Efficient systolic designs for 1- and 2-dimensional DFT of general transform-lengths for high-speed wireless communication applications

Meher, Pramod Kumar; Patra, Jagdish Chandra; Vinod, Achutavarrier Prasad

2010

Meher, P. K., Patra, J. C., & Vinod, A. P. (2010). Efficient Systolic Designs for 1- and 2-Dimensional DFT of General Transform-Lengths for High-Speed Wireless Communication Applications. *Journal of Signal Processing Systems*, 60(1), 1-14.

<https://hdl.handle.net/10356/94362>

<https://doi.org/10.1007/s11265-008-0328-x>

© 2010 Springer. This is the author created version of a work that has been peer reviewed and accepted for publication by *Journal of Signal Processing Systems*, Springer. It incorporates referee's comments but changes resulting from the publishing process, such as copyediting, structural formatting, may not be reflected in this document. The published version is available at: [DOI: <http://dx.doi.org/10.1007/s11265-008-0328-x>].

Downloaded on 26 Apr 2023 20:13:00 SGT

Efficient Systolic Designs for 1- and 2-Dimensional DFT of General Transform-Lengths for High-Speed Wireless Communication Applications

Pramod K. Meher • Jagdish C. Patra • A. P. Vinod*

*School of Computer Engineering,
Nanyang Technological University, 50 Nanyang Avenue,
Singapore 639798, Singapore*

**e-mail: aspkmeher@ntu.edu.sg*

J. C. Patra e-mail: aspatra@ntu.edu.sg

A. P. Vinod e-mail: asvinod@ntu.edu.sg

ABSTRACT

In wireless communication, multiple receive-antennas are used with orthogonal frequency division multiplexing (OFDM) to improve the system capacity and performance. The discrete Fourier transform (DFT) plays an important part in such a system since the DFTs are required to be performed for the output of all those antennas separately. This paper presents area-time efficient systolic structures for one-dimensional (1-D) and two-dimensional (2-D) DFTs of general lengths. A low-complexity recursive algorithm based on Clenshaw's recurrence relation is formulated for the computation of 1-D DFT. The proposed algorithm is used further to derive a linear systolic array for the DFT. The concurrency of computation has been enhanced and complexity is minimized by the proposed algorithm where an N -point DFT is computed via four inner-products of real-valued data of length $\approx (N/2)$. The proposed 1-D structure offers significantly lower latency, twice the throughput, and involves nearly the same area-time complexity of the corresponding existing structures. The proposed algorithm for 1-D DFT is extended further to obtain a 2-D systolic structure for the 2-D DFT without involving any transposition operation.

Keywords

DFT • VLSI • Systolic array

1. INTRODUCTION

The discrete Fourier transform (DFT) has been widely used for the frequency-domain analysis and design of signals and systems in digital signal processing (DSP), image processing communication applications [1]. For example, in wireless communications, the use of multiple receive antennas with orthogonal frequency division multiplexing (OFDM) significantly improves the system capacity and performance [2]. Multi-carrier software defined radio (SDR) implementation of OFDM receiver is gaining popularity because it facilitates the creation of an adaptive OFDM transceiver system capable of

dynamically changing system parameters in order to maximize the channel capacity and minimize the power consumption. But, the complexity of such combination of multiple receive antennas with OFDM is very large because multiple discrete Fourier transforms (DFTs), (one for each receive antenna), are required to take full advantage of the space diversity. Even though DFT can be implemented using the fast Fourier transform (FFT), its complexity is still a major concern for the receiver implementation [3]. Besides, the processing of N communication channels requires the computation of N -point DFT, and FFT cannot be used if N is not a power of two [4]. Furthermore, a large number of DFT blocks demands more power, larger chip area and longer processing time for the signal processing. This is a disadvantage for power critical, size constrained and high-speed applications such as those in mobile wireless communications.

Several algorithms are, therefore, suggested by various researchers [5–7] for efficient computation of the DFT in general-purpose machines. The algorithms suggested for programmable-computers are, however, not well suited for very large-scale integration (VLSI) due to their global communication requirement. Moreover, fast computation of the DFT to meet the requirement of real-time applications is considered as a challenging task, and many algorithms and architectures are, thus, suggested for efficient implementation of the DFT in dedicated VLSI [8–11]. Amongst the existing VLSI systems, systolic architectures have been extensively popular owing not only to the simplicity of their design using repetitive identical processing elements (PE) having regular and local interconnections; but also for their potential of using high level of pipelining in small chip-area with low power consumption [12]. Efforts have, therefore, been made for efficient systolization of one-dimensional (1-D) and two-dimensional (2-D) DFT [13–27].

The radix-2 FFT-based systolic-like structures [13] have time-complexity $O(N)$ and involve $O(\log_2 N)$ multiply-accumulators along with $O(N + \log_2 N)$ delay elements for N -point transform, but they can be used only for transform-lengths $N = 2^p$. The radix-4 FFT-based structure of [14] offers still lower hardware-complexity over the radix-2 structures, but can be used for power-of-4 transform-lengths only. Besides, the FFT-based structures do not have the advantages of systolic designs due to lack of simplicity, uniformity and regularity of structure. Prime-factor approach [15–18] using linear systolic arrays offers attractive solutions with hardware- and time-complexity $O(N^{1/2})$ for composite transform-length $N = N_1 \times N_2$, where N_1 and N_2 are relatively prime and $N_1 \approx N_2$. The structures of [18–21] also offer reduced-hardware and high-throughput solutions by decomposition of the computation for implementation in linear systolic arrays for DFT of smaller transform-lengths. It is, therefore, important to reduce the hardware- and time-complexities of the linear systolic arrays for computing the DFT so as to use them as optimal building blocks for the structures based on various decomposition schemes. Several different linear systolic arrays are also reported in the literature [22–29] which differ mainly in terms of their area-complexity, time-complexity and I/O requirement.

Power-of-2 transform-length is not a natural choice of the applications. For example, the DFT filter bank is employed for extracting individual radio channels from the wideband input signal in a digital wireless communication receiver. The signal in this

case is split into M uniform-bandwidth subbands by the DFT filter bank, where M -point DFTs are required to be computed, M being not a power-of-two number very often. For implementing DFT through FFT algorithm, M -point input is padded with zeros to derive input of power-of-2 lengths. Similarly, in many applications, power-of-2 and power-of-4 lengths are chosen because most algorithms for software implementation are based on radix-2, radix-4 or split-radix decomposition. When the transform-length is constrained to be power-of-2 or power-of-4, the choice becomes very much limited particularly for large transform-lengths. As example, if we decide on power-of-2 lengths, then we cannot have a length in between 128 and 256, or between 256 and 512 or between 512 and 1024. The difference between two consecutive allowed lengths increases exponentially. But, we can design efficient hardware even if the transform-length is not a power-of-2 so that the system designer would have flexibility to choose the transform-length according to the natural choice of the applications on hand. An efficient algorithm and systolic design based on a low-complexity convolutional formulation is presented in [29] for computation of 1-D DFT, where transform-length is restricted to be even multiple of prime. Recently, we have proposed some novel and area-time efficient systolic designs for 1-D and 2-D DFT of general lengths [30–32]. In this paper, we extend those work further to derive novel reduced-complexity recursive formulation and fully-pipelined systolic architectures for 1-D and 2-D DFT of general transform-lengths.

