ISSN 2047-3338

Implementation of FNT Processor for OFDM-Based Communication System

S. Ram Babu, Rajeswara Mahidar P., T. Narendra Babu and T.J.V.S. Rao
Sasi Institute of Tech and Engineering, Tadepalligudem, Andhra Pradesh-534101, India

***Abstract*– Orthogonal frequency division multiplexing (OFDM) is becoming the chosen modulation technique for wireless communications. OFDM can provide large data rates with sufficient robustness to radio channel impairments. Multiple Carrier Modulation (MCM) technology using FFT and IFFT. The Discrete Fourier Transform (DFT) plays an important role in DMTIOFDM systems. Because of the computational complexity of N-points DFT is $O(N^2)$,it takes large operation time and power consumption to perform DFT directly, especially in large transform size. Hence, various FFT algorithms have been proposed to reduce the computational complexity This paper presents a new VLSI architecture for computing the *N* point discrete Fourier transform (DFT) of real data and the corresponding inverse (IDFT) based on Fermat Number Transform (FNT).**

***Index Terms*– OFDM, FFT, IFFT and FNT**

I. INTRODUCTION

ORTHOGONAL Frequency-Division Multiple Access (OFDMA) is a multi-user version of the popular Orthogonal frequency-division multiplexing (OFDM) digital modulation scheme.[1] Multiple access is achieved in OFDMA by assigning subsets of sub carriers to individual users as shown in the illustration below. This allows simultaneous low data rate transmission from several users. Due to its robustness against frequency selective fading or narrowband interference, the OFDM technology has been widely implemented in many digital communications such as wireless local area network (WLAN, IEEE 801.11a/g)[1]. The basic block diagram of OFDMA system shown in Fig. 1.

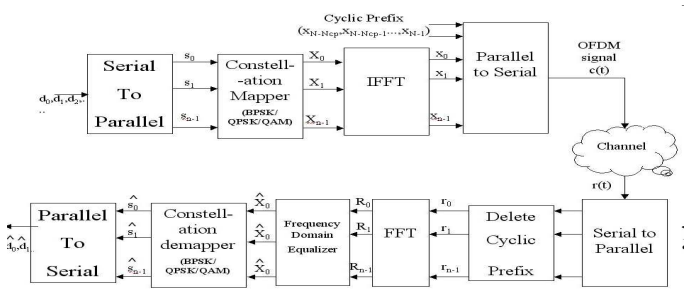

Fig. 1. Block Diagram of OFDMA System

In general, the fast Fourier transform (FFT) and inverse FFT (IFFT) operations are used as the modulation/demodulation kernel in the OFDM systems, and the sizes of FFT/IFFT operations are varied in different applications of OFDM systems. Because of the computational complexity of N-points DFT is $O(N^2)$, it takes large operation time and power consumption to perform DFT directly, especially in large transform size. Hence, various FFT algorithms have been proposed to reduce the computational complexity, such as Discrete Heartly Transform (DHT) [2], where the kernel of FFT is $e^{-j2\Pi kn/N}$. It requires number of exponential arithmetic operations. After that proposed Discrete Heartly Transform which have kernel of CAS function which is nothing but (COS+SIN). Simple it eliminates the exponential operations and contains real arithmetic operations. Even though to implement the DHT using VLSI it requires special algorithms as Cardiac algorithm [2].

To simplify all these here proposed FFT based on Fermat Number Transform (FNT) in the diminished-1 number system which is very easy to implement.

II. FERMAT NUMBER TRANSFORM (FNT)

FNT is proposed in the case the principle root of unity is equal to 2 or its integer power. The FNT and the IFNT are accomplished by the Code Conversion without Addition (CCWA) and the Butterfly Operation without Addition (BOWA) mainly.

The FNT of a sequence of length *N* $\{x_i\}$ (i= 0,1, …N- 1) is defined as :

$$X_k = \sum_{i=0}^{N-1} x_i \alpha_N^{(ik)} \bmod(n) F_t (k = 0,1....N-1)$$

where Ft=2^{2t}+1, the t^{th} Fermat; *N* is a power of 2 and α is an *N*th root unit (*i.e.*, $\alpha^N{}_N \bmod Ft=1$ and $\alpha^M{}_N \bmod Ft \neq 1$,$1 \leq m < N$). The notation < *ik* > means *ik* modulo *N*.

