Complexity of the Composite Length FFT Algorithms

Rafayel, Barseghyan
Institute for Informatics and Automation Problems of NAS of RA
Yerevan, Armenia
e-mail: rafayelbarseghyan@ipia.sci.am

ABSTRACT
In this paper logarithmic formula is derived which allows to compute exact number of necessary operations for computing discrete Fourier transform (DFT) of composite ($q \times 2^p$, where q is an arbitrary odd integer) length. Developed expressions allow to compute the number of arithmetic operations for both 2/4 and 2/8 split-radix algorithms.

Keywords
Fast Fourier transform (FFT), split-radix algorithm, computational complexity

1. INTRODUCTION
The discrete Fourier transform has a wide range of applications in many fields of science and engineering[1],[2],[3]. The main reason for its popularity is the existence of various algorithms which allows to significantly reduce the computational complexity. These algorithms are generally known as the fast Fourier transforms (FFT). Fast algorithm for efficient computation of DFT was first introduced by Culey and Tukey by their historical paper in 1965 [4]. FFT algorithms allows to compute DFT of size N with $O(N \lg N)$ operations in opposite of direct form computation which require $O(N^2)$ operations. There are number of FFT algorithm, but most popular methods are based on fixed-radix and split-radix approaches. Split-radix algorithms have been considered to be the most computationally efficient and structurally regular.

Split-radix algorithm was introduced first by Yavne [5] in 1969 and later by various authors Vetterli, Duhamel [9],[15]. Split-radix algorithm allows to compute DFT of $N = 2^m$ with $4N \lg N - 6N + 8$ arithmetic operations. In recent years by various authors [6], a new modification of split-radix algorithm was developed which allows to perform DFT of $N = 2^m$ with slightly reduced number of arithmetic operations.

For applications which need to perform DFT of sizes $N \neq 2^m$ usually specialists use zero padding technique. It means that input sequence is filled by zeros until it becomes power of two length for performing any available FFT algorithm. Such method significantly decreases required number of arithmetic operations. Because DFT for input sequences of non power two of length required in many practical applications it is important problem.

Algorithm for computing DFT for sizes $q \times 2^p$, where q is an odd integer, was first introduced by Bi and Chen in 1998 [7]. Algorithm has a 2/4 split-radix structure and in case of $q = 1$ has a same complexity as conventional split-radix FFT algorithm. After that in 2004 by Bouguezel and et.al. In [12] presented new improved algorithm for $q \times 2^p$ length DFT. Algorithm was based on 2/8 split-radix FFT algorithm scheme and improves such important factors as data transfer, address generation, twiddle factor computation and access to the lookup table, but number of arithmetic operations has not been reduced. In 2010 by Bi and Chen [8] published a new paper where authors presented unified method for generation of $2/2a$ (where a is a integer and $a > 1$) split-radix algorithms for $q \times 2^p$ length DFTs.

In this paper we developed general logarithmic formula for calculating number of arithmetic operations for 2/4 and 2/8 split-radix algorithms for $q \times 2^p$ length DFTs. For all $q < 20$ special cases developed formulas for counting exact number of arithmetic operations.

2. GENERAL ALGORITHM
Let $x = \{x_0, x_1, \ldots, x_{N-1}\}^T$ be a complex valued column-vector of length N, where $N = q \times 2^p$ and q is an odd integer. The DFT of this vector is defined as

$$X[k] = \sum_{k=0}^{N-1} x[n] W_N^{nk} \qquad (1)$$

where

$$0 \leq k \leq N-1,\ {W_N}^n = \exp\left(-j\frac{2\pi}{N}n\right) = \cos\left(\frac{2\pi}{N}n\right) - j\sin\left(\frac{2\pi}{N}n\right),$$

$$j = \sqrt{-1}.$$

Below algorithm from [8] is presented. Even indices of the transform are computed by

$$X[2k] = \sum_{n=0}^{N/2-1} (x[n] + x[n + N/2]]) W_{N/2}^{nk} \qquad (2)$$

where $X[2k]$ is a $N/2$ DFT. The odd indices are defined by

$$X[2ak + l] = \sum_{n=0}^{N-1} x[n] W_N^{n(2ak+l)} \qquad (3)$$

where $0 \leq k \leq (N/2a) - 1$, and a is an integer ($a > 1$) and l has a selected odd values so that $2ak + l$ generates $N/2$ odd integers that can be uniquely matched to all the odd index values between 0 and N. With some manipulations based on the periodic and symmetric properties of $W_N^{n(2ak+l)}$, (3) can be represented as