Two-dimensional DFT is traditionally computed by a 3-stage approach. N number of N -point 1-D DFT of the columns of the input array are computed in the first stage to obtain an intermediate matrix to calculate the 2-D DFT of an array of size $(N \times N)$. The intermediate matrix is transposed in the second stage; and in the third stage, N number of N -point 1-D DFT of the columns of the transposed intermediate matrix are computed to obtain the desired 2-D DFT. The transposition operation involved in stage-two, is quite unattractive for VLSI. Not only it requires considerable hardware, but also introduces significant delay because it is not possible to perform the computation of stage-three unless the blocks of desired input are generated by the transposition unit. Lim and Swartzlander [17] have proposed a 2-D array for computing 2-D DFT by a two-stage approach without involving any transposition. The key idea used there is to combine two different types of semi-systolic arrays in an orthogonal way to process the two stages of computation in mutually orthogonal directions instead of using transposition of the intermediate matrix. The rows of the input, as well as, the multiplying coefficients to be used in the second stage are fed to this computing structure in an interleaved manner. Therefore, it involves additional controls for the interleaving, and additional memory for storing the coefficients. Besides, it requires $2N$ cycles to compute a complete set of 2-D DFT of size $(N \times N)$, and consequently it cannot support arrival rate of an input array of size $(N \times N)$ in less than $2N$ cycles. A different transposition-free architecture is suggested in [28] for high-throughput computation of the 2-D DFT in a fully-pipelined systolic VLSI, with flexible choice of implementation using either CORDIC circuits or memory-based multiplication units or hardwired-multiplier. In this paper, we aim at presenting a systolic design for transposition-free flexible computation of 2-D DFT like that of [28] with lower area-time complexity than the latter. Each of the proposed design performs multiplications with fixed constants, such that they can be implemented by any of the constant multiplication methods or hardwired multiplier like that of [28].

The rest of the paper is organized as follows. The derivation of proposed reduced-complexity algorithm and subsequent recursive formulation for 1-D DFT are discussed in Section 2. Proposed linear array structure for 1-D transform is described in Section 3. Section 4 deals with the development of low-complexity algorithm for 2-D DFT. The proposed structure for 2-D DFT is described in Section 5. Hardware- and time-complexities of the structures are discussed and compared with those of the existing structures in Section 6. Conclusions and scope for future work are presented in Section 7.

2. FORMULATION OF PROPOSED ALGORITHM FOR ONE-DIMENSIONAL DFT

We derive here a reduced-complexity algorithm for computation of 1-D DFT, and obtain an efficient recursive formulation based on Clenshaw's recurrence relation [33] for the computation of the matrix-vector products by a systolic hardware.

2.1 Formulation of a Low-Complexity Algorithm for 1-D DFT

The DFT of a sequence $\{x(n) | 0 \leq n \leq N-1\}$ is defined as

$$X(k) = \sum_{n=0}^{N-1} x(n) \left[\cos \frac{2\pi kn}{N} - j \sin \frac{2\pi kn}{N} \right], \quad (1)$$

for $k = 0, 1, \dots, N-1$.

Supposing N to be an even number, the DFT of Eq. 1 can be expressed by two summation terms as given by the following equations.

$$X(k) = A(k) + B(k) \quad (2a)$$

where

$$A(k) = \sum_{n=0}^{M-1} a(n) \left[\cos \frac{\pi k(2n+1)}{M} - j \sin \frac{\pi k(2n+1)}{M} \right] \quad (2b)$$

$$B(k) = \sum_{n=0}^{M-1} b(n) \left[\cos \frac{2\pi kn}{M} - j \sin \frac{2\pi kn}{M} \right], \quad (2c)$$

for $M = N/2$, $a(n) = x(2n+1)$ and $b(n) = x(2n)$, for $n = 0, 1, \dots, M-1$, and for $k = 0, 1, \dots, N-1$. (Note that we can also have a decomposition similar to Eq. 2 for odd values of N , where the upper limit of summation index would be different.)

$A(k)$ and $B(k)$ in Eqs. 2b and 2c, respectively, can be broken into even components, $A_e(k)$ and $B_e(k)$; and odd components $A_o(k)$ and $B_o(k)$ as follows.

$$A(k) = A_e(k) - jA_o(k) \quad (3a)$$

$$B(k) = B_e(k) - jB_o(k), \quad (3b)$$

where

$$A_e(k) = \sum_{n=0}^{M-1} a(n) \left[\cos \frac{\pi k(2n+1)}{M} \right] \quad (3c)$$

$$A_o(k) = \sum_{n=0}^{M-1} a(n) \left[\sin \frac{\pi k(2n+1)}{M} \right] \quad (3d)$$

$$B_e(k) = \sum_{n=0}^{M-1} b(n) \left[\cos \frac{2\pi kn}{M} \right] \quad (3e)$$

$$B_o(k) = \sum_{n=0}^{M-1} b(n) \left[\sin \frac{2\pi kn}{M} \right], \quad (3f)$$

for $k = 0, 1, \dots, N-1$.

Using symmetry properties of sine and cosine functions, from Eqs.3c–3f, it can be found that

$$A_e(M-k) = -A_e(k) \quad (4a)$$

$$A_o(M-k) = A_o(k) \quad (4b)$$

$$B_e(M-k) = B_e(k) \quad (4c)$$

$$B_o(M-k) = -B_o(k), \quad (4d)$$

for $k = 1, 2, \dots, K$, where $K = M/2 - 1$, and

$$A_e(N-k) = A_e(k) \quad (4e)$$

$$A_o(N-k) = -A_o(k) \quad (4f)$$

$$B_e(N-k) = B_e(k) \quad (4g)$$

$$B_o(N-k) = -B_o(k), \quad (4h)$$

for $k = 1, 2, \dots, M-1$.

For $k = 0, M/2, M$ and $3M/2$, directly from Eq. 2, one can find that

$$X(0) = \sum_{n=0}^{M-1} a(n) + \sum_{n=0}^{M-1} b(n) \quad (5a)$$

$$X(M/2) = \sum_{n=0}^{M-1} (-1)^n b(n) - j \sum_{n=0}^{M-1} (-1)^n a(n) \quad (5b)$$

$$X(M) = \sum_{n=0}^{M-1} b(n) - \sum_{n=0}^{M-1} a(n) \quad (5c)$$

$$X(3M/2) = \sum_{n=0}^{M-1} (-1)^n b(n) + j \sum_{n=0}^{M-1} (-1)^n a(n). \quad (5d)$$

Using Eqs.2, 3 and 4a–4d, rest of the first half of the DFT components can be obtained as

$$X(k) = (A_e(k) + B_e(k)) - j(A_o(k) + B_o(k)) \quad (6a)$$

$$X(M - k) = (B_e(k) - A_e(k)) - j(A_o(k) - B_o(k)), \quad (6b)$$

for $k = 1, 2, \dots, K$.

