Galois Field Fourier Transform Implementation Using XOR MUX Full Adder

Dr M SURENDRA KUMAR , B ANUSHA
KLR COLLEGE OF ENGINEERING & TECHNOLOGY

Abstract: **A special technique, the Galois field Fourier transform, is used to convert the symbols from vector representation to energy representation, thus eliminating the need for complexities. On latest FPGA implementation along a full-size exorcism regarding on-chip distributed embedded memory. The suggested approach to this task was to develop a Galois Field Fourier Transform using the fewest possible logic gates in an attempt to reduce the complexity of the FPGA implementation while increasing the factor speed of the resulting circuit. Complete adder design was used instead of the standard complete adder design, and the method was expanded to include Serial In Parallel Out, Parallel In Serial Out, and Optimized Serial In Parallel Out architectures, while still taking into account all relevant parameters in terms of area, length, and power consumption.**

Index Terms: **Galois Field , Cyclotomic Fast Fourier Transform**

I. INTRODUCTION

The Fourier transform on a (finite) Galois field (GFT) is one of the most computationally expensive functions in developing Bose-Chaudhury - Hoquenghem (BCH) and Reed-Solomon (RS) tokens. An obstacle in implementing GFT is the domain-limited multiplicity, thus many approaches have been suggested in the academic literature to reduce the number of complexities. For primes longer than 2, a strategy based on substitution and pre-computation is provided in [7] for computing the GFT over GF(2^m) for arbitrary m, which saves around a fifth of the multiplications compared to a brute-force implementation [2], [5], [8]. In an attempt to cut down on the sheer amount of multiplications required, researchers have looked into implementing GFT in the fashion of the Fast Fourier Transform.

Polynomial equations that can be solved by radicals are characterised by properties of the permutation group of their roots, which can be studied in the Galois field for studying roots of polynomials. If it can be expressed in a formula that uses only integers, nth roots, and the four basic arithmetic operations, the roots of the equation can be solved by radicals. A generic polynomial of degree at least five cannot be solved by radicals, according to this. Fourier transform considers a set of values as a set of coefficients to a polynomial, and then uses that polynomial to evaluate a set of input values {1, α, α^2, ..., α^{n-1}} to produce a new set of values [1]. Inverse Fourier transform or more generally, reverses the process. Consider Fourier transform as a mapping and Inverse Fourier Transform as the inverse mapping of a set of values to another set of values. A Fourier transform can be applied repeatedly, and then Inverse Fourier Transform can be applied the same number of times or vice versa to end up with the original set of values. A Fourier transform or it's inverse will not transform a code word from one encoding scheme to another.

The implementation regarding GFT in accordance with reduce the quantity concerning multiplications. Galois's cyclotomic approach [4] reduces the number of multiplications beyond (n^2) by a factor of (1/4)n($\log_2 n$)^2 on top of GF(p^m), where p is a prime number, and then by n$\log_2$ n due to GF($2^{(2r)}$), where r is an unbounded, i.e. arbitrary, value. cyclotomic Fast Fourier transform (CFFT) hardware design and implementation. Change GF(2^m) on the upper floor by rephrasing the solution given.

II. GALOIS FIELD USING MULTIPLIER

As indicated in Table 1, the main idea behind our action is correct: By converting a GFT number from a vector representation to a force representation in an observational representation method, we can transform a feature using convoluted overhead addition. The Galois Field (Finite Field) of size p" [2] is called GF(p"), or just F for short, if p is a prime number and n is an even number.

If m is a large positive integer and 2^m - 1 = n, then GF(2^m) stands for the Galois field on 2 ^{m}. Let us assume that a vector over GF(2^m) with stability (a_0, a_1,..., a_{n-1}) = a exists, in which case there exists a primitive element in GF(2^m) such that quantity n = 1, or every nonzero component in GF(2^m) can be represented as β_j ,0 ≤ j ≤ n − 1. Since 0 ≤ t ≤ n, the t h component of the GFT for an is the vector (b_0, b_1,..., b_{n-1}) = b with the following form:

$$a_0 \beta^0 + a_1 \beta^t + \ldots\ldots + a_{n-1} \beta^{(n-1)t} = b_t = \sum_{k=0}^{n-1} a_k \beta^{kt} \quad (1)$$

This is the interior multiplication concerning durability $(1, \beta^t, \beta^{2t}, ..., \beta^{(n-1)t})$ yet $(a_0, a_1, ..., a_{n-1})$ [1] . A element or symbol in GF(2^m) may both maintain in power form or its equivalent vector shape(form) demonstrated into Table 1. Although basic XOR gates may be used to build the vector representation for addition in hardware, a hardware implementation of multiplication is more challenging due to the need for a finite field multiplier. However, the representation of force used in multiplication is as follows

Table 1 : GF(2^3) Vector and Power representation for

Primitive Polynomial $p(a) = 1 + a + a^3$

Vector Representation	Power Representation
000	-1
001	β^0
010	β^1
100	β^2
011	β^3
110	β^4
111	β^5
101	β^6