The inverse FNT is given by:

$$x_t = \frac{1}{N}\sum_{k=0}^{N-1} x_k \alpha_N^{-(ik)} \bmod F_t (i = 0,1....N-1) \quad (1)$$

Where $1/N$ is an element in the finite field or ring of integer and satisfies the following condition: $(N.1/N) \bmod F_t = 1$

Parameters α, *Ft, N* must be chosen carefully and some conditions must be satisfied so that the FNT possesses the cyclic convolution property. In this project, $\alpha=2$, $Ft=2^{2t}+1$ and $N=2.2^t$ where t is an integer.

III. IMPORTANT OPERATIONS

Important operations of the cyclic convolution based on FNT with the unity root 2 include the CCWA, the BOWA and the MPPM. The CCWA and the BOWA both consist of novel modulo 2^n+1 4-2 compressors mainly which are composed of the 4-2 compressor.

A. Code Conversion without Addition

The CC converts normal binary numbers into their diminished-1 representation. It is the first stage in the FNT. To reduce the cost, we propose the CCWA that is performed by the modulo 2^n+1 4-2 compressor shown in fig 2. Let *A* and *B* represent two operands whose widths are no more than 2*n* bits. We define two new variables:

$$A = 2^n A_H + A_L$$

$$B = 2^n B_H + B_L$$

and

$$M_0 = (2^n - 1) - A_H = \bar{A}H$$

$$M_1 = (2^n - 1) - B_H = BH$$

$$M_2 = (2^n - 1) - B_L = BL$$

If the subsequent operation of CC is modulo 2^n+1 addition, assign A_L, M_0, B_L and M_1 to *I*0, *I*1, *I*2, *I*3 in the modulo 2^n+1 4-2 compressor respectively. *I*0, *I*1, *I*2, *I*3 are defined as follows:

$$I_0 = I_{0\,(n-1)}\, I_{0\,(n-2)}.....I_{01}I_{00}$$

$$I_1 = I_{1\,(n-1)}\, I_{1\,(n-2)}.....I_{11}I_{10}$$

$$I_2 = I_{2\,(n-1)}\, I_{2\,(n-2)}.....I_{21}I_{20}$$

$$I_3 = I_{3\,(n-1)}\, I_{3\,(n-2)}.....I_{31}I_{30}$$

We obtain the sum vector H_O^* and carry vector H_1^* in the diminished-1 number system. The most significant bit of H_1^* is complemented and connected back to its least significant bit. That is to say

$$H_O^* = H_{0(n-1)}\, H_{0(n-2)}.....H_{01}H_0$$

$$H_1^* = H_{1(n-2)}.....H_{11}H_{10}H_{1(n-1)}$$

After CCWA, we obtain the result consisting of two diminished-1 numbers. The result also includes the information of modulo 2^n+1 addition or subtraction in the first stage of previous BO.

B. Butterfly Operation without Addition

After the CCWA, we obtain the results of modulo 2^n+1 addition and subtraction in the diminished-1 representation. Each result consists of two diminished-1 values. The subsequent butterfly operation involves four operands. The proposed BOWA involves two modulo 2^n+1 4-2 compressors, a multiplier and some inverters as shown in Fig 3. The BOWA can be performed without the carry propagation chain so as to reduce the delay and the area. *K**, *L**, *M**, *N** are corresponding to two inputs and two outputs of previous BO in the diminished-1 number system respectively and given by:

$$M^* = |M_0^* + M_1^*|_{2^n+1} = |K_0^* + K_1^* + L_0^* \times 2^i + L_1^* \times 2^i|_{2^n+1} = |K^* + L^* \times 2^i|_{2^n+1}$$

$$N^* = |N_0^* + N_1^*|_{2^n+1} = |K_0^* + K_1^* - L_0^* \times 2^i - L_1^* \times 2^i|_{2^n+1} = |K^* - L^* \times 2^i|_{2^n+1}$$

