# Implementation of a New Class of FFT Algorithms on Transputer Computational Structures

Domingo Rodríguez
Electrical and Computer Engineering Department
University of Puerto Rico, Mayaguez PR 00681-5000
E-mail: domingo@rmece01.upr.clu.edu

Nayda G.Santiago
Electrical and Computer Engineering Department
University of Puerto Rico, Mayaguez PR 00681-5000
E-mail: nayda@rmece01.upr.clu.edu

Carlos Vélez
Electrical and Computer Engineering Department
University of Puerto Rico, Mayaguez PR 00681-5000
E-mail: cuso@rmece01.upr.clu.edu

**Abstract** - This work presents the theoretical formulation of a new class of fast Fourier transform algorithms and the experimental results obtained from the implementation of these algorithms on transputers computational structures. The theoretical formulation is based on a new methodology for the tensor or Kronecker product decomposition of the discrete Fourier transform matrix into a product of sparse matrices, with not all matrices having the same order. Special attention is given to what is termed a general twiddle or phase factor operator. Conforming this generalized operator, and the other factors present in a Kronecker product formulation of an algorithm, to the specific topology of a given transputer computational structure results in more efficient software implementations.

## I. INTRODUCTION

Two phenomena are motivating the search for fast parallel algorithms to compute the discrete Fourier transform (DFT). On the one hand, high performance computer technology is allowing the use of computationally efficient algorithms in some signal processing tasks for large scale applications. On the other hand, advances in communications technology are requiring more and more fast signal processing algorithms to solve very large and complex applications. Signal processing itself can be defined as the mathematical treatment of signals with the objective of extracting information important to a user. This definition points to the very fact that the endeavor of efficient signal processing algorithm design and development has become a vital component in the process of obtaining solutions to signal processing engineering and scientific applications. At the core of many of these signal processing algorithm developments is the efficient computation of the ubiquitous discrete Fourier transform operator.

This work presents a new class of parallel algorithms to compute the DFT. These fast Fourier transform (FFT) algorithms are of the additive type. They are called additive because they use the group theoretic additive properties of the input and output data indexing sets. They form a new class because they do not belong to the commonly known Cooley-Tukey[1] type additive FFT's. For instance, a set of these algorithms, called the 4S (for four stages) group, always has four stages of computation in addition to the permutation and twiddle or phase factor stages. This differs from Cooley-Tukey FFT's whose computational stages grow linearly when the length of the data grows. as powers of two.

This new class of algorithms is the result of a new methodology for the tensor or Kronecker decomposition of the Fourier matrix representing the discrete Fourier transform (DFT) operator into a product of sparse matrices, with not all matrices having the same order. The methodology is based on new techniques for indexing the input and output data sets resulting in different mapping techniques of one-dimensional data array to multidimensional data arrays. The work presents the theoretical formulation of this new class of algorithms and describes a methodology for mapping the algorithms to transputer computational structures. A transputer is a microprocessor specially designed to operate as a component within a parallel array of similar processors[2]. It has four bi-directional links for interprocessor communication. These links provide the user the capability of placing the transputers in any of the interconnection topologies used in parallel structures, that is, linear chain, tree, two-dimensional mesh and hypercube. A transputer computational structure (TCS) works as a multiple instruction multiple data (MIMD) structure (see Fig. 3). This implies that it can operate on different data using different commands at the same time. The data is transmitted from one transputer to another through special "channels" which synchronize the transmission.

An analysis of the data flow and the computational complexity is presented. It is shown that this new class of algorithms can be expressed with fewer number of computational stages, and for some algorithms, the data flow diagram has been simplified. Tensor or Kronecker product algebra, a branch of finite dimensional multilinear algebra, is used in this work for the mathematical formulation of this new class of FFT algorithms. It is shown that this algebra can be used as tool to assist in the implementation and modification of these algorithms on transputer computational structures. In [3], a methodology is presented for the analysis, design, implementation and modification of Fourier transform algorithms on various architectures. In [4], properties of tensor or Kronecker product algebra are used to design a new FFT algorithm. The work presented here is an extension of that previous work. Before describing this new class of algorithms, a set of basic definitions will be presented which will assist in the mathematical formulation of the algorithms.

