Efficient Algorithms for Computing the New Mersenne Number Transform

Mounir T. Hamood[a,*] **and Said Boussakta**[b]

[a] Department of Electrical Engineering, University of Tikrit, Tikrit, P. O. BOX 42, IRAQ
m.t.hamood@tu.edu.iq

[b] School of Electrical and Electronic Engineering, Newcastle University, Newcastle upon Tyne, NE1 7RU, UK
s.boussakta@ncl.ac.uk

Abstract

The new Mersenne number transform (NMNT) has proved to be an important number theoretic transform (NTT) used for error-free calculation of convolutions and correlations. Its main feature is that for a suitable Mersenne prime number (p), the allowed power-of-two transform lengths can be very large. In this paper, efficient radix-2^2 decimation-in-time and in-frequency algorithms for fast calculation of the NMNT are developed by deriving the appropriate mathematical relations in finite field and applying principles of the twiddle factor unscrambling technique. The proposed algorithms achieve both the regularity of radix-2 algorithm and the efficiency of radix-4 algorithm and can be applied to any powers of two transform lengths with simple bit reversing for ordering the output sequence. Consequently, the proposed algorithms possess the desirable properties such as simplicity and in-place computation. The validity of the proposed algorithms has been verified through examples involving large integer multiplication and digital filtering applications, using both the NMNT and the developed algorithms.

Keywords—Number theoretic transforms (NTTs), new Mersenne number transform (NMNT), radix-2^2 algorithm.

1. Introduction

Convolutions and correlations are the most fundamental mathematical tools used for enormous area of digital signal/image processing and other diverse applications [1, 2]. For instance, convolutions are widely used in the design and implementation of the finite impulse response (FIR) as well as the infinite impulse response (IIR) digital filters. Moreover, it is well known that the DFT of prime lengths can be computed by converting it to a cyclic convolution using 'Rader's convolution algorithm' [3]. Correlation differs from convolution only by a simple inversion of one of the input sequences [4], therefore developments for the convolutions algorithms are equally applicable to the correlation also.

By proper scaling of the convolution's inputs, they can be always converted to a set of integers, and the convolution can be performed modulo a prime number M in the finite (Galois) field $GF(M)$. If the scaling factor is such that the convolution output has never exceeded $M/2$, then the convolution output has the identical values modulo M that would be obtained in the normal field. Under these conditions, the calculation of the convolution can be simplified by introducing a new family of transforms defined in finite field, known as number theoretic transforms (NTTs) [5, 6], that have the same structure as the DFT but with complex operations replaced by an exact integer operations performed modulo M. NTTs first presented by Pollard [7], are discrete transforms defined over residue class fields or rings of integers, which were introduced for efficient calculation of error-free convolution and correlation without truncation or round-off errors.

NTTs have been firmly recognized within the field of signal processing [2]. Interesting applications of NTTs are found in the areas of digital filtering, image processing [8, 9], fast coding and decoding [10], large integer and matrix multiplication [11, 12], cryptography [13], and deconvolution [14]. This is owing to their contributing ability to perform error-free calculations over a field or a ring of integers whilst maintaining the cyclic convolution property (CCP). This is in contrast to other methods of calculation, such as the DFT which involves complex arithmetic with rounding and/or truncation errors in its calculations; errors also arise in the multiplication with cosine and sine functions which are irrational, preventing exact representation in a finite precision machine [15].

The most recognised NTTs are the Fermat (FNT) [16] and Mersenne (MNT) [6] number transforms. However, for standard signal processing applications the main drawback of these transforms is the stringent relationship between word length (the number of bits in the modulus), obtainable transform length, and a limited choice of possible word lengths. To retain the advantages of NTTs, the New Mersenne Number Transform (NMNT) was introduced [17, 18], which alleviate this relationship. The NMNT is defined modulo the Mersenne numbers, where arithmetic operations are simple equivalent to 1's complement and has the cyclic convolution property; hence, it can be used for fast calculation of error-free convolutions and correlations. The NMNT is a particularly interesting NTT as it has a long powers of two lengths up to 2^p, making it amenable to fast algorithms.

Various Cooley-Tukey algorithms for the fast calculations of the NMNT have been developed based on both DIT and DIF approaches such as radix-2 [17, 18], radix-4 [19, 20] and split-radix [21, 22] algorithms. However, for any transform to stand as a good candidate for real applications, its complete fast algorithms need to be developed.

Over the last years, a new hardware-oriented FFT algorithm known as radix-2^2 [23-25], as well as its variants algorithms [26-29], has been recognized as one of the most powerful structures used in pipeline architectures. It achieves at the same time both a simple and regular butterfly structure as radix-2 algorithm and a reduced number of twiddle factor multiplication provided by radix-4 algorithm. Therefore, it is desirable to generalize this algorithm to other discrete transforms such as the NMNT.

Therefore, the aim of this paper is to introduce new radix-2^2 decimation-in-time (DIT) and in-frequency (DIF) NMNT algorithms. The derivation of the proposed algorithms is based on the principle of the twiddle factor unscrambling technique [30-32], which is different from the conventional multidimensional index mapping technique [18]. The development of the presented algorithms has rested mainly on the observation that a radix-4 algorithm can be modified so that the output is in bit-reversed order; if a normal radix-4 butterfly is used, the output is in base-4 reversed order. However, if the outputs of the four short length butterflies are modified to have their outputs in bit-reversed order, the output of the total radix-4 algorithm will be in bit-reversed order and not base-4 reversed order.

The remaining contents of this paper are organised as follows: Section 2 reviews the NMNT and its cyclic convolution property. In sections 3 and 4, we propose radix-2^2 DIT and DIF NMNT algorithms, respectively. In section 5, we study the performance of the proposed algorithms by analysing their arithmetic complexity and comparing them with existing NMNT algorithms. Section 7 introduces two examples for the presented algorithms. A conclusion is then given in section 8.

2. The New Mersenne Number Transform

2.1 Transform Definition

Let p be a prime and $Mp=2^p-1$ Mersenne numbers, which are primes for p=2,3,5,7,13,17,19,……, etc. The NMNT of an integer sequence $x(n)$ of length N is given by [17, 18]:

$$X(k) = \langle \sum_{n=0}^{N-1} x(n)\beta(nk)\rangle_{Mp} \qquad k = 0,1,\ldots\ldots\ldots..,N-1 \tag{1}$$

and its inverse has exactly the same form:

$$x(n) = \langle N^{-1}\sum_{k=0}^{N-1} X(k)\beta(nk)\rangle_{Mp} \qquad n = 0,1,\ldots\ldots\ldots..,N-1 \tag{2}$$

where:

$$\beta(nk) = \beta_1(nk) + \beta_2(nk) \tag{3}$$

$$\beta_1(nk) = \langle Re(\alpha_1 + j\alpha_2)^{nk}\rangle_{Mp} \tag{4}$$

$$\beta_2(nk) = \langle Im(\alpha_1 + j\alpha_2)^{nk}\rangle_{Mp} \tag{5}$$

Also: $\propto_1 = \pm\langle 2^q\rangle_{Mp};\ \ \propto_2 = \pm\langle -3^q\rangle_{Mp};\quad q = 2^{p-2}$ (6)

$\langle\ \rangle_{\boldsymbol{Mp}}$ represents modulo Mp.

