Radix-2/4 Streamlined Real Factor FFT Algorithms

Shaik Qadeer(1), Mohammed Zafar Ali Khan(2) and Syed A. Sattar(3)

(1) Department of Electrical and Electronics engineering, MJ College of Engineering, Hyderabad, India
{email: haqbei@gmail.com}
(2) Department of Electrical engineering, Indian Institute of Technology, Hyderabad, India
{email: zafar@iith.ac.in}
(3) Department of Electronics and communication engineering, RITS, Hyderabad, India
{email: syedabdulsattar1965@gmail.com}

Abstract — In this paper three Real Factor FFT algorithms are presented. Two of them are based on Radix-2 and one on Radix-4. The computational complexity of Radix-2 and Radix-4 is shown as order \( N^{\log_2 N - \frac{1}{2}} \) and \( N^{\log_4 N - \frac{1}{2}} \) respectively unlike their standard counterparts \( N^{\frac{5}{2}} \) and \( N^{\frac{5}{4}} \). Moreover, the proposed algorithms also require fewer multiplications than their standard FFTs. We then show that the fixed point implementation of 'real factor' FFT can be modified, with unique scaling procedure, so that its noise to signal ratio (NSR) is lower than the NSR of standard FFT. Finally, implementation issues are presented which verify the suitability of the proposed 'real factor' FFT's.

Keywords - Discrete Fourier Transform (DFT), Decimation in frequency (DIF), Fast Fourier Transform (FFT), and real-factor FFT

I. INTRODUCTION

The Discrete Fourier transform (DFT) is among the most fundamental operations in digital signal processing [7, 9]. When considering the alternate implementations, the FFT/IFFT algorithm should be chosen keeping in view the execution speed, hardware complexity, flexibility and precision [8, 9]. Most of the above mentioned parameters depend on the exact count of arithmetic operations (real additions and multiplications), herein called flops (floating-point operation), required for a DFT of a given size N which remains an intriguing unsolved mathematical question. Among the numerous developments that followed Cooley and Tukey's [1] original contribution are the Winograd Fourier transform algorithm (WFTA) [2] and the real factor algorithms [3, 4] for reduction in the order of the multiplicative complexity. However both WFTA and 'real-factor' FFT did not meet expectations once implemented as the number of additions (and data transfers) also matter in the implementation. In additional Rader and Brenner's 'real factor' FFT's are ill-conditioned i.e., "small computational errors lead to large output errors"[3] due to the large values that the twiddle factor can take.

In this paper the solutions for both the problems of Rader and Brenner's 'real factor' FFT algorithm is presented. We first show that the arithmetic complexity (Multiplications plus additions, also known as the 'Flop count') of the Rader and Brenner's 'real factor' FFT can be reduced to about \( N^{\frac{5}{2}} \) which is less than the arithmetic complexity of Cooley-Tukey radix-2 FFT, the corresponding Radix-4 is also discussed which takes \( N^{\frac{5}{4}} \) which is again lesser than its counterpart. It is then shown that the modified 'real factor' FFT is free from ill-conditions as the magnitude of all the twiddle factors is less than 1 and hence has lower NSR than radix-2 Cooley-Tukey FFT. As a result the proposed FFT is more suitable as it requires only half the number of real multiplications.

The rest of the paper is organized as follows: In section II, modified DIT version of Rader and Brenner Radix-2 FFT that requires \( 5N\log_2 N \) flops is first presented in subsection II-A. In subsection II-B we derive the corresponding radix-2 DIF FFT. In subsection II-C, it is shown that the flop count can be further reduced to \( 2N\log_2 N \). Notably this is the best known flop count that can be achieved by a radix-2 FFT. In subsection II-D equivalent Radix-4 FFT algorithm is covered. We then present in section III implementation issues of real factor FFT based on the radix-2 FFTs developed in this paper and conclusions are given in section IV.