## II. MATHEMATICAL PRELIMINARIES

The discrete Fourier transform (DFT) of a complex sequence, say $x$, of length $N$, is defined as another complex sequence, say $y$, also of the same length $N$, whose $k$-th term is given by the expression

$$y[k] = \sum_{l=0}^{N-1} x[l]\omega_N^{kl}, \quad \omega_N = e^{-j\frac{2\pi}{N}}, \quad j = \sqrt{-1}$$

CH 3381-1/93/$01.00 ©1993 IEEE

This is equivalent to the matrix-vector computation

$$y = F_N \cdot x,$$

where $F_N$ denotes the matrix representation of the DFT operator and is given by

$$F_N = \left[\omega_N^{kl}\right]_{0 \le k,\, l < N}, \quad \omega_N = e^{-j\frac{2\pi}{N}}$$

The set of all $N$-dimensional complex sequences form a vector space which denoted by $C^N$. Associated with this space is the standard basis set

$$\delta^N = \{\delta^N_{\{0\}}, \delta^N_{\{1\}}, \delta^N_{\{2\}}, \ldots, \delta^N_{\{N-1\}}\}$$

where $\delta^N_{\{k\}}[m] = \begin{cases} 1, \text{ if } k = m \\ 0, \text{ if } k \neq m \end{cases}$

Given two vector spaces, say $C^R$ and $C^M$, a new vector space of dimension $RM$ can be formed as the Kronecker product $C^R \otimes C^M$ with standard basis

$$\delta^R \otimes \delta^M = \{\delta^R_{\{k\}} \otimes \delta^M_{\{l\}} : k \in Z_R, l \in Z_M\}$$

Here, $Z_R = \{0, 1, 2, \ldots, R-1\}$.

The vector space $C^R \otimes C^M$ can be identified with the vector space $C^{RM}$ through the mapping

$$\delta^R_{\{k\}} \otimes \delta^M_{\{l\}} \leftrightarrow \delta^{RM}_{\{kM+l\}}$$

Another identification can be made between the vector space $C^R \otimes C^M$ and the vector space $C^{R \times M}$ of $R \times M$ matrices with standard basis

$$\Delta^{R \times M} = \left\{\Delta^{R \times M}_{\{k,\, l\}} : k \in Z_R, l \in Z_M\right\}$$

where the $\Delta^{R \times M}_{\{k,\, l\}}$ is equal to $1$ in the $\{k, l\}$ position and zeros elsewhere. Thus, the following mapping can be established

$$\delta^R_{\{k\}} \otimes \delta^M_{\{l\}} \leftrightarrow \Delta^{R \times M}_{\{k,\, l\}}$$

The identifications established above allow the following definition of the tensor or Kronecker product of linear transformations. Let $A$ be a linear transformation in the vector space $C^R$, and $B$ a linear transformation in the vector space $C^M$. Then,

$$(A \otimes B)(x \otimes y) = Ax \otimes By$$

If the symbols $A$ and $B$ are also used to denote the matrix representation of the linear transformation with respect to the standard basis, the following matrix expression is obtained:

$$A \otimes B = [a_{kl} \cdot B]$$

These definitions allow the study of actions of linear operators over complex sequences of finite length representing finite duration discrete signals. As a linear operator, the discrete Fourier transform has many properties which appear as redundancies in the Fourier matrix $F_N$. Advantages of these redundancies can be taken when a matrix-vector computation is being performed. Fast algorithms for computing the discrete Fourier transform are obtained when the matrix $F_N$, representing the DFT operator, is decomposed into a series of factors which can be mapped to specific machine architectures through efficient software implementations. With tensor or Kronecker product algebra a mathematical environment can be created in which to analyze algorithms in a unified manner. One of the advantages of using Kronecker product language to describe FFT algorithms is that this mathematical language can be used as an analytic tool to study algorithm formulations for hardware and software implementations as well as the identification of new algorithms. A Kronecker product formulation of a new class of parallel algorithm is now presented.