$$= |K^* + \overline{L^* \times 2^i}|_{2^n+1} \quad (2)$$

Where $K_*^* = |K_0^* + K_1^*|_{2^n+1}$, $L_*^* = |L_0^* + L|_{2^n+1}$

C. Modular Addition in GF

Most of algorithms describing the addition modulo $(2^n + 1)$ are performed in the diminished-one number system, where a number *x* is represented by $x' = x - 1$ and the number 0 is not used or treated as a special case. Since the operators performing the complex addition processes the numbers in normal representation, the best way to perform a modular addition is to keep the same architecture to get the reconfigurability at lower costs. Let us now study the modulo $(2^n + 1)$ addition of two numbers in normal representations. In the author described some algorithms that return the desired results increased by one. Nevertheless this property facilitates the design of the circuit.

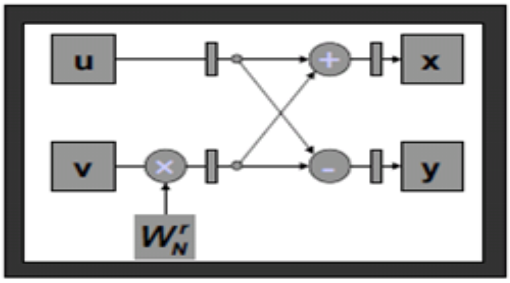

Fig. 2. The Complex butterfly

The modulo $(2^n + 1)$ addition is defined by:

$$\bmod (2^n + 1) = \begin{cases} 2^n \text{ if } x = 2^n \text{ and } y = 2^n \quad (x+y+1) \\ (x + y) \bmod 2^n + c_{out} \quad \text{if } 0 \le x+y \le 2^{n+1} \end{cases} \quad (3)$$

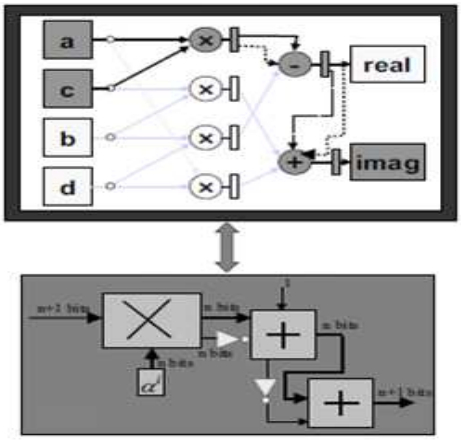

Fig. 3. The Modulo (2^n+1) Multiplier

In a direct implementation of equation (3) is presented. The circuit is shown in Fig 2 to improve the implementation, the author suggests an alternative architecture suppressing the multiplexer Fig. 3 the modulo (2^n+1) addition is expressed as:

$$(x+ y+1) \bmod (2^n + 1) = (x+ y) \bmod (2^n + s_{n+1}2^n + s_{n+1}2^n \text{ v } s_n \quad (4)$$

To perform an addition that returns directly the desired result, we propose an alternative adder. We define s^1; s^2 the sums at the first and second adders respectively with the (n+2)-bit integer $s^1 = [s^1{}_{n+1}s^1{}_{n.....}s0] = x + y$. The modulo ($2^n$ + 1) addition can be expressed as: The Butterfly is configured to operate over C and one wants to perform a calculation over *GF* (*Ft*).

$$(x+ y) \bmod (2^n + 1) = \begin{cases} (x+ y) \bmod 2^n, & \text{if } 0 \le x+y \le 2^n \\ (x+ y) \bmod 2^n + 2^{n-1}, & \text{if } 2^n \le x+y \le 2^{n+}1 \end{cases} \quad (5)$$

To do this, the Butterfly should download the primitive element *i*, activate the different logic gate (AND, OR and the multiplexers) and reconfigure the connection inter-operators the global architecture of the FNT is presented.

IV. THE FNT ARCHITECTURE

In the previous sections, we have presented the reconfiguration at a rather low level. The Butterfly constitutes a high parameterized function level. The fact to have this parameterized function allows designing a reconfigurable operator who's Butterfly forms the highest level operator. Fig 4 depicts the global reconfigurable operator. Over C it is called FFT and over GF (*Ft*) is called FNT. This architecture has been validated by software. A simple test of calculation of FFT and IFFT, showed the validity of this structure.