α_1 and α_2 are of order $N=2^{p+1}$. For transform length N/d where d is an integer power of two, β_1 and β_2 are given by:

$$\beta_1(nk) = \langle Re((\alpha_1 + j\alpha_2)^d)^{nk}\rangle_{Mp} \tag{7}$$

$$\beta_2(nk) = \langle Im((\alpha_1 + j\alpha_2)^d)^{nk}\rangle_{Mp} \tag{8}$$

Re(.) and Im(.) denote real and imaginary parts of the enclosed term respectively, (N^{-1}) exists and is given by (2^{p-d}), where $N=2^d$ and d is an integer, $0 \le d \le p$.

2.2 NMNT Cyclic Convolution Property

The NMNT has the cyclic convolution property; if $x(n)$ and $h(n)$ are two sequences to be convolved and [$y(n)=x(n)\circledast h(n)$], is the convolution result, then

$$Y(k) = X(k)\boldsymbol{\Gamma}H(k) = X(k)\bullet H_{\mathrm{ev}}(k) + X(N-k)\bullet H_{\mathrm{od}}(k) \tag{9}$$

where $\circledast$ is the cyclic convolution operator and $\bullet$ is point-by-point multiplication. $X(k)$, $H(k)$ and $Y(k)$ stand for the NMNT transforms of $x(n)$, $h(n)$ and $y(n)$ respectively. $H_{ev}(k)$ and $H_{od}(k)$ stand for even and odd parts of $H(k)$ respectively, which are given by:

$$H_{ev}(k) = \langle (H(k) + H(N-k)) \times 2^{p-1} \rangle_{Mp} \tag{10}$$

$$H_{od}(k) = \langle (H(k) - H(N-k)) \times 2^{p-1} \rangle_{Mp} \tag{11}$$

If both $x(n)$ and $h(n)$ are properly padded with zeros, their circular convolution given in (9) will be equivalent to their linear convolution. To avoid overflow, the modulus, Mp must be chosen so that $y(n)$ does not exceed Mp, one upper bound is given by [5, 18]:

$$|y(n)| \le |x(n)|_{max} \sum_{n=0}^{N-1} |h(n)| \le Mp/2 \tag{12}$$

The process of calculation of the convolution via the NMNT is shown in Fig. 1, where the operator $\mathbf{\Gamma}$ is given in (9).

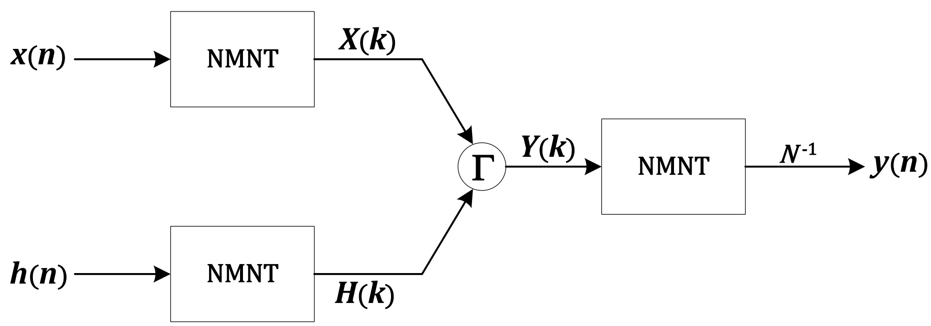


Fig. 1. Fast convolution using the NMNT

3. Decimation in Time Algorithm

The development of radix-2^2 algorithms starts by decomposing (1) into four partial sums and replacing (n) with $(4n+l)$ for n=0,1,……, N/4-1 and l=0,1,2,3 as follows:

$$X(k) = \langle \sum_{l=0}^{3} \sum_{n=0}^{\frac{N}{4}-1} x(4n+l)\beta((4n+l)k) \rangle_{Mp} \tag{13}$$

According to (13), the input sequence $x(n)$ is decimated into four sets so that each partial sum represents NMNT of size N/4. The output sequence $X(k)$ is computed as four separate parts, and each part denoted by $X(k+\lambda N/4)$ has $(N/4)$ consecutive elements indexed by k for k=0,1,……, N/4-1 and λ=0,1,2,3. Therefore, (13) becomes:

$$X\left(k + \lambda \frac{N}{4}\right) = \langle \sum_{l=0}^{3} \sum_{n=0}^{\frac{N}{4}-1} x(4n+l)\beta((4n+l)(k+\lambda \frac{N}{4})) \rangle_{Mp} \tag{14}$$

Using NMNT identities given below, which have been proved in [5]:

$$\beta(m+n) = \beta_1(m)\beta(n) + \beta_2(m)\beta(-n) \tag{15}$$

$$\beta_1(m+n) = \beta_1(m)\beta_1(n) - \beta_2(m)\beta_2(n) \tag{16}$$

$$\beta_2(m+n) = \beta_1(m)\beta_2(n) + \beta_2(m)\beta_1(n) \tag{17}$$

$$\beta_1(-n) = \beta_1(n) \tag{18}$$

$$\beta_2(-n) = -\beta_2(n) \tag{19}$$

$\beta(.)$ term in (14) can be simplified as follows:

$$\beta\left(\left(k+\lambda\frac{N}{4}\right)(4n+l)\right) = \beta\left((4nk+\lambda Nn) + \left(kl+\lambda l\frac{N}{4}\right)\right) \tag{20}$$

Using (15) and the periodicity property of NMNT, the right hand side of (20) becomes:

$$\beta\left((4nk+\lambda Nn) + \left(kl+\lambda l\frac{N}{4}\right)\right) = \beta_1\left(kl+\lambda l\frac{N}{4}\right)\beta(4nk) + \beta_2\left(kl+\lambda l\frac{N}{4}\right)\beta(-4nk) \tag{21}$$

Using (16) and (17), $\beta_1(.)$ and $\beta_2(.)$ terms in (21) can be simplified further to yield:

$$\beta_1\left(kl+\lambda l\frac{N}{4}\right) = \beta_1(kl)\beta_1\left(\lambda l\frac{N}{4}\right) - \beta_2(kl)\beta_2\left(\lambda l\frac{N}{4}\right) \tag{22}$$

$$\beta_2\left(kl+\lambda l\frac{N}{4}\right) = \beta_1(kl)\beta_2\left(\lambda l\frac{N}{4}\right) + \beta_2(kl)\beta_1\left(\lambda l\frac{N}{4}\right) \tag{23}$$

Substituting (21)-(23) into (20), we get:

$$\beta\left(\left(k+\lambda\frac{N}{4}\right)(4n+l)\right) = \left[\beta_1(kl)\beta_1\left(\lambda l\frac{N}{4}\right) - \beta_1(kl)\beta_1\left(\lambda l\frac{N}{4}\right)\right]\beta(4nk) + \left[\beta_1(kl)\beta_2\left(\lambda l\frac{N}{4}\right) + \beta_2(kl)\beta_1\left(\lambda l\frac{N}{4}\right)\right]\beta(-4nk) \tag{24}$$