Similarly, using Eqs. 4e-4h, rest of the second half of the DFT components can also be obtained as

$$X(N - k) = (A_e(k) + B_e(k)) + j(A_o(k) + B_o(k)) \quad (7a)$$

$$X(M + k) = (B_e(k) - A_e(k)) + j(A_o(k) - B_o(k)), \quad (7b)$$

for $k = 1, 2, \dots, K$.

An N -point DFT may thus be computed from $A_e(k)$, $A_o(k)$, $B_e(k)$ and $B_o(k)$ for $k = 1, 2, \dots, (N/4) - 1$ of Eq. 3, along with $X(0)$, $X(M/2)$, $X(M)$ and $X(3M/2)$ of Eq. 5. N -point 1-D DFT can, therefore, be computed from four matrix-vector products, where each matrix is of size $K \times M$ of real-valued data and each vector is of length M of complex-valued data, which involves nearly N^2 real-valued multiplications and additions. A straight-forward implementation of N -point DFT would, however, involve N^2 complex-multiplications and nearly the same number of complex-additions.

2.2 Recursive Formulation for 1-D DFT

Let us consider a sum-of-products given by

$$S(k) = \sum_{n=L}^H C_n \cdot F_n(k) \quad (8)$$

where L and H are nonnegative integers, and for some functions $\alpha(n, k)$ and $\beta(n, k)$, $F_n(k)$ obeys the recurrence relation:

$$F_{n+1}(k) = \alpha(n, k) \cdot F_n(k) + \beta(n, k) \cdot F_{n-1}(k). \quad (9)$$

Using Clenshaw's recurrence formula [33], $S(k)$ can be recursively computed by

$$S(k) = C_H \cdot F_H(k) - \beta(H, k) \cdot F_{H-1}(k) \phi_{H-1}^k - F_H \phi_{H-2}^k \quad (10)$$

where ϕ_n^k can be determined by the following recursive expression

$$\phi_n^k = \frac{1}{\beta(n+1, k)} \left[\phi_{n-2}^k - \alpha(n, k) \cdot \phi_{n-1}^k - C_n \right], \quad (11)$$

for $L \leq n \leq H$ and $\phi_{L-1}^k = \phi_{L-2}^k = \dots = 0$.

Substituting H for n in Eq. 11, one can find

$$C_H = \phi_{H-2}^k - \alpha(H, k) \cdot \phi_{H-1}^k - \beta(H+1, k) \cdot \phi_H^k. \quad (12)$$

Replacing the C_H in Eq. 10 by that of Eq. 12, one can have

$$S(k) = -\beta(H+1, k) \cdot F_H(k) \cdot \phi_H^k - F_{H+1}(k) \cdot \phi_{H-1}^k. \quad (13)$$

The sum-of-products of the form given by Eq. 8, can be computed according to Eq. 13 through $(H - L + 1)$ recursions of Eq. 11, which can be used for efficient computation of the four matrix-vector-products to find $A_e(k)$, $A_o(k)$, $B_e(k)$ and $B_o(k)$ for $k=1, 2, \dots, (N/4)-1$ of Eqs.3c-3d.

2.2.1 Computation of $A_e(k)$ for $k = 1, 2, \dots, K$

To obtain a recurrence relation for computation of $A_e(k)$, we can compare Eq. 3c with Eq. 8 and find

$$C_n = a(n) \quad (14a)$$

and

$$F_n(k) = \cos \left[\frac{1}{2}(2n+1)\theta_k \right] \text{ for } \theta_k = \left[\frac{2\pi k}{M} \right] \quad (14b)$$

where, $F_n(k)$ can be found to satisfy a recurrence relation

$$F_{n+1}(k) = 2 \cos(\theta_k) \cdot F_n(k) - F_{n-1}(k). \quad (14c)$$

Comparing Eq. 14c with Eq. 9, we can have $\alpha(n, k) = 2 \cos(\theta_k)$ and $\beta(n, k) = -1$. Using these values of $\alpha(n, k)$ and $\beta(n, k)$, Eqs. 11 and 13 can, respectively, be written as

$$S(k) = F_H(k) \cdot \phi_H^k - F_{H+1}(k) \cdot \phi_{H-1}^k, \quad (15)$$

for

$$\phi_n^k = 2 \cos(\theta_k) \cdot \phi_{n-1}^k - \phi_{n-2}^k + a(n). \quad (16)$$

Using the value $F_n(k)$ of Eq. 14b, for $H = M - 1$ and $S(k) = A_e(k)$, Eq. 15 can be written as

$$A_e(k) = \cos \frac{\pi k}{M} \cdot (\phi_{M-1}^k - \phi_{M-2}^k). \quad (17)$$

2.2.2 Computation of $A_o(k)$, $B_e(k)$ and $B_o(k)$ for $k = 1, 2, \dots, K$

To obtain a recursive formulation for computation of $A_o(k)$, similarly, we can compare Eq. 3d with Eq. 8, and find $C_n = a(n)$, $F_n(k) = \sin \left[\frac{1}{2}(2n+1)\theta_k \right]$, for $\theta_k = \left[\frac{2\pi k}{M} \right]$. It can also be found that $F_n(k)$ satisfies the recurrence relations of Eqs. 14c and 16. Substituting the value of $F_n(k)$ on Eq. 15, for $H = M - 1$ and $S(k) = A_o(k)$, one can have

$$A_o(k) = -\sin \frac{\pi k}{M} \cdot (\phi_{M-1}^k + \phi_{M-2}^k). \quad (18)$$

Similarly, one can also obtain $B_e(k)$ and $B_o(k)$, of Eqs. 3e and 3f, respectively, as

$$B_e(k) = \cos \frac{2\pi k}{M} \cdot \phi_{M-1}^k - \phi_{M-2}^k \quad (19)$$

and

$$B_o(k) = -\sin \frac{2\pi k}{M} \cdot \phi_{M-1}^k \quad (20)$$

where

$$\varphi_n^k = 2 \cos(\theta_k) \cdot \varphi_{n-1}^k - \varphi_{n-2}^k + b(n). \quad (21)$$

From Eqs.17 and 18, it can be noticed that both $A_e(k)$ and $A_o(k)$ can be computed from the same values of ϕ_{M-1}^k and ϕ_{M-2}^k for $1 \leq k \leq K$. Similarly, from Eqs.19 and 20, it can be seen that $B_e(k)$ can be obtained from the values of φ_{M-1}^k and φ_{M-2}^k for $1 \leq k \leq K$, and the same values of φ_{M-1}^k can be used to compute $B_o(k)$. The core computation involved in an N -point DFT thus gets reduced to computation of $(\phi_{M-1}^k, \phi_{M-2}^k)$ and $(\varphi_{M-1}^k, \varphi_{M-2}^k)$ for $1 \leq k \leq K$ through $M-1$ number of recursions of Eqs. 16 and 21, respectively.

