A 64-Point Fourier Transform Chip for Video Motion Compensation Using Phase Correlation

Colin Chiu Wing Hui, Tiong Jiu Ding, John V. McCanny, *Senior Member, IEEE,* and Roger F. Woods, *Member, IEEE*

Abstract—**Details of a new low power fast Fourier transform (FFT) processor for use in digital television applications are presented. This has been fabricated using a 0.6-μm CMOS technology and can perform a 64 point complex forward or inverse FFT on real-time video at up to 18 Megasamples per second. It comprises 0.5 million transistors in a die area of 7.8 $\times$ 8 mm^2 and dissipates 1 W. The chip design is based on a novel VLSI architecture which has been derived from a first principles factorization of the discrete Fourier transform (DFT) matrix and tailored to a direct silicon implementation.**

I. Introduction

VIDEO motion compensation methods based on the use of phase correlation algorithms are now being successfully used in equipment for professional television applications. These include video slow-motion systems and digital TV standards converters. These contrast with more conventional motion compensation methods, in that motion estimation is performed in the frequency domain, rather than on blocks of pixels in the spatial domain. These methods produce high quality images suitable for studio broadcasting systems [1]–[4]. A key requirement in TV phase correlation systems is the need to perform two-dimensional (2-D) fast Fourier transform (FFT) computations in real-time. Typical computation rates required are in the region of several billion multiply/accumulates per second (MAC's) in standard digital TV (DTV) applications and over 10 billion MAC's in digital HDTV systems.

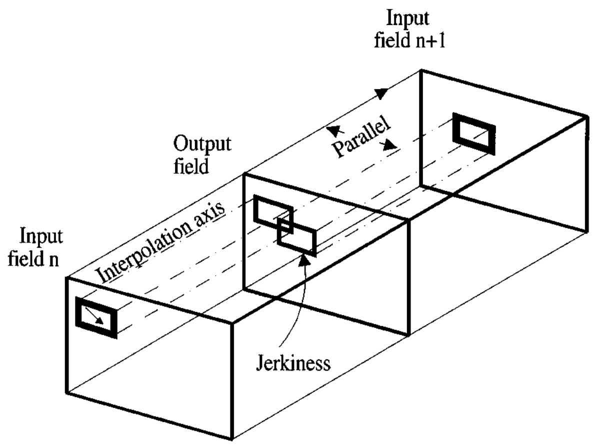
This paper describes a novel 64-point FFT chip which has been developed as part of a larger research program to develop new digital TV standards converters and slow motion systems based on the use of phase correlation algorithms.

The chip, which has been successfully fabricated and tested, performs a forward or inverse 64-point FFT on complex two's complement video data supplied at a rate of 13.5 MHz (one real 16 b word and one imaginary 16 b word, at a clock rate of 27 MHz) and can operate at 18 Megasamples per second (36 MHz clock rate) making it suitable for wide screen TV. It has been fabricated using VLSI Technology's 0.6-μm, double-layer metal, standard cell CMOS technology, contains 535 000 transistors and uses an internal 3.3 V power supply. It has an area of 7.8 $\times$ 8 mm^2, dissipates 1 W, has 48 I/O pins and is housed in a 84-pin PLCC package, leading to a low power, cost-effective silicon solution.

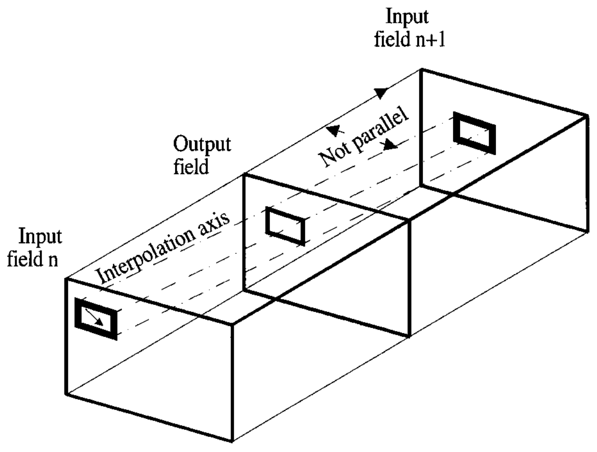
Manuscript received April 8, 1996; revised July 9, 1996. This work was supported in part by the U.K. Science and Engineering Research Council under grant GR/G64909.

The authors are with the Institute of Advanced Microelectronics, The Ashby Building, Queen's University of Belfast, Belfast BT9 5AH, N. Ireland.

Publisher Item Identifier S 0018-9200(96)07940-1.



(a)



(b)

Fig. 1. (a) Conventional standards conversion. (b) Motion compensated standards conversion.

II. Phase Correlation—A Brief Overview

Conventionally, in TV standards conversion, picture resolution conversion is performed by an averaging (filtering) process, whereas frame rate conversion is achieved by repeating fields on the time axis. This type of frame rate conversion works well with still images but can cause significant jerkiness if the program material contains motion. However, with motion compensation, the motion of an object in a scene is tracked so that a new frame is interpolated along the motion trajectory, rather than the time axis, as illustrated in Fig. 1.

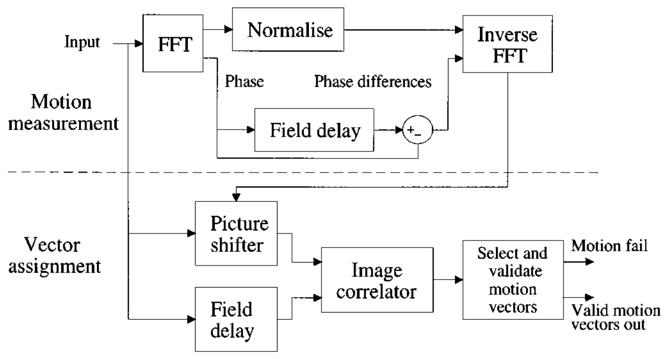
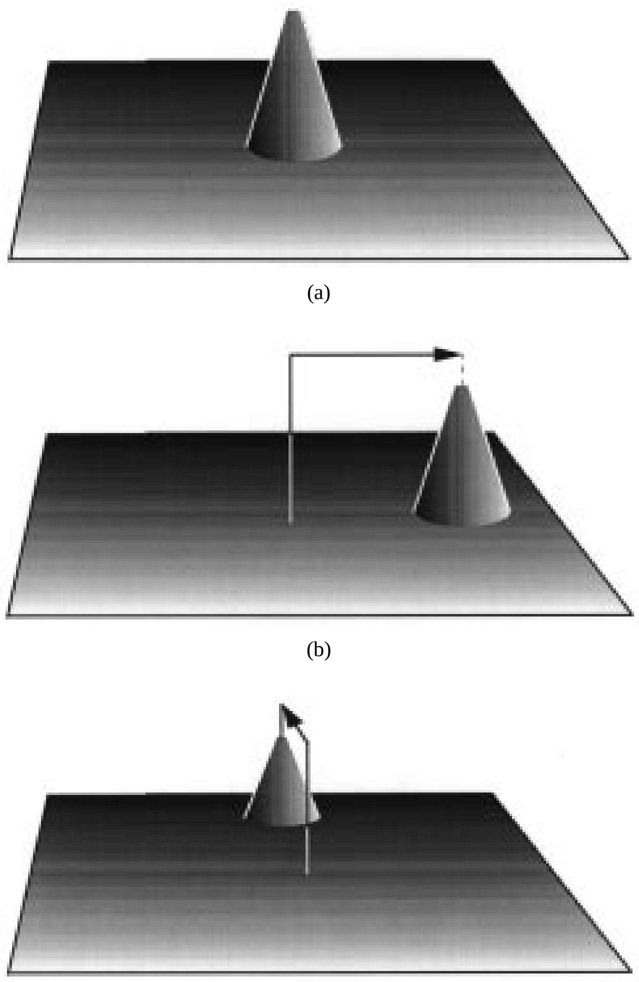


Fig. 2. Phase correlated motion estimator.

A block diagram of a phase correlated motion estimator is shown in Fig. 2. An image is first partitioned into blocks which are typically 64 pixels by 64 lines. Each image block, in two successive fields, is then phase correlated. This is achieved by first applying a 2-D FFT to each block. The phase of each transformed block is then subtracted from the corresponding block of the previous field. Meanwhile, the amplitudes are normalized to eliminate any variations in illumination which may confuse the motion measurement. The phase differences and the normalized amplitudes are then subjected to a 2-D inverse FFT to produce a phase correlation surface. The surface obtained contains peaks, whose coordinates from the center correspond to the vertical and horizontal velocity components of dominant motions in the scene. A list of trial motion vectors is therefore compiled by locating the peaks. Fig. 3 demonstrates some examples of the correlation surface.