The epsilon exponents β in a row (column) are distinct for each prime of length n. $(k \times t) \bmod (2^m - 1)$ is determined as the powers of β . The row vector of the input is mutable but the values in the squared matrix follow a fixed pattern and are stored as a lookup table in ROM for use in calculations. For example, consider the GFT of vector length 7 with n = 7 and m = 3 in GF (2^3). Calculated values and matrix notation using Eq. (1) are as follows.

$$[b_0\ b_1\ b_2\ \ldots\ldots\ b_{n-1}] = [a_0\ a_1\ a_2\ \ldots\ldots\ a_{n-1}]$$

$$x \begin{bmatrix} \beta^0 & \beta^0 & \beta^0 & \ldots\ldots\ldots\beta^0 = 1 \\ \beta^0 & \beta^1 & \beta^{2\,x\,1} & \ldots\ldots\ldots\beta^{(n-1)\,x\,1} \\ \beta^0 & \beta^2 & \beta^{2\,x\,2} & \ldots\ldots\ldots\beta^{(n-1)\,x\,2} \\ : & & & \\ \beta^0 & \beta^{(n-2)} & \beta^{2\,x\,(n-1)} & \ldots\ldots\beta^{(n-1)\,x\,(n-2)} \\ \beta^0 & \beta^{1\,x\,(n-1)} & \beta^{2\,x\,(n-1)} & \ldots\ldots\ldots\beta^{(n-1)\,x\,(n-1)} \end{bmatrix} \quad (2)$$

Galois Field GF(2^3) where $2^3 = 8$ i.e. total of 8 elements represented as

$$GF(2^3) = \{0, 1, \beta^1, \beta^2, \beta^3, \beta^4, \beta^5, \beta^6\}$$

Where 0 is the zero element in the group and remaining are the nonzero element in the group, primitive element β. Minimal, generator polynomials of BCH and RS codes may be computed with the help of primitive elements. On the off chance that the most un-positive number n by which p(x) separates is not exactly m, then, at that point, p(x) is supposed to be a basic polynomial of degree m. $x^n + 1$ is $2^m - 1 = n$.

Table 2 Power representation of $\beta^j * \beta^z = \beta^{j+z}$

j \ z	0	1	…	n-1
0	β^0	β^1		β^{n-1}
1	β^1	β^2		$\beta^n=1$
2	β^2	β^3		β^{n+1}
:				
n-1	β^{n-1}	β^0		β^{n-2}

Table3 Power representation of β^{j+z} for GF(2^3)

$\beta^0 = \beta^7 = 1$

i \ z	0	1	2	3	4	5	6
0	β^0	β^1	β^2	β^3	β^4	β^5	β^6
1	β^1	β^2	β^3	β^4	β^5	β^6	1
2	β^2	β^3	β^4	β^5	β^6	1	β^1
3	β^3	β^4	β^5	β^6	1	β^1	β^2
4	β^4	β^5	β^6	1	β^1	β^2	β^3
5	β^5	β^6	1	β^1	β^2	β^3	β^4
6	β^6	1	β^1	β^2	β^3	β^4	β^5

Table 4 Vector representation of β^{j+z} for GF(2^3)

β^j / z	1	2	4	3	6	7	5
0	1	2	4	3	6	7	5
1	2	4	3	6	7	5	1
2	4	3	6	7	5	1	2
3	3	6	7	5	1	2	4
4	6	7	5	1	2	4	3
5	7	5	1	2	4	3	6
6	5	1	2	4	3	6	7

Traditionally, a GF multiplicator has been used to do the calculation ak x β^t, where both ak and β^t have been represented as vectors. The suggested hardware implementation involves a transformation from the vector representation of the input (symbol) to the power representation of each ak and β^t, as illustrated in Table 1. Wrap round carry-addition is then used to add the abilities. After converting back to vector form from the power representation, the rule illustration is complete. Imagine you are doing the calculation for ak$*$ β^5 where ak = 010 and β^5 = 111. Instead regarding par GF quality we desire convert ak of its limit representation beside the vector representation as hand over us β^0 according after Table 1. So, ak$*$ β^5 turns into $\beta^1 * \beta^5$ as is β^6 among the rule representation. Using Table 1 the 101 vector representation is converted through its β^6 power representation.