## III. KRONECKER PRODUCT FORMULATION OF PARALLEL ALGORITHMS

In [5] it was demonstrated how to use Kronecker products to formulate Cooley-Tukey type algorithms for the computation of the discrete Fourier transform of an $N$-point complex sequence whenever $N$ is a composite integer. When $N$ is a composite of the form

$$N = a_{k-1}2^{k-1} \cdot a_{k-2}2^{k-2} \ldots a_2 2^2 \cdot a_1 2 \cdot a_0$$

for $a_{k-1}, a_{k-2}, \ldots, a_2, a_1, a_0$ any integers, mappings of the input and output indexing sets of a DFT computation can be identified which associate these one-dimensional arrays of indexing sets to multidimensional arrays. These mappings, as the ones described above in the section on mathematical preliminaries, are used to obtain the new class of FFT algorithms as will be shown now. For the sake of clarity, the two factor case $N = RM$, $M = RS$ has been chosen.

Start by considering the expression

$$y[k] = \sum_{l=0}^{N-1} x[l]\omega_N^{kl}, \quad \omega_N = e^{-j\frac{2\pi}{N}}, \quad j = \sqrt{-1}$$

Using the mappings

$$k \leftrightarrow k_1 + k_0 M, \qquad l \leftrightarrow l_1 + l_0 M$$

Results in the product expansion

$$k = k_1 l_1 + k_1 k_0 M + k_0 l_0 + k_0 l_0 M^2$$

for $k_1, l_1 \in Z_M$, $k_0, l_0 \in Z_R$. This substitution produces the following result

$$y[k_1, k_0] = \sum_{l_1=0}^{M-1} \omega_R^{k_0 l_1} \left\{ \omega_{RM}^{k_1 l_1} \left[ \sum_{l_0}^{R-1} \omega_R^{k_1 l_0} x_1[l_0, l_1] \right] \right\}$$

If the arrays $x[l_1, l_0]$ and $x_1[l_0, l_1]$ are identified with elements in $C^{RM}$, then they can be related through the permutation operation

$$x_1[l_0, l_1] = P_{N,M}\, x[l_1, l_0]$$

This permutation, of order $N$, is termed a stride by $M$ permutation matrix. This matrix can be defined through the identity

$$P_{N,M}(x \otimes y) = (y \otimes x)$$

for $x \in C^R$, and $y \in C^M$.

Using the identifications

$$\delta^R_{\{k\}} \otimes \delta^M_{\{l\}} \leftrightarrow \delta^{RM}_{\{kM+l\}}$$

$$\delta^R_{\{k\}} \otimes \delta^M_{\{l\}} \leftrightarrow \Delta^{R \times M}_{\{k,l\}}$$

defined above, the following important mapping can be established

$$\Delta^{R \times M}_{\{k,l\}} \leftrightarrow \delta^{RM}_{\{kM+l\}}$$

Consider an association established between the permuted input data set $x_1[l_0, l_1] \in C^{R \times M}$, ordered lexicographically, and an element, say $x_1[l_0 l_1]$ in $C^{RM}$. Then, consider a matrix representation of the linear transformation

$$y_1[k_1, l_1] = \sum_{l_0=0}^{R-1} \omega_R^{k_1 l_0} x_1[l_0, l_1]$$

which was given as the inner bracket of the previous summation expression. This matrix representation can be expressed in Kronecker product notation by noticing that the matrix vector operation amounts to the computation of $M$ R-point discrete Fourier transforms. Thus, the following result is obtained:

$$y_1[k_1 l_1] = (I_M \otimes F_R) x_1[l_0 l_1]$$

Following the same procedure, and after some Kronecker product manipulations, the following general expression is obtained [5] for the DFT computation of a complex sequence of length $N = RM$.

