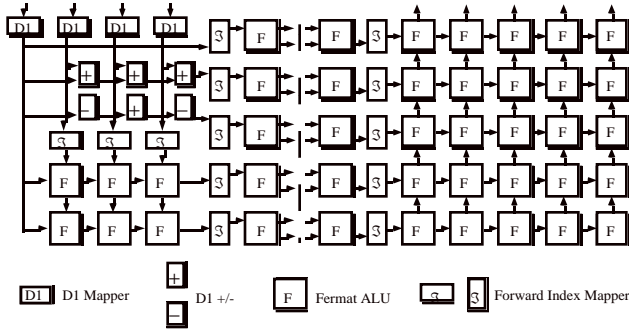


Fig. 2 Floorplan of original Fermat 257 Array

3. Enhanced Input Mapping

For a chosen indeterminate $X=2^r$ the simplest polynomial mapping of the binary data, would be to group the data into r -bit segments, each segment representing a coefficient. In this case the range of the coefficients will be $[-X+1, X-1]$

The redundant property of the polynomial map is used to optimize the input data representation. If we allow a mix of positive and negative coefficients to represent any number, regardless of sign, we can reduce the maximum value of the coefficient by as much as half (this is analogous to using CSD representation to reduce the number of non-zero digits). In order to achieve a reduction in the coefficients we may have to increase the degree of the polynomial. In the

enhanced input mapping, the input polynomial will be at most one degree higher than the simple mapping method and the coefficient of this higher degree indeterminate will only be 1 or -1. This means that in the input mapping process, the trivial map of the binary number to a polynomial will be replaced with a lookup table to select the coefficient. The following mapping stages will not change in spite of the possible increase in the degree of the polynomial. The effect of the extra coefficient is realized by a polynomial addition, the results of which will be summed with the output of the final addition stage. This approach allows us to implement reasonable filter lengths using only a Mod 257 ALU; in practice this means that we can implement the equivalent of a 24-bit FIR data path with completely independent, identically constructed, 8-bit computational paths. The effect on clock spike reduction and ease of testing are well documented.

Let use represent the input polynomial from the simple map as below:

$$a_2x^2 + a_1x^1 + a_0x^0 \quad (3)$$

The conversion process is performed in sequence, starting from the ‘least significant’ coefficient; the ROMs store the following function (Fig. 3):

$$\begin{aligned} \text{for } (|a_i| < 4) \quad & a'_i = a_i \quad C_i = 0 \\ \text{for } (|a_i| > 4) \quad & a'_i = \pm X + a_i \quad C_i = \pm 1 \end{aligned} \quad (1)$$

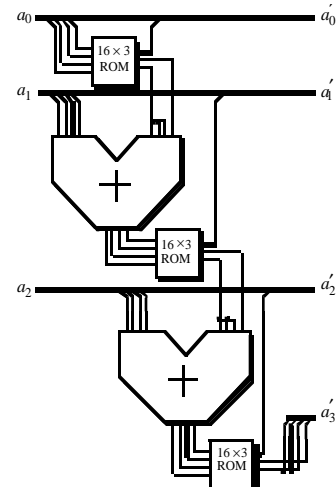


Fig. 3 Block diagram of the input polynomial mapper

The Fermat ALU performs inner product computation, other words polynomial multiplication and addition. With the simple mapping the input polynomials are of degree 2, hence the multiplication will result in polynomial of degree 4. With the enhanced mapping the polynomial may be of degree 3 and so the product polynomial will be of a maximum degree of 6. But this multiplication can be represented in terms of a second degree polynomial multiplication as follows:

$$\begin{aligned}
 A(x) &= a_2x^2 + a_1x^1 + a_0x^0 \\
 B(x) &= b_2x^2 + b_1x^1 + b_0x^0 \\
 (a_3x^3 + A(x))b_3x^3 + B(x) & \text{ where } a_3, b_3 \in \{-1, 0, 1\} \\
 a_3b_3x^6 + a_3(B(x)) + b_3(A(x)) + A(x)B(x)
 \end{aligned}
 \tag{4}$$

The first three terms are simply a polynomial addition. The last term is a polynomial multiplication of two second degree polynomial, the same operation that the Fermat ALU performs. The polynomial addition is performed outside of the RNS system and summed with the final addition stage result. This allows to use the existing multiply accumulators (MACs) while increasing the dynamic range of the computation. A block diagram of the new FIR array is shown in Fig. 4