$$a_0\beta^0 + a_1\beta^0 + a_2\beta^0 + a_3\beta^0 + a_4\beta^0 + a_5\beta^0 + a_6\beta^0 = b_0$$

$$a_0\beta^0 + a_1\beta^1 + a_2\beta^2 + a_3\beta^3 + a_4\beta^4 + a_5\beta^5 + a_6\beta^6 = b_1$$

$$a_0\beta^0 + a_1\beta^2 + a_2\beta^4 + a_3\beta^6 + a_4\beta^1 + a_5\beta^3 + a_6\beta^5 = b_2$$

$$a_0\beta^0 + a_1\beta^3 + a_2\beta^6 + a_3\beta^2 + a_4\beta^5 + a_5\beta^1 + a_6\beta^4 = b_3$$

$$a_0\beta^0 + a_1\beta^4 + a_2\beta^1 + a_3\beta^5 + a_4\beta^2 + a_5\beta^6 + a_6\beta^3 = b_4$$

$$a_0\beta^0 + a_1\beta^5 + a_2\beta^3 + a_3\beta^1 + a_4\beta^6 + a_5\beta^4 + a_6\beta^2 = b_5$$

$$a_0\beta^0 + a_1\beta^6 + a_2\beta^5 + a_3\beta^4 + a_4\beta^3 + a_5\beta^2 + a_6\beta^1 = b_6$$

III. GALOIS FIELD USING XOR MUX

In this work, two designs—serial in parallel out and parallel in serial out—are proposed, and all of their characteristics—including area, reaction time, and power—are compared. 1, 2, and 3. Up to the approximation level of RS and BCH codes, the Galois Field Fourier Transform (GFT) is a fundamental operation in signal processing or digital communications applications. Even while there are several ways to decrease the number of finite field multipliers needed to create a resilient FPGA, this is particularly true in this case. We demonstrate whether on-chip intelligence can effectively implement GFTs in this application area without turning to complexity..

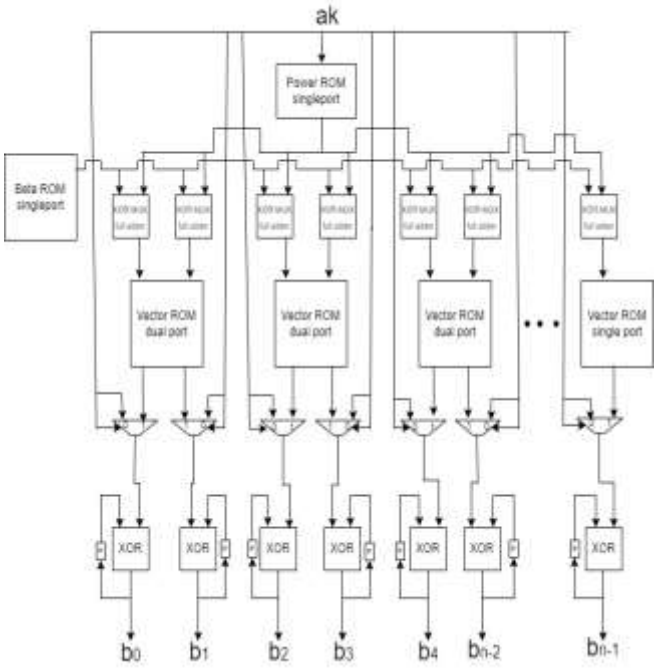


Figure 1 Serial In Parallel Out (SIPO)Architecture

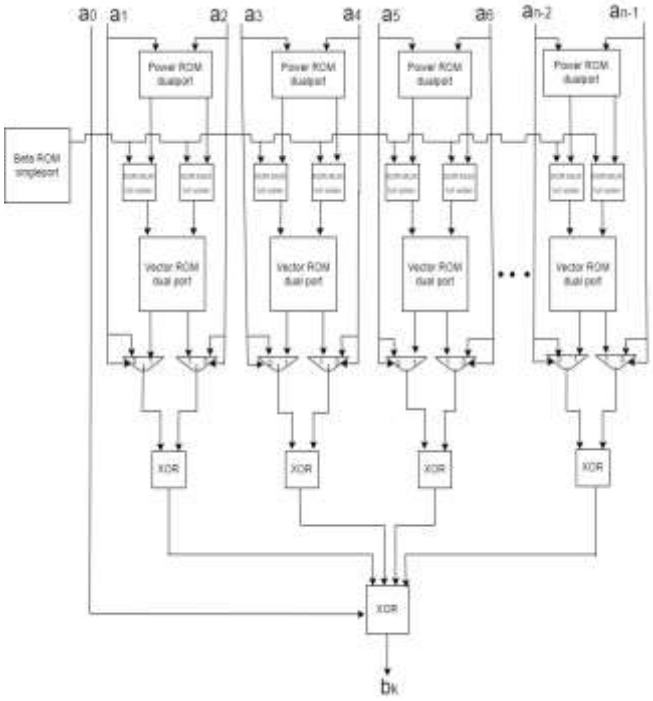


Figure 2 Parallel In Serial Out (PISO) Architecture

We demonstrate that the suggested designs are successful after factor-based overall implementation with regards to latency, throughput, power consumption, and register/LUT needs. An a-by-t-vector b (b_0, b_1,..., b_{n-1}) is the IGFT, where t is the t-th component, because $0 \leq t \leq n$ is addicted by, $b_t = \sum_{k=0}^{n-1} a_k \beta^{-kt} = a_0 \beta^0 + a_1 \beta^{-t} + \ldots\ldots + a_{n-1} \beta^{-(n-1)t}$ hence the same structure then implementation scheme and perform IGFT as much well.