II. NEW FFT

In this section, we first modify Rader and Brenner DIT FFT that requires \( 5N\log_2 N \) flops. The corresponding DIF version is derived and the algorithm is further modified so that the flop count is reduced down to \( 2N\log_2 N \). Finally a similar radix-4 DIF FFT algorithm is also derived.
A. DIT version of Rader and Brenner FFT:

The essence of Rader and Brenner's DIT FFT is as follows: Let \( \{A_k\} \) denotes \( N\)-point DFT of the sequence \( a_n \) of \( N=2^M \) i.e. \( \{A_k\}=\text{DFT}_{N/2}\{a_n\} \). Then radix-2 DIT FFT is given by

\[
A_k = B_k + W_N^k \cdot D_k, \quad k = 0, N-1
\]

where \( W_N^k=\exp(-j2\pi k/N) \) and the \( N/2 \)-point sequences are defined as \( \{b_n\} = \{a_{2n}\} \) and \( \{d_n\} = \{a_{2n+1}\} \). Note that for a \( 2N \)-point sequence, if the index \( k, n \) is greater than \( N-1 \) we assume \( k, n \mod N/2 \). Since in general \( D_k \) and \( W_N^k \) are both complex in nature, the basic butterfly operation indicated in (1) requires '4' real multiplications and '4' real additions. Rader and Brenner introduced the following sequence \([3]\)

\[
q_n = \sum_{m=0}^{N/2-1} d_m \Rightarrow n=0, N/2-1
\]

where \( q_n = \frac{N}{2} \sum_{m=0}^{N/2-1} d_m \). It follows that (1) can be rewritten as

\[
A_0 = B_0 + C_0, \quad A_{N/2} = B_0 - C_0, \quad \text{and}
A_k = B_k - \frac{j}{2} \csc\left(\frac{2\pi k}{N}\right)C_k, \quad k \neq 0 \& N/2
\]

where \( \{C_k\}=\text{DFT}_{N/2}\{c_n\} \) and \( \csc(x) \) is the cosecant of \( x \).

If we change the sign plus to minus in the definition of \( c_n \) then the coefficient of \( C_k \) in (3) is \( 1/2\sec(2\pi k/N) \) where \( \sec(x) \) is the secant of \( x \). In either case, (3) requires '4' real multiplications and '4' real additions. But this is at the expense of more additions to compute \( c_n \) from \( b_n \). Therefore total real additions \( (A(N)) \) will be \( A(N)=2N+N+2N=5N \). The real multiplications \( (M(N)) \) are given by \( M(N)=N \) and the total flop count \( (T(N)) \) is given by \( T(N)=6N\log_2 N \).

To further reduce the arithmetic complexity here we modify \( c_n \) as

\[
c_n = d_n - d_{n+1}, \quad n=0, N/2-1
\]

It follows that (3) is still valid with the exception that \( C_0 \) is defined as

\[
c_0 = \sum_{n=0}^{N/2-1} c_n.
\]

Denote \( P(n,N) = \frac{j}{2} \csc\left(\frac{2m}{N}\right) \), then the DIT computation for the new algorithm is schematically represented for \( N=16 \) is shown in Figure 1. The arithmetic complexity of this DIT FFT will become \( T(N)=5N\log_2 N \) due to the reduction in the number of addition to compute \( c_n \) using (4) instead of (2). Note that the difference between Rader and Brenner’s FFT and our proposed FFT is in the computation of \( C_0 \).

B. Derivation of DIF version of New FFT

According to the principle of DIF FFT, the even and odd DFT coefficients are given by

\[
A_{2k} = \text{DFT}_{N/2}\left\{a_n + a_{n+N/2}\right\}, \quad k=0, N/2-1
\]

\[
A_{2k+1} = \text{DFT}_{N/2}\left\{a_n - a_{n+N/2}\right\} W_N^k, \quad k=0, N/2-1
\]

Fig.1 The decomposition of \( N \) point DFT into two \( \frac{N}{2} \) point DFTs of new DIT Radix-2 FFT algorithm for \( N=16 \).

Fig.2 The decomposition of \( N \) point DFT into two \( \frac{N}{2} \) point DFTs for new DIF Radix-2 FFT algorithm for \( N=16 \).
To eliminate the complex multiplication factor from (7), we define the sequence \( \{c_n\} \) as

\[
c_0 = \left\{ a_{n} - a_{n+1} \right\} \frac{N}{2}, n = 0, \frac{N}{2} - 1
\]

where \( \delta(n) \) is unit impulse function. Equation (8) can be simplified as

\[
c_0 = \left\{ a_{n} - a_{n+1} \right\} ,
\]

(9)

\[
c_n = -\frac{j}{2} \csc \left( \frac{2m}{N} \right) \left\{ a_{n} - a_{n+1} \right\}, n = 1, \ldots, \frac{N}{2} - 1
\]

(10)

Applying \( \frac{N}{2} \)-point DFT on both sides of (8) we get

\[
C_k = C_{k+1}^{+} + c_0 = A_{2k+1}, k = 0, \frac{N}{2} - 1
\]

(11)

where \( C_k = \text{DFT} \left\{ c_n \right\} \). The Radix-2 DIF computation for the new algorithm is schematically represented for \( N=16 \) is shown in Figure 2.