V. SIMULATION RESULTS

The modulo 2^n+1 addition for the diminished-1 number system is the crucial operation which contains a standard *n*-bit carry propagation computation such as a parallel-prefix adder with a carry-logic block and a zero indicator of the diminished-1 operand to determine whether to perform

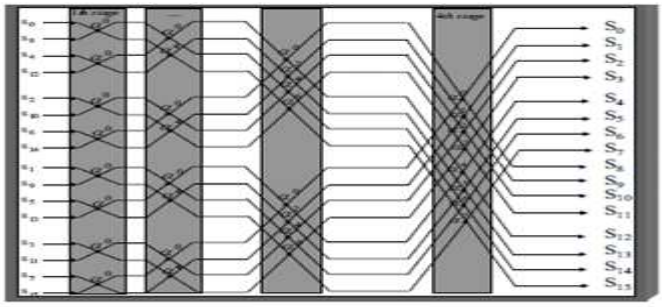

Fig. 4.The architecture of FNT operator

subsequent operations. It produces the longest execution delay and requires large area in the previous solution. The proposed CCWA and BOWA overcome the disadvantage of the carry-propagation adder and don't require a zero indicator. Thus our architecture is faster and more efficient than the existing one.

The delay and the area estimations of modulo 2^n+1 adder and modulo 2^n+1 multiplier in the cyclic convolution are given in Table 1 as a function of the operand size *n*. To obtain more accurate results, we describe the proposed parallel cyclic convolution in verilog for *Ft*=2^8+1, 2^{16}+1, 2^{32}+1. The validated Verilog code is synthesized using a 0.13-μm CMOS standard cells library in the worst operating condition by the Synopsys Design Compiler. EDA software can't provide a faster design. The results for the fastest derived implementation are listed in Table 2. Table 1 and 2 indicate that for values of $F_t \ge 2^8 + 1$, the proposed architecture comprising the CCWA and BOWA require less delay and area than the previous one.

VLSI implementations using a 0.13 um standard cell library show the proposed parallel architecture can attain lower area and delay than that of the existing solution when the modulus is no less than 2^8+1. In this architecture we used modulo adders and modulo multipliers to reduce the round off errors and also to reduce the delay and also that CCWA and BOWA blocks are used for further reduction of the delay Based on the number of operands as the butterfly stages generally increases there is no other alternative for finding the convolution. The changes can be made either for the blocks like multiplication by multiplier in order to reduce the size. This is the high speed architecture; we further work on for the size or power consumption and others. The simulation is done using Xilinx 12.1ISE, the simulation results are shown in Fig. 5 – Fig. 8.