The trial motion vectors are then passed to a vector assignment unit as candidate vectors and are used to derive a valid motion vector for each pixel in the scene. The two image fields are spatially shifted by each candidate vector in turn and correlated by an image correlator. During the spatial correlation, the modulus of the luminance difference is calculated and analyzed. The vector which gives the lowest value is regarded as the valid vector.

Other applications of phase correlation include slow motion generation, noise reduction, correction of film unsteadiness, and HDTV bandwidth reduction. The availability of cost effective real-time FFT VLSI chips therefore becomes a prerequisite for the hardware implementation of such systems.



(a)

(b)

(c)

Fig. 3. Correlation surface. (a) No motion, (b) pan right, and (c) tilt.

III. FFT Architecture

Prior work on Fourier transform systems typically falls into one of two categories a) methods based on direct discrete Fourier transform implementations [5]–[7] and b) methods based on direct hardware implementations of established FFT signal flow graphs [8]–[18]. The problem with such solutions is that the approach adopted at algorithmic level typically takes little account of implications at architecture, data flow, or chip design levels. Consequently, many such designs may be irregular, dominated by wiring and may have heavy overheads in terms of data storage [12]–[15].

The approach adopted here has been to develop algorithm-to-architecture mappings directly from first principles with the aim of producing an efficient silicon solution. In particular, the inherent structure within the DFT matrix has been exploited to tailor the organization and flow of data to that best suited to candidate architectural solutions. In the case of a radix-4 matrix factorization, this involves partitioning the DFT matrix into regular blocks of smaller matrices, with these then being mapped onto a regular silicon structure. The merits of using a radix-4 decomposition (rather than other radices) are well documented [19]. These include regularity, simplicity, and a reduction in the number of complex multiplications required. Conventional Cooley–Tukey radix-4 FFT's algorithms [20] are based on *either* the decimation of the time *or* the frequency samples, leaving the other in a natural order. However, investigations undertaken in the course of this research suggested that greater architectural advantage can be achieved by using a base-4 decomposition in which *both* the time and frequency transform data are reversed in order prior to the matrix factorization [21]. This *decimation-in-time-and-frequency* (DITF) algorithm leads not only to computational efficiencies similar to that of more conventional Cooley–Tukey algorithms but also to regular architectures where data flow is tailored to that required in typical real-time image processing applications. As will be discussed, it also leads to important reductions in data storage requirements.

The approach adopted here can be explained by considering the example of a 16-point DFT matrix, as seen in (1) at the bottom of the page.

In order to exploit the inherent structure within the matrix, both the time and frequency sequences can be permuted into a base-4 digit-reversed order. Equation (1) can hence be rewritten as (2), shown at the bottom of the page, where

$$[A]=\begin{bmatrix}1&1&1&1\\1&1&1&1\\1&1&1&1\\1&1&1&1\end{bmatrix},\quad [B]=\begin{bmatrix}1&-j&-1&j\\1&-j&-1&j\\1&-j&-1&j\\1&-j&-1&j\end{bmatrix},$$

$$[C]=\begin{bmatrix}1&-1&1&-1\\1&-1&1&-1\\1&-1&1&-1\\1&-1&1&-1\end{bmatrix},\quad [D]=\begin{bmatrix}1&j&-1&-j\\1&j&-1&-j\\1&j&-1&-j\\1&j&-1&-j\end{bmatrix}$$

and

$$\begin{bmatrix}d_1\\d_2\\d_3\\d_4\end{bmatrix}_D=\begin{bmatrix}d_1&0&0&0\\0&d_2&0&0\\0&0&d_3&0\\0&0&0&d_4\end{bmatrix}.$$

Since each of the rows within each of the individual matrices $\boldsymbol{A}$, $\boldsymbol{B}$, $\boldsymbol{C}$, and $\boldsymbol{D}$ are the same, it follows that the multiplication of these matrices with an input sequence x in (2) can effectively be replaced by a 4-by-4 matrix multiplication as shown in (3)

$$\begin{bmatrix}a_0&a_1&a_2&a_3\\b_0&b_1&b_2&b_3\\c_0&c_1&c_2&c_3\\d_0&d_1&d_2&d_3\end{bmatrix}=\begin{bmatrix}1&1&1&1\\1&-j&-1&j\\1&-1&1&-1\\1&j&-1&-j\end{bmatrix}\begin{bmatrix}x_0&x_1&x_2&x_3\\x_4&x_5&x_6&x_7\\x_8&x_9&x_{10}&x_{11}\\x_{12}&x_{13}&x_{14}&x_{15}\end{bmatrix}\tag{3}$$

where a, b, c, and d are the elements with each of the rows of new matrices $\boldsymbol{A}'$, $\boldsymbol{B}'$, $\boldsymbol{C}'$, and $\boldsymbol{D}'$ which result from

$$\begin{bmatrix}X_0\\X_1\\X_2\\X_3\\X_4\\X_5\\X_6\\X_7\\X_8\\X_9\\X_{10}\\X_{11}\\X_{12}\\X_{13}\\X_{14}\\X_{15}\end{bmatrix}=\begin{bmatrix}
1&1&1&1&1&1&1&1&1&1&1&1&1&1&1&1\\
1&W&W^2&W^3&-j&W^5&W^6&W^7&-1&-W&-W^2&-W^3&j&-W^5&-W^6&-W^7\\
1&W^2&-j&W^6&-1&-W^2&j&-W^6&1&W^2&-j&W^6&-1&-W^2&j&-W^6\\
1&W^3&W^6&-W&j&-W^7&W^2&W^5&-1&-W^3&-W^6&W&-j&W^7&-W^2&-W^5\\
1&-j&-1&j&1&-j&-1&j&1&-j&-1&j&1&-j&-1&j\\
1&W^5&-W^2&-W^7&-j&-W&-W^6&W^3&-1&-W^5&W^2&W^7&j&W&W^6&-W^3\\
1&W^6&j&W^2&-1&-W^6&-j&-W^2&1&W^6&j&W^2&-1&-W^6&-j&-W^2\\
1&W^7&-W^6&W^5&j&W^3&-W^2&W&-1&-W^7&W^6&-W^5&-j&-W^3&W^2&-W\\
1&-1&1&-1&1&-1&1&-1&1&-1&1&-1&1&-1&1&-1\\
1&-W&W^2&-W^3&-j&-W^5&W^6&-W^7&-1&W&-W^2&W^3&j&W^5&-W^6&W^7\\
1&-W^2&-j&-W^6&-1&W^2&j&W^6&1&-W^2&-j&-W^6&-1&W^2&j&W^6\\
1&-W^3&W^6&W&j&W^7&W^2&-W^5&-1&W^3&-W^6&-W&-j&-W^7&-W^2&W^5\\
1&j&-1&-j&1&j&-1&-j&1&j&-1&-j&1&j&-1&-j\\
1&-W^5&-W^2&W^7&-j&W&-W^6&-W^3&-1&W^5&W^2&-W^7&j&-W&W^6&W^3\\
1&-W^6&j&-W^2&-1&W^6&-j&W^2&1&-W^6&j&-W^2&-1&W^6&-j&W^2\\
1&-W^7&-W^6&-W^5&j&-W^3&-W^2&-W&-1&W^7&W^6&W^5&-j&W^3&W^2&W
\end{bmatrix}\begin{bmatrix}x_0\\x_1\\x_2\\x_3\\x_4\\x_5\\x_6\\x_7\\x_8\\x_9\\x_{10}\\x_{11}\\x_{12}\\x_{13}\\x_{14}\\x_{15}\end{bmatrix}\tag{1}$$

$$\begin{bmatrix}X_0\\X_4\\X_8\\X_{12}\\X_1\\X_5\\X_9\\X_{13}\\X_2\\X_6\\X_{10}\\X_{14}\\X_3\\X_7\\X_{11}\\X_{15}\end{bmatrix}=\begin{bmatrix}
\begin{bmatrix}1\\1\\1\\1\end{bmatrix}_D[A] & \begin{bmatrix}1\\-j\\-1\\j\end{bmatrix}_D[A] & \begin{bmatrix}1\\-1\\1\\-1\end{bmatrix}_D[A] & \begin{bmatrix}1\\j\\-1\\-j\end{bmatrix}_D[A]\\
\begin{bmatrix}1\\1\\1\\1\end{bmatrix}_D[B] & \begin{bmatrix}W\\-jW\\-W\\jW\end{bmatrix}_D[B] & \begin{bmatrix}W^2\\-W^2\\W^2\\-W^2\end{bmatrix}_D[B] & \begin{bmatrix}W^3\\jW^3\\-W^3\\-jW^3\end{bmatrix}_D[B]\\
\begin{bmatrix}1\\1\\1\\1\end{bmatrix}_D[C] & \begin{bmatrix}W^2\\-jW^2\\-W^2\\jW^2\end{bmatrix}_D[C] & \begin{bmatrix}-j\\j\\-j\\j\end{bmatrix}_D[C] & \begin{bmatrix}W^6\\jW^6\\-W^6\\-jW^6\end{bmatrix}_D[C]\\
\begin{bmatrix}1\\1\\1\\1\end{bmatrix}_D[D] & \begin{bmatrix}W^3\\-jW^3\\-W^3\\jW^3\end{bmatrix}_D[D] & \begin{bmatrix}W^6\\-W^6\\W^6\\-W^6\end{bmatrix}_D[D] & \begin{bmatrix}W^9\\jW^9\\-W^9\\-jW^9\end{bmatrix}_D[D]
\end{bmatrix}\begin{bmatrix}x_0\\x_4\\x_8\\x_{12}\\x_1\\x_5\\x_9\\x_{13}\\x_2\\x_6\\x_{10}\\x_{14}\\x_3\\x_7\\x_{11}\\x_{15}\end{bmatrix}\tag{2}$$