3. PROPOSED STRUCTURE FOR ONE-DIMENSIONAL DFT

The dependence graph (DG) for the computation of $(\phi_{M-1}^k, \phi_{M-2}^k)$ for $1 \leq k \leq K$ is shown in Fig. 1a and that of $(\varphi_{M-1}^k, \varphi_{M-2}^k)$ is shown in Fig. 1b. As shown in the figures, each of these DGs consists of KM number of nodes arranged in M rows and K columns. Function of the nodes of the DGs is depicted in Fig. 1c. Each of the nodes performs a multiplication of the first vertical input with a fixed constant, subtracts the second vertical input from the product value, and finally adds its left input with the result of subtraction to deliver the first vertical output. The first vertical input is transferred to the node below as the second vertical output. The left input is propagated across a row of nodes without modifications.

Since, both the DGs are identical in structure and function, they can be merged together and can be projected along $[0 \ 1]^T$ with a default schedule [9] to obtain a linear systolic array consisting of $K = (N/4 - 1)$ PEs, shown in Fig. 2a as part of the proposed structure for 1-D DFT. The function of the PEs are depicted in Fig. 2b. Each PE performs a pair of multiplications of its complex-valued input with a real-valued multiplying coefficient followed by two pairs of complex-additions during a cycle period to implement a pair of recursions of Eqs.16 and 21. The duration of cycle period of a PE depends on the number of multipliers and adders provided to the PEs. If each PE is provided with four multipliers and four adders then the duration of a cycle period would be $T_1 = (T_M + T_A)$, where T_M and T_A are the times required to perform a multiplication and an addition, respectively. Instead of that, if each PE is provided with two multipliers and two adders, then the duration of a cycle period would be $T_2 = 2(T_M + T_A)$. The input samples are propagated across the PEs of the systolic array and finally fed to an adder cell [shown in Fig. 2c]. During every cycle period, the adder cell performs four additions to compute $X(0)$, $X(M/2)$, $X(M)$ and $X(3M/2)$ according to Eq. 5. To obtain the DFT components, it is required to compute $A_e(k)$, $A_o(k)$, $B_e(k)$ and $B_o(k)$ for $k = 1, 2, \dots, K$ according to Eqs.17–20 followed by the additions of Eqs.6 and 7. The output cell of the structure (Fig. 2d) performs those operations accordingly to find the 1-D DFT components. An array of $(K - 1)$ latch cells is used to flush out the intermediate results $\phi_M^k, \phi_{M-1}^k, \varphi_M^k, \varphi_{M-1}^k$ to be processed finally in the output cell. Function of the latch cells is depicted in Fig. 1e. The output from the adder cell and the first four output from the output cell are obtained $(3N/4 - 1)$ cycles after the first pair of input values enter the structure at the left most PE. The

proposed structure receives a set of N input samples in every $N/2$ cycles and produces N DFT components in every $N/2$ cycles.

4. ALGORITHM FORMULATION FOR 2-D DFT

The 2-D DFT of an array $[x(m, n)]$ of size $(N \times N)$ is defined as

$$X(k, l) = \sum_{m=0}^{N-1} \sum_{n=0}^{N-1} W_N^{km} \cdot W_N^{ln} \cdot x(m, n), \quad (22)$$

for $k, m, l, n = 0, 1, \dots, N - 1$, where

$$W_N^{ab} = e^{-i2\pi ab/N}. \quad (23)$$

The 2-D DFT of Eq. 22 can be decomposed into two stages of computation as

$$X(k, l) = \sum_{m=0}^{N-1} W_N^{km} \cdot V(m, l), \quad 0 \leq k, l \leq N - 1 \quad (24)$$

where

$$V(m, l) = \sum_{n=0}^{N-1} W_N^{ln} \cdot x(m, n), \quad 0 \leq m, l \leq N - 1. \quad (25)$$

The 2-D DFT of size $(N \times N)$, given by Eqs.24 and 25, can be computed in the following two stages:

Stage-1 Compute N -point 1-D DFT of each row of the input matrix to obtain the intermediate matrix $[V(m, l)]$ of size $(N \times N)$.

Stage-2 Compute N -point 1-D DFT of each row of the transposed form of intermediate matrix $[V(m, l)]$ to find the desired 2-D DFT coefficients.

To avoid the transposition operation, the intermediate matrix should be made available to the structure in column-wise manner for the computation of stage-two. Keeping this in view, we derive here a low-complexity formulation for concurrent computation of stage-one, which would perform the row-DFT processing to generate the columns of intermediate output, such that the DFTs of those columns of intermediate output can be implemented efficiently, without further transposition, by the 1-D DFT modules derived in Sections 2 and 3.

The 1-D DFT of the rows of input matrix given by Eq. 25 may also be expressed as:

$$\begin{aligned}
V(m, l) = & \left(x(m, 0) + (-1)^l x(m, M) \right) \\
& + \sum_{n=1}^{M-1} \left(W_N^{ln} \cdot x(m, n) + W_N^{-ln} \cdot x(m, N - n) \right)
\end{aligned} \tag{26a}$$

and

$$\begin{aligned}
V(m, N-l) = & \left(x(m, 0) + (-1)^l x(m, M) \right) \\
& + \sum_{n=1}^{M-1} \left(W_N^{-ln} \cdot x(m, n) + W_N^{ln} \cdot x(m, N - n) \right),
\end{aligned} \tag{26b}$$

for $l=1, \dots, M-1$,

$$\begin{aligned}
V(m, 0) = & \left(x(m, 0) + x(m, M) \right) \\
& + \sum_{n=1}^{M-1} \left(x(m, n) + x(m, N - n) \right)
\end{aligned} \tag{26c}$$

and

$$\begin{aligned}
V(m, M) = & \left(x(m, 0) + x(m, M) \right) \\
& + \sum_{n=1}^{M-1} (-1)^n \left(x(m, n) + x(n, N - n) \right)
\end{aligned} \tag{26d}$$

where $m = 0, 1, \dots, N - 1$. N and M are assumed to be even numbers, but it would also be possible to suitably modify Eq. 26 for odd values of N and M .