$$y = (F_R \otimes I_M)((U^T{}_S \otimes I_R) \otimes I_M) T_{L,K,N,M} (I_M \otimes (U_S \otimes I_R)) \cdot (I_M \otimes F_R) P_{N,M}\, x$$

where $I_M$ is the identity matrix of order $M$, $U^T{}_M = \{1, 1, 1, \ldots, 1\}$ is a row vector of length $M$, and $T_{L,K,N,M}$ is the generalized twiddle factor which is a matrix of order $L$, given as the direct sum of diagonal matrices, expressed as follows

$$T_{L,K,N,M} = \sum_{k=0}^{K-1} \oplus D^k{}_{N,M},$$

$$D_{N,M} = diag\,[1, \omega_N, \omega^2{}_N, \ldots, \omega^{M-1}{}_N]$$

## IV. MAPPING ALGORITHMS TO COMPUTATIONAL STRUCTURES

In the previous section a general Kronecker product formulation of a new class of parallel algorithms was presented. This section discusses a methodology for the mapping of these algorithms to computational hardware structures. It is important to point out that the main idea behind a Kronecker product formulation of an algorithm is to use Kronecker Product algebra as a tool to assist in the implementation process. For instance, consider the Kronecker product decomposition of the DFT matrix $F_N$ which was presented in the previous section:

$$F_N = (F_R \otimes I_M)((U^T{}_S \otimes I_R) \otimes I_M) T_{L,K,N,M} (I_M \otimes (U_S \otimes I_R)) \cdot (I_M \otimes F_R) P_{N,M}$$

Since the DFT matrix is symmetric, taking the Fourier on both sides of the expression above results in the following mathematical variant

$$F_N = P_{N,R} (I_M \otimes F_R)(I_M \otimes (U T^T{}_S \otimes I_R)) T_{L,K,N,M} ((U_S \otimes I_R) \otimes I_M) \cdot (F_R \otimes I_M)$$

A brief description of the methodology for mapping mathematical algorithms to computer hardware structures for software implementation is now presented. First, the concept of a computational mathematics environment (CME) is introduced [6]. This environment basically consists of an input data set, an output data set, a set of mathematical operators acting on these data sets, and a set of rules for these operators. Under this environment, computational mathematics frameworks can be developed. A given computational mathematics framework is composed of an FFT Kronecker product formulation, termed the canonical form, and a set of variants of this formulation obtained through the use of properties of Kronecker product algebra. When, for example, a computational hardware structure is given, as depicted in figure 1and 2 on the appendix below, inherent attributes of this structure are expressed as parameters and used in the computational mathematics framework to assist in the mapping of the algorithm. What follows is a search for variants of this algorithm (elements of a defined computational mathematics framework) which best conform to the inherent attributes of the computational structure.

The problem of mapping mathematical algorithms to computer hardware structures for software implementation has a central point the identification of the basic mathematical expressions, termed mathematical functional primitives (MFP) (which appear as factors in any given Kronecker product formulation of an algorithm) with basic software operations, termed software functional primitives (SFP). An important class of mathematical functional primitives are stride permutation operations. These are called stride because they permute a given data set at a constant stride or length. These permutations arise as Kronecker product bases are permuted. Kronecker products of stride permutations govern the addressing between the stages of a Kronecker product decomposition. For instance, the following theorem, known as the Kronecker product commutation theorem [7], allows to interchange expressions of the form $(I_M \otimes F_R)$, known as parallel Fourier factor operations, with expressions of the form $(F_R \otimes I_M)$, known as vector Fourier factor operations.

*Theorem:*
Let $A$ and $B$ be arbitrary matrices of order $R$ and $M$, respectively. If $N = RM$, then

$$P_{N,R}(A \otimes B) P_{N,M} = (B \otimes A)$$

If $A = F_R$ and $B = I_M$, then

$$P_{N,R}(F_R \otimes I_M) P_{N,M} = (I_M \otimes F_R)$$

For the case of Kronecker product formulation of this new class of FFT algorithms, the following mathematical functional primitives can be identified from the general expression given above: permutation operation $P_{N,M}$, parallel Fourier factor operation $(I_M \otimes F_R)$, parallel addressing assignment operation $(I_M \otimes (U_S \otimes I_R))$, generalized twiddle or phase factor operation $T_{L,K,N,M}$, vector addressing assignment operation $((U^T{}_S \otimes I_R) \otimes I_M)$, and vector Fourier factor operation $(F_R \otimes I_M)$.