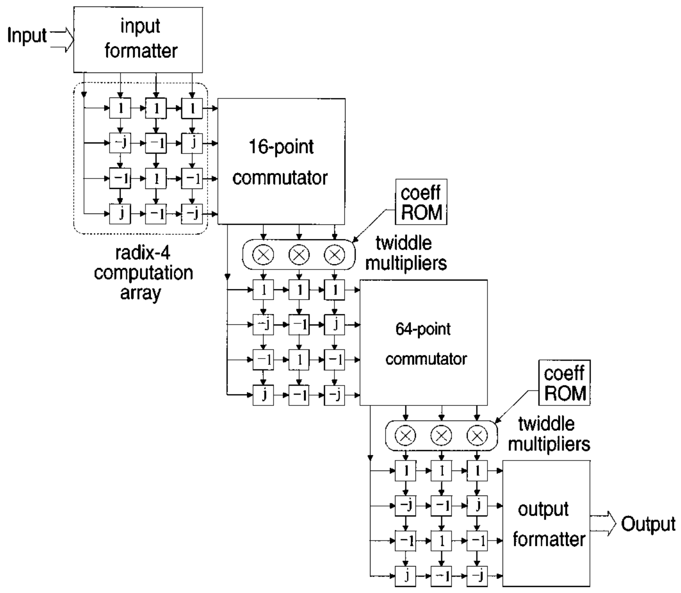


Fig. 4. FFT chip architecture.

the multiplication of the respective submatrices $\boldsymbol{A}$, $\boldsymbol{B}$, $\boldsymbol{C}$, and $\boldsymbol{D}$ with the appropriate four elements from the $\boldsymbol{x}$ vector. From (3), it follows that the computation in (2) can written as

$$\begin{bmatrix} X_1 & X_1 & X_2 & X_3 \\ X_4 & X_5 & X_6 & X_7 \\ X_8 & X_9 & X_{10} & X_{11} \\ X_{12} & X_{13} & X_{14} & X_{15} \end{bmatrix} = \begin{bmatrix} 1 & 1 & 1 & 1 \\ 1 & -j & -1 & j \\ 1 & -1 & 1 & -1 \\ 1 & j & -1 & -j \end{bmatrix} \begin{bmatrix} a_0 & b_0 & c_0 & d_0 \\ a_1 & Wb_1 & W^2c_1 & W^3d_1 \\ a_2 & W^2b_2 & W^4c_2 & W^6d_2 \\ a_3 & W^3b_3 & W^6c_3 & W^9d_3 \end{bmatrix}. \tag{4}$$

A similar factorization and decimation of a 64-point DFT matrix can be achieved as outlined in the Appendix. This has been used directly to derive the architecture shown in Fig. 4. This architecture consists of a repetitive cascade of three building blocks: a) a radix-4 computation array (R4CA); b) twiddle multipliers; and c) data commutator circuits. It also contains appropriate input and output formatters to allow the direct interface with DTV signals input in natural order. From an implementation point of view, it was decided that real and imaginary data should be multiplexed onto the same data bus in order to reduce the number of I/O pins by half. The R4CA blocks perform the radix-4 operations described and involve only additions and subtractions (see Fig. 5). As data streams are clocked through the circuit, they are reordered using different commutator circuits and then multiplied by twiddle factors stored in ROM. The computed results are then clocked through the system and emerge in natural order at the output formatter.

Detailed investigations were undertaken in order to ascertain the best tradeoffs between a) computational requirements; b) silicon area; and c) data organization/data flow. This led to

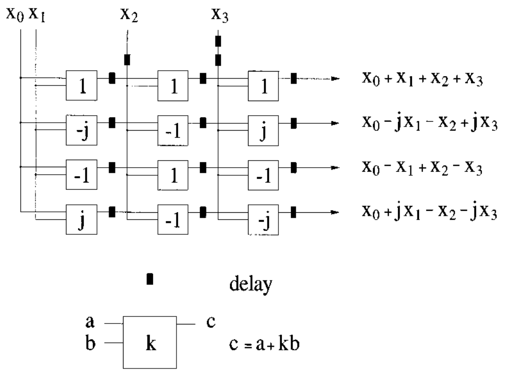


Fig. 5. Radix-4 computation array.

an internal circuit organization in which data is processed in a word-parallel, digit-serial manner. In this case, data input in a conventional word-serial, bit-parallel format is reorganized into eight parallel (four real and four imaginary) streams of 3-b wide digits. Use of this data organization means that all computational blocks are 100% utilized. This contrasts with the Bi and Jones [22] architecture in which the multiplier blocks exhibit 75% efficiency. The Gold–Bially [14] architecture can achieve 100% efficiency but requires all four data words to be processed to be available in parallel. For the application considered (as in many other FFT systems), where the FFT processor must be interfaced to a continuous word serial stream, it is only possible to achieve a 25% computational efficiency as there is a 4 : 1 mismatch between between the bandwidth of the input data and that of the processor.

The bit-sliced nature of the architecture facilitates tradeoffs between digit size and throughput rates. In this case, a 3-b wide digit stream was found to be sufficient to allow the required chip sampling rates to be achieved in the fabrication technology used. Fig. 6 describes the data conversion and reordering performed by the input formatter. This consists of a converter and a delay commutator. For the purposes of illustration, each digit is numbered in Fig. 6(a). As illustrated, the data converter distributes the block of 64 bit-parallel samples into four subblocks of 16 digit-serial samples, in effect performing a matrix transposition. The output formatter has the same basic function as the input formatter and is implemented using a similar circuit.

It should be noted that the architecture presented has a storage requirement of $1.75N$ complex word registers. This is $0.5N$ and $0.25N$ less than that of Gold and Bially [14] and Bi and Jones [22], respectively, and represents an important savings. A more detailed comparison of the relative complexity of the three architectures is given in Table I. In this it is assumed that all architectures are interfaced with word-serial, bit-parallel I/O samples in a natural order. The notation $\{\ \}$ denotes the value when $N = 64$ and $d = n/4$. Here n is the number of bits per input sample, and d the digit size. The architecture described internally employs digit-serial adders/multipliers. This contrasts with the Gold and Bially and Bi and Jones circuits which use bit-parallel multiplier/adders.