Define two sequences $X_l(k)$ and $X_l(N/4\text{-}k)$ for l=0,1,2,3 as:

$$X_l(k) = \langle\sum_{n=0}^{\frac{N}{4}-1} x(4n+l)\beta(4nk)\rangle_{Mp} \qquad k = 0,1,\ldots\ldots\ldots..,\frac{N}{4}-1 \tag{25}$$

and:

$$X_l\left(\frac{N}{4}-k\right) = \langle\sum_{n=0}^{\frac{N}{4}-1} x(4n+l)\beta(-4nk)\rangle_{Mp} \qquad k = 0,1,\ldots\ldots\ldots..,\frac{N}{4}-1 \tag{26}$$

Substituting (24)-(26) into (14):

$$X\left(k+\lambda\frac{N}{4}\right) = \sum_{l=0}^{3}\left[X_l(k)\beta_1(kl)\beta_1\left(\lambda l\frac{N}{4}\right) - X_l(k)\beta_2(kl)\beta_2\left(\lambda l\frac{N}{4}\right)\right] + \left[X_l(\frac{N}{4}-k)\beta_1(kl)\beta_2\left(\lambda l\frac{N}{4}\right) + X_l(\frac{N}{4}-k)\beta_2(kl)\beta_1\left(\lambda l\frac{N}{4}\right)\right] \tag{27}$$

Rearranging (27), we get:

$$X\left(k+\lambda\frac{N}{4}\right) = \sum_{l=0}^{3}\left[X_l(k)\beta_1\left(\lambda l\frac{N}{4}\right) + X_l\left(\frac{N}{4}-k\right)\beta_2\left(\lambda l\frac{N}{4}\right)\right]\beta_1(kl) + \left[X_l\left(\frac{N}{4}-k\right)\beta_1\left(\lambda l\frac{N}{4}\right) - X_l(k)\beta_2\left(\lambda l\frac{N}{4}\right)\right]\beta_2(kl) \tag{28}$$

Applying the unscrambling mapping technique, by interchanging the locations of the intermediate twiddle factors and re-indexing (***l***) of $\beta_1(kl)$ and $\beta_2(kl)$ according to bit reversed order, (28) can be written as:

$$X\left(k+\lambda\frac{N}{4}\right)=X_0(k)+\left[X_1(k)\beta_1\left(\lambda\frac{N}{4}\right)+X_1\left(\frac{N}{4}-k\right)\beta_2\left(\lambda\frac{N}{4}\right)\right]\beta_1(2k)+\left[X_1\left(\frac{N}{4}-k\right)\beta_1\left(\lambda\frac{N}{4}\right)-X_1(k)\beta_2\left(\lambda\frac{N}{4}\right)\right]\beta_2(2k)$$
$$+\left[X_2(k)\beta_1\left(\lambda\frac{N}{2}\right)+X_2\left(\frac{N}{4}-k\right)\beta_2\left(\lambda\frac{N}{2}\right)\right]\beta_1(k)+\left[X_2\left(\frac{N}{4}-k\right)\beta_1\left(\lambda\frac{N}{2}\right)-X_2(k)\beta_2\left(\lambda\frac{N}{2}\right)\right]\beta_2(k) \quad (29)$$
$$+\left[X_3(k)\beta_1\left(\lambda\frac{3N}{4}\right)+X_3\left(\frac{N}{4}-k\right)\beta_2\left(\lambda\frac{3N}{4}\right)\right]\beta_1(3k)+\left[X_3\left(\frac{N}{4}-k\right)\beta_1\left(\lambda\frac{3N}{4}\right)-X_3(k)\beta_2\left(\lambda\frac{3N}{4}\right)\right]\beta_2(3k)$$

Equation (29) is a general decomposition formula for the radix-2^2 NMNT-DIT algorithm; expanding it gives the desired output points. These points are derived by considering the relations given below for integer (***v***).

$$\beta_1\left(\upsilon\frac{N}{2}\right)=(-1)^{\upsilon} \quad (30)$$

$$\beta_2\left(\upsilon\frac{N}{2}\right)=0 \quad (31)$$

$$\beta_1\left(\upsilon\frac{N}{4}\right)=\begin{cases}(-1)^{\frac{\upsilon}{2}} & \upsilon{:}Even\\ 0 & \upsilon{:}Odd\end{cases} \quad (32)$$

$$\beta_2\left(\upsilon\frac{N}{4}\right)=\begin{cases}0 & \upsilon{:}Even\\ (-1)^{\frac{\upsilon-1}{2}} & \upsilon{:}Odd\end{cases} \quad (33)$$

The proof of (30)-(33) is given in the Appendix.

Therefore, $X(k)$, X (k+N/4), X (k+N/2) and X (k+3N/4) points can be written as:

$$X(k)=X_0(k)+\left[X_1(k)\beta_1(2k)+X_1\left(\frac{N}{4}-k\right)\beta_2(2k)\right]+\left[X_2(k)\beta_1(k)+X_2\left(\frac{N}{4}-k\right)\beta_2(k)\right]+\left[X_3(k)\beta_1(3k)+X_3\left(\frac{N}{4}-k\right)\beta_3(3k)\right] \quad (35)$$

$$X\left(k+\frac{N}{4}\right)=X_0(k)-\left[X_1(k)\beta_1(2k)+X_1\left(\frac{N}{4}-k\right)\beta_2(2k)\right]+\left[X_2\left(\frac{N}{4}-k\right)\beta_1(k)-X_2(k)\beta_2(k)\right]-\left[X_3\left(\frac{N}{4}-k\right)\beta_1(3k)-X_3(k)\beta_3(3k)\right] \quad (36)$$

$$X\left(k+\frac{N}{2}\right)=X_0(k)+\left[X_1(k)\beta_1(2k)+X_1\left(\frac{N}{4}-k\right)\beta_2(2k)\right]-\left[X_2(k)\beta_1(k)+X_2\left(\frac{N}{4}-k\right)\beta_2(k)\right]-\left[X_3(k)\beta_1(3k)+X_3\left(\frac{N}{4}-k\right)\beta_3(3k)\right] \quad (37)$$

$$X\left(k+\frac{3N}{4}\right)=X_0(k)-\left[X_1(k)\beta_1(2k)+X_1\left(\frac{N}{4}-k\right)\beta_2(2k)\right]-\left[X_2\left(\frac{N}{4}-k\right)\beta_1(k)-X_2(k)\beta_2(k)\right]+\left[X_3\left(\frac{N}{4}-k\right)\beta_1(3k)-X_3(k)\beta_3(3k)\right] \quad (38)$$

Combining eight points together gives an in-place butterfly of the radix-2^2 DIT-NMNT algorithm, as shown in Fig. 2.