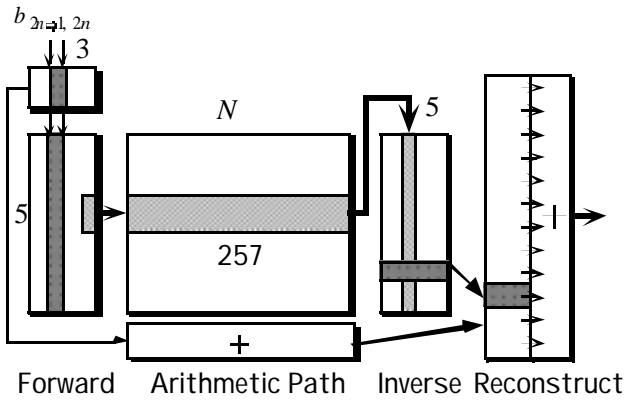


Fig. 4 New floorplan of Fermat ALU Array

4. Hardware Design

In this section we will discuss the hardware requirements and changes made to reduce the area of the Fermat ALU, focussing on low power architectures. In the original design a 256x8 dynamic ROM was used which exhibits much lower power dissipation compared to static ROMs. The purpose of the ROM is to lookup the inverse index in diminished-1 representation. Since the ROM occupies the most area in the Fermat ALU any reductions in its size will greatly affect the overall design area. With this in mind the contents of the ROM were examined to see if a pattern existed in lookup table. As it turns out the first half of the lookup table is a bit inverse of the second half. This effectively means that only half the look up table needs to be stored in the ROM, with the inverse or true values being selected by the most significant bit (Fig. 5).

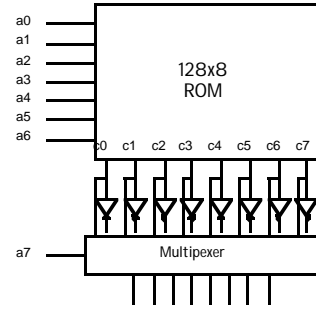


Fig. 5 Minimized ROM

The dynamic ROM was also custom designed to take advantage of the triple metal process, which allowed for a more denser ROM core design. The low power dissipation of the ROMs was achieved through the design of dynamic sense amplifiers and decoder.

Fig. 6 shows the dynamic sense-amp circuit. Signal X is a word line from the decoder. M0 is the selected ROM cell, and M1-31 are the unselected ROM cells, which are parallel connected with M0 to the same bit-line. MN5 and MN6 are column selector, they are connected to VDD, means this bit line are selected by column decoding.

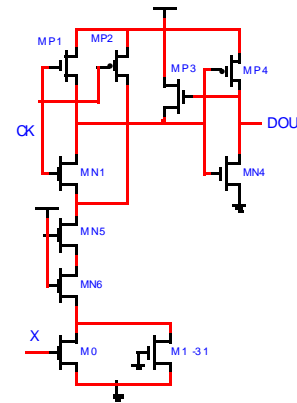


Fig. 6 Dynamic sense amplifier used in the ROM design

The adders used in the Fermat ALU, the polynomial mapper and polynomial adder are dynamic EMODL (Enhanced Multiple Output Domino Logic) adders. These adders feature a low power dissipation, a good speed (when the bit length is not bigger that 16), and design with modularity.

The low power dissipation/good speed attributes of EMODL are due to the elimination the p transistors in the MODL evaluation phase and the use of a sort of new XOR gate. Design modularity means that with the increasing of the bit length, the Domino discharging tree (EMODL tree) grows in modularity. Each extra bit can be grown from the low-bit adder.

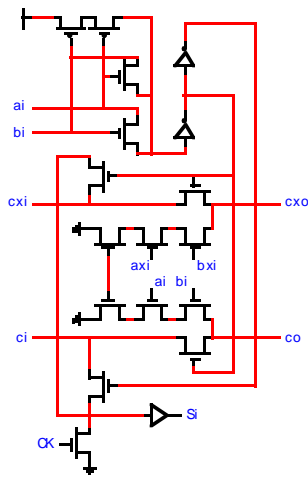


Fig. 7 1 bit position adder module

The small lookup tables for the input polynomial mapper were designed using minimized switching trees, as the layout would be more efficient as shown in Fig. 8.