Using complex conjugate behaviour of the kernel coefficients, from Eqs.26a and 26b, we can have

$$\begin{aligned}
V(m, l) = & \left(x(m, 0) + (-1)^l x(m, M) \right) \\
& + \left(V_1(m, l) - V_2(m, l) \right) \\
& + j \left(V_3(m, l) + V_4(m, l) \right)
\end{aligned} \tag{27a}$$

and

$$\begin{aligned}
V(m, N-l) = & \left(x(m, 0) + (-1)^l x(m, M) \right) \\
& + \left(V_1(m, l) + V_2(m, l) \right) \\
& + j \left(V_3(m, l) - V_4(m, l) \right) \quad (27b)
\end{aligned}$$

where

$$V_1(m, l) = \sum_{n=1}^{M-1} \text{Re}[s(m, n)] \cdot \text{Re}[W_N^{ln}] \quad (28a)$$

$$V_2(m, l) = \sum_{n=1}^{M-1} \text{Im}[d(m, n)] \cdot \text{Im}[W_N^{ln}] \quad (28b)$$

$$V_3(m, l) = \sum_{n=1}^{M-1} \text{Im}[s(m, n)] \cdot \text{Re}[W_N^{ln}] \quad (28c)$$

$$V_4(m, l) = \sum_{n=1}^{M-1} \text{Re}[d(m, n)] \cdot \text{Im}[W_N^{ln}] \quad (28d)$$

$$s(m, n) = (x(m, n) + x(m, N-n)) \quad (28e)$$

$$d(m, n) = (x(m, n) - x(m, N-n)), \quad (28f)$$

for $l, n = 1, \dots, M-1, m = 0, 1, \dots, N-1$,

$$V(m, 0) = (x(m, 0) + x(m, M)) + \sum_{n=1}^{M-1} s(m, n), \quad (28g)$$

and

$$\begin{aligned}
V(m, M) = & (x(m, 0) + x(m, M)) \\
& + \sum_{n=1}^{M-1} (-1)^n s(m, n), \quad (28h)
\end{aligned}$$

for $m = 0, 1, \dots, N-1$. $\text{Re}[\cdot]$ and $\text{Im}[\cdot]$, respectively, denote the real-part and the imaginary-part of a complex number, $[\cdot]$.

The core computation of the stage-one can be computed according to Eqs.28a–28d through $4N(M-1)$ number of $(M-1)$ -point real-valued inner-products, which amounts

to nearly N^3 real-valued multiplications and nearly the same number of real-additions. Note that a straight-forward implementation of stage-one of 2-DFT of size $N \times N$ involves N^3 complex-valued multiplications and nearly the same number of complex-additions.

5. PROPOSED STRUCTURE FOR 2-D DFT

The dependence graph DG-1 for computation of $V_1(m, l)$ and $V_3(m, l)$ for $l = 1, 2, \dots, M - 1$ and for a given value of m is shown in Fig. 3a and the dependence graph DG-2 of $V_2(m, l)$ and $V_4(m, l)$ is shown in Fig. 3b. As shown in the figures, each of these DGs consists of $(M - 1) \times (M - 1)$ number of nodes arranged in $M - 1$ rows and equal number of columns. Function of the nodes of both the DGs is depicted in Fig. 3c. Each of the nodes performs a pair of multiplications of the real and imaginary parts of the vertical input with a fixed constant and adds the pair of input available from its left to deliver a pair of output. The vertical input is propagated across a column of nodes without modifications. Since, both the DGs are identical in structure and function, they can be merged together and each of the nodes of the merged DG can be assigned to a PE to obtain a 2-D systolic array consisting of $(M - 1) \times (M - 1)$ number of PEs arranged in $(M - 1)$ rows and equal number of columns as shown in Fig. 4a. Function of the PEs is depicted in Fig. 4b.

A pair of input values from a row of the matrices $[s(m, n)]$ and $[d(m, n)]$ are fed to the PEs on the top row of the structure such that the input to a PE is staggered by one time-step with respect to the PE on its left to maintain the data-dependency requirement. For conventional multiplier-accumulator-based implementation, each PE stores a pair of constants, and during each cycle period, it performs two pairs of real-multiplications of the real and the imaginary parts of the pair of complex-valued vertical input with its stored pair of constants. It adds the four product values with the four real-valued input available from its left and delivers four real-valued output to its right. Rows of the pair of input matrices $[s(m, n)]$ and $[d(m, n)]$ may be processed one after the next to generate four matrices $[V_1(m, l)]$, $[V_2(m, l)]$, $[V_3(m, l)]$ and $[V_4(m, l)]$ of size $N \times (M - 1)$.

The structure for computation of stage-1 is shown in Fig. 5. The major part of this structure is constituted by the 2-D systolic array of $(M - 1) \times (M - 1)$ PEs for computation of intermediate sub-matrices $[V_1(m, l)]$, $[V_2(m, l)]$, $[V_3(m, l)]$ and $[V_4(m, l)]$ (described in Fig. 4). In addition to this, the structure consists of $(M - 1)$ Adder Cell-1 (AC-1), one Adder Cell-2 (AC-2) and $(M - 1)$ Adder Cell-3 (AC-3). The functions of AC-1, AC-2 and AC-3 are depicted in Fig. 5b. During each cycle period, each of the adder-cells AC-1 receives a pair of input values from the top such that the i -th AC-1 receives the i -th and $(N - i)$ -th value of the rows of input matrix $[x(m, n)]$ such that the input pair to the AC-1s are staggered by one cycle period with respect to the AC-1 on its left. AC-1 performs the add/subtract operations of Eqs.28e and 28f to generate the input values $s(m, n)$ and $d(m, n)$ to be processed by the 2-D systolic array. AC-2, similarly, receives a pair of input values $x(m, 0)$ and $x(m, M)$ from the top and performs the add/subtract operations of Eqs.28g and 28h for computing the DC components $V(m, 0)$ and middle component $V(m, M)$. Each of the adder-cells 'AC-3' performs the eight

additions of Eq. 27 during a cycle period in three pipelined stages. AC-2 yields a pair of components $N/2$ cycles after the first input arrives at the structure. Each of the $(N/2 - 1)$ AC3s, one after the other, during each of the next $(N/2 - 1)$ cycles start yielding a pair of output. Once the pipeline is filled in completely in the first $(N - 1)$ cycles, the structure yields N output in every cycle.

The hardware involved in a PE and the duration of cycle period depend mostly on how the four multiplications are implemented in the PEs. If each PE is provided with four multipliers and four adders, then the duration of a cycle period will be $T_1 = (T_A + T_M)$. The complete structure for 2-D DFT is shown in Fig. 6. It consists of the structure for the computation of stage-one followed by N number of 1-D systolic array (described in Sections 2 and 3). The output of the structure of stage-one, which is generated in column order. The each PE of the proposed structure for stage-one delivers one output per cycle period of duration $T_1 = (T_A + T_M)$, while each of the 1-D DFT structures of stage-two takes a pair of consecutive values of an intermediate output sequence for each recursion in its PEs. The the output of stage-one is thus transferred to the 1-D arrays of stage-two through a 1–2 lines multiplexer where the output line is alternated in a period of duration of T_1 . To match the input-output rates of both the stages of computation, and to have full hardware utilization, each PE of the 1-D DFT arrays should have two multipliers and two adders so as to perform each recursion in a cycle period of duration $T_2 = 2(T_A + T_M)$.