$$\begin{aligned} X[2ak+l] &= \sum_{n=0}^{(N/2a)-1} x'[n]W_N^{nl}W_{N/2a}^{nk} + \dots \\ &+ \sum_{n=0}^{(N/2a)-1} x'[n+N/2a]W_N^{(n+N/2a)l}W_{N/2a}^{nk} + \dots \\ &+ \sum_{n=0}^{(N/2a)-1} x'[n+\frac{(a-1)N}{2a}]W_N^{(n+\frac{(a-1)N}{2a})l}W_{N/2a}^{nk} \end{aligned} \quad (4)$$

where $n = 0, 1, 2 \dots, N/2-1$ and

$$x'[n] = x[n] - x[n+N/2]. \quad (5)$$

It is seen that (4) is a length-$N/2a$ DFT whose input sequence is the result from the computation inside the brackets of (4) for $n = 0, 1, 2 \dots, N/2-1$. In summary, the even indexed outputs of (1) are obtained from one length-$N/2$ DFT defined in (2) based on the radix-2 decomposition, and the odd indexed outputs are obtained from a length-$N/2a$ DFTs based on the radix-$2a$ decomposition. The complexity of algorithm can be computed by the following expressions

$$\begin{aligned} C_N^{\times} &= C_{N/2}^{\times} + aC_{N/2a}^{\times} + \tfrac{N}{2a}C^{\times} + 2N - C_t^{\times}, \\ C_N^{+} &= C_{N/2}^{+} + aC_{N/2a}^{+} + \tfrac{N}{2a}C^{+} + 3N - C_t^{+}, \end{aligned} \quad (6)$$

where by $C^{\times}$ and C^{+} denote number of real multiplications and additions that are used for each of the inner sums defined in (4), and $C_t^{\times}$ and C_t^{+} are the number of real multiplications and additions saved from all the trivial twiddle factors W_N^{nl} in (4).

3. 2/4 SPLIT-RADIX ALGORITHM

For $a = 2$ the algorithm becomes a modified version of conventional 2/4 split-radix algorithm. Inserting $a = 2$ in (4) we get

$$\begin{aligned} X[4k+l] &= \sum_{n=0}^{N/4-1} W_N^{nl}(\sum_{n=0}^{1} x'[n+i\frac{N}{4}]W_{N/4}^{il})W_{N/4}^{nk} = \\ &= \sum_{n=0}^{N/4-1} W_N^{nl}(x'[n] + (-j)^l x'[n+\frac{N}{4}])W_{N/4}^{nk} \end{aligned} \quad (7)$$

From (7) we can see that it reduces to conventional split-radix algorithm which is reported in [5],[9],[10]. To cover all odd indices we set $l = \{-1, 1\}$. In this case we have arithmetic computational gain only in case of $n = 0$ and $n = N/8$ (W_N^0 and $W_N^{lN/8}$ twiddle factors become trivial).

Now it is easy to see that the number of arithmetic operations are

$$\begin{aligned} C_N^{+} &= C_{N/2}^{+} + 2C_{N/4}^{+} + 4N - 4q \\ C_N^{\times} &= C_{N/2}^{\times} + 2C_{N/4}^{\times} + 2N - 12q \end{aligned} \quad (8)$$

Using difference equations theory [11] we developed the software system which allows us to get the number of arithmetic operations required for computation of (7) in logarithmic form

$$\begin{aligned} C_N^{+} &= \tfrac{8}{3}pq2^p - \tfrac{2^p(28q - 3C_{2q}^{+} - 3C_q^{+})}{9} + \\ &+ \tfrac{(-1)^p(10q - 3C_{2q}^{+} + 6C_q^{+})}{9} + 2q, \\ C_N^{\times} &= \tfrac{4}{3}pq2^p - \tfrac{2^p(44q - 3C_{2q}^{\times} - 3C_q^{\times})}{9} - \\ &- \tfrac{(-1)^p(10q + 3C_{2q}^{\times} - 6C_q^{\times})}{9} + 6q \end{aligned} \quad (9)$$

where by C_q and C_{2q} are denoted the complexities of q and $2q$ length DFTs, respectively. Using methods from [7] for computing $2q$-length DFT we have

$$\begin{aligned} C_{2q}^{+} &= 2C_q^{+} + 4q, \\ C_{2q}^{\times} &= 2C_q^{\times} \end{aligned} \quad (10)$$

finally puting (10) into (9) and $2^p = \frac{N}{q}$ we get

$$\begin{aligned} C_N^{+} &= \tfrac{8}{3}pq2^p - \tfrac{2^p}{9}(16q - 9C_q^{+}) - \tfrac{2}{9}q(-1)^p + 2q, \\ C_N^{\times} &= \tfrac{4}{3}pq2^p - \tfrac{2^p}{9}(44q - 9C_q^{\times}) - \tfrac{10}{9}q(-1)^p + 6q \end{aligned} \quad (11)$$