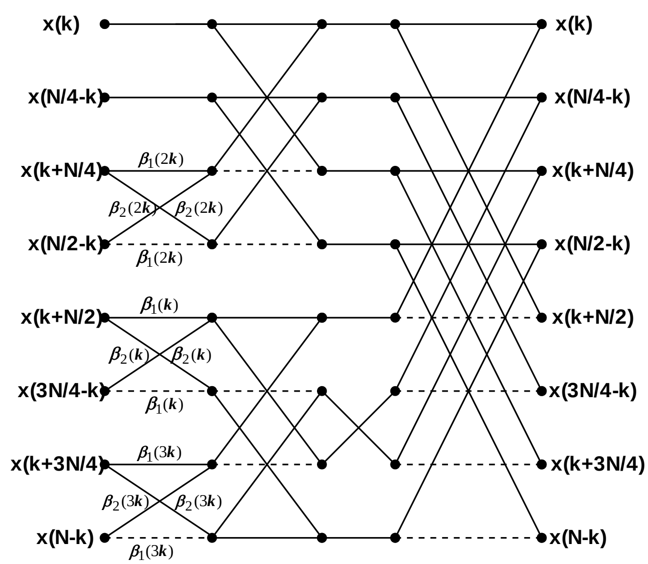


Fig. 2. An in-place butterfly structure of the radix-$2^{2,}$ NMNT DIT algorithm; where solid and dotted lines stand for addition and subtraction respectively

4. Decimation in Frequency Algorithm

To derive the radix-2^2 NMNT algorithm using the DIF approach, we replace the variables *n* and *k* in (1) by:

$$n + \lambda \frac{N}{4} \qquad n = 0,1,\ldots\ldots,\frac{N}{4} - 1;\ \lambda = 0,1,2,3$$

$$4k + l \qquad k = 0,1,\ldots\ldots,\frac{N}{4} - 1;\ l = 0,1,2,3 \qquad (39)$$

Thus, (1) becomes:

$$X(4k + l) = \langle \sum_{n=0}^{\frac{N}{4}-1} \sum_{l=0}^{3} x\left(n + \lambda \frac{N}{4}\right) \beta((n + \lambda \frac{N}{4})(4k + l)) \rangle_{Mp} \qquad (40)$$

Using similar mathematical manipulations given by (20)-(23), $\beta(.)$ term in (40) can be simplified as:

$$\beta\left(\left(n + \lambda \frac{N}{4}\right)(4k + l)\right) = \left[\beta_1\left(\lambda l \frac{N}{4}\right)\beta(4nk) + \beta_2\left(\lambda l \frac{N}{4}\right)\beta(-4nk)\right]\beta_1(nl) + \left[\beta_1\left(\lambda l \frac{N}{4}\right)\beta(-4nk) - \beta_2\left(\lambda l \frac{N}{4}\right)\beta(4nk)\right]\beta_2(nl) \qquad (41)$$

Substituting (41) into (40) and using the following relations:

$$\sum_{n=0}^{\frac{N}{4}-1} x\left(n + \lambda \frac{N}{4}\right)\beta_1(nl)\beta(-4nk) = \sum_{n=0}^{\frac{N}{4}-1} x\left(\lambda \frac{N}{4} - n\right)\beta_1(nl)\beta(4nk) \qquad (42)$$

$$\sum_{n=0}^{\frac{N}{4}-1} x\left(n + \lambda \frac{N}{4}\right)\beta_2(nl)\beta(-4nk) = -\sum_{n=0}^{\frac{N}{4}-1} x\left(\lambda \frac{N}{4} - n\right)\beta_2(nl)\beta(4nk) \qquad (43)$$

The proof of these relations is obtained by applying (18) and (19) to (42) and (43) respectively, we get:

$$X(4k + l) = \sum_{n=0}^{\frac{N}{4}-1} y(l,n)\beta(4nk) \qquad (44)$$

where $y(l,n)$ is given by:

$$y(l,n) = \sum_{l=0}^{3}\left[x\left(n+\lambda l\frac{N}{4}\right)\beta_1\left(\lambda l\frac{N}{4}\right) + x\left(\lambda\frac{N}{4}-k\right)\beta_2\left(\lambda l\frac{N}{4}\right)\right]\beta_1(nl) - \left[x\left(\lambda\frac{N}{4}-k\right)\beta_1\left(\lambda l\frac{N}{4}\right) + x\left(n+\lambda l\frac{N}{4}\right)\beta_2\left(\lambda l\frac{N}{4}\right)\right]\beta_2(nl) \quad (45)$$

Applying the unscrambling mapping method, by interchanging the locations of the intermediate twiddle factors and re-indexing (l) of $\beta_1(nl)$ and $\beta_2(nl)$ according to bit reversed order, (45) can be written as:

$$y(l,n) = \begin{cases} \left[x(n) + x\left(n+\frac{N}{4}\right)\beta_1\left(l\frac{N}{4}\right) + x\left(\frac{N}{4}-n\right)\beta_2\left(l\frac{N}{4}\right) + x\left(n+\frac{N}{2}\right)\beta_1\left(l\frac{N}{2}\right) + x\left(n+\frac{3N}{4}\right)\beta_1\left(l\frac{3N}{4}\right) + x\left(\frac{3N}{4}-n\right)\beta_2\left(l\frac{3N}{4}\right)\right]\beta_1(nl) \\ \left[x(N-n) + x\left(\frac{N}{4}-n\right)\beta_1\left(l\frac{N}{4}\right) + x\left(n+\frac{N}{4}\right)\beta_2\left(l\frac{N}{4}\right) + x\left(\frac{N}{2}-n\right)\beta_1\left(l\frac{N}{2}\right) + x\left(\frac{3N}{4}-n\right)\beta_1\left(l\frac{3N}{4}\right) + x\left(n+\frac{3N}{4}\right)\beta_2\left(l\frac{3N}{4}\right)\right]\beta_2(nl) \end{cases} \quad (46)$$

Equation (46) is a general decomposition formula for the radix-2^2 NMNT-DIF algorithm; expanding it gives the desired output points. These points are derived by substituting (30)-(33) in (46). Therefore, $X(4k)$, $X(4k+1)$, $X(4k+2)$ and $X(4k+3)$ points can be written as:

$$X(4k) = \sum_{n=0}^{\frac{N}{4}-1}\left\{x(n) + x\left(n+\frac{N}{4}\right) + x\left(n+\frac{N}{2}\right) + x\left(n+\frac{3N}{4}\right)\right\}\beta(4nk) \quad (47)$$

$$X(4k+1) = \sum_{n=0}^{\frac{N}{4}-1}\left\{\left[x(n) - x\left(n+\frac{N}{4}\right) + x\left(n+\frac{N}{2}\right) - x\left(n+\frac{3N}{4}\right)\right]\beta_1(2n) - \left[x(N-n) - x\left(\frac{N}{4}-n\right) + x\left(\frac{N}{2}-n\right) - x\left(\frac{3N}{4}-n\right)\right]\beta_2(2n)\right\}\beta(4nk) \quad (48)$$

$$X(4k+2) = \sum_{n=0}^{\frac{N}{4}-1}\left\{\left[x(n) + x\left(\frac{N}{4}-n\right) - x\left(n+\frac{N}{2}\right) - x\left(\frac{3N}{4}-n\right)\right]\beta_1(n) - \left[x(N-n) + x\left(n+\frac{N}{4}\right) - x\left(\frac{N}{2}-n\right) - x\left(n+\frac{3N}{4}\right)\right]\beta_2(n)\right\}\beta(4nk) \quad (49)$$

$$X(4k+3) = \sum_{n=0}^{\frac{N}{4}-1}\left\{\left[x(n) - x\left(\frac{N}{4}-n\right) - x\left(n+\frac{N}{2}\right) + x\left(\frac{3N}{4}-n\right)\right]\beta_1(3n) - \left[x(N-n) - x\left(n+\frac{N}{4}\right) - x\left(\frac{N}{2}-n\right) + x\left(n+\frac{3N}{4}\right)\right]\beta_2(3n)\right\}\beta(4nk) \quad (50)$$

Combining eight points together gives an in-place butterfly of the radix-2^2 DIF-NMNT algorithm, as shown in Fig. 3.