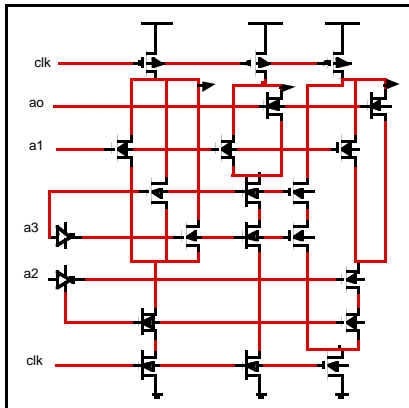


Fig. 8 Switching tree ROM

The final adder evaluates the output polynomial, by substituting the indeterminate and summing the terms and it is implemented by a CSA array.

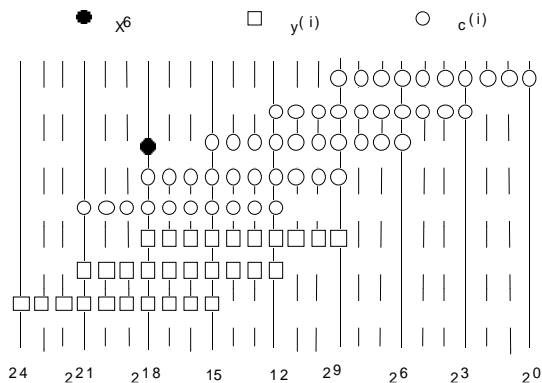


Fig. 9 CSA Array for final polynomial addition

True Single Phase Clock (TSPC) strategy is used throughout the whole design. In the whole design of Fermat unit, all of the stages have the form of Fig. 10. and all of the logic is implemented in the NMOS blocks of the Domino stages which precede the TSPC latch. The transistors in the latch were sized to overcome the self-skewing problem of TSPC designs.

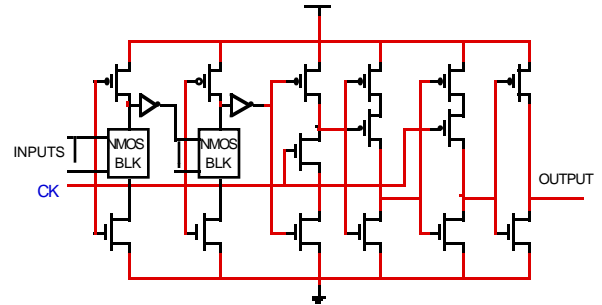


Fig. 10 Domino stages with TSPC latch

Fig. 11 shows the layout of the Fermat ALU designed in a 0.35µm, triple metal process.

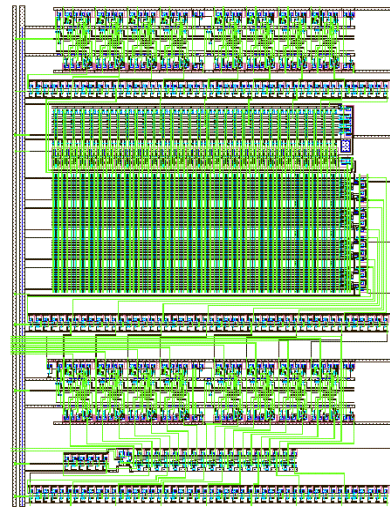


Fig. 11 Layout of Fermat 257

Table 1 shows a power estimation for the various components of the Fermat ALU in both 0.5µm and 0.35µm. The next table shows a comparison of the Fermat IPSP(5 replication of Fermat ALU) with a carefully designed binary multiply/accumulator.

Table 1 Power Estimation

	Power(mW/100MHz)	
	0.5µm	0.35µm
8-bits adder	1.24	1.18
256x8 bits ROM	1.44	0.95
DFF	0.05	0.04
Fermat275	5.50	4.70

Table 2 Power Comparison

	Power(mW/100MHz)	
	0.5 μ m	0.35 μ m
FermatIPSP(257x17)	38.5	32.0
Fermat IPSP (257)	27.5	23.5
MAC with Booth algorithm	41.0	34.0

The output delay of the 8-bits dynamic adder is 3.65ns and 3.25ns for 0.5 μ m and 0.35 μ m designs respectively. As discussed above the speed of adder is the limitation of maximum clock frequency for the pipeline IPSP structure, so the IPSP can work at the maximum clock frequencies 120MHz and 133 MHz for 0.5 μ m and 0.35 μ m designs respectively.