$4q$-length DFT can be computed by

$$C_{4q}^{+} = 4C_q^{+} + 16q$$

$$C_{4q}^{\times} = 4C_q^{\times}$$

Using that and (8) we finally get the formula which shows the number of real arithmetic operations of $q \times 2^p$

$$\begin{aligned} C_N^{+} &= \tfrac{8}{3}pq2^p - \tfrac{2^p}{9}(16q - 9C_q^{+}) - \tfrac{2}{9}q(-1)^p + 2q = \\ &= \tfrac{8}{3}N\log_2\left(\tfrac{N}{q}\right) - N\left(\tfrac{16}{9} - \tfrac{1}{q}C_q^{+}\right) - \\ &- \tfrac{2}{9}q(-1)^{\log_2\left(\frac{N}{q}\right)} + 2q, \\ C_N^{\times} &= \tfrac{4}{3}pq2^p - \tfrac{2^p}{9}(38q - 9C_q^{\times}) + \tfrac{2}{9}q(-1)^p + 6q = \\ &= \tfrac{4}{3}N\log_2\left(\tfrac{N}{q}\right) - N\left(\tfrac{38}{9} - \tfrac{1}{q}C_q^{+}\right) + \\ &+ \tfrac{2}{9}q(-1)^{\log_2\left(\frac{N}{q}\right)} + 6q \end{aligned} \quad (12)$$

It is interesting to see that if $q = 1$ and therefore $C_1^{+} = 0$ and $C_1^{+} = 0$ from (12) we can get

$$\begin{aligned} C_N^{+} &= \tfrac{8}{3}N\log_2 N - \tfrac{16}{9}N - \tfrac{2}{9}(-1)^{\log_2 N} + 2, \\ C_N^{\times} &= \tfrac{4}{3}N\log_2 N - \tfrac{38}{9}N + \tfrac{2}{9}(-1)^{\log_2 N} + 6 \end{aligned} \quad (13)$$

(13) is the same as the number of real arithmetic operations count required by conventional split-radix algorithm. Doing some optimization from [7] we get following recurrent expressions for computing $8q$ length DFTs.

$$\begin{aligned} C_{8q}^{+} &= C_{4q}^{+} + 4C_q^{+} + 36q = 8C_q^{+} + 52q, \\ C_{8q}^{\times} &= C_{4q}^{\times} + 2C_q^{\times} + 2C_{sq}^{\times} = 6C_q^{\times} + 2C_{sq}^{\times} \end{aligned} \quad (14)$$

where by $C_{sq}^{\times}$ the number of arithmetic operations required by divided DFT [7] is denoted. Using (14) we get an improvement in the number of arithmetic operations

$$\begin{aligned} C_N^{+} &= \tfrac{8}{3}pq2^p - \tfrac{2^p}{9}(16q - 9C_q^{+}) - \tfrac{2}{9}q(-1)^p + 2q = \\ &= \tfrac{8}{3}N\log_2\left(\tfrac{N}{q}\right) - N\left(\tfrac{16}{9} - \tfrac{1}{q}C_q^{+}\right) - \\ &- \tfrac{2}{9}q(-1)^{\log_2\left(\frac{N}{q}\right)} + 2q, \\ C_N^{\times} &= \tfrac{4}{3}pq2^p - \tfrac{2^p}{18}(82q - 3(5C_q^{\times} + C_{sq}^{\times})) + \\ &+ \tfrac{2}{9}(7\,q + 3\,(C_q^{\times} - C_{sq}^{\times}))(-1)^p + 6q = \\ &= \tfrac{4}{3}N\log_2\left(\tfrac{N}{q}\right) - \tfrac{N}{18}\left(82 - \tfrac{3}{q}(5C_q^{\times} + C_{sq}^{\times})\right) + \\ &+ \tfrac{2}{9}(7q + 3(C_q^{\times} - C_{sq}^{\times}))(-1)^{\log_2\left(\frac{N}{q}\right)} + 6q \end{aligned} \quad (15)$$