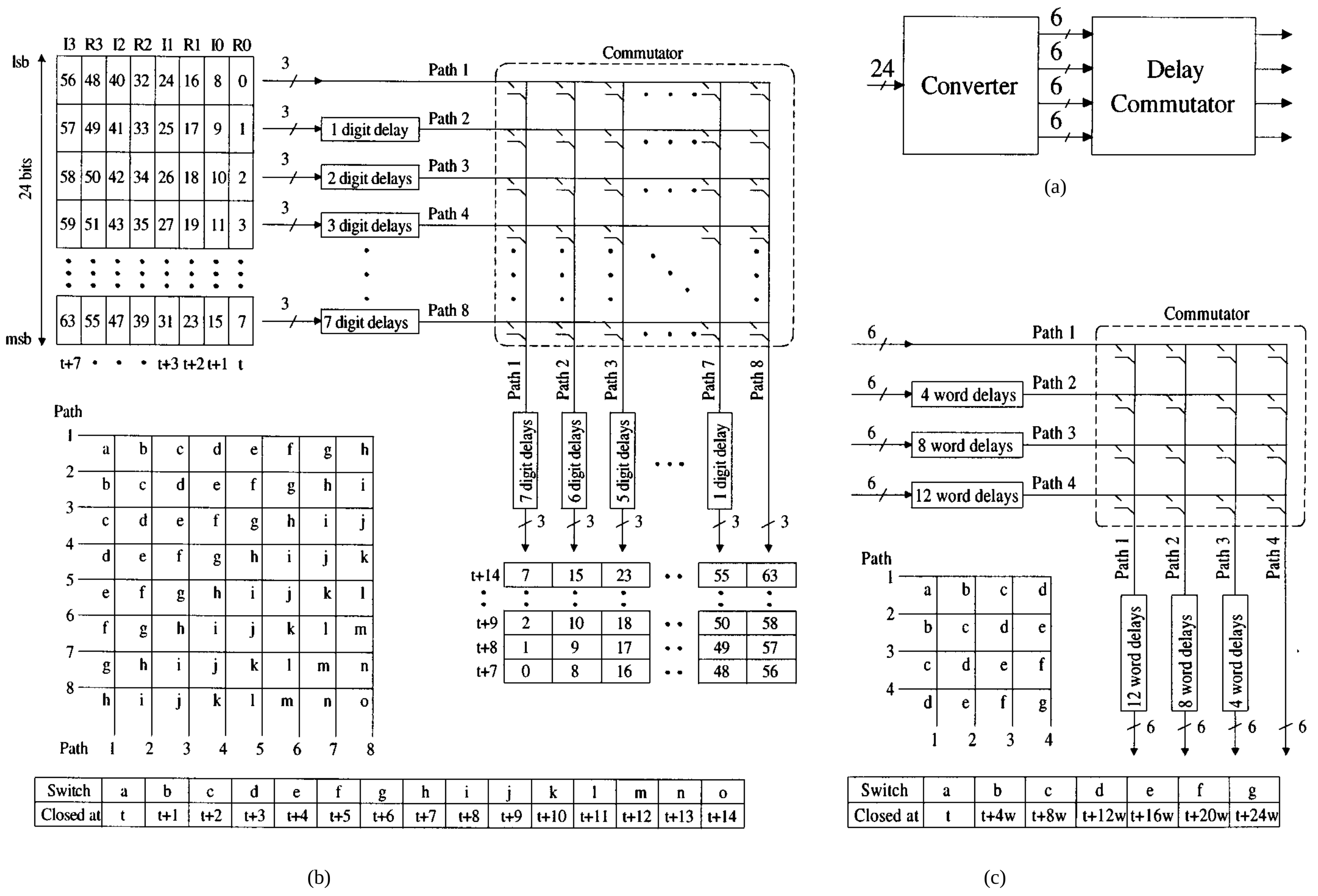


Fig. 6. (a) Input formatter, (b) converter, and (c) delay commutator.

TABLE I
COMPARISON OF PIPELINED FFT ARCHITECTURES

	Gold & Bially	Bi & Jones	New architecture
Pipelined algorithm	Gold-Bially, radix-4	'Single multiplication', radix-4	New algorithm, radix-4
Radix computational element (CE)	butterfly	butterfly	array
No. of complex adders	$8log_4N$ *	$3log_4N$ *	$12log_4N$ **
Complexity per adder	$O(n)$	$O(n)$	$O(d)$
Total complexity of adders	$\{O(24n)\}$	$\{O(9n)\}$	$\{O(9n)\}$
No. of complex multipliers	$3(log_4N-1)$ *	log_4N-1 *	$3(log_4N-1)$ **
Complexity per multiplier	$O(nK)$	$O(nK)$	$O(dK)$
Total complexity of multipliers	$\{O(6nK)\}$	$\{O(2nK)\}$	$\{O(1.5nK)\}$
Memory requirement for data shuffle ***	$n(3.25N-10)$ $\{198n\}$	$n(2.75N-2)$ $\{174n\}$	$n(2.5N+3)$ $\{163n\}$
Wiring overhead	$O(4n)$	$O(n)$	$O(4d)$ $\{O(n)\}$
Computational efficiency	25%	add/sub 100%, mult.75%	100%

* bit-parallel ** digit-serial

*** memory requirement includes input and output data commutation for natural ordering

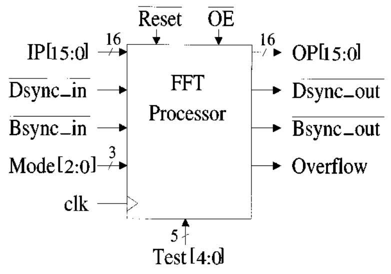


Fig. 7. FFT chip block diagram.

IV. FFT CHIP DESIGN

A. Overall Architecture

A block diagram of the overall chip showing its various input and output interfaces is illustrated in Fig. 7, the chip layout is shown in Fig. 8, and a chip photograph presented in Fig. 9. These interfaces were developed to facilitate the integration of the chip with existing video processing systems and to ensure that no off-chip data permutation or format conversion circuitry is required. A block diagram of the device (including test circuitry) is shown in Fig. 10. The chip is synchronized to the start of the image frame (Dsync) and the image block (Bsync), which means that it can be directly interfaced to a raster scan video stream. This also allows two chips to be cascaded directly to perform a 2-D transform. The reliability and testability of the chip are also enhanced using built-in overflow control and test circuitry. Four optional output modes are available. One of these corresponds to the normal operation of the processor, whereas the others are provided for the purposes of testing.

B. Signal-to-Noise Ratio Optimization

Although a block-floating point format can provide a large dynamic range [19], it is not suited to 2-D transform applications because of the undesirable intermediate normalization which tends to degrade processor accuracy. A fixed-point format has therefore been chosen for its simplicity. The effect of finite arithmetic on the FFT output signal-to-noise ratio (SNR) was investigated [21]. Mathematical error models for various radix-4 FFT configurations were developed and the error analysis was verified through extensive simulations in which real image data was processed using a detailed HDL description of the chip. A dynamic range of 90–96 dB at the FFT output—required in studio DTV applications—means that an output word accuracy of 15–16 b (fixed point) is necessary. This requirement was achieved using 24-b internal precision. Detailed investigations showed that the scaling-after-multiplication, stage-by-stage truncated shifting scheme used produces a better SNR when compared with other alternatives examined. These included overall scaling schemes and stage-by-stage shifting schemes (such as those used in decimation-in-frequency algorithms in which scaling is applied before multiplication).

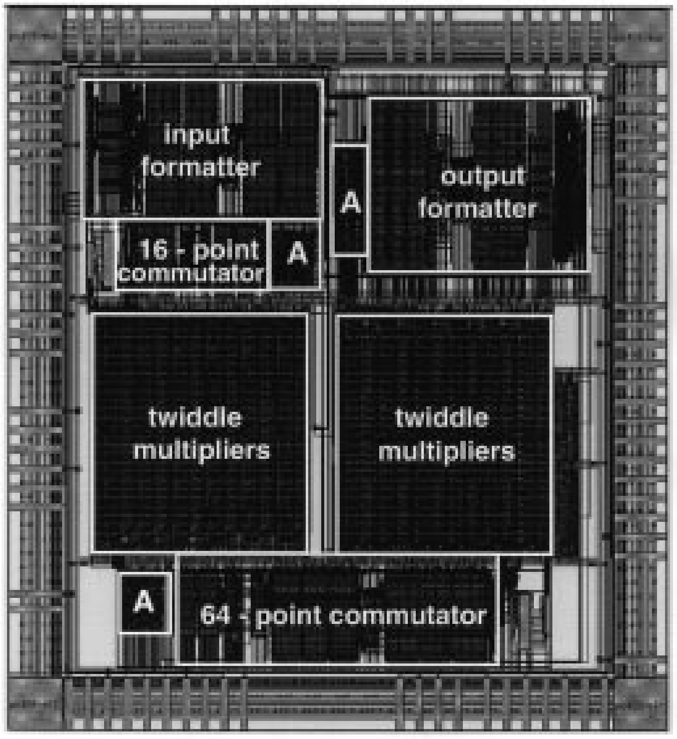


Fig. 8. FFT chip layout.

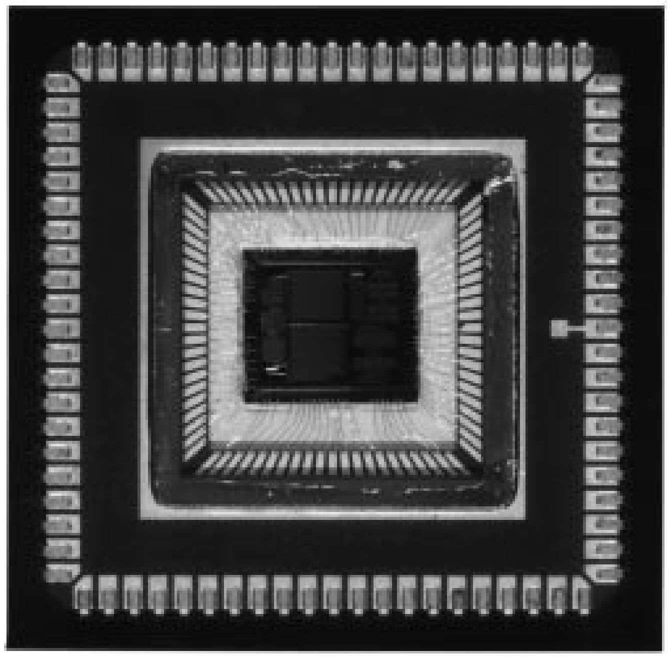


Fig. 9. Photograph of the FFT chip.