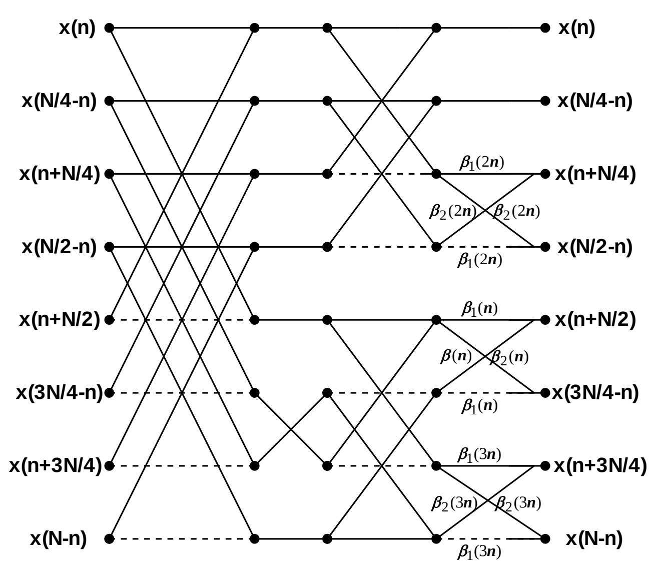


Fig. 3. An in-place butterfly structure of the radix-$2^{2,}$ NMNT DIF algorithm; where solid and dotted lines stand for addition and subtraction respectively.

5. Arithmetic Complexity

In this section, the performances of the proposed algorithms are analysed by calculating their number of multiplications and additions. Since the proposed DIT and DIF algorithms are based on the same decomposition approach, their arithmetic complexities are exactly the same. Therefore, the analysis of the arithmetic complexity of only one is sufficient. Let us consider the arithmetic complexity of the proposed DIT algorithm given in section 3. (35)-(38) represent the radix-2^2 DIT decomposition formula.

In general, the radix-2^2 algorithm needs ($\log_2 N$) stages of butterfly computation. Each stage uses ($3N/2$) integer multiplications and ($11N/4$) integer additions. In addition, four ($N/4$)-point NMNTs have to be calculated, thus the whole radix-2^2 NMNT complexity satisfies the following equations:

$$M(N) = 4M\left(\frac{N}{4}\right) + \frac{3N}{2} - M_t \tag{51}$$

$$A(N) = 4A\left(\frac{N}{4}\right) + \frac{11N}{4} - A_t \tag{52}$$

where $M(N)$ and $A(N)$ are the number of integer multiplications and additions, respectively, needed by the radix-2^2 algorithm for a length-N NMNT, and M_t and A_t are the number of multiplications and additions saved from trivial twiddle factors. According to (30)-(34), when n=0 and n=N/2, the twiddle factors become (0) or (±1) so that eight multiplications and four additions can be saved, and when n=N/4 and n=3N/4, two multiplications and additions are also saved. If all trivial twiddle factors are considered, then M_t=10 and A_t=6. The computational complexities in (51) and (52) are recursive. To obtain the complexity for different transform sizes, the initial values of these complexities are needed. In this case, the initial values can be the number of operations that are needed by length-4 and length-8 NMNTs, which in this case equal to $M(4)$=0 and $A(4)$=8; $M(8)$=4 and $A(8)$=26. Therefore, the overall arithmetic complexity for the radix-2^2 NMNT algorithm is given as:

$$M(N) = 4M\left(\frac{N}{4}\right) + \frac{3N}{2} - 12 \tag{53}$$

$$A(N) = 4A\left(\frac{N}{4}\right) + \frac{11N}{4} - 10 \tag{54}$$

Substituting the initial values for $M(4)$, $M(8)$ in (53), $A(4)$ and $A(8)$ in (54) gives the arithmetic complexities of the radix-2^2 NMNT algorithm, as shown in first column of Table I.

A comparison has been made among radix-2, radix-4 and the developed algorithm in terms of the number of multiplications and additions, as shown in Table I. The results of this comparison have revealed that the developed algorithm involves less arithmetic operations than radix-2 or radix-4.

TABLE I
COMPARISON BETWEEN RADIX-2, RADIX-4, AND RADIX-2^2 NMNT ALGORITHMS, WHERE *M*(*N*) AND *A*(*N*) ARE THE NUMBER OF INTEGER MULTIPLICATIONS AND ADDITIONS RESPECTIVELY

Length	Proposed Radix-2^2 NMNT Algorithm			Radix-2 NMNT Algorithm			Radix-4 NMNT Algorithm		
N	*M*(*N*)	*A*(*N*)	Total	*M*(*N*)	*A*(*N*)	Total	*M*(*N*)	*A*(*N*)	Total
8	2	22	24	4	26	30	-	-	-
16	12	66	78	20	74	94	14	70	84
32	44	166	208	68	194	262	-	-	-
64	132	430	562	196	482	678	142	450	592
128	356	1006	1362	516	1154	1670	-	-	-
256	900	2414	3314	1284	2690	3974	942	2498	3440
512	2180	5422	7602	3076	6146	9222	-	-	-
1024	5124	12462	17586	7172	13826	20998	5294	12802	18096
2048	12462	28674	41136	16388	30722	47110	-	-	-
4096	26628	61102	87730	36868	67586	104454	27310	62466	89776

Moreover, owing to the symmetrical properties of the NMNT transform, the computational complexity of the proposed radix-2^2 algorithms can be further reduced, by observing the symmetry of the NMNT kernel parameters. A view of the proposed algorithm operation is illustrated by the structure shown in Figure 4 below, which represents a partial part of the signal flow graph extracted from the whole NMNT graph at a spe

cific length. It can be proved that Fig.4a and Fig.4b are equivalent at (γ=N/8) as follows:

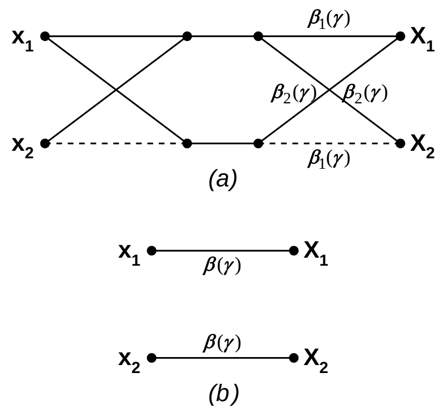


Fig. 4. Partial signal flow graph for the (a) radix-4, and (b) radix-2^2 NMNT algorithms.

From Fig.4a:

$$X_1= (x_1+x_2)\,\beta_1(\,\gamma) +(\,x_1-x_2)\,\beta_2\,(\,\gamma) \tag{55}$$

$$X_2= (x_1+x_2)\,\beta_2\,(\gamma) - (x_1-x_2)\,\beta_1\,(\,) \tag{56}$$

For (γ =N/8), $\beta_1(\gamma)$=$\beta_2(\gamma)$ we get:

$$X_1=x_1\,\beta_1\,(\gamma)+\beta_2\,(\gamma)=x_1\beta(\gamma) \tag{57}$$

$$X_2=x_2\,\beta_1\,(\gamma)\,+\beta_2\,(\gamma)=x_2\beta(\gamma) \tag{58}$$

Hence (55) is identical to (57) and (56) is identical to (58) when γ=N/8, which means that Fig.4a and Fig.4b are also identical.