Inputs and outputs are handled differently in the two suggested architectures, Serial In Parallel Out (SIPO) Fig1 and Parallel In Serial Out Architecture (PISO) Fig2. The inputs a0 and a1 occur in serial in the SIPO architecture (shown in Fig. 1). PowerROM changes the vector format into the power format. The ROM has a depth of n + 1, but its breadth is just m = $\log_2(n + 1)$. PowerROM size = (n + 1) $\log_2(n + 1)$ difficulty. The inputs and the BetaROM's contents are intended to be supplied simultaneously. BetaROM has a size of (n+1)($\log_2$(n+1)n). The BetaROM settings are predetermined in both

architectures. Wrap-around carry summation, in which the carry is brought back in accordance to arrive at the final result, is performed by the adder block. Then, the VectorROMs will adapt their side-control mechanism to resemble a vector graphic. Each VectorROM has a size of $(n + 1) \log_2(n + 1)$. With a (n/2)-fold increase in memory size for each port, dual-port VectorROMs provide unprecedented flexibility. The multiplexers receive the output of the VectorROM. When the input is 0, the Multiplexer chooses the output over the VectorROM, and vice versa. XOR gates are used in the design to collect these by using Galois subject addition.

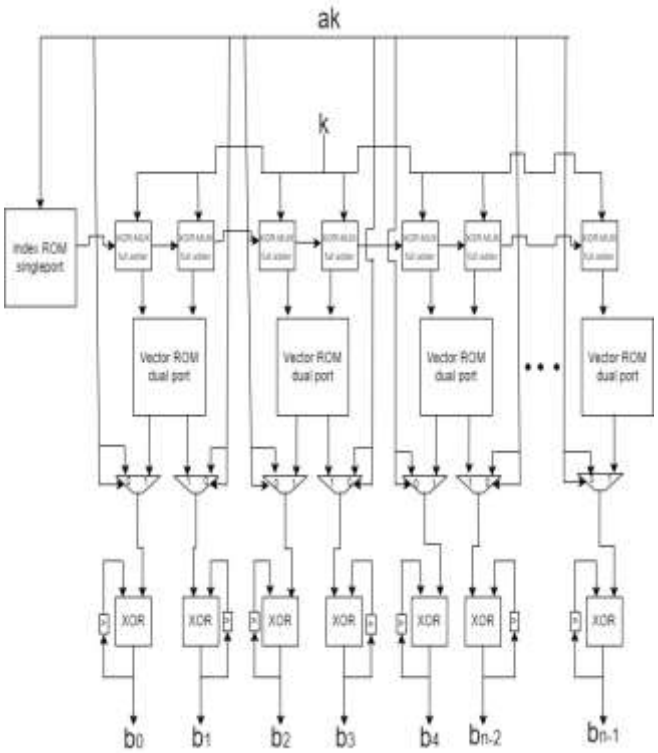


Figure 3 Optimized Serial In Parallel Out Architecture

The cyclic feature on Galois field is used to restrict the suggested architectures' use of device memory while computing GFTs with prime lengths. The notion right here is relating to build a table about the quality implications regarding $ak*\beta^{kt}$ among (1) above $GF(2^m)$ are diverse values. A $GF(2^m)$ item may still be written as β_j, as stated before ak. For simplicity such as $\beta^{kt} = \beta^z$. Table 2 illustrates the limit representation for $\beta^j * \beta^z = \beta^{j+z}$ because of $0 \leq z, j < n$ or Table 3 illustrates the $\beta j+z$ since $GF(2^3)$ then primitive polynomial $p(a) = 1 + a + a^3$. The result is also zero when $\beta_j = 0$. Take note that the results for $1 \leq j < n$ in Table 2 are derived by cyclic variation on the j = 0 column with j as the varying variable. Tables 2 and 3 each provide many columns with unique numerical values. Since this adoration is genuine, n must be first. The β^{j+z} s vector representations for Table 3 are shown in Table 4. The material about each pillar is the shifted version over the 2nd coloumn (that pertains in imitation of $\beta^0 = 1$), via j. For instance the input is equivalent to 5, the second column is moved by means of 6, owing to the fact the power representation about 5 is β^6. The vector representation of 0–6 that we have access to in the VectorROM is shown in the second column of Table 4. As a result, in an improved technique, betaROMs or PowerROMs can still be removed from the implementation shown in figure 1 and modified with a smaller ROM of size n containing pointers around the vectorROM. We named this structure IndexROM. So, if ak, k ,j is given, we look at the index value above the input in IndexROM, which is really th k item relative to IndexROM, then according to ak k j s due to $1 \leq j < n$, we just compile the index after k j and finally look at VectorROM to get the result. Since $GF(2^3)$, or the example below, highlights the key elements of the optimization, we can see the optimised design in Fig. 3. Let's pretend ak = 2, and type in k = 3. To replicate the results of Equation (1), we must continue multiplying a3 by $[\beta^0, \beta^3, \beta^6, \beta^2, \beta^5, \beta^1, \beta^4]$. In order to improve upon the functionality of the VectorROM, we develop IndexROM as a carrier of indices. Here is a rundown of everything you'll find in both the VectorROM and the IndexROM. You may write VectorROM as [1, 2, 4, 3, 6, 7, 5], and IndexROM as [1, 2, 4, 3, 7, 5, 6]. In IndexROM, this position is indicated by the number 2, as a3 = 2. Since VectorROM[2]=2, the vector portrayal of a3=0 is the second feature of VectorROM. For the rest of be determined, the moved worth, k j, should be embedded after 2. (the index). Since 5 is the result of adding the index based on k (k = 3), we get VectorROM [5]= $a3 * \beta^3 = 6$. Similarly, adding 2k to the index yields 1 when modulo 7 is applied, yet VectorROM[1]=1 since $a3 * \beta^6 = 1$.