The number of multiplications can be calculated from (10). For each value of \( n \), '2' real multiplications are needed and the total number of real multiplications with given two \( \frac{N}{2} \)-points DFTs is \( (N-2) \). Therefore the total number of real multiplications for a \( N \)-point DFT, is given by

\[
M(N) = \begin{cases} 
\frac{2M(N/2)}{2} + N - 2, & \text{if } N = 2^n \geq 2 \\
0, & \text{if } N = 1
\end{cases}
\]

(12)

Solving (12) by repeated substitutions we have

\[
M(N) = N \log_2 N - N
\]

(13)

The number of real additions given two \( \frac{N}{2} \)-points DFTs is calculated as follows

- \( 2(N/2) \) real additions for evaluating (6) for \( n=0..N/2-1 \).
- \( 2(N/2+1) \) real additions for evaluating (8) for \( n=0..N/2-1 \).
- \( 4(N/2) \) real additions for evaluating (11) for \( n=0..N/2-1 \).

It follows that

\[
A(N) = \begin{cases} 
\frac{2A(N/2)}{2} + 4N, & \text{if } N = 2^n \geq 2 \\
0, & \text{if } N = 1
\end{cases}
\]

(14)

Solving (14) by repeated substitutions we have

\[
A(N) = 4N \log_2 N
\]

(15)

The total flop count is then given by \( T(N) = 5N \log_2 N - N \). Importantly, the DIF version of the FFT presented in sub section II-A and II-B requires the same number of computations (real adds + real multiplies) as normal Radix-2 DIT or DIF FFT while requiring lesser multiplies. Also as compared to DIT version of Rader and Brenner’s FFT the modified DIT shown in subsection II-A and DIF in II-B requires \( N \) fewer real additions.

C. Modified DIF version of new FFT

In this sub section we show that the computational complexity can be reduced down further. If we add \( c_0 \) to (10) then for \( n= (N/4) \),

\[
c_n = -\frac{j}{2} \csc(\pi / 2) \left\{ a_{n} / 4 - a_{n+1} / 4 \right\} + c_0
\]

(16)

And rest of the sequence \( \{c_n\}, n = 0..N/4-I, N/4 + I, ..., N/2-I \) is given by (9)-(10). Equation (11) for this case modifies to

\[
A_{2k+1} = \begin{cases} 
C_k - C_{k+1}^{+} + 2c_0, & \text{for even values of } k \\
C_k - C_{k+1}^{+}, & \text{otherwise}
\end{cases}
\]

(17)

The DIF computation for the new algorithm is schematically represented for \( N=16 \) are shown in Figure 2.
and Figure 3. With these substitutions it is obvious that the numbers of real multiplications are same. However, as compared to the sub section II-A and II-B the number of real additions are reduced by N/2-2. Accordingly, the number of real additions is given by

\[ A(N) = \begin{cases} 2N & \text{if } N = 2^n \geq 4 \\ 0 & \text{if } N = 1 \end{cases} \]

(18)

Solving (18) by repeated substitutions we have

\[ A(N) = \frac{7}{2} N \log_2 N - N \]

(19)

The total flop count is given by

\[ T(N) = 4 \frac{1}{2} N \log_2 N - 2N \].

Importantly a flop count of this Radix-2 FFT is less than any other Radix-2 FFTs (including standard DIT and DIF Radix-2 FFTs).

D. New Radix-4 DIF FFT:

According to the principle of Radix-4 DIF FFT, the even and odd DFT coefficients are given by

\[ a_{4k} = DFT_N \left( \frac{a_n + a_{N-n}}{2} + \frac{a_{N-n} + a_n}{2} \right), \]

(20)

\[ a_{4k+2} = DFT_N \left( \frac{a_n - a_{N-n}}{2} - \frac{a_{N-n} - a_n}{2} \right) W_N^2, \]

(21)

\[ a_{4k+4} = DFT_N \left( \frac{a_n + a_{N-n}}{2} + j \left( \frac{a_{N-n} - a_n}{2} \right) \right) W_N^3, \]

(22)

for \( k = 0, \ldots, N/4 - 1 \)