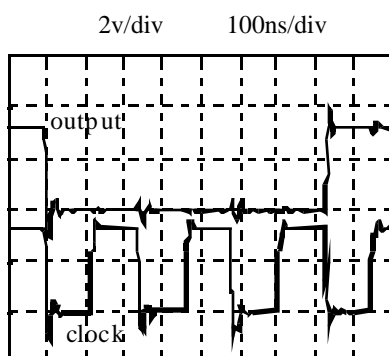


Fig. 12 Low cycle rate test

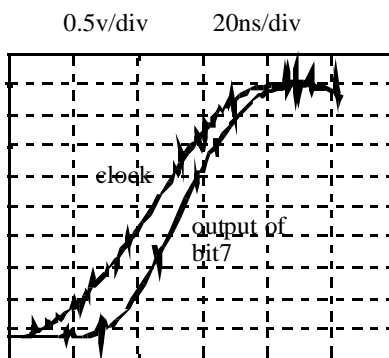


Fig. 13 Delay test

5. Conclusions

6. Acknowledgment

The authors would like to thank the Natural Sciences and Engineering Council (NSERC) of Canada and micronet for their financial support and the Canadian Microelectronics Corporation (CMC) for their equipment loan and fabrication services.

7. References

- [1] Soderstrand, M.A., Jenkins, W.K., Jullien, G.A. and Taylor, F.J., 1986. "Residue Number System Arithmetic: Modern Applications in Digital Signal Processing." IEEE Press. New York, NY.
- [2] G.A. Jullien, W. Luo, and N.M. Wigley, 1996, "High Throughput VLSI DSP Using Replicated Finite Rings", Journal of VLSI Signal Processing, (Invited), 14, pp. 207-220.
- [3] G.A. Jullien, S. Bizzan, N.M. Wigley, 1994, "Using Redundant Finite Rings for Fault Tolerant Signal Processors", Proc. SPIE, Advanced Signal Processing: Algorithms, Architectures and Implementations V, Paper 2296-75 (Invited Session on Algorithmic Fault Tolerance).
- [4] Jullien, G.A., Taheri, M., Bandyopadhyay, S. and Miller, W.C., 1990. "A Low-Overhead Scheme for Testing a Bit Level Finite Ring Systolic Array." *Journal of VLSI Signal Processing.*, Vol 2.3 pp. 131-138.
- [5] N.M. Wigley, G.A. Jullien and D. Reaume, 1994, "Large Dynamic Range Computations over Small Finite Rings." IEEE Trans. on Computers, Vol. 43, No.1, pp. 78-86.
- [6] L.M. Leibowitz, "A Simplified Binary Arithmetic for the Fermat Number Transform." IEEE Trans. Acoustics, Speech and Signal Processing, Vol. ASSP-24, No. 5., Oct, 1976.
- [7] L.-I. Alfredson, Properties of Zech's logarithms over Fermat prime fields, Proc. of the 6th Joint Swedish-Russian Symp. on Information Theory, pp. 310-314, 1993.
- [8] Zelniker, G. and Taylor, F.J., 1991. "A Reduced-Complexity Finite Field ALU." *IEEE Trans. on CAS.*, Vol. 38, No. 12 pp.1571-1573.
- [9] Mellott, J.D., Smith, J.C. and Taylor, F.J., 1993. "The Gauss Machine: A Galois-Enhanced Quadratic Residue Number System Systolic Array." *Proceedings of the 11th IEEE Symposium on Computer Arithmetic.*, Windsor, Canada. pp. 156-162.
- [10] G. A. Jullien, W.C. Miller, R. Grondin, L. Del Pup, S. Bizzan and D. Zhang, 1994, "Dynamic Computational Blocks for Bit-Level Systolic Arrays," *IEEE J. of Solid-State Circuits*, 29, 1, pp. 14-22.
- [11] J. Wang, Z. Wang, G.A. Jullien, S.S. Bizzan, W. Luo and W.C. Miller, "Circuit Driven Delay Optimization of EMODL Carry Lookahead Adders," *Proceedings of the Asilomar Conference on Signals, Systems and Computers*, November, 1994, pp. 550-554
- [12] Afghahi, M. and Svensson, C., 1990. "A Unified Single-Phase Clocking Scheme for VLSI Systems." *IEEE J. Solid-State Circuits*. 25, Feb., pp. 225-233