C. Overflow Control

To avoid overflow in the computation, each input data value is right shifted by 2 b before entering each R4CA. There are three R4CA's in the processor. The first stage shifting is simply incorporated at the input interface connection without additional hardware, whereas the second and third stage shifting are implemented by inserting a digit-serial shifter at the output of the twiddle multipliers. In addition, when 24-b

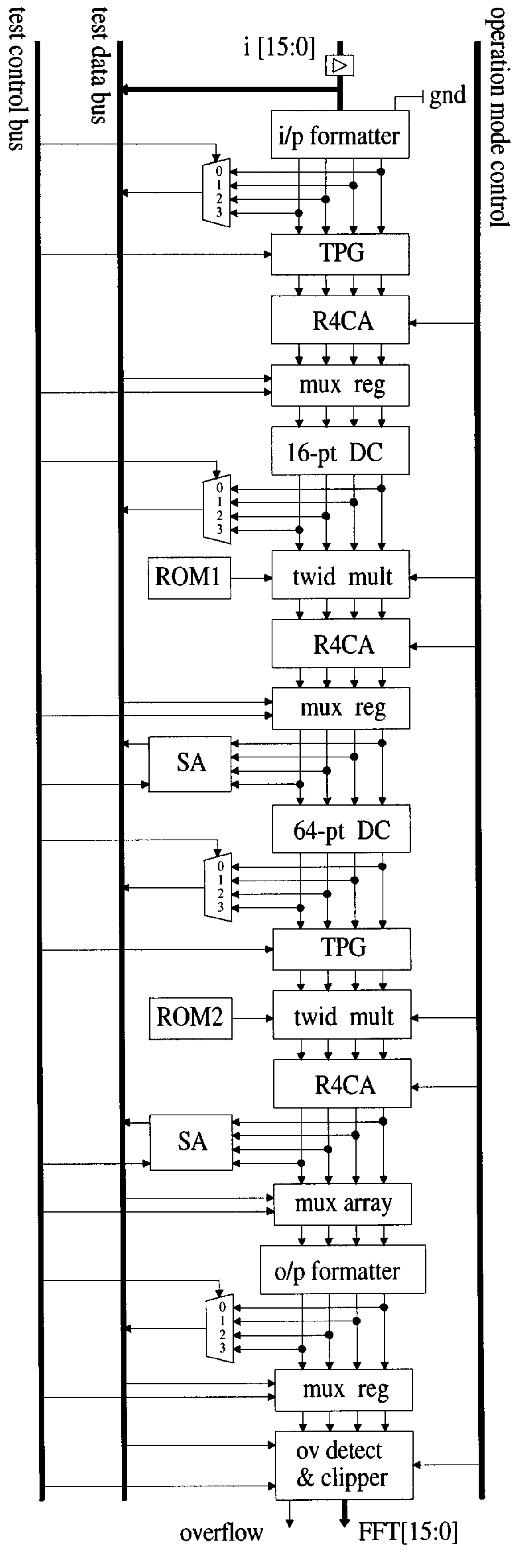


Fig. 10. Detailed block diagram including test circuitry.

TABLE II
CHIP SPECIFICATIONS

	27 MHz Clock		36 MHz Clock	
Function	Sample rate MHz	Time µs	Sample rate MHz	Time µs
64 Complex pt. FFT	13.5	4.74	18	3.56
2×64 Real pt. FFT	27	4.74	36	3.56

Characteristic	Min	Typ	Max	Units
Clock frequency	0	27	40	MHz
Clock high period	8.3			ns
Clock low period	8.3			ns
Clock to output delay		9.2	17	ns
Output hold time	4			ns
Input set-up time	8			ns
Input hold time			0	ns
Reset duration	25			ns

internal data is rounded to 16 b at the output interface, overflow may result. For this, an overflow detector and saturation circuit are incorporated into the chip. Any value less than $-\boldsymbol{P}$ is set to $-\boldsymbol{P}$; whereas, any value larger than $\boldsymbol{P}$ is set to $\boldsymbol{P}$.

D. Test

The inherently linear structure of the architecture used greatly facilitates chip testing. A combination of a bus-oriented ad-hoc technique and built-in self test (BIST) [24] was used to test the device. This was done by partitioning the design into blocks, as shown in Fig. 10. Around 80 K test vectors were sufficient to provide a comprehensive test for the chip. The vectors used include data for setting up the initial test circuitry, as well as actual image data. The latter was also used to test the full dynamic range of the device. The entire testing procedure is controlled by five test control pins, Test[4 : 0]. Multiplexers have been inserted to direct the test outputs to the output pins. Modules such as I/O formatters, delay commutators, the overflow detector and the saturation (clipper) circuitry involve little or no computation, and therefore test patterns for these modules can be applied externally. However, for the computational modules (i.e., the twiddle multipliers and the R4CA's), a BIST technique was used. Both the TPG's and signature analyzers use 25-stage linear feedback shift registers. In addition, the chip has been extensively and successfully tested in a full video phase correlation test system using 200 K test vectors of demanding video sequences. This has been done under a wide range of temperatures (i.e., −20°C to 120°C at 5 V ± (5%), as well as the extremes of timing requirements, detailed in Table II.

A number of FFT chip designs have been developed to date. These include the GEC Plessey Semiconductors PS-DSP16510 FFT processor [24] and a recent 8 K point FFT chip described by Bidet *et al* [25]. A detailed comparison with these devices is complicated by differences in a) transform size; b) wordlengths; c) technology; and d) implementation methods used (e.g., standard cell *v* custom design). However, work undertaken by Hui [21], which attempts to measure the effective number of normalized FFT computations per unit silicon area per watt, indicates that the performance of the new chip is significantly greater than the GEC Plessey Device. It also compares very favorably with the Bidet chip, despite the fact that this is a 0.5-μm CMOS custom design, while that presented is based on a 0.6-μm standard cell CMOS technology. This performance is a direct result of the novel architecture used.

V. Conclusions

A novel radix-4 FFT architecture suitable for real-time video processing applications has been presented. This architecture is based on a direct factorization of the DFT matrix so that the resulting algorithm-to-architecture mapping is well suited to silicon implementation. It exhibits numerous attractive features from a VLSI point of view. These include regularity, modularity, simple wiring, and high data computation rates. The data flow is ideally suited to real-time digital TV/video processing applications.

A new, low-power, high-performance FFT chip for digital television applications has been successfully designed and fabricated based on the architecture described. Its functionality has been successfully verified across a wide range of temperatures and voltages, as well as extremes of timing requirements. It has also been used in the implementation of prototype systems and shown to be fully operational. The chip includes all necessary data and coefficient storage elements to interface directly with video signals so that no off-chip permutation of data is required. The output accuracy of the chip results in a signal-to-noise ratio in excess of 90 dB. If measured in terms of direct DFT computational requirements, the device performs the equivalent of 3.5 billion multiplications and additions per second, yet dissipates only 1 W when clocked at 36 MHz. This, coupled with the low I/O requirements of the architecture used, allows it to be housed in an inexpensive 84-pin PLCC plastic package. The new chip has resulted in a considerable reduction in cost, size, and power consumption of existing phase correlation hardware where the real-time computation of 2-D Fourier transforms represent the main computational bottleneck. The work undertaken demonstrates how combining higher level systems requirements with algorithm-to-architecture methods and advanced DSP chip design can produce efficient silicon solutions for very high quality video processing systems.