To eliminate the complex multiplication factor from (21), we similarly define the sequence \( \{c_n\} \) as

\[ c_n = 1 - W_N^4 \delta(n) = \left( \frac{a_n + a_{N-n}}{2} \left( \frac{a_{N-n} + a_n}{2} \right) \right) W_N^2, \]

(24)

\[ n = 0, \ldots, N/4 - 1 \]

where \( \delta(n) \) is unit impulse function. Equation (24) can be simplified as

\[ c_n = \frac{1}{2} \left( \frac{a_n + a_{N-n}}{2} - \left( \frac{a_{N-n} + a_n}{2} \right) \right), \]

(25)

Applying N/4 –point DFT on both sides of (24) we get

\[ c_{k+1} = 0 \quad \text{for } k = 0, \ldots, N/4 - 1 \]

(26)

where \( C_k = DFT_{N/4} \{ c_n \} \) and \( C_k = DFT_{N/4} \{ c_n \} \). As in subsection II-C, if we add \( c_0 \) to (25) then for \( n = N/8 \)

\[ c_N = \frac{-j}{2} \cdot \text{csc} \left( \frac{\pi}{4} \right) \left( \frac{a_n + a_{N-n}}{2} - \left( \frac{a_{N-n} + a_n}{2} \right) \right) + c_0, \]

and for the rest of the sequence \( \{c_n\}, n = 0 \ldots N/8-1, N/8 + 1, \ldots, N/4-1 \) is given by (25). Equation (26) for this case modifies to
Solving (28) by repeated substitutions we have

\[ T(N) = \begin{cases} 
4T(N) + 16L \frac{N}{2}, & \text{if } N = 4^a \geq 4 \\
16, & \text{if } N = 4 
\end{cases} \]  

(28)

Solving (28) by repeated substitutions we have

\[ T(N) = \frac{4^4}{8} \log_2 N 

(28.1)

E. Comparison of computational Complexity of various FFT algorithms:

The computational complexity comparison for Radix-2/4 FFTs is given in Table I (Note that RF-FFT means Real Factor while RB-FFT means Rader and Brenner's FFT). Observe that the new DIF version of modified real factor FFT presented in section II requires not only lesser multiplications but also the total flop counts (T(N)).

III. IMPLEMENTATION ISSUES OF PROPOSED FFT

In this section we consider the implementation issues of 'real factor' FFTs presented in section II (Note that we cover here only Radix-2 DIF and other can be similarly obtained [11, 12, and 15]). The noise to signal ratio (NSR) of Rader and Brenner's algorithm is known to be very poor because of the large values of \( \text{csc}(\frac{2\pi}{N}) \) [3]. Accordingly we first modify the real factor FFT for fixed point implementation in section III-A to eliminate the problem of ill computation; then we discussed the remaining implementation issues in III-B and expected execution times is given in section III-C.

A. Modification to real factor FFT for fixed point implementation

To eliminate the problem of ill-conditioned we proposed a scaling method to (9)-(10) as

\[ \hat{c}_0 = \frac{c_0}{2} = \left\{ \begin{array}{ll}
\frac{a_0}{2} - \frac{a_N}{2} & \text{if } n = 1 \\
-\frac{a_0}{2} - \frac{a_N}{2} & \text{if } n = 2
\end{array} \right. \]  

(29)

\[ \hat{c}_n = \frac{c_n}{2} = \left\{ \begin{array}{ll}
\frac{a_0}{2} - \frac{a_N}{2} & \text{if } n = 1 \\
-\frac{a_0}{2} - \frac{a_N}{2} & \text{if } n = 2
\end{array} \right. \]  

(30)

where \( l \) is chosen such that \( \text{csc}(2\pi/N) \leq 2^{l+1} \). It follows that \( \text{csc}(2\pi/N)\leq 2^{l+1} \), for all \( n \). This modification ensures that the real multiplication factors are less than 1. Applying DFT on both sides of (8), with this modification we get

\[ \frac{\hat{c}_k}{\hat{c}_{k+1}} = A_{2k+1}, \quad k = 0, \frac{N}{2} - 1 \]  

(31)

where \( \hat{c}_k = \text{DFT} \frac{N}{2} \hat{c}_n \) and \( \text{DFT} \frac{N}{2} \hat{c}_n = \hat{c}_0 \).

The purely imaginary twiddle factor is now given by \( P(n, N) = \frac{-j}{2^{l+1}} \text{csc}(\frac{2\pi}{N}) \). A similar algorithm with purely real twiddle factor can be similarly developed. Note that in both the cases \( 0 < P(n, N) < 1 \). The advantage of the presented fixed point algorithm can be observed from (31) wherein the effect of large values of \( \text{csc}(\theta) \) has been captured in \( l \) and the effective precision at the output is \( l + b \) bits [6, 10].