6. HARDWARE- AND TIME-COMPLEXITIES

We discuss here the hardware- and time-complexities of the proposed structures for 1-D and 2-D DFTs, and compare them with those of the existing systolic structures.

6.1 Complexity Considerations of 1-D DFT

For computation of the 1-D DFT of length N , the proposed structure, in general, will contain $(N/4 - 1)$ PEs along with an adder cell and one output cell. During each cycle period, each PE performs four real-multiplications and eight real-additions. To have a cycle period of duration $T_1 = (T_M + T_A)$, each PE is required to have four multipliers and four adders, the adder cell is required to have eight adders, and each output cell is required to have eight multipliers and eight adders. The input samples are fed in pair to the left most PE of the structure and propagated across the array in successive cycles. The first set of output of the structure is obtained after a latency of $(3N/4 - 1)$ cycles after the first pair of input values enter the structure at the left most PE and produces N DFT components in every $N/2$ cycles after the latency period. The hardware- and time-complexities of the proposed structure are estimated accordingly and listed in Table 1 along with those of the existing structures [22–25]. The orders of area-complexities of the structures are calculated here from the number of multipliers, since most area of a structure is contributed by the multipliers only. It is found that the proposed structure involves nearly the same number of multipliers and offers nearly half the average computation-time (ACT),¹ and half the area-time complexity compared with the best of the existing structures.

6.2 Complexity Considerations of 2-D DFT

Stage-one of the proposed structure for 2-D DFT of size $(N \times N)$ involves a 2-D systolic array of $(N/2 - 1)^2$ number of PEs along with $(N/2 - 1)$ AC-1, one AC-2 and $(N/2 - 1)$ AC-3. Each PE consists of four multipliers and four adders to perform four multiplications and four additions during a cycle period $T_1 = (T_M + T_A)$. Each AC-1 and AC-2 require four adders, while AC-3 requires eight adders. The structure of stage-one thus involves $(N - 2)^2$ multipliers and $(N^2 + 2N + 4)$ adders. Stage-two of the proposed 2-D DFT structure involves N number of 1-D DFT arrays with cycle period $T_2 = 2(T_M + T_A)$, where each of the $(N/4 - 1)$ of its PEs is required to have two multipliers and two adders, each adder cell is required to have four adders and each output cell is required to have four multipliers and four adders. The structure of stage-two thus involves $N(N + 4)/2$ multipliers and $N(N + 12)/2$ adders. The first set of intermediate output are available from the structure of stage-one, $N/2$ cycles after the first input arrives at the structure.

The pipeline is filled in completely in the first $(N - 1)$ cycles, and the structure yields N output in every cycle thereafter. The first set of output of stage-two of the structure is obtained after a latency of $(3N/4 - 1)$ cycles of duration $T_2 = 2(T_M + T_A)$ after the first pair of input values enter stage to at the left most PE of the top-row, and produces a complete set of N^2 components of 2-D DFT in every $N/2$ cycles of duration T_2 after the pipeline is filled in completely in the next $(N/2 - 1)$ cycles. The proposed 2-D structure, therefore, would have a latency of $2(N - 2)$ cycles and average computation of time of N cycles of duration $T_1 = (T_M + T_A)$. The hardware- and time-complexities of the proposed structure are estimated accordingly and listed in Table 2 along with those of the existing structures [22–25]. The orders of area-complexities of the structures are calculated here from the number of multipliers, since most area of a structure is contributed by the multipliers only. It is found that the proposed structure provides the same ACT, but involves nearly $(3/4)$ -th the number of multipliers and adders, involves nearly the same latency with 75% of the area-time complexity compared with the best of the corresponding existing structures.

7. CONCLUSION

A reduced-complexity recursive solution based on Clenshaw's recurrence relation is derived for concurrent of computation of the 1-D DFT. The proposed computational algorithm is used further to obtain a simple, regular and locally connected fully pipelined linear array architecture for systolic implementation of the DFT. It is shown that the proposed structure requires nearly the same number of multipliers and adders and offers nearly twice the throughput and involves nearly half the area-time complexity as that of the best of the corresponding existing DFT structures.

A simple 2-D systolic architecture is also derived for highly concurrent pipelined implementation of the 2D-DFT. Stage-one of the proposed 2-D DFT structure performs the row DFT processing and generates the elements of the columns of intermediate matrix such that the output of stage-one is processed by 1-D DFT arrays directly to have transposition-free computation of 2-D DFT. It is found that the proposed structure

involves nearly (3/4)-th the number of multipliers and adders and involves nearly the same latency and the same ACT with 75% of the area-time complexity compared with the best of the existing 2-D DFT structures [30]. The proposed 2-D DFT structure can also be used for computation of long-length 1-D DFT by prime-factor approach for composite transform-length $N = N_1 \times N_2$, where N_1 and N_2 are relatively prime.

Since the multiplying coefficient used by each of the PEs is fixed, the multiplications may be implemented by any of the constant multiplication method, e.g., CORDIC multiplier, ROM-based multiplier, adder-based multiplier using CSD representation or a hardwired multiplier. Unlike some of the existing structures, the proposed one can be used for the DFT of any transform-length and does not involve tag-bit control. It would, therefore, have potential scope to be used in real-time signal processing and communication applications.