VI. Appendix

In calculating a 64-point DFT using the decimation-in-time-and-frequency algorithm, the time (m) and frequency (k) sequences of the transform and hence the transform matrix, are first permuted into a base-4, digit-reversed order using (5)

$$\begin{aligned} x(m) &= x(m_1 + 4m_2 + 16m_3) \\ X(k) &= X(k_1 + 4k_2 + 16k_3) \\ &\begin{cases} m_1 = 0,1,2,3 & k_1 = 0,1,2,3 \\ m_2 = 0,1,2,3 & k_2 = 0,1,2,3. \\ m_3 = 0,1,2,3 & k_3 = 0,1,2,3 \end{cases} \end{aligned} \tag{5}$$

From (6), shown at the bottom of the page, the permuted 64-by-64 matrix, which is not appropriate to be shown explicitly here, can be partitioned into four 16-by-16 matrices as illustrated in (7)–(10), shown on this and the following page

$$\begin{bmatrix} a_0 & a_1 & a_2 & a_3 & a_4 & a_5 & a_6 & a_7 & a_8 & a_9 & a_{10} & a_{11} & a_{12} & a_{13} & a_{14} & a_{15} \\ b_0 & b_1 & b_2 & b_3 & b_4 & b_5 & b_6 & b_7 & b_8 & b_9 & b_{10} & b_{11} & b_{12} & b_{13} & b_{14} & b_{15} \\ c_0 & c_1 & c_2 & c_3 & c_4 & c_5 & c_6 & c_7 & c_8 & c_9 & c_{10} & c_{11} & c_{12} & c_{13} & c_{14} & c_{15} \\ d_0 & d_1 & d_2 & d_3 & d_4 & d_5 & d_6 & d_7 & d_8 & d_d & d_{10} & d_{11} & d_{12} & d_{13} & d_{14} & d_{15} \end{bmatrix}$$
$$= \begin{bmatrix} 1 & 1 & 1 & 1 \\ 1 & -j & -1 & j \\ 1 & -1 & 1 & -1 \\ 1 & j & -1 & -j \end{bmatrix} \begin{bmatrix} x_0 & x_4 & x_8 & x_{12} & x_1 & x_5 & x_9 & x_{13} & x_2 & x_6 & x_{10} & x_{14} & x_3 & x_7 & x_{11} & x_{15} \\ x_{16} & x_{20} & x_{24} & x_{28} & x_{17} & x_{21} & x_{25} & x_{29} & x_{18} & x_{22} & x_{26} & x_{30} & x_{19} & x_{23} & x_{27} & x_{31} \\ x_{32} & x_{36} & x_{40} & x_{44} & x_{33} & x_{37} & x_{41} & x_{45} & x_{34} & x_{38} & x_{42} & x_{46} & x_{35} & x_{39} & x_{43} & x_{47} \\ x_{48} & x_{52} & x_{56} & x_{60} & x_{49} & x_{53} & x_{57} & x_{61} & x_{50} & x_{54} & x_{58} & x_{62} & x_{51} & x_{55} & x_{59} & x_{63} \end{bmatrix} \tag{6}$$

$$\begin{bmatrix} X_0 \\ X_{16} \\ X_{32} \\ X_{48} \\ X_4 \\ X_{20} \\ X_{36} \\ X_{52} \\ X_8 \\ X_{24} \\ X_{40} \\ X_{56} \\ X_{12} \\ X_{28} \\ X_{44} \\ X_{60} \end{bmatrix} = \begin{bmatrix} \begin{bmatrix} 1 \\ 1 \\ 1 \\ 1 \end{bmatrix}_D [E] & \begin{bmatrix} 1 \\ -j \\ -1 \\ j \end{bmatrix}_D [E] & \begin{bmatrix} 1 \\ -1 \\ 1 \\ -1 \end{bmatrix}_D [E] & \begin{bmatrix} 1 \\ j \\ -1 \\ -j \end{bmatrix}_D [E] \\ \begin{bmatrix} 1 \\ 1 \\ 1 \\ 1 \end{bmatrix}_D [F] & \begin{bmatrix} W^4 \\ -jW^4 \\ -W^4 \\ jW^4 \end{bmatrix}_D [F] & \begin{bmatrix} W^8 \\ -W^8 \\ W^8 \\ -W^8 \end{bmatrix}_D [F] & \begin{bmatrix} W^{12} \\ jW^{12} \\ -W^{12} \\ -jW^{12} \end{bmatrix}_D [F] \\ \begin{bmatrix} 1 \\ 1 \\ 1 \\ 1 \end{bmatrix}_D [G] & \begin{bmatrix} W^8 \\ -jW^8 \\ -W^8 \\ jW^8 \end{bmatrix}_D [G] & \begin{bmatrix} -j \\ j \\ -j \\ j \end{bmatrix}_D [G] & \begin{bmatrix} W^{24} \\ jW^{24} \\ -W^{24} \\ -jW^{24} \end{bmatrix}_D [G] \\ \begin{bmatrix} 1 \\ 1 \\ 1 \\ 1 \end{bmatrix}_D [H] & \begin{bmatrix} W^{12} \\ -jW^{12} \\ -W^{12} \\ jW^{12} \end{bmatrix}_D [H] & \begin{bmatrix} W^{24} \\ -W^{24} \\ W^{24} \\ -W^{24} \end{bmatrix}_D [H] & \begin{bmatrix} W^{36} \\ jW^{36} \\ -W^{36} \\ -jW^{36} \end{bmatrix}_D [H] \end{bmatrix} \begin{bmatrix} a_0 \\ a_1 \\ a_2 \\ a_3 \\ a_4 \\ a_5 \\ a_6 \\ a_7 \\ a_8 \\ a_9 \\ a_{10} \\ a_{11} \\ a_{12} \\ a_{13} \\ a_{14} \\ a_{15} \end{bmatrix} \tag{7}$$

where

$$[E] = \begin{bmatrix}1&1&1&1\\1&1&1&1\\1&1&1&1\\1&1&1&1\end{bmatrix}, \quad [F] = \begin{bmatrix}1&-j&-1&j\\1&-j&-1&j\\1&-j&-1&j\\1&-j&-1&j\end{bmatrix},$$

$$[G] = \begin{bmatrix}1&-1&1&-1\\1&-1&1&-1\\1&-1&1&-1\\1&-1&1&-1\end{bmatrix}, \quad [H] = \begin{bmatrix}1&j&-1&-j\\1&j&-1&-j\\1&j&-1&-j\\1&j&-1&-j\end{bmatrix}$$

and

$$\begin{bmatrix}d_1\\d_2\\d_3\\d_4\end{bmatrix}_D \equiv \begin{bmatrix}d_1&0&0&0\\0&d_2&0&0\\0&0&d_3&0\\0&0&0&d_4\end{bmatrix}.$$

Again, since each of the rows within each of the individual matrices $\boldsymbol{E}$, $\boldsymbol{F}$, $\boldsymbol{G}$, and $\boldsymbol{H}$ are the same, the multiplications of these matrices with the appropriate four elements from the intermediate vectors $\boldsymbol{a}$, $\boldsymbol{b}$, $\boldsymbol{c}$, and $\boldsymbol{d}$ in (7)–(10) can effectively be calculated using (11), shown at the top of the next page.

$$\begin{bmatrix}X_1\\X_{17}\\X_{33}\\X_{49}\\X_5\\X_{21}\\X_{37}\\X_{53}\\X_9\\X_{25}\\X_{41}\\C_{57}\\X_{13}\\X_{29}\\X_{45}\\X_{61}\end{bmatrix} = \begin{bmatrix}
\begin{bmatrix}1\\1\\1\\1\end{bmatrix}_D & [E] & \begin{bmatrix}W\\-jW\\-W\\jW\end{bmatrix}_D & [E] & \begin{bmatrix}W^2\\-W^2\\W^2\\-W^2\end{bmatrix}_D & [E] & \begin{bmatrix}W^3\\jW^3\\-W^3\\-jW^3\end{bmatrix}_D & [E] \\
\begin{bmatrix}1\\1\\1\\1\end{bmatrix}_D & [F] & \begin{bmatrix}W^5\\-jW^5\\-W^5\\jW^5\end{bmatrix}_D & [F] & \begin{bmatrix}W^{10}\\-W^{10}\\W^{10}\\-W^{10}\end{bmatrix}_D & [F] & \begin{bmatrix}W^{15}\\jW^{15}\\-W^{15}\\-jW^{15}\end{bmatrix}_D & [F] \\
\begin{bmatrix}1\\1\\1\\1\end{bmatrix}_D & [G] & \begin{bmatrix}W^9\\-jW^9\\-W^9\\jW^9\end{bmatrix}_D & [G] & \begin{bmatrix}W^{18}\\-W^{18}\\W^{18}\\-W^{18}\end{bmatrix}_D & [G] & \begin{bmatrix}W^{27}\\jW^{27}\\-W^{27}\\-jW^{27}\end{bmatrix}_D & [G] \\
\begin{bmatrix}1\\1\\1\\1\end{bmatrix}_D & [H] & \begin{bmatrix}W^{13}\\-jW^{13}\\-W^{13}\\jW^{13}\end{bmatrix}_D & [H] & \begin{bmatrix}W^{26}\\-W^{26}\\W^{26}\\-W^{26}\end{bmatrix}_D & [H] & \begin{bmatrix}W^{39}\\jW^{39}\\-W^{39}\\-jW^{39}\end{bmatrix}_D & [H]
\end{bmatrix} \begin{bmatrix}b_0\\W^4b_1\\W^8b_2\\W^{12}b_3\\b_4\\W^4b_5\\W^8b_6\\W^{12}b_7\\b_8\\W^4b_9\\W^8b_{10}\\W^{12}b_{11}\\b_{12}\\W^4b_{13}\\W^8b_{14}\\W^{12}b_{15}\end{bmatrix} \tag{8}$$