As it can be seen from above figures, at each stage there are reductions in multiplications by a factor of 2, and in additions by a factor of 4 recursively. Therefore, the saving in the arithmetic complexity compared to radix-4 algorithm are $[(N-4)/6]$ multiplications and $[(N-4)/3]$ additions respectively.

6. Applications of the Proposed Algorithms

In order to proof and test the validity of the developed algorithms, the following example illustrating the NMNT application for the calculation of large integer multiplication in modular arithmetic[33, 34], which is the foundation of most public-key cryptosystems, specifically RSA [34] is given. In RSA algorithm the modulus used for private and public keys equals to the product of two primes P and Q, which means that the word length of the RSA algorithm is (P×Q)**.** For the sake of demonstration and without loss of generality, let P (126-digits) and Q (127-digits) are two primes [35] to be multiplied, such that:

P=235,723, 375,373, 223,233,257,277,337,353,373,523,557,577,727,733,757,773,223,722,732,333,235,723,772,557, 275,327,773,253,325,733,233,373,353,335,573,727,373,352,335,237.

and,

Q=1,631,576,853,416,450,450,376, 889, 988,725,553, 548,134,047, 486,329,585, 349,843,022,397,649,864, 136,156, 162,979,036,439,091,121,153,232,606,890,925,336,730,106,285,793,281.

The procedure is based on fast polynomial multiplication [12, 36], and can be summarized in the following steps:

step.1: Express the two numbers in polynomial forms as:

$$P(x) = \sum_{i=0}^{N_1} a_i x^i \tag{59}$$

$$Q(x) = \sum_{i=0}^{N_2} b_i x^i \tag{60}$$

where the coefficients a_i and b_i represent the digits of P, Q and N_1, N_2 represents polynomial degrees of $P(x)$ and $Q(x)$ respectively, in this example N_1=125 and N_2=126.

step.2: Choose N as the minimum power of two, greater than the product of the two polynomials. Since their product degree is 251, then N=256 is the nearest power of two length.

step.3: Pad (N- N_1) zero coefficients to $P(x)$ and (N- N_2) zero coefficients to $Q(x)$, to obtain new sequences $x(n)$ and $h(n)$ of length N, follows that $|x(n)|_{max}$=7 and $\sum|h(n)|$=574. According to (12), Mp must be greater than 4018, so 13 bits Mersenne number (Mp=8191), will be enough to calculate this multiplication.

step.4: $x(n)$ and $h(n)$ are transformed into their NMNT domain using the transform parameters (Mp,N,α_1, α_2)= (8191, 256, 336, 1198) producing two 256-points integer sequences $X(k)$ and $H(k)$.

step.5: Compute the convolution of $x(n)$ and $h(n)$ using the NMNT convolution property described in section 2.2, yields the following convolution output $y(n)$:

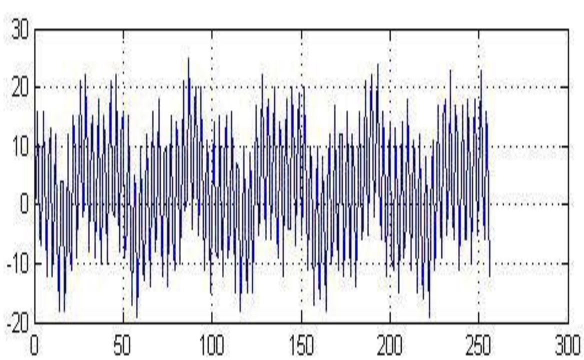
[7 59 40 48 119 119 121 166 166 180 168 201 189 221 270 293 301 338 351 362 372 396 393 486 420 451 489 480 447 512 533 527 565 519 600 482 601 598 623 629 677 653 701 643 670 762 810 864 779 901 801 884 851 931 957 1035 960 1050 954 1100 1031 1120 1119 1179 1182 1110 1139 1124 1158 1193 1238 1259 1369 1213 1347 1287 1412 1361 1448 1529 1443 1494 1556 1511 1550 1600 1544 1724 1762 1674 1727 1783 1770 1744 1809 1869 1801 1831 1850 1958 1932 1968 1969 2122 2132 2032 2128 2168 2082 2103 2190 2100 2251 2144 2219 2156 2285 2270 2197 2338 2392 2286 2447 2409 2440 2416 2425 2315 2405 2338 2247 2319 2301 2204 2243 2221 2247 2243 2218 2216 2193 2153 2113 2088 2208 2135 2192 2069 2050 1970 1969 1995 2009 1882 1923 1917 1918 1942 1915 1906 1848 1864 1889 1825 1811 1840 1789 1767 1689 1733 1685 1627 1561 1551 1643 1643 1560 1658 1601 1551 1476 1532 1481 1415 1422 1344 1430 1449 1262 1328 1290 1155 1188 1329 1241 1230 1192 1171 1170 1170 1131

1098 1049 1012 1001 1022 995 909 895 838 757 739 748 726 699 709 688 690 695 681 639 653 612 587 606 504 444 470 401 338 311 298 269 242 272 257 285 257 226 226 213 193 189 182 140 92 70 72 48 29 15 2 0 0 0 0]

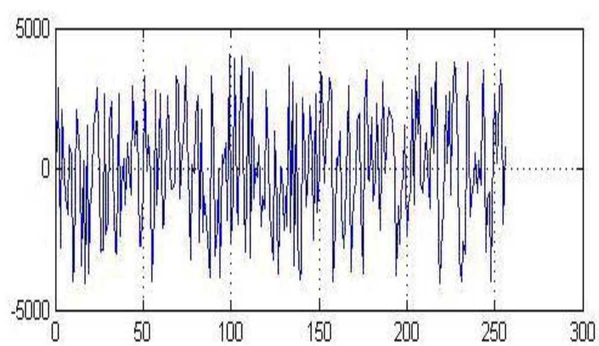
step.6: The final multiplication result can be computed by applying the adjust carry method [37] with the decimal base, the multiplication result has 252-digits length, and it is equal to:

P×Q=384,600,803,068,148, 369,222,933,011, 154,448,166,699,040,769,833,914,100,388,707,870,270,200,068, 942, 245,524,715,631,445,999,051,035,038,811,990,326,927,239,897, 974,343,679,210,292,518,252,352, 348, 607, 283,317, 930,743,916, 118,315,189,655,338,601,303,123,251, 697,409,583,984, 336,203,767,935,781,359,203, 882, 967,208,132,420,978,394,142,597.

Another example deals with the digital filtering application of the NMNT using the developed algorithms shown in Figures 5-7. In this example, the input signal to the convolution process consists of multi sinusoidal of different frequencies and these are convolved using this technique with a low pass filter. The modulus chosen for this calculation is 8191 and the transform parameters used are ($Mp, N, \alpha_1, \alpha_2$) = (8191, 256, 336, 1198). The input signal, with its multi frequencies components, is shown in Fig. 4(a) and its NMNT transform in Fig. 4(b). The impulse response of the seventh order Butterworth low pass filter is shown in Fig. 5(a) and its transform in Fig. 5(b). Fig. 6 shows the convolution result from equations (9)-(11) and it clearly shows that the filtering operation has extracted the low frequencies components from the multi frequencies input signal. This confirms the validity of NMNT transform in digital filtering applications [38].