A. In-place computation of proposed FFT algorithm:

Figure 5 shows the In-Place computation diagram of the proposed real factor FFT, which implements (6) through (17) for \( N=8 \). It is observed that Part-I of algorithm is similar to Cooley-Tukey's radix-2 DIF FFT algorithms, [1] except the twiddle factor \( P(n, N) \), which is defined earlier (but not shown in algorithm ). Once a butterfly operation is performed on a pair of complex number \((a, b)\) to produce \((A, B)\), there is no need to save the input pair \((a, b)\), except the first value of vector 'c' as shown in Figure 5. Hence we can store the results \((A, B)\) in the same locations as \((a, b)\).
Consequently we require a fixed number $2N + \left( \frac{N}{2} - 1 \right)$ of storage registers in order to store the results (note that N is a complex number) of the computations at each stage. Since $2N + \left( \frac{N}{2} - 1 \right)$ storage locations are used throughout the computation of the N-Point DFT, we can say that the computations are done in-place. A second observation is need of data shuffling (DS) in Part-II of proposed algorithm.

<table>
<thead>
<tr>
<th>S.No</th>
<th>Problem Length</th>
<th>C-T in m sec</th>
<th>Proposed R2DIF in m sec</th>
<th>Proposed R4DIF in m sec</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>$2^1$</td>
<td>5Nlog$_2$N</td>
<td>$\frac{1}{2}N\log_2N - 2N$</td>
<td>$\approx 4\frac{1}{2}N\log_2N$</td>
</tr>
<tr>
<td>2</td>
<td>4</td>
<td>0.0005*</td>
<td>0.0005</td>
<td>0.0013</td>
</tr>
<tr>
<td>3</td>
<td>8</td>
<td>0.0020</td>
<td>0.0014</td>
<td>0.0013</td>
</tr>
<tr>
<td>4</td>
<td>16</td>
<td>0.0060</td>
<td>0.0046</td>
<td>0.0042</td>
</tr>
<tr>
<td>5</td>
<td>32</td>
<td>0.00160</td>
<td>0.00128</td>
<td>0.00116</td>
</tr>
<tr>
<td>6</td>
<td>64</td>
<td>0.00400</td>
<td>0.00328</td>
<td>0.00298</td>
</tr>
<tr>
<td>7</td>
<td>128</td>
<td>0.00960</td>
<td>0.00800</td>
<td>0.00728</td>
</tr>
<tr>
<td>8</td>
<td>256</td>
<td>0.01240</td>
<td>0.02400</td>
<td>0.01720</td>
</tr>
<tr>
<td>9</td>
<td>512</td>
<td>0.01520</td>
<td>0.09856</td>
<td>0.08992</td>
</tr>
<tr>
<td>10</td>
<td>1024</td>
<td>0.02500</td>
<td>2.21016</td>
<td>2.09096</td>
</tr>
<tr>
<td>11</td>
<td>2048</td>
<td>0.03400</td>
<td>4.63000</td>
<td>4.44156</td>
</tr>
<tr>
<td>12</td>
<td>4096</td>
<td>0.06280</td>
<td>10.6450</td>
<td>9.72800</td>
</tr>
<tr>
<td>13</td>
<td>8192</td>
<td>0.12620</td>
<td>23.1424</td>
<td>21.1456</td>
</tr>
<tr>
<td>14</td>
<td>16384</td>
<td>0.25340</td>
<td>49.9712</td>
<td>45.6704</td>
</tr>
</tbody>
</table>

*Note that our algorithms works for N>2.

IV. CONCLUSION

In this paper we have shown that the arithmetic complexity of real factor FFTs is significantly less than that of standard FFT. We have shown that these FFTs have a significant advantage compared to corresponding standard FFTs. When it comes to fixed point implementation, as it has lower number of noise sources (real multiplications) and hence lower NSR. This algorithm is of practical importance as Radix-2/4 FFTs are generally preferred in ASIC implementation and for software defined radio (cognitive radio) applications due to ease of re-configurability. Over all these factors are in favor of real factor FFTs. However, the disadvantage is additional memory storage of $C_0$ required for each stage. But as the cost of memory is decreasing storage may not be a significant factor.

A part of this paper was presented at EUSIPCO 2010, [14].
REFERENCES