mum product and division errors occur as percentages of the operands, these algorithms are suited for high-speed hardware rather than GP computer applications.

One such application is real time digital filtering. The computations involved are usually sums of products of a variable and constant coefficients. Using the log-antilog algorithms gives complete freedom of coefficient selection. For single multiplications a cobweb array multiplier is simpler. However, for other configurations, such as a parallel digital filter bank, the log-antilog technique is less complex. Also, there are applications, such as multiplicative digital filters, where log and exponent conversions are necessary.

References

[1] J. N. Mitchell, Jr., "Computer multiplication and division using binary logarithms," *IRE Trans. Electronic Computers*, vol. EC-11, pp. 512–517, August 1962.

[2] M. Combet, H. Van Zonneveld, and L. Verbeek, "Computation of the base two logarithm of binary numbers," *IEEE Trans. Electronic Computers*, vol. EC-14, pp. 863–867, December 1965.

[3] E. L. Hall, D. D. Lynch, and R. E. Young, "A digital modified discrete Fourier transform Doppler radar processor," *1968 EASCON Rec.*, pp. 150–159.

[4] J. F. Kaiser and F. Kuo, *System Analysis by Digital Computer*. New York: Wiley, 1966, pp. 218–277.

[5] C. M. Rader and B. Gold, "Digital filter design techniques in the frequency domain," *Proc. IEEE*, vol. 55, pp. 149–171, February 1967.

[6] A. V. Oppenheim, R. W. Schafer, and T. G. Stockham, Jr., "Nonlinear filtering of multiplied and convolved signals," *Proc. IEEE*, vol. 56, pp. 1264–1291, August 1968.

A Generalization of the Fast Fourier Transform

J. A. GLASSMAN

***Abstract*—A procedure for factoring of the $N \times N$ matrix representing the discrete Fourier transform is presented which does not produce shuffled data. Exactly one factor is produced for each factor of N, resulting in a fast Fourier transform valid for any N. The factoring algorithm enables the fast Fourier transform to be implemented in general with four nested loops, and with three loops if N is a power of two. No special logical organization, such as binary indexing, is required to unshuffle data. Included are two sample programs, one which writes the equations of the matrix factors employing the four key loops, and one which implements the algorithm in a fast Fourier transform for N a power of two. The algorithm is shown to be most efficient for N a power of two.**

***Index Terms*—Cooley-Tukey algorithm, discrete Fourier transform, fast Fourier transform, mixed radix, spectral analysis.**

THE fast Fourier transform, extensively covered in current journals [1]–[7] and popularly termed the Cooley–Tukey algorithm [4], can be generalized for any number of coefficients N by a factoring which does not shuffle the data. This factoring has three major effects on the fast Fourier transform. First, the transform is more easily explained, since the complexity of the tree graph to trace data is eliminated. Second, the mechanization is simplified since the final data need not be unshuffled, and third, the fast Fourier transform may be conveniently applied to any number of coefficients, although an application to anything but a power of two or four may be only of academic interest. The application of the fast Fourier transforms to any N, which may be termed a mixed radix algorithm, has appeared in recent articles [7]–[9] but no other algorithm has been found which does not involve scrambled data. This paper develops the basic matrix, demonstrates the algorithm for its factorization, and illustrates the factoring and the resulting fast Fourier transform with programs for time-sharing operation.

Manuscript received January 16, 1969; revised August 4, 1969.
The author is with Hughes Aircraft Company, Canoga Park, Calif.

The Discrete Fourier Transform

The Fourier transform $Y(w)$ of a function $x(t)$ is defined by the relation

$$Y(w) = \int_{-\infty}^{\infty} x(t)e^{-jwt}dt. \tag{1}$$

If $x(t)$ is sampled μ times, a sampled function $X^*(t)$ is produced, defined by

$$X^*(t) = \sum_{k=0}^{\mu-1} x(t)\delta(t - kT) \tag{2}$$

where $\delta(t)$ is the Dirac delta function. The Fourier transform of $X^*(t)$ is the discrete Fourier transform $Y^*(w)$:

$$\begin{aligned} Y^*(w) &= \int_{-\infty}^{\infty} x^*(t)e^{-jwt}dt \\ &= \int_{-\infty}^{\infty} \sum_{k=0}^{\mu-1} x(t)\delta(t - kT)e^{-jwt}dt \\ Y^*(w) &= \sum_{k=0}^{\mu-1} x(kT)e^{-jwkT}. \end{aligned} \tag{3}$$

If $Y^*(w)$ is expressed at frequencies which are integer multiples n of $1/NT$, and if μ is selected as some multiple m of N, the transform becomes

$$Y^*\left(\frac{n2\pi}{NT}\right) = \sum_{k=0}^{mN-1} x(kT)e^{-j(2\pi nk/N)} \tag{4}$$

which may be expressed as

$$Y^*\left(\frac{n2\pi}{NT}\right) = \sum_{k=0}^{N-1}\left(\sum_{i=0}^{m-1} x((k+iN)T)\right)e^{-j(2\pi nk/N)} \tag{5}$$

using the periodic property of the exponential function. For simplicity, the inner summation may be replaced by a derived function $x'(t)$, so that

$$Y^*\left(\frac{n2\pi}{NT}\right) = \sum_{k=0}^{N-1} x'(kT)e^{-j(2\pi nk/N)}. \tag{6}$$

The function of $x'(t)$ is a folding of the function $x(t)$, implied by (5). If $\mu = N$, $x'(t)$ is identically $x(t)$. If $x(t)$ is periodic with period NT and if it is sampled without noise, no advantage accrues to folding.