IV. COMBINATIONAL CIRCUITS

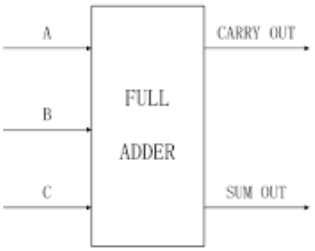
WRAP AROUND CARRY ADDITION

Wrap Around Carry run-on includes the blocks on full adders or half adders the place the output disclosed via the full adder is wrapped round i.e. given feedback with the same whole adder or the generated output is given as much input in accordance with the half adder. The Table 5 indicates about wrap around carry.

Table 5 Three-bit binary fixed-point numbers in two's complement.

Decimal	Binary
-4	100
-3	101
-2	110
-1	111
0	000
1	001
2	010
3	011

An adder that takes in two operand bits and a carry among bit is called a full adder. The Carry among (Cin) bit, enter operand bits A and B, and generates the Sum output (Sout) bit and carry output (Cout) concerning a complete adder designed using XOR , AND , OR gate .The internal block of Full Adder contains two half adders which generates the outputs.



A	B	C	SUM OUT	CARRY OUT
0	0	0	0	0
0	0	1	1	0
0	1	0	1	0
0	1	1	0	1
1	0	0	1	0
1	0	1	0	1
1	1	0	0	1
1	1	1	1	1

Figure 4 Block Diagram and Truth Table of Full Adder

MULTIPLEXER:

A multiplexer (or mux; sometimes spelt like multiplexer) is a device that takes in a large number of analogue or digital signals and routes just the ones that are picked to a single output line. A different group of digital inputs, called select lines, guides the decision-making process. As can be seen in Fig.5, a multiplexer with 2n inputs and n selection lines may be used to selectively transmit signals from any of the inputs to the output.

Table 6 Truthtable of MUX 2 x1

Selection line(s)	Output(y)
0	D0
1	D1

Figure 5 MUX 2 x 1

MOD XOR:

The flip-flop, xor-block, and feedback link are all carried by the Mod XOR gate. Binary integers are added up, one digit at a time, using modulo 2 arithmetic. Each digit is considered separately from the others. Credit card and phone numbers are no longer shared or borrowed. Addition by modulo 2 is carried out by performing an exclusive OR (XOR) operation on the appropriate binary digits of each operand.