8. REFERENCES

- [1] Bracewell, R. N. (2000). *The Fourier transform and its applications* (3rd ed.). Boston: McGraw-Hill.
- [2] Wong, K. K., Cheng, R. S. K., Letaief, K. B., & Murch, R. D. (2001). Adaptive antennas at the mobile and base stations in an OFDM/TDMA system. *IEEE Transactions on Communications*, 49(1), 195–206.
- [3] Grass, E., et al. (2001). On the single-chip implementation of a hiperlan/2 and IEEE 802.11a capable modem. *IEEE Personal Communications*, 8(6), 48–57.
- [4] Mahesh, R., & Vinod, A. P. (2008). Coefficient decimation approach for realizing reconfigurable finite impulse response filters. In *Proc. IEEE international symposium on circuits & systems, ISCAS '08* (pp. 81–84), May.
- [5] Duhamel, P., & Vetterli, M. (1990). Fast Fourier transforms: A tutorial review and a state of the art. *IEEE Transactions on Signal Processing*, 19, 259–299.
- [6] Sundararajan, D., Ahmad, M. O., & Swamy, M. N. S. (1998). Vector computation of the discrete Fourier transform. *IEEE Transactions on Circuits and Systems. II: Analog and Digital Signal Processing*, 45(4), 449–461.
- [7] Silverman, H. (1977). An introduction to programming the Winograd Fourier transform algorithm (WFTA). *IEEE Transactions on Acoustics, Speech, and Signal Processing*, 25(2), 152–165.
- [8] Swartzlander, Jr., E. E. (1986). *VLSI signal processing systems*. Boston: Kluwer.
- [9] Kung, S. Y. (1988). *VLSI array processors*. Englewood Cliffs: Prentice Hall.
- [10] Guo, J.-I., Liu, C.-M., & Jen, C.-W. (1992). The efficient memory-based VLSI array design for DFT and DCT. *IEEE Transactions on Circuits and Systems. II: Analog and Digital Signal Processing*, 39(10), 723–733.
- [11] Canaris, J. (1993). A VLSI architecture for the real time computation of discrete trigonometric transforms. *Journal of VLSI Signal Processing*, 5, 95–104.
- [12] Kung, H. T. (1982). Why systolic architectures? *IEEE Computer*, 15, 37–45.
- [13] Chang, C. H., Wang, C. L., & Chang, Y. T. (2000). Efficient VLSI architectures for fast computation of the discrete Fourier transform and its inverse. *IEEE Transactions on Signal Processing*, 48(11), 3206–3216.
- [14] Hsiao, S. F., & Shiue, W. R. (2000). Low-cost unified architectures for the computation of discrete trigonometric transforms. In *Proc. IEEE international conf. on acoust. speech. and signal process (ICASSP'2000)* (Vol. 6, pp. 3299–3302), June.
- [15] Jones, K. J. (1990). High-throughput, reduced-hardware systolic solution to prime factor discrete Fourier transform algorithm. *IEE Proceedings E. Computers and Digital Techniques*, 137(3), 191–196.
- [16] Sediljom, S. G. (1994). A new systolic architecture for pipeline prime factor DFT-algorithm. In *Proc. fourth great lakes symp on design automation of high performance VLSI systems* (pp. 40–45), March.
- [17] Lim, H., & Swartzlander, Jr., E. E. (1999). Multidimensional systolic arrays for the implementation of discrete Fourier transforms. *IEEE Transactions on Signal Processing*, 47(5), 1359–1370.

- [18] Nash, J. G. (2005). Computationally efficient systolic architecture for computing the discrete Fourier transform. *IEEE Transactions on Signal Processing*, 53(12), 4640–4651.
- [19] Nash, J. (2002). Hardware efficient base-4 systolic architecture for computing the discrete Fourier transform. In *Proc. IEEE workshop on signal processing systems (SiPS '02)* (pp. 87–92), May.
- [20] Baghaie, R., & Hartimo, I. (1993). An efficient new systolic architecture for the solution of discrete Fourier transform. In *Proc. VI workshop on VLSI signal processing* (pp. 453–461), October.
- [21] Wang, L., Hartimo, I., & Laakso, T. (1992). A novel double-decomposition method for systolic implementation of DFT. In *Proceedings of IEEE international symposium on circuits and systems (ISCAS '92)* (Vol. 3, pp. 1085–1088), May.
- [22] Kar, D. C., & Rao, V. V. B. (1993). A new systolic realization for the discrete Fourier transform. *IEEE Transactions on Signal Processing*, 41(5), 2008–2010.
- [23] Murthy, N. R., & Swamy, M. N. S. (1994). On the real-time computation of DFT and DCT through systolic architectures. *IEEE Transactions on Signal Processing*, 42(4), 988–991.
- [24] Liu, C. M., & Jen, C. W. (1992). On the design of VLSI arrays for discrete Fourier transform. *IEE Proceedings G. Circuits, Devices and Systems*, 139(4), 541–552.
- [25] Fang, W. H., & Wu, M. L. (1997). An efficient unified systolic architecture for the computation of discrete trigonometric transforms. In *Proc. IEEE symp. circuits and syst.* (Vol. 3, pp. 2092–2095).
- [26] Fang, W. H., & Wu, M. L. (1999). Unified fully-pipelined implementations of one- and two-dimensional real discrete trigonometric transforms. *IEICE Transactions on Fundamentals of Electronics, Communications and Computer Science*, E82-A(10), 2219–2230, October.
- [27] Chang, T. S., Guo, J. I., & Jen, C. W. (2000). Hardware-efficient DFT designs with cyclic convolution and subexpression sharing. *IEEE Transactions on Circuits and Systems. II: Analog and Digital Signal Processing*, 47(9), 886–892.
- [28] Meher, P. K. (2005). Design of a fully-pipelined systolic array for flexible transposition-free VLSI of 2-D DFT. *IEEE Transactions on Circuits and Systems. II: Express Briefs*, 52(2), 85–89.
- [29] Meher, P. K. (2006). Efficient systolic implementation of DFT using a low-complexity convolution-like formulation. *IEEE Transactions on Circuits and Systems. II: Express Briefs*, 53(8), 702–706.
- [30] Meher, P. K. (2006). Highly concurrent reduced-complexity 2-D systolic array for discrete Fourier transform. *IEEE Signal Processing Letters*, 13(8), 481–484.
- [31] Meher, P. K., Patra, J. C., & Vinod, A. P. (2006). A 2-D systolic array for high-throughput computation of 2-D discrete Fourier transform. In *Proc. IEEE Asia-Pacific conference on circuits and systems (APCCAS-06)* (pp. 1929–1932), December.
- [32] Meher, P. K., Patra, J. C., & Vinod, A. P. (2007). Novel recursive solution for area-time efficient systolization of discrete Fourier transform. In *2007 IEEE International Symposium on Circuits and Systems, ISCAS '07*, May.

- [33] Press, W., Teukolsky, S. A., Vetterling, W. T., & Flannery, B. P. (1992). *Numerical recipes in C: The art of scientific computing*. Cambridge: Cambridge University Press.

List of Tables

Table 1	Hardware- and time-complexities of proposed systolic structure for 1-D DFT and existing structures.
Table 2	Hardware- and time-complexities of proposed systolic structure for 2-D DFT and existing structures.

List of Figures

- Figure 1 **a** DG for computation of ϕ_{M-1}^k and ϕ_{M-2}^k for $1 \leq k \leq K$. **b** ϕ_{M-1}^k and ϕ_{M-2}^k for $1 \leq k \leq K$. **c** Function of each node of the DGs. $\omega = [2 \cos(\frac{2\pi k}{M})]$ for all the nodes of k -th column.
- Figure 2 The Proposed linear systolic array structure for the computation of 1-D DFT. **a** The linear array structure. **b** Function of the each PE of the structure. **c** Function of the Adder Cell. **d** Function of the Output Cell. **e** Function of the Latch Cell.
 $\theta = (\frac{2\pi k}{M})$ for the k -th PE.
- Figure 3 DGs for computation of $V_1(m, l)$, $V_2(m, l)$, $V_3(m, l)$, and $V_4(m, l)$ for $l = 1, 2, \dots, M-1$ and for a given value of m . **a** DG-1 for computation of $V_1(m, l)$ and $V_3(m, l)$. **b** DG-2 for computation of $V_2(m, l)$ and $V_4(m, l)$. **c** Function of each node of the DGs. $\gamma = [\cos(\frac{2\pi ln}{N})]$ and $\gamma = [-\sin(\frac{2\pi ln}{N})]$ for the node on l -th row and n -th column of DG-1 and DG-2, respectively.
- Figure 4 2-D systolic structure for the computation of intermediate sub-matrices $[V_1(m, l)]$, $[V_2(m, l)]$, $[V_3(m, l)]$ and $[V_4(m, l)]$ (**b**) Function of the processing element PEs. $C = [\cos(\frac{2\pi ln}{N})]$ and $S = [-\sin(\frac{2\pi ln}{N})]$ for the PE on l -th row and n -th column, respectively.
- Figure 5 Proposed structure for stage-one of the 2-D DFT **b** Function of Adder Cells AC-1, AC-2 and AC-3. TAG=1 and -1 for odd and even numbered AC-1.
- Figure 6 Complete structure for the 2-D DFT.