3.1 Comparison of Arithmetic Complexities

The number of additions and multiplications required for computing DFT for complex input vector for various lengths is presented in Table 1. As an example a range from 256 to 2048 is chosen. Using conventional algorithm for 2^p we can only compute DFT of $256, 512, 1024, 2048$ sizes. If the size doesn't equal to these values we need to pad the input data up to next 2^p. The $q \times 2^p$ algorithm allows to cover the range $256 - 1024$ with 27 new points. This approach allows to significantly reduce the number of arithmetic operations. To find out the q for which the algorithm becomes the most efficient in terms of the number of arithmetic operations, first of all we cut the values of q for which $C_{N_1} > C_{N_2}$, but $N_1 < N_2$, where $C_N = C_N^+ + C_N^\times$. It is easy to see what only for $1, 3, 5, 9, 13, 15$ condition presented above is true. For getting more accurate results we compare the value of $E_N = \frac{C_N}{N}$. Finaly we get

$$E_N(9\times 2^p) < E_N(9\times 3^p) < E_N(9\times 15^p) < E_N(1\times 2^p) <$$

$$< E_N(5 \times 2^p) < E_N(15 \times 2^p)$$

These results are graphically illustrated in Figure 1.

Table 1: Number of arithmetic operations required by 2/4 split-raidx algorithm for DFT length 256-1024

N	q	p	Add.	Mul.	Count
256	1	8	5380	1284	6664
272	17	4	6832	1720	8552
288	9	5	6036	1196	7232
304	19	4	9200	1672	10872
320	5	6	6736	1880	8616
352	11	5	9468	2204	11672
360	45	3	7812	1140	8952
384	3	7	8028	2192	10220
416	13	5	10852	2372	13224
448	7	6	10992	2760	13752
480	15	5	10956	2112	13068
512	1	9	12292	3076	15368
544	17	5	15092	4052	19144
576	9	6	13584	3136	16720
608	19	5	19996	4028	24024
640	5	7	15172	4580	19752
704	11	6	20784	5288	26072
720	45	4	17424	3000	20424
768	3	8	18096	5396	23492
832	13	6	23888	5784	29672
896	7	7	24364	6668	31032
960	15	6	24432	5460	29892
1024	1	10	27652	7172	34824
1088	17	6	33040	9464	42504
1152	9	7	30228	7724	37952
1216	19	6	43184	9576	52760
1280	5	8	33744	10840	44584
1408	11	7	45308	12380	57688
1440	45	5	38628	7620	46248
1536	3	9	40284	12816	53100
1664	13	7	52196	13700	65896
1792	7	8	53488	15688	69176
1920	15	7	53964	13344	67308
2048	1	11	61444	16388	77832

4. 2/8 SPLIT-RADIX ALGORITHM

In case of $a = 4$ the algorithm becomes 2/8 split-radix algorithm.

$$X[8k+l] = \sum_{n=0}^{N/4-1} W_N^{nl} \left(\sum_{n=0}^{3} x'[n + i\frac{N}{8}] W_8^{il}\right) W_{N/8}^{nk} \tag{16}$$

The total number of real multiplications and real additions required by the algorithm are

$$\begin{aligned} C_N^+ &= C_{N/2}^+ + 4C_{N/8}^+ + \tfrac{11}{2}N - C_t^+ \\ C_N^\times &= C_{N/2}^\times + 4C_{N/8}^\times + \tfrac{5}{2}N - C_t^\times \end{aligned} \tag{17}$$

Below the number of arithmetic operations in logarithmic form are presented

$$\begin{aligned} C_N^+ = {} & \tfrac{11}{4}pq2^p - \tfrac{55}{16}q2^p - \tfrac{1}{8}2^p \left(C_t^+ - (2C_q^+ + C_{2q}^+ C_{4q}^+)\right) + \\ & + \tfrac{1}{4}C_t^+ + (-1)^p 2^{p/2} \\ & [7(12C_q^+ - 2(C_{2q}^+ + C_{4q}^+ + C_t^+) + 55q)\cos(\alpha) \\ & + \sqrt{7}(4C_q^+ - 2(11C_{2q}^+ - 5C_{4q}^+ - C_t^+) - 99q)\sin(\alpha)] \end{aligned}$$