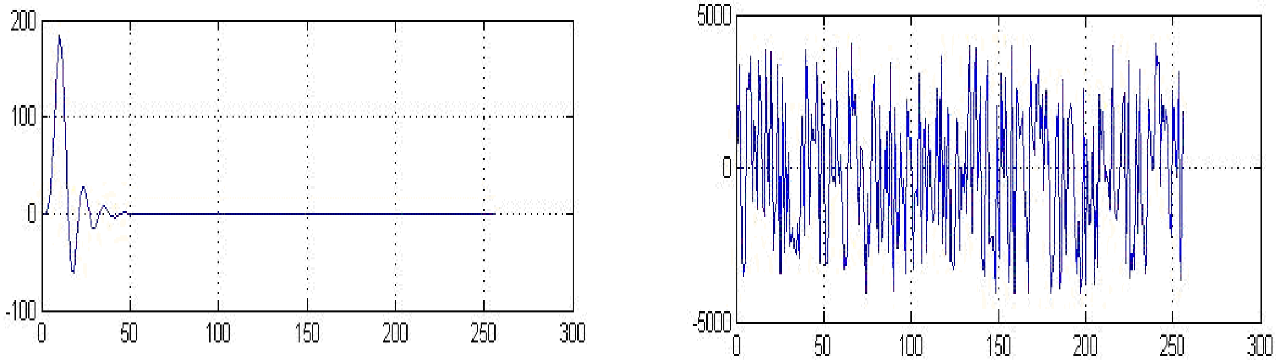


(a)



(b)

Fig. 5. (a) The 256-point multi frequency input signal; (b) Transform of the signal using 8191 as modulus.



(a) (b)

Fig. 6. (a) The 256-point impulse response of seventh order Butterworth lowpass filter; (b) The NMNT transform of filter using 8191 as modulus.

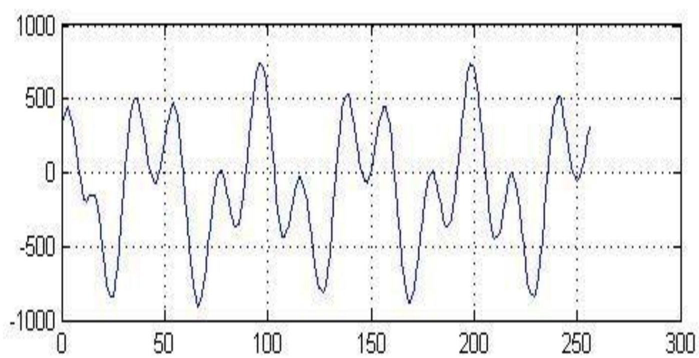


Fig. 7. Convolution results for seventh order Butterworth filter with the input signal.

7. Conclusion

In this paper, a new approach based on unscrambling technique of twiddle factors and proper divide-and-conquer relations in finite field for computing radix-2^2 DIT and DIF NMNT algorithms has been presented, and its advantages relative to the conventional multidimensional index map approach have been verified. The proposed algorithms are analysed and implemented, and their computational complexities are calculated for different transform lengths. Comparisons are carried out between the developed algorithms and the existing NMNT algorithms. These comparisons have shown that the new algorithms outperform all radix based algorithms with fewer operations. Also, the developed algorithms have significantly reduced the structural complexities with better indexing schemes make them suitable for pipeline implementations. The efficiency and validity of these algorithms are demonstrated by examples for large integer multiplication and digital filtering applications. Furthermore, the developed approach can lead to the vector-radix (VR-2^2) algorithms for multidimensional NMNT in a forward manner and provides the necessity to implement these algorithms efficiently.

8. Appendix

Proof of (30)-(33)

Since $\beta(N)$ is a root of unity of order N, then

$$\beta(N) = \langle \beta_1(N) + \beta_2(N) \rangle_{Mp} = 1 \quad \text{(A.1)}$$

From the definition of β_1 and β_2 given in (4) and (5) respectively

$$\langle \beta_1(N) \rangle_{Mp} = 1 \quad \text{(A.2)}$$

$$\langle \beta_2(N) \rangle_{Mp} = 0 \quad \text{(A.3)}$$

According to theorem-6 given in [39], β is a primitive Nth root of unity if and only if: $\beta(N/2)$= -1 mod Mp;

$$\beta\left(\frac{N}{2}\right) = \langle (\alpha_1 + j\alpha_2)^{N/2} \rangle_{Mp} = -1 \quad \text{(A.4)}$$

Firstly, from (A.4):

$$\beta_1\left(\frac{N}{2}\right) = \langle Re(\alpha_1 + j\alpha_2)^{N/2} \rangle_{Mp} = -1 \quad \text{(A.5)}$$

$$\beta_2\left(\frac{N}{2}\right) = \langle Im(\alpha_1 + j\alpha_2)^{N/2} \rangle_{Mp} = 0 \quad \text{(A.6)}$$

For integer (υ):

$$\beta_1\left(\upsilon\frac{N}{2}\right) = \langle Re\left((\alpha_1 + j\alpha_2)^{N/2}\right)^{\upsilon} \rangle_{Mp} = (-1)^{\upsilon} \quad \text{(A.7)}$$

$$\beta_2\left(\upsilon\frac{N}{2}\right) = \langle Im\left((\alpha_1 + j\alpha_2)^{N/2}\right)^{\upsilon} \rangle_{Mp} = 0 \quad \text{(A.8)}$$

Thus (A.7) and (A.8) are the proof of (30) and (31) respectively.

Secondly, from (A.4):

$$\beta\left(\frac{N}{4}\right) = \langle \left((\alpha_1 + j\alpha_2)^{N/2}\right)^{1/2} \rangle_{Mp} = (-1)^{1/2} = j \quad \text{(A.9)}$$

$$\beta_1\left(\frac{N}{4}\right) = \langle Re\left(\beta(\frac{N}{4})\right) \rangle_{Mp} = 0 \quad \text{(A.10)}$$

$$\beta_2\left(\frac{N}{4}\right) = \langle Im\left(\beta(\frac{N}{4})\right) \rangle_{Mp} = 1 \quad \text{(A.11)}$$

For integer (υ):

$$\beta\left(v\frac{N}{4}\right) = \langle\left((\alpha_1 + j\alpha_2)^{N/4}\right)^{v}\rangle_{Mp} = (j)^{v} \tag{A.12}$$

Since:

$$(j)^{v} = \begin{cases} (-1)^{\frac{v}{2}} & v{:}Even \\ j(-1)^{\frac{(v-1)}{2}} & v{:}Odd \end{cases} \tag{A.13}$$

Yields the proof of (32) and (33).

9. References

[1] S. Gudvangen, "Practical applications of number theoretic transforms," in *Procceding of NORSIG*, Asker, 1999, pp. 102-107.

[2] J. H. McClellen and C. M. Rader, *Number theory in digital signal processing*: Prentice Hall Professional Technical Reference, 1979.

[3] C. M. Rader, "Discrete Fourier transforms when the number of data samples is prime," *Proceedings of the IEEE,* vol. 56, pp. 1107-1108, 1968.

[4] H. J. Nussbaumer, *Fast Fourier transform and convolution algorithms*: Springer-Verlag, 1982.