Structures	Multipliers	Adders	Cycle-time	Latency	ACT	Area-time
Structure of [22]	$2(N + 1)$	$2(N + 1)$	$T_M + T_A$	$(N + 1)(T_M + T_A)$	$N(T_M + T_A)$	$2N(N + 1)(T_M + T_A)$
Structure of [23]	$4(N + 1)$	$2(N + 1)$	$T_M + 2T_A$	$(N + 1)(T_M + 2T_A)$	$N(T_M + 2T_A)$	$4N(N + 1)(T_M + 2T_A)$
Structure of [24]	$2N$	$2N$	$T_M + T_A$	$(3N - 4)(T_M + T_A)$	$(N - 1)(T_M + T_A)$	$2N(N - 1)(T_M + T_A)$
Structure of [25]	$N + 2$	$N + 2$	$T_M + T_A$	$N(T_M + T_A)$	$N(T_M + T_A)$	$N(N + 1)(T_M + T_A)$
Proposed	$N + 4$	$N + 12$	$T_M + T_A$	$(3N/4 - 1)(T_M + T_A)$	$(N/2)(T_M + T_A)$	$(N/2)(N + 4)(T_M + T_A)$

Table 1

Structures	Multipliers	Adders	Latency	ACT	Area-time
Structure of [17]	$8N^2$	$4N^2$	$2N(T_M + 2T_A)$	$N(T_M + 2T_A)$	$(8N^3)(T_M + 2T_A)$
Structure of [28]	$4N^2$	$2N^2$	$2N(T_M + 2T_A)$	$2N(T_M + 2T_A)$	$(8N^3)(T_M + 2T_A)$
Structure of [30]	$2(N - 1)^2$	$2N^2 - N + 1$	$2N(T_M + T_A)$	$N(T_M + T_A)$	$2N(N - 1)^2(T_M + T_A)$
Proposed	$(3N^2/2 - 2N + 4)$	$3N^2/2 + 8N + 4$	$2(N - 2)(T_M + T_A)$	$N(T_M + T_A)$	$N(3N^2/2 - 2N + 4)(T_M + T_A)$

Table 2

Figure 1

Figure 2

Figure 3

Figure 4

Figure 5

Figure 6

A. P. Vinod received his B Tech degree in instrumentation and control engineering from University of Calicut, India in 1994 and the M. Engg and PhD degrees in computer engineering from Nanyang Technological University, Singapore in 2000 and 2004 respectively. He was with Kirloskar, India, from October 1993 to October 1995, and Tata Honeywell, India, from November 1995 to May 1997. During June 1997 to November 1998, he was associated with Shell Singapore. From September 2000 to September 2002, he was a lecturer in the School of Electrical and Electronic Engineering at Singapore Polytechnic, Singapore. He was a lecturer in the School of Computer Engineering at Nanyang Technological University (NTU), Singapore, from September 2002 to November 2004, and since December 2004, he has been an assistant professor in NTU. His research interests include digital signal processing, low power and reconfigurable DSP circuits, software radio, cognitive radio and brain-computer interface.

Jagdish C. Patra obtained Bachelors' (with first class honors) and Masters' degrees in Electronics and Telecommunication Engineering from Sambalpur University, India, in 1978 and 1988, respectively. He worked in various industry, teaching and R&D organizations (of India) for about 8 years. In 1987, he joined as a Lecturer at Regional College of Engineering, Rourkela, India, where he was promoted to Assistant Professor, subsequently. He was awarded with PhD degree from Indian Institute of Technology, Kharagpur, India in 1997. In 1999, he had been to the Technical University, Delft, The Netherlands, as a Guest Faculty, for six months. Subsequently, he joined as a Research Fellow at the School of EEE, Nanyang Technological University (NTU), Singapore. Currently, he is serving as an Assistant Professor in the School of Computer Engineering, NTU, Singapore.

Primary research interest of Dr. Jagdish includes complex problem solving using artificial neural networks, in the broad areas of pattern classification, digital communication channel equalization, nonlinear system identification, intelligent sensor processing and image processing. He is also active in the research areas of bioinformatics (DNA microarray analysis, disease gene prediction, drug design and discovery), digital document security (watermarking), web document and image retrieval, nonlinear ICA and BSS algorithms, and VLSI implementation. He has about 100 journal and conference publications in these areas. He has made few short time (each of 6–8 weeks) research visits to reputed international universities, such as, University of Iwate, Japan, University of Tampere, Finland, and University of Applied Sciences, HEIG-VD, Yverdon les Bains, Switzerland. He is a member of IEEE.

Pramod Kumar Meher (SM'03) received the first class degrees of B.Sc. (Honours) and M.Sc. in Physics, and Ph.D. in Science from Sambalpur University, Sambalpur, India in 1976, 1978, and 1996, respectively.

He has a wide scientific and technical background covering Physics, Electronics and Computer Engineering. Currently, he is a Senior Fellow in the School of Computer Engineering, Nanyang Technological University, Singapore. He was a Professor at Utkal University, Bhubaneswar, India since 1997–2002, a Reader in Electronics at Berhampur University, Berhampur, India during 1993–1997, and a Lecturer in Physics in various Government Colleges (in India) during 1981–1993. His research interest includes design of dedicated and reconfigurable architectures for computation-intensive algorithms pertaining to signal processing, image processing, secure communication and bioinformatics. He has published more than 100 technical papers in various reputed journals and conference proceedings. Currently, he is serving as Associate Editor of the *IEEE Transactions on Circuits and Systems-II: Express Briefs* and *The Journal of Circuits and Systems for Signal Processing*.

Dr. Meher was conferred with the Samanta Chandrasekhar Award for excellence in research in Engineering & Technology for the year 1999. He is a Chartered Engineer of the Engineering Council of United Kingdom, a Senior Member of IEEE, a Fellow of The Institution of Electronics and Telecommunication Engineers (IETE) of India, and a Fellow of the Institution of Engineering and Technology (IET), (formerly known as the Institution of Electrical Engineers), UK.