## V. CONCLUSION

The formulation of a new of class parallel FFT algorithms has been presented. This formulation has been accomplished using the algebra of Kronecker products. Kronecker product algebra has been used as a tool to assist in the implementation of variants of the algorithm to computational hardware structures. Kronecker product algebra was also used as a language to identify FFT variants from the canonical representation of a given algorithm as well as the identification of similarities and differences between these variants. Stride permutation operations were introduced as mathematical functional primitives which control the data communication aspects between the given stages of an algorithm implementation.

## REFERENCES


[1] J. W. Cooley, J. W. Tukey, "An Algorithm for the Machine Calculation of Complex Fourier Series," Math. Comp. Vol. 19, pp. 297-301, 1965.

[2] M. Homewood, et al, "The IMS T800 Transputer," IEEE Micro, Vol. 7, No. 5, pp. 10-26, Oct. 1987.

[3] J. Johnson, R. Johnson, D. Rodríguez, R. Tolimieri, "A Methodology for Designing, Modifying, and Implementing Fourier Transform Algorithms on Various Architectures," Journal of Circuits, Systems and Signal Processing, Birkhäuser, Vol. 9, No. 4, 1990.

[4] D. Rodríguez, "A New FFT Algorithm and Its Implementation on the DSP96002," Proceedings of the IEEE International Conference on Acoustics Speech and Signal Processing, Vol. 3, pp. 2189-2192, Toronto, Canada, 1991.

[5] D. Rodríguez, "Tensor Product Algebra as a Tool for the VLSI Implementation of the Discrete Fourier Transform," IEEE International Conference on Acoustics Speech and Signal Processing, Vol. 2, pp. 1025-1028, Toronto, Canada, 1991.

[6] D. Rodríguez, J. Seguel, E. Cruz, "Algebraic Methods for the Analysis and Design of Time-Frequency Signal Processing Algorithms," 1993 International Conference on Circuits and Systems, Vol. 1, pp. 196-199, Chicago, 1993.

[7] D. Rodríguez, "Tensor Products Formulations of Additive Fast Fourier Transform Algorithms and Their Implementations," Ph. D. Thesis, City University of New York, Feb. 1988.


## APPENDIX

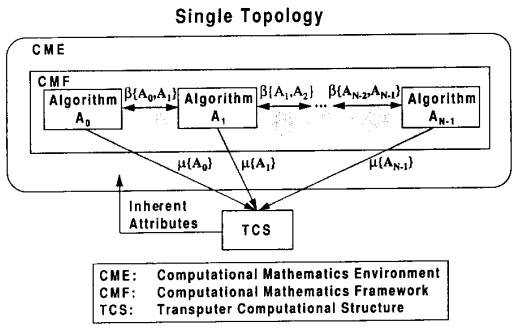


Figure 1: Implementations of variants of a canonical algorithm formulation on a transputer computational structure (fixed hardware topology).

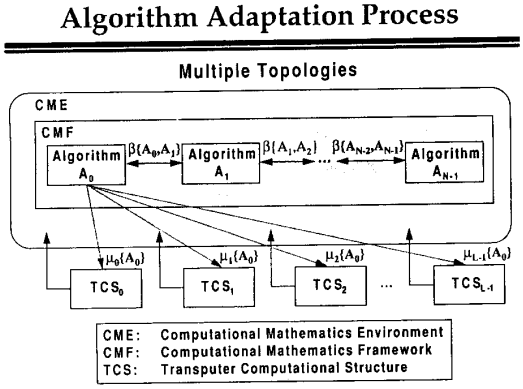


Figure 2: Implementation of a canonical algorithm formulation on transputer computational structures (variable hardware topology).

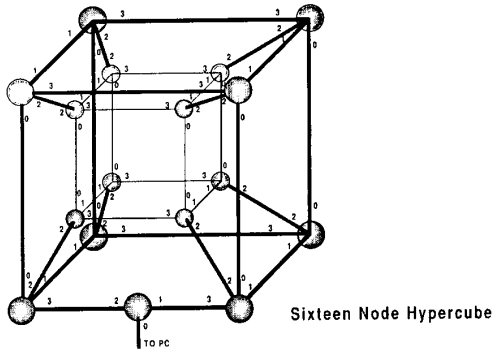


Figure 3: A hardware computational structure consisting of transputers in a hypercube topology plus a node to access the external world.


This work is partially supported under NSF Grant CDA 8913486.