[5] R. C. Agarwal and C. S. Burrus, "Number theoretic transforms to implement fast digital convolution," *Proceedings of the IEEE,* vol. 63, no.4, pp. 550-560, 1975.

[6] C. M. Rader, "Discrete convolutions via Mersenne transforms," *IEEE Transactions on Computers.,* vol. C-21, pp. 1269-1273, 1972.

[7] J. M. Pollard, "The fast Fourier transform over the finite fields," *Mathematics of Computation,* vol. 25, pp. 365-374, 1971.

[8] S. Boussakta and A. G. J. Holt, "Number theoretic transforms and their applications in image processing," in *Advances in Imaging and Electron Physics*. vol. Volume 111, ed: Elsevier, 1999, pp. 1-90.

[9] I. S. Reed, T. K. Truong, Y. S. Kwoh, and E. L. Hall, "Image processing by transforms over a finite field," *IEEE Transactions on Computers,* vol. C-26, pp. 874-881, 1977.

[10] R. E. Blahut, *Algebraic codes for data transmission*. Oxford: Cambridge University Press, 2003.

[11] D. Anindya, P. K. Piyush, S. Chandan, and S. Ramprasad, "Fast integer multiplication using modular arithmetic," presented at the Proceedings of the 40th annual ACM symposium on Theory of computing, Victoria, British Columbia, Canada, 2008.

[12] A. E. Yagle, "Fast algorithms for matrix multiplication using pseudo-number-theoretic transforms," *IEEE Transactions on Signal Processing,* vol. 43, pp. 71-76, 1995.

[13] I. E. Shparlinski, *Number theoretic methods in cryptography: complexity lower bounds*: Birkhauser book, Springer, 1999.

[14] G. Drauschke and M. Tasche, "Exact deconvolution using number-theoretic transforms," *Computers and Mathematics with Applications,* vol. 15, pp. 757-768, 1988.

[15] A. V. Oppenheim and C. J. Weinstein, "Effects of finite register length in digital filtering and the fast Fourier transform," *Proceedings of the IEEE,* vol. 60, pp. 957-976, 1972.

[16] R. Agarwal and C. Burrus, "Fast convolution using Fermat number transforms with applications to digital filtering," *IEEE Transactions on Acoustics, Speech and Signal Processing,* vol. 22, no. 2, pp. 87-97, 1974.

[17] S. Boussakta and A. G. J. Holt, "New number theoretic transform," *Electronics Letters,* vol. 28, pp. 1683-1684, 1992.

[18] S. Boussakta and A. G. J. Holt, "New transform using the Mersenne numbers," *IEE Proceedings -Vision, Image and Signal Processing.,* vol. 142, pp. 381-388, 1995.

[19] O. Alshibami, S. Boussakta, and M. Aziz, "Radix-4 algorithm for the new Mersenne number transform," in *Proceedings of the 5th International Conference on Signal Processing , WCCC-ICSP 2000.* , 2000, pp. 54-56 vol.1.

[20] S. Boussakta, O. Alshibami, and A. Bouridane, "Radix-4 decimation-in-frequency algorithm for the new Mersenne number transform," in *Proceedings of the 10th IEEE International Conference on Electronics, Circuits and Systems, ICECS 2003.* , 2003, pp. 1133-1136 Vol.3.

[21] O. Alshibami, S. Boussakta, M. Aziz, and D. Xu, "Split-radix algorithm for the new Mersenne number transform," in *The 7th IEEE International Conference on Electronics, Circuits and Systems, ICECS 2000.* , 2000, pp. 583-586 vol.1.

[22] M. T. Hamood and S. Boussakta, "Decimation-in-frequency split-radix algorithm for computing new Mersenne number transform," in *IEEE International Symposium onSignal Processing and Information Technology (ISSPIT),* 2010 pp. 298-302.

[23] A. Cortes, I. Velez, and J. F. Sevillano, "Radix-r^k FFTs: Matricial representation and SDC/SDF pipeline implementation," *IEEE Transactions on Signal Processing.,* vol. 57, pp. 2824-2839, 2009.

[24] H. Shousheng and M. Torkelson, "A new approach to pipeline FFT processor," in *Proceedings of the 10th International on Parallel Processing Symposium, IPPS '96.*, 1996, pp. 766-770.

[25] H. Shousheng and M. Torkelson, "Design and implementation of a 1024-point pipeline FFT processor," in *Proceedings of the IEEE Custom Integrated Circuits Conference.*, 1998, pp. 131-134.

[26] O. Jung-yeol and M. S. Lim, "New radix-2 to the 4th power pipeline FFT processor," *IEICE Transactions on Electronics,* vol. E88, pp. 1740-1746, 2005.

[27] M. Young-jin and K. Young-il, "A mixed-radix 4-2 butterfly with simple bit reversing for ordering the output sequences," in *The 8th International Conference on Advanced Communication Technology, ICACT 2006.* , 2006, pp. 1771-1774.

[28] J. Yunho, Y. Hongil, and K. Jaeseok, "New efficient FFT algorithm and pipeline implementation results for OFDM/DMT applications," *IEEE Transactions on Consumer Electronics.,* vol. 49, pp. 14-20, 2003.

[29] J. Yunho, T. Yonji, K. Jaeseok, P. Junhyun, K. Dongkyu, and P. Hyuncheol, "Efficient FFT algorithm for OFDM modulation," in *Proceedings of IEEE Region 10 International Conference on Electrical and Electronic Technology, TENCON.* , 2001, pp. 676-678 vol.2.

[30] C. Burrus, "Bit reverse unscrambling for a radix-2^M FFT," in *IEEE International Conference on Acoustics, Speech, and Signal Processing, ICASSP '87.*, 1987, pp. 1809-1810.

[31] C. S. Burrus, "Unscrambling for fast DFT algorithms," *IEEE Transactions on Acoustics, Speech and Signal Processing.,* vol. 36, pp. 1086-1087, 1988.

[32] P. E. Papamichalis and C. S. Burrus, "Conversion of digit-reversed to bit-reversed order in FFT algorithms," in *International Conference on Acoustics, Speech, and Signal Processing, ICASSP-89.* , 1989, pp. 984-987 vol.2.

[33] G. Brassard and P. Bratley, *Algorithmics, theory and practice*: Prentice-Hall, Englewwod Cliffs., 1988.

[34] K. Kalach and J. P. David, "Hardware implementation of large number multiplication by FFT with modular arithmetic," in *The 3rd International IEEE-NEWCAS Conference.* , 2005, pp. 267-270.

[35] C. K. Crandall and G. L. Honaker. [Online]. Available: http://primes.utm.edu/curios/index.php

[36] E. Chu and A. George, *Inside the FFT black-box: Serial and parallel fast Fourier transform algorithms*: CRC Press, 2000.

[37] R. E. Crandall and C. Pomerance, *Prime numbers: A computational perspective*: Springer, 2005.

[38] S. Boussakta and A. G. J. Holt, "Filtering employing a new transform," in *Proceedings of the OCEANS '94.* , 1994, pp. I/547-I/553 vol.1.

[39] R. Creutzberg and M. Tasche, "Parameter determination for complex number theoretic transforms using cyclotomic polynomials " *Mathematics of computation,* vol. 52, pp. 189-200, 1989.