V. RESULTS

Schematic:-

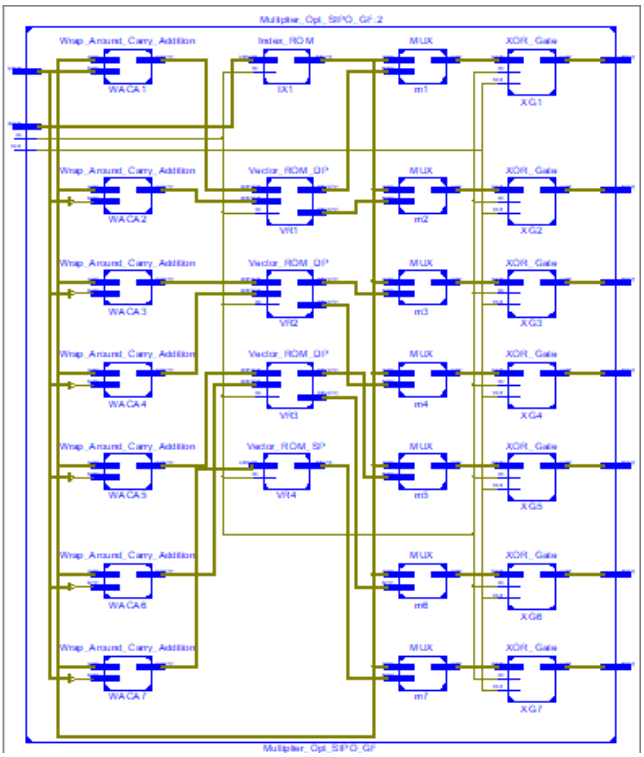


Figure 6. RTL Schematic of Multiplier opt SIPO GF

The RTL schematic is specifically condensed at the daybook movement stage in order to identify the purposeful drawing in accordance with the best engineering as we work to develop. This permits the engineering-related graph. A rundown of the engineering is given in imitation of how the functionality is defined by the use of the coding language, Verilog or VHDL, even though the HDL sound is employed for trading on the account. The RTL diagram also identifies the intestine connecting blocks for better research. The parent mentioned below makes a recommendation for the RTL schematic layout of the suggested engineering.