$$\begin{bmatrix}X_2\\X_{18}\\X_{34}\\X_{50}\\X_6\\X_{22}\\X_{38}\\X_{54}\\X_{10}\\X_{26}\\X_{42}\\X_{58}\\X_{14}\\X_{30}\\X_{46}\\X_{62}\end{bmatrix} = \begin{bmatrix}
\begin{bmatrix}1\\1\\1\\1\end{bmatrix}_D & [E] & \begin{bmatrix}W^2\\-jW^2\\-W^2\\jW^2\end{bmatrix}_D & [E] & \begin{bmatrix}W^4\\-W^4\\W^4\\-W^4\end{bmatrix}_D & [E] & \begin{bmatrix}W^6\\jW^6\\-W^6\\-jW^6\end{bmatrix}_D & [E] \\
\begin{bmatrix}1\\1\\1\\1\end{bmatrix}_D & [F] & \begin{bmatrix}W^6\\-jW^6\\-W^6\\jW^6\end{bmatrix}_D & [F] & \begin{bmatrix}W^{12}\\-W^{12}\\W^{12}\\-W^{12}\end{bmatrix}_D & [F] & \begin{bmatrix}W^{18}\\jW^{18}\\-W^{18}\\-jW^{18}\end{bmatrix}_D & [F] \\
\begin{bmatrix}1\\1\\1\\1\end{bmatrix}_D & [G] & \begin{bmatrix}W^{10}\\-jW^{10}\\-W^{10}\\jW^{10}\end{bmatrix}_D & [G] & \begin{bmatrix}W^{20}\\-W^{20}\\W^{20}\\-W^{20}\end{bmatrix}_D & [G] & \begin{bmatrix}W^{30}\\jW^{30}\\-W^{30}\\-jW^{30}\end{bmatrix}_D & [G] \\
\begin{bmatrix}1\\1\\1\\1\end{bmatrix}_D & [H] & \begin{bmatrix}W^{14}\\-jW^{14}\\-W^{14}\\jW^{14}\end{bmatrix}_D & [H] & \begin{bmatrix}W^{28}\\-W^{28}\\W^{28}\\-W^{28}\end{bmatrix}_D & [H] & \begin{bmatrix}W^{42}\\jW^{42}\\-W^{42}\\-jW^{42}\end{bmatrix}_D & [H]
\end{bmatrix} \begin{bmatrix}c_0\\W^8c_1\\W^{16}c_2\\W^{24}c_3\\c_4\\W^8c_5\\W^{16}c_6\\W^{24}c_7\\c_8\\W^8c_9\\W^{16}c_{10}\\W^{24}c_{11}\\c_{12}\\W^8c_{13}\\W^{16}c_{14}\\W^{24}c_{15}\end{bmatrix} \tag{9}$$

$$\begin{bmatrix}X_3\\X_{19}\\X_{35}\\X_{51}\\X_7\\X_{23}\\X_{39}\\X_{55}\\X_{11}\\X_{27}\\X_{43}\\X_{59}\\X_{15}\\X_{31}\\X_{47}\\X_{63}\end{bmatrix} = \begin{bmatrix}
\begin{bmatrix}1\\1\\1\\1\end{bmatrix}_D & [E] & \begin{bmatrix}W^3\\-jW^3\\-W^3\\jW^3\end{bmatrix}_D & [E] & \begin{bmatrix}W^6\\-W^6\\W^6\\-W^6\end{bmatrix}_D & [E] & \begin{bmatrix}W^9\\jW^9\\-W^9\\-jW^9\end{bmatrix}_D & [E] \\
\begin{bmatrix}1\\1\\1\\1\end{bmatrix}_D & [F] & \begin{bmatrix}W^7\\-jW^7\\-W^7\\jW^7\end{bmatrix}_D & [F] & \begin{bmatrix}W^{14}\\-W^{14}\\W^{14}\\-W^{14}\end{bmatrix}_D & [F] & \begin{bmatrix}W^{21}\\jW^{21}\\-W^{21}\\-jW^{21}\end{bmatrix}_D & [F] \\
\begin{bmatrix}1\\1\\1\\1\end{bmatrix}_D & [G] & \begin{bmatrix}W^{11}\\-jW^{11}\\-W^{11}\\jW^{11}\end{bmatrix}_D & [G] & \begin{bmatrix}W^{22}\\-W^{22}\\W^{22}\\-W^{22}\end{bmatrix}_D & [G] & \begin{bmatrix}W^{33}\\jW^{33}\\-W^{33}\\-jW^{33}\end{bmatrix}_D & [G] \\
\begin{bmatrix}1\\1\\1\\1\end{bmatrix}_D & [H] & \begin{bmatrix}W^{15}\\-jW^{15}\\-W^{15}\\jW^{15}\end{bmatrix}_D & [H] & \begin{bmatrix}W^{30}\\-W^{30}\\W^{30}\\-W^{30}\end{bmatrix}_D & [H] & \begin{bmatrix}W^{45}\\jW^{45}\\-W^{45}\\-jW^{45}\end{bmatrix}_D & [H]
\end{bmatrix} \begin{bmatrix}d_0\\W^{12}d_1\\W^{24}d_2\\W^{36}d_3\\d_4\\W^{12}d_5\\W^{24}d_6\\W^{36}d_7\\d_8\\W^{12}d_9\\W^{24}d_{10}\\W^{36}d_{11}\\d_{12}\\W^{12}d_{13}\\W^{24}d_{14}\\W^{36}d_{15}\end{bmatrix} \tag{10}$$

$$\begin{bmatrix} e_0 & e_1 & e_2 & e_3 & e_4 & e_5 & e_6 & e_7 & e_8 & e_9 & e_{10} & e_{11} & e_{12} & e_{13} & e_{14} & e_{15} \\ f_0 & f_1 & f_2 & f_3 & f_4 & f_5 & f_6 & f_7 & f_8 & f_9 & f_{10} & f_{11} & f_{12} & f_{13} & f_{14} & f_{15} \\ g_0 & g_1 & g_2 & g_3 & g_4 & g_5 & g_6 & g_7 & g_8 & g_9 & g_{10} & g_{11} & g_{12} & g_{13} & g_{14} & g_{15} \\ h_0 & h_1 & h_2 & h_3 & h_4 & h_5 & h_6 & h_7 & h_8 & h_9 & h_{10} & h_{11} & h_{12} & h_{13} & h_{14} & h_{15} \end{bmatrix} = \begin{bmatrix} 1 & 1 & 1 & 1 \\ 1 & -j & -1 & j \\ 1 & -1 & 1 & -1 \\ 1 & j & -1 & -j \end{bmatrix}$$
$$\cdot \begin{bmatrix} a_0 & a_4 & a_8 & a_{12} & b_0 & b_4 & b_8 & b_{12} & c_0 & c_4 & c_8 & c_{12} & d_0 & d_4 & d_8 & d_{12} \\ a_1 & a_5 & a_9 & a_{113} & W^4 b_1 & W^4 b_5 & W^4 b_9 & W^4 b_{13} & W^8 c_1 & W^8 c_5 & W^8 c_9 & W^8 c_{13} & W^{12} d_1 & W^{12} d_5 & W^{12} d_9 & W^{12} d_{13} \\ a_2 & a_6 & a_{10} & a_{14} & W^8 b_2 & W^8 b_6 & W^8 b_{10} & W^8 b_{14} & W^{16} c_2 & W^{16} c_6 & W^{16} c_{10} & W^{16} c_{14} & W^{24} d_2 & W^{24} d_6 & W^{24} d_{10} & W^{24} d_{14} \\ a_3 & a_7 & a_{11} & a_{15} & W^{12} b_3 & W^{12} b_7 & W^{12} b_{11} & W^{12} b_{15} & W^{24} c_3 & W^{24} c_7 & W^{24} c_{11} & W^{24} c_{15} & W^{36} d_3 & W^{36} d_7 & W^{36} d_{11} & W^{36} d_{15} \end{bmatrix} \quad (11)$$