$$\begin{aligned} C_N^\times = {} & \tfrac{5}{4}pq2^p - \tfrac{25}{16}q2^p - \tfrac{1}{8}2^p \left(C_t^\times - (2C_q^\times + C_{2q}^\times C_{4q}^\times)\right) + \\ & + \tfrac{1}{4}C_t^\times + (-1)^p 2^{p/2} \\ & [7(2(C_t^\times - 6C_q^\times + C_{2q}^\times + C_{4q}^\times) - 25q)\cos(\alpha) \\ & + \sqrt{7}(C_t^\times + 4C_q^\times - 22C_{2q}^\times + 5C_{4q}^\times - 45q)\sin(\alpha)] \end{aligned} \tag{18}$$

where with C_q,C_{2q} and C_{4q} denote number of arithmetic operations required for computation of q, $2q$ and $4q$ length DFTs respectively.

$$\alpha = p\arctan(\sqrt{7})$$

For computing $2q$ and $4q$ length DFT we can use methods described in previous section, which allow us to rewrite (18) as

$$\begin{aligned} C_N^+ = {} & \tfrac{11}{4}pq2^p - \tfrac{15}{16}q2^p - \tfrac{1}{8}2^p \left(C_t^+ - 8C_q^+\right) + \tfrac{1}{4}C_t^+ + \\ & + (-1)^p 2^{p/2} \quad [7(15q - 2C_t^+)\cos(\alpha) + \\ & + \sqrt{7}(2C_t^+ - 27q)\sin(\alpha)] \end{aligned}$$

$$\begin{aligned} C_N^\times = {} & \tfrac{5}{4}pq2^p - \tfrac{25}{16}q2^p - \tfrac{1}{8}2^p \left(C_t^\times - 8C_q^\times\right) + \tfrac{1}{4}C_t^\times + \\ & + (-1)^p 2^{p/2} \quad [7(25q - 2C_t^\times)\cos(\alpha) + \\ & + \sqrt{7}(2C_t^\times - 45q)\sin(\alpha)] \end{aligned} \tag{19}$$

5. CONCLUSION

In case of looking for a computationally efficient algorithm in terms of number of multiplications in general case we need to choose 2/8 split-radix FFT algorithm, because of coefficient of $N \log_2 N$ is a fewer. Efficiency of algorithms in terms of total number of arithmetic operations is discussed below.

The total number of arithmetic operations required by 2/4 split-radix algorithm can be computed using (12) and presented below

$$C_N(2/4) = 4pq2^p - 2^p(6q - C_q^+) + 8q \tag{20}$$

The total number of arithmetic operations required for computation 2/8 split-radix algorithm can be retrieved from (19)

$$C_N(2/8) = 4pq2^p - 2^p(\tfrac{5}{2}q - (\tfrac{1}{8}C_t - C_q)) + 8q+$$
$$+2(-1)^p 2^{p/2} \times [7(20q - C_t)\cos(\alpha)+$$
$$+\sqrt{7}(C_t - 36q)\sin(\alpha)] \quad (21)$$

For getting the computational efficient algorithm we need to subtract (21) from (20). For simplicity only coefficients for 2^p and $q \times 2^p$are included

$$C_N(2/4) - C_N(2/8) = (\tfrac{7}{2}q - \tfrac{1}{8}C_t)2^p < 0$$
$$C_t > 28q \quad (22)$$

In other words we can say that if trade-off C_t is greater than $28q$, then 2/4 split-radix algorithm becomes more efficient than 2/8 split-radix algorithm.

Table 2: Number of additions required by 2/8 split-raidx algorithm for DFT length 256-1024

N	q	p	Add.	Mul.	Count
256	1	8	5380	1284	6664
272	17	4	6832	1720	8552
288	9	5	6036	1196	7232
304	19	4	9200	1672	10872
320	5	6	6736	1880	8616
352	11	5	9468	2204	11672
384	3	7	8028	2192	10220
416	13	5	10852	2372	13224
448	7	6	10992	2760	13752
480	15	5	10956	2112	13068
512	1	9	12292	3076	15368
544	17	5	15092	4052	19144
576	9	6	13584	3136	16720
608	19	5	19996	4028	24024
640	5	7	15172	4580	19752
704	11	6	20784	5288	26072
768	3	8	18096	5396	23492
832	13	6	23888	5784	29672
896	7	7	24364	6668	31032
960	15	6	24432	5460	29892
1024	1	10	27652	7172	34824
1088	17	6	33040	9464	42504
1152	9	7	30228	7724	37952
1216	19	6	43184	9576	52760
1280	5	8	33744	10840	44584
1408	11	7	45308	12380	57688
1536	3	9	40284	12816	53100
1664	13	7	52196	13700	65896
1792	7	8	53488	15688	69176
1920	15	7	53964	13344	67308
2048	1	11	61444	16388	77832