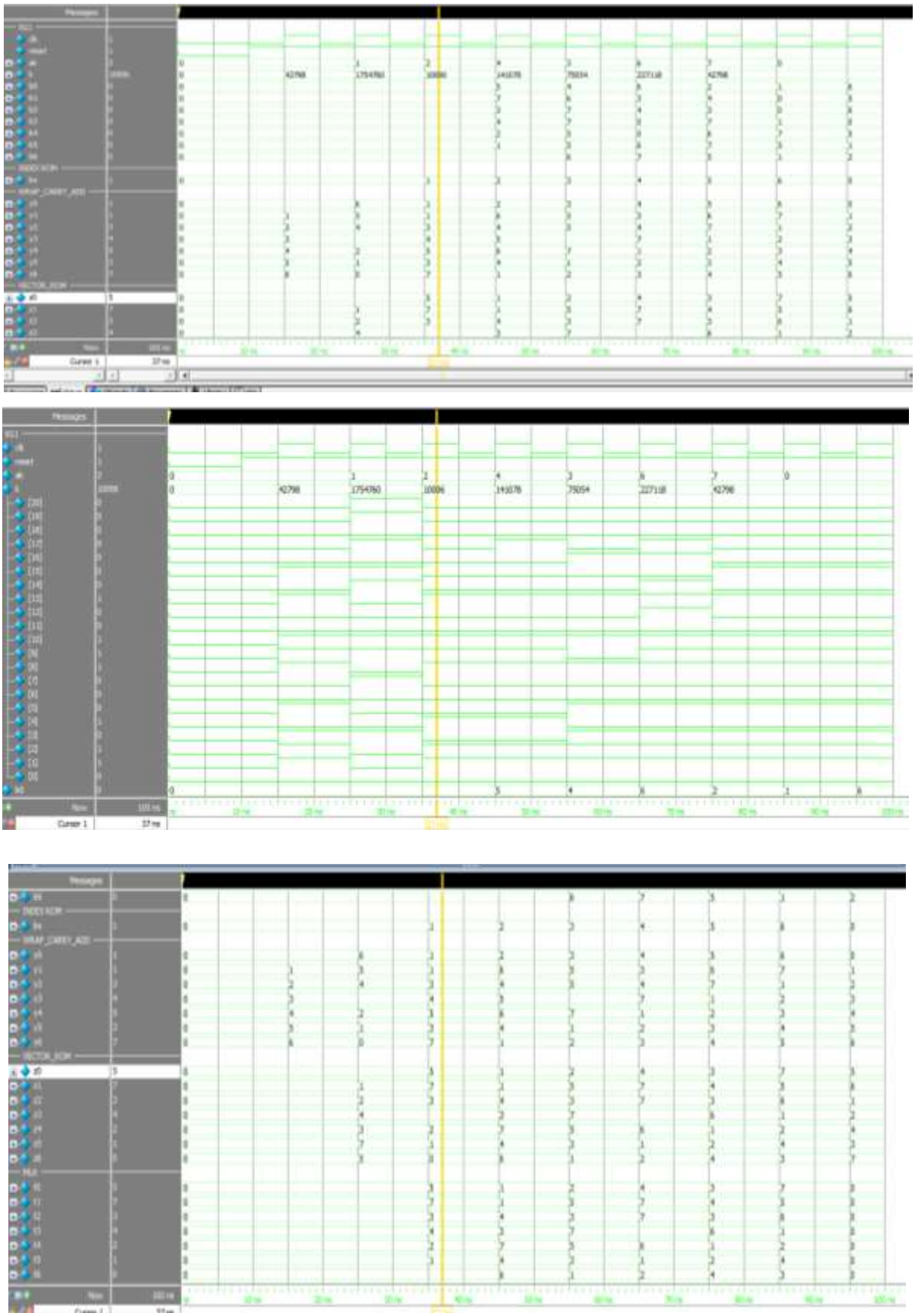


Figure 7 Simulated Wave form of Multiplier Opt SIPO GF

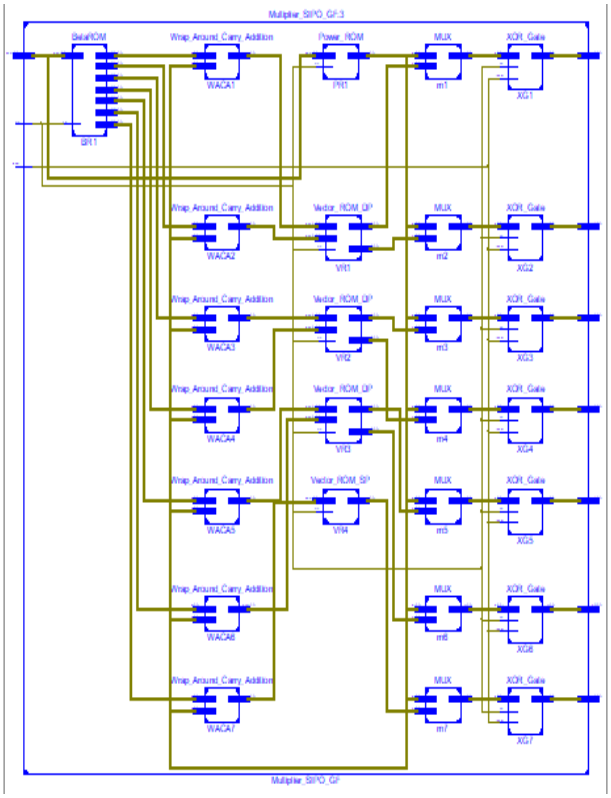


Figure 8. RTL Schematic of Multiplier SIPO GF

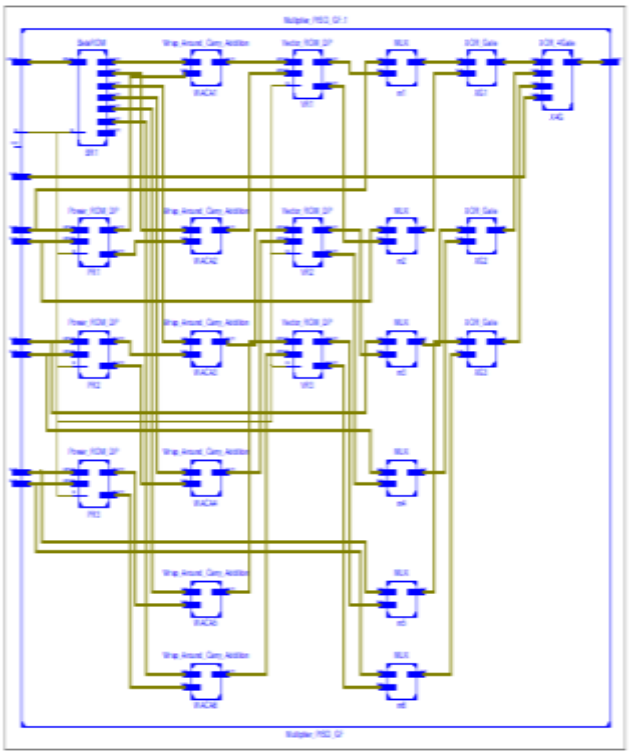


Figure 9. RTL Schematic of Multiplier PISO GF

Table 7: Comparision parameters

	Analyzing Vertex -5 FPGA Multiplier-free Galois Field Fourier Transform Implementations					
	SIPO Architecture		PISO Architecture		Optimized SIPO Architecture	
	XoR-Mux Full Adder	Conventional Full Adder	XoR-Mux Full Adder	Conventional Full Adder	XoR-Mux Full Adder	Conventional Full Adder
Slice Registers	21	21	0	0	21	21
Slice LUTs	42	98	32	75	42	84
Occupied Slices	23	46	14	30	22	42
Number of IOBs	26	26	28	28	47	47
Block RAMs	3	4	4	4	3	3
Max Freq (MHz)	500	500	500	500	500	500
Delay (ns)	0.894	4.331	5.471	5.139	0.898	0.898
Power (mW)	3.295	3.295	3.294	3.294	3.296	3.296