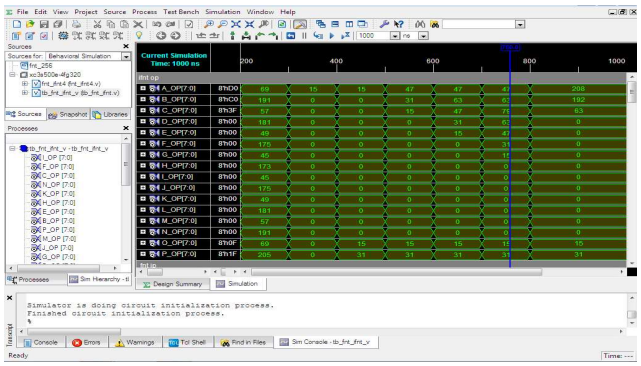

Fig. 5. IFNT output simulation results

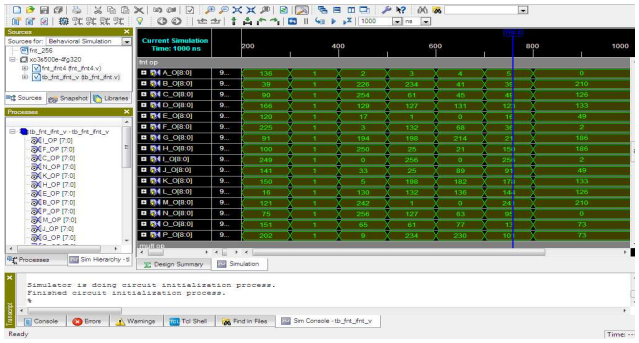

Fig. 6. FNT s output simulation results

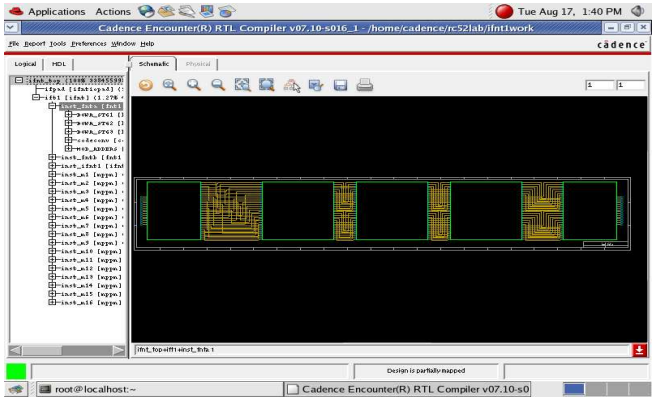

Fig.7. FNT frontend internal schematics

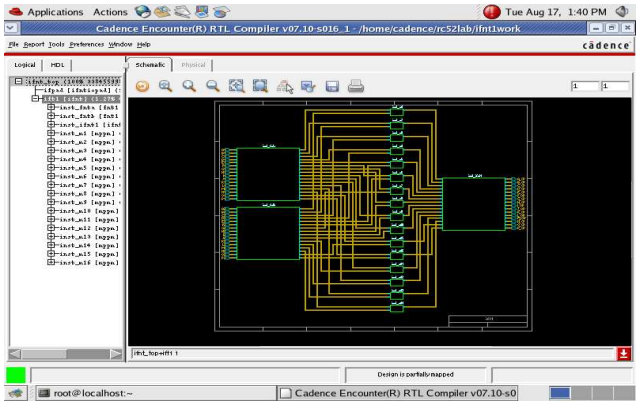

Fig.8. IFNT frontend internal schematics

VI. CONCLUSION

A novel architecture for the FNT processor for OFDMA is proposed in the case the principle root of unity is equal to 2 or its integer power. The FNT and the IFNT are accomplished by the CCWA and the BOWA mainly. The point wise multiplication is performed by the modulo 2^n+1 partial product multiplier. A theoretical model was applied to access the efficiency independently of the target technology. VLSI implementations using a 0.13 um standard cell library show the proposed parallel architecture can attain lower area and delay than that of the existing solution when the modulus is no less than 2^8+1. In this architecture we used modulo adders and modulo multipliers to reduce the round off errors and also to reduce the delay and also that CCWA and BOWA blocks are used for further reduction of the delay Based on the number of operands as the butterfly stages generally increases there is no other alternative for finding the convolution. The changes can be made either for the blocks like multiplication by multiplier in order to reduce the size. This is the high speed architecture; we further work on for the size or power consumption and others.

REFERENCES

[1] Jen-Chih Kuo, Ching-HuaWen, Chih-Hsiu Lin, An-Yeu (Andy)Wu, "VLSI Design of a Variable-Length FFT/IFFT Processor for OFDM-Based Communication Systems" EURASIP Journal on Applied Signal Processing 2003:13, 1306–1316.

[2] Chin-Liang Wang, Ching-Hsien Chang, "A Variable Length DHT-Based FFT/IFFT PROCESSOR FOR VDSL/ADSL SYSTEMS"The IEEE Asia-Pacific Conference on Circuits and Systems,December 2004, 6-9,pp.381-384.

[3] C. Cheng, K.K. Parhi, "Hardware efficient fast DCT based on novel cyclic convolution structures", IEEE Trans. Signal processing, 2006, 54(11), pp. 4419- 4434

[4] R. V. Nee and R. Prasad, "OFDM Wireless Multimedia Communications",Norwood, MA: Artech House, 2000.

[5] Lars-Inge Alfredsson,"VLSI Architectures and Arithmetic Operations with Application to the Fermat Number Transform", Link ¨oping Studies in Science and Technology, 1996.