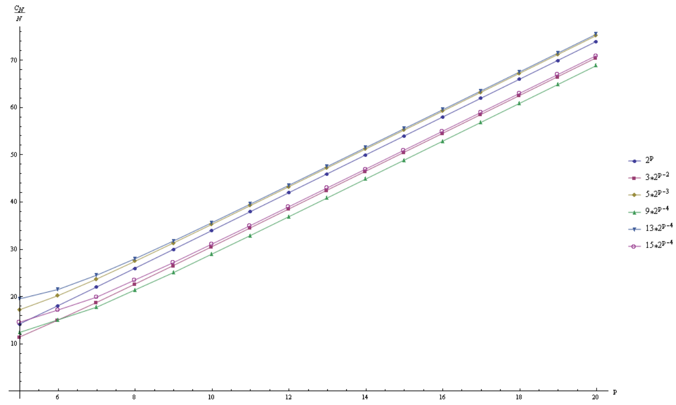

Figure 1: Comparisons on the total number of arithmetic operations for $2^p, 3 \times 2^{p-2}, 5 \times 2^{p-3}, 9 \times 2^{p-4}, 13 \times 2^{p-4}, 15 \times 2^{p-4}$ length DFTs (vertical axis presents the total number of arithmetic operations divided by length of DFT).

REFERENCES

[1] E. O. Brigham, The Fast Fourier applications. Englcwood Cliffs, NJ, Prentice-Hall, 1988.

[2] N. Ahmed, K. R. Rao, Orthogonal Transforms for Digital Signal Processing, 1973.

[3] J.K.Ersoy,Fourier-Related Transforms. Fast Algorithms and Applications. Englewood Cliffs, NJ, Prentice-Hall,1997.

[4] J. W. Cooley and J. W. Tukey, "An algorithm for the machine computation of the complex Fourier series," Math. Computation - 1965, pp. 297-301.

[5] R. Yavne, "An economical method for calculating the discrete Fourier transform," in Proc. AFIPS, vol. 33, 1968, pp. 115-125.

[6] M. Frigo and S. G. Johnson, "A modified split-radix FFT with fewer arithmetic operations, IEEE Trans. Signal Processing" - vol. 55, pp. 111-119, 2207.

[7] G. Bi and Y. Q. Chen, "Fast DFT algorithm for length $N = q \times 2^m$," IEEE Trans. Circuits and Systems II, vol. 45, no. 6, pp. 685 - 690, 1998.

[8] G. Bi, G. Li and X. Li, "A Unified Expression for Split-radix DFT Algorithms," pp. 323 - 326, 2010.

[9] P. Duhamel, "Implementation of split-radix FFT algorithms for complex, real, and real-symmetric data." IEEE Trans. Acoust., Speech, Signal Process., vol. 34, pp. 285 - 295, April, 1986.

[10] H. V. Sorensen, M. T. Heideman and C. S. Burrus, "On computing the split-radix FFT," IEEE Trans. Acoust., Speech, Signal Process., vol. 34,pp. 152 - 156, Feb. 1986.

[11] Petrovski I.G., Lectures on the theory of ordinary differential equations (in russian),1984

[12] S. Bouguezel, M. Omair and M. N. S. Swamy, "A new radix-2/8 FFT algorithm for length-$q \times 2^m$ DFTs," IEEE Trans. Circuits and Systems I, vol. 51, no. 1, pp. 1723-1732, 2004.

[13] S. Bouguezel, M. Omair and M. N. S. Swamy, "A general class of split-radix FFT algorithms for the computation of the DFT of length-2^m," IEEE Trans. Signal Processing vol. 55, no. 8, pp. 4127 - 4138, 2007.

[14] I. Selesnick, S. Burrus, "Programs for Prime Length FFTs," http://cnx.org/content/m18137/1.5/.

[15] M. Vetterli and P. Duhamel, "Split-radix algorithms for length- p^m DFT's," IEEE Trans. Acoust., Speech, Sig- nal Processing, vol. 37, pp.57 -64, 1989.