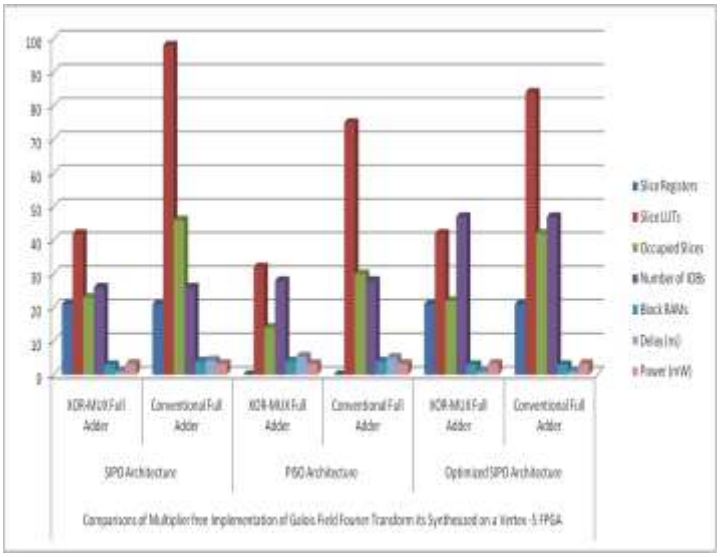


Figure 10. Comparision view

Consider in VLSI the boundaries treated are region, deferral, recurrence and power, in view of these boundaries one can pass judgment on one design to another. Here the thought of region and power utilizations are viewed as the boundaries are gotten by utilizing the instrument XILINX 14.7 and the HDL is Verilog language. When recurrence is something else for any plan it will speed up plan as shown in Table 7.

VI. CONCLUSION

Whether you're dealing with analogue signals or digital ones, the Galois Field Fourier Transform (GFT) is an integral part of any signal processing or digital communication workflow using RS or BCH codes. Though in that position have been several breakthroughs inside decreasing the wide variety regarding multipliers necessary in imitation of compute the GFT that is nonetheless a bottleneck in particular when that occurs after FPGA implementation. We demonstrate the limits of on-chip embedded memory's ability to persistently store data necessary to enforce GFTs without resorting to

multipliers. We demonstrate that the suggested designs comply well with the register then LUT specifications for latency, throughput, and governmental blasting in collecting using a multiplier as the primary implementation mechanism. Because zero $\leq t \leq n$ is dependent on, durability lifespan, the t th component of the vector $b = (b_0, b_1,..., b_{n-1})$ describing an is the Inverse Galois Fourier seriously change (IGFT) on a, $b_t = \sum_{k=0}^{n-1} a_k \beta^{-kt} = a_0 \beta^0 + a_1 \beta^{-t} + \ldots\ldots + a_{n-1} \beta^{-(n-1)t}$, consequently the same structure and implementation plan do keep back because of IGFT as properly.

REFERENCES

[1] "Low complexity bit parallel architectures for polynomial basis multiplication over gf (2m)," IEEE Transactions on Computers, vol. 53, no. 8, pp. 945-959, 2004.

[2] "Prime factor cyclotomic fourier transforms with reduced complexity over finite fields," in Signal Processing Systems (SIPS), 2010 IEEE Workshop on, pp. 450-455, IEEE, 2010. X. Wu, Z. Yan, N. Chen, and M. Wagh.

[3] Dr. Rameshwar Rao and T. Esther Rani, "Area and Power Optimized Multipliers with Minimum Leakage," in 3rd International Conference on Electronics Computer Technology (ICECT 2011).

[4] "On algorithms and complexities of cyclotomic fast fourier transforms over arbitrary finite fields," IEEE Transactions on Signal Processing, vol. 60, no. 3, 2012, pp. 1149–1158.

[5] S. Fedorenko and P. Trifonov, "A method for fast computation of the Fourier transform over a finite field," Problems of Information Transmission, vol. 39, no. 3, pp. 231-238, 2003.

[6] "Iterative soft-decision decoding of reed-Solomon codes of prime lengths," Information Theory (ISIT), 2017 IEEE International Symposium on, pp. 341-345, IEEE, 2017. S. Lin, K. Abdel-Ghaffar, J. Li, and K. Liu.

[7]International Journal of Communication and Computer Technologies Retheesh.D " Analysis on FPGA Designs of Parallel High Performance Multipliers" Volume 02, Issue: 01 Page 11

[8] W. Gappmair, "An efficient prime-length dft algorithm over finite fields gf (2m)," Transactions on Emerging Telecommunications Technologies, vol. 14, no. 2, 2003, pp. 171–176.

[9] On the fpga implementation of the fourier transform over finite fields gf (2m), Y. Louet, A. Nafkha, and J. Palicot, Communications and Information Technologies, 2007. International Symposium on, pp. 146-151, IEEE, 2007.

[10] "A fast algorithm for the fourier transform over finite fields and its vlsi implementation," IEEE Journal on Selected Areas in Communications, vol. 6, no. 3, pp. 572-577, 1988.

[11] "Fpga designs of parallel high performance gf (2233) multipliers.," in ISCAS (2), pp. 268-271, Citeseer, 2003. C. Grabbe, M. Bednara, J. Teich, J. von zur Gathen, and J. Shokrollahi.

[12] Esther Rani Thuraka and Rasmitha Thota, "CMOS Processing Using Non-Volatile Memory for Energy Harvesting Applications," JASC: Journal of Applied Science and Computations Volume V, number XI, November 2018, pages 2503-2514, ISSN NO: 1076-5131.

"The interaction algorithm and practical fourier analysis," Journal of the Royal Statistical Society, Series B, pp. 361-372, 1958.