$$\begin{bmatrix} X_0 & X_4 & X_8 & X_{12} & X_1 & X_5 & X_9 & X_{13} & X_2 & X_6 & X_{10} & X_{14} & X_3 & X_7 & X_{11} & X_{15} \\ X_{16} & X_{20} & X_{24} & X_{28} & X_{17} & X_{21} & X_{25} & X_{29} & X_{18} & X_{22} & X_{26} & X_{30} & X_{19} & X_{23} & X_{27} & X_{31} \\ X_{32} & X_{36} & X_{40} & X_{44} & X_{33} & X_{37} & X_{41} & X_{45} & X_{34} & X_{38} & X_{42} & X_{46} & X_{35} & X_{39} & X_{43} & X_{47} \\ X_{48} & X_{52} & X_{56} & X_{60} & X_{49} & X_{53} & X_{57} & X_{61} & X_{50} & X_{54} & X_{58} & X_{62} & X_{51} & X_{55} & X_{59} & X_{63} \end{bmatrix} = \begin{bmatrix} 1 & 1 & 1 & 1 \\ 1 & -j & -1 & j \\ 1 & -1 & 1 & -1 \\ 1 & j & -1 & -j \end{bmatrix}$$
$$\cdot \begin{bmatrix} e_0 & f_0 & g_0 & h_0 & e_4 & f_4 & g_4 & h_4 & e_8 & f_8 & g_8 & h_8 & e_{12} & f_{12} & g_{12} & h_{12} \\ e_1 & W^4 f_1 & W^8 g_1 & W^{12} h_1 & W e_5 & W^5 f_5 & W^9 g_5 & W^{13} h_5 & W^2 e_9 & W^6 f_9 & W^{10} g_9 & W^{14} h_9 & W^3 e_{13} & W^7 f_{13} & W^{11} g_{13} & W^{15} h_{13} \\ e_2 & W^8 f_2 & W^{16} g_2 & W^{24} h_2 & W^2 e_6 & W^{10} f_6 & W^{18} g_6 & W^{26} h_6 & W^4 e_{10} & W^{12} f_{10} & W^{20} g_{10} & W^{28} h_{10} & W^6 e_{14} & W^{14} f_{14} & W^{22} g_{14} & W^{30} h_{14} \\ e_3 & W^{12} f_3 & W^{24} g_3 & W^{36} h_3 & W^3 e_7 & W^{15} f_7 & W^{27} g_7 & W^{39} h_7 & W^6 e_{11} & W^{18} f_{11} & W^{30} g_{11} & W^{42} h_{11} & W^9 e_{15} & W^{21} f_{15} & W^{33} g_{15} & W^{45} h_{15} \end{bmatrix} \quad (12)$$

Substituting (11) into (7)–(10), the 64-point Fourier transform is hence given as (12), shown at the top of the page.

References

[1] G. Thomas, "Motion estimation and its application to HDTV transmission and up-conversion using DATV," *SMPTE J.*, pp. 987–992, 1990.

[2] ——, "Application of motion compensation to video processing in the TV studio," in *IEE Colloquium on Video Processing in the TV Studio*, 1991, vol. 148, pp. 5/1–5/4.

[3] J. Watkinson, *The Engineer's Guide to Motion Compensation*. Snell and Wilcox, 1994.

[4] T. Borer, "Motion estimation," Television Standards Conversion, University of Surrey, 1992, Ph.D. thesis, ch. 7.

[5] C. Chen and L. Wang, "A new efficient systolic architecture for the 2-D discrete Fourier-transform," *IEEE Int. Symp. Circuits and Systems*, 1992, vol. 6, ch. 732, pp. 689–692.

[6] M. Ghouse, "2D grid architectures for the DFT and the 2D DFT," *J. VLSI Signal Processing*, vol. 5, pp. 57–74, 1993.

[7] H. Lim and E. Swartzlander, "A systolic array for 2-D DFT and 2-D DCT," in *Int. Conf. Application Specific Array Processors*, 1994, pp. 123–131.

[8] H. Yeh, "Highly pipelined VLSI architecture for computation of fast Fourier transforms," in *Proc. SPIE—Int. Society Optical Eng.*, 1993, vol. 2027, pp. 112–121.

[9] V. Di Lecce and E. Di Sciascio, "A Compact Bit-Serial Array Machine for DSP," in *EPE"91 4th European Conf. Power Electronics and Applications*, 1991, vol. 3, pp. 527–531.

[10] K. Liu and K. Yao, "On uniform one-chip VLSI design considerations for some discrete orthogonal transforms," in *ICASSP*, 1988, pp. 2136–2139.

[11] A. Vacher *et al.*, "A VLSI implementation of parallel fast Fourier transform," in *Proc. European Design and Test Conf.*, 1994, pp. 250–255.

[12] N. Zhang, X. Zhang, and C. Zhang, "Two kinds of array structure for FFT processor," in *Proc. Int. Conf. Signal Processing*, 1990, vol. 2, ch. 355, pp. 1161–1162.

[13] J. Muller and D. Trystram, "Perfect shuffle VLSI implementation: Applications to sorting and FFT computation," in *CONPAR 88*, 1989, pp. 197–204.

[14] B. Gold and T. Bially, "Parallelism in fast Fourier transform hardware," *IEEE Trans. Audio Electroacoustics*, vol. 21, no. 1, pp. 5–16, 1973.

[15] L. Rabiner and B. Gold, *Theory and Application of Digit Signal Processing*. Englewood Cliffs, NJ: Prentice-Hall, 1975.

[16] E. Swartzlander, W. Young, and S. Joseph, "A radix 4 delay commutator for fast Fourier transform processor implementation," *IEEE J. Solid-State Circuits*, vol. 19, no. 5, pp. 702–709, 1984.

[17] E. Swartzlander, V. Jain, and H. Hikawa, "A radix-8 wafer scale FFT processor," *J. VLSI Signal Processing*, vol. 4, pp. 165–176, 1992.

[18] V. Szwarc and L. Desormeaux, "A chip set for pipeline and parallel pipeline FFT architectures," *J. VLSI Signal Processing*, vol. 8, pp. 253–265, 1994.

[19] A. Oppenheim and R. Schafer, *Discrete-Time Signal Processing* Englewood Cliffs, NJ: Prentice-Hall International, 1989.

[20] J. Cooley and J. Tukey, "An algorithm for the machine calculation of complex Fourier series," *Mathematics of Computation*, vol. 19, pp. 297–301, 1965.

[21] C. C. W. Hui, "VLSI architectures for digital television applications," Ph.D. Thesis, Department of Electrical and Electronic Engineering, The Queen's University of Belfast, Nov. 1995.

[22] G. Bi and E. Jones, "A pipelined FFT processor for word-sequential data," *IEEE Trans. Acoustics, Speech, Signal Processing*, vol. 37, no. 12, pp. 1982–1985, 1989.

[23] Plessey PSDSP16510 Stand alone FFT Processor Data Sheet, Aug., 1990.

[24] N. Weste and K. Eshraghian, *Principles of CMOS VLSI Design: A Systems Perspective*. Reading, MA: Addison-Wesley, 2nd ed., 1993.

[25] E. Bidet, D. Castelain, C. Joanblanq, and P. Senn, "A fast single-chip implementation of 8192 complex point FFT," *IEEE J. Solid-State Circuits*, vol. 30, no. 3, Mar. 1995.

Colin Chiu Wing Hui graduated from Queen's University Belfast with a first class honors degree in electrical and electronic engineering in 1991 and obtained the Ph.D. from the same university in 1996.

From 1991 to 1996 he worked as a Research Assistant in the VLSI Signal Processing Research Group at Queen's University where his main interest was on VLSI architectures and advanced chip design for digital television applications. He joined Nortel (N. Ireland) in 1996 where he is currently working as a Hardware Development Engineer. His current interests include broadband network development and synchronous transmission systems.

Tiong Jiu Ding received a first class B.Eng. degree in electrical and electronic engineering from Queen's University of Belfast in 1993.

Since then, he has been a research assistant in the VLSI Signal Processing Research Group at Queens University, where he is also currently pursuing a Ph.D. degree. His main research interests are VLSI architectures and VLSI systems design for TV and video applications.

John V. McCanny (M'89–SM'95) holds a personal Chair in Microelectronics Engineering at the Queen's University of Belfast, where he leads a broad based digital signal processing and telecommunications research group. His main research interests include algorithms, VLSI architectures and design methodologies for application specific DSP chips. He has published over 130 scientific papers, holds ten patents and has edited several books in the field of VLSI architectures and design. He is also a director of Audio Processing Technology Ltd. and Integrated Silicon Systems Ltd., two spin-off companies which he helped to found.

Dr. McCanny is a Fellow of the Royal Academy of Engineering, a Fellow of the Institution of Electrical Engineers, and a Fellow of the Institute of Physics. In 1996, he was awarded the Royal Academy of Engineering's Silver Medal for "Outstanding Contributions to UK Engineering."

Roger F. Woods (M'95) obtained the B.Sc. and Ph.D. from the Queen's University of Belfast in 1985 and 1990, respectively

He is a senior lecturer in the School of Electrical Engineering and Computer Science at the Queen's University of Belfast. His main research interests are in the areas of VLSI architectures for digital signal processing applications, including digital filtering and numerical processors, VLSI architectural synthesis, and the use of field programmable gate arrays in custom computing applications. He has published over 50 papers and holds two patents.

Dr Woods is a member of the IEE.