If N Fourier coefficients $n=0$ to $N-1$ are computed, the array of N coefficients may be expressed by

$$Y^* = M \cdot X \tag{7}$$

where

$$Y^* = \begin{bmatrix} Y^*(0) \\ Y^*(2\pi/NT) \\ Y^*(4\pi/NT) \\ \vdots \\ Y^*((N-1)2\pi/NT) \end{bmatrix}, \tag{8}$$

$$X = \begin{bmatrix} x'(0) \\ x'(T) \\ x'(2T) \\ \vdots \\ x'((N-1)T) \end{bmatrix}, \tag{9}$$

and M is the $N \times N$ symmetric matrix which, with $W = e^{-j2\pi/N}$, is given by

$$M = \begin{bmatrix} 1 & 1 & 1 & \cdots & 1 \\ 1 & W^1 & W^2 & \cdots & W^{N-1} \\ 1 & W^2 & W^4 & \cdots & W^{N-2} \\ & & & \vdots & \\ 1 & W^{N-1} & W^{N-2} & \cdots & W^1 \end{bmatrix} \tag{10}$$

for which the periodic property of the exponential has again been employed.

The Factoring Algorithm

If the Fourier transform is computed directly from (7), $(N-1)^2$ complex multiplications are required. However, the M matrix may be factored producing one sparse matrix for each factor of N, which significantly reduces the number of complex multiplications required.

In general, if r is any factor of N, then letting $N' = N/r$, the jth row of the M matrix may be written as

$$\begin{aligned} m_j]^t = [&m_{j1}\, m_{j2} \cdots m_{jr}\, W^{r(j-1)} \cdot m_{j1}\, W^{r(j-1)} \cdot m_{j2} \\ &\cdots W^{r(j-1)} \cdot m_{jr} \cdots W^{N'r(j-1)} \cdot m_{j1} \\ &\cdot W^{N'r(j-1)} \cdot m_{j2} \cdots W^{N'r(j-1)} \cdot m_{jr}] \end{aligned} \tag{11}$$

where] denotes a column matrix and t denotes the transpose. To simplify the notation, let the partial row $m_{j1}\, m_{j2} \cdots m_{jr}$, equal γ_j,

$$m_j]^t = [\gamma_j\, W^{r(j-1)} \cdot \gamma_j \cdots W^{N'r(j-1)} \cdot \gamma_j]. \tag{12}$$

The periodic property of W may be shown by writing

$$W^{r(j-1)} = W^{\{r(j-1)\}_{\text{MOD } N}}. \tag{13}$$

Since

$$\{r(j-1)\}_{\text{MOD } N} = r(j-1)_{\text{MOD } N'}, \tag{14}$$

(12) has N' distinct forms, one for each value of $(j-1)_{\text{MOD } N'}$. Thus,

$$m_j]^t = \begin{cases} [\gamma_j & \gamma_j & \cdots & \gamma_j], & (j-1)_{\text{MOD } N'} = 0 \\ [\gamma_j & W^r \cdot \gamma & \cdots & W^{r(N'-1)} \cdot \gamma_j], & (j-1)_{\text{MOD } N'} = 1 \\ [\gamma_j & W^{2r} \cdot \gamma_j & \cdots & W^{2r(N'-1)} \cdot \gamma_j], & (j-1)_{\text{MOD } N'} = 2 \\ & \vdots & & & \\ [\gamma_j & W^{(N'-1)r} \cdot \gamma_j & \cdots & W^{(N'-1)r(N'-1)} \cdot \gamma_j], & (j-1)_{\text{MOD } N'} = N'-1. \end{cases} \tag{15}$$

The N' relations of (15) yield to an obvious representation as the product of two matrices. Letting ϕ_r be the row matrix of r zeros, the first N' values of j in (15) are equivalent to

$$\begin{bmatrix} \gamma_1 & \phi_r & \phi_r & \cdots & \phi_r \\ \phi_r & \gamma_2 & \phi_r & \cdots & \phi_r \\ \phi_r & \phi_r & \gamma_3 & \cdots & \phi_r \\ \vdots & \vdots & \vdots & & \vdots \\ \phi_r & \phi_r & \phi_r & \cdots & \gamma_{N'} \end{bmatrix} \begin{bmatrix} U_r & U_r & U_r & \cdots & U_r \\ U_r & W^{1r} \cdot U_r & W^{2r} \cdot U_r & \cdots & W^{(N'-1)r} \cdot U_r \\ U_r & W^{2r} \cdot U_r & W^{4r} \cdot U_r & \cdots & W^{(N'-2)r} \cdot U_r \\ \vdots & \vdots & \vdots & & \vdots \\ U_r & W^{(N'-1)r} \cdot U_r & W^{(N'-2)r} \cdot U_r & \cdots & W^r \cdot U_r \end{bmatrix} \tag{16}$$

where U_r is the $r \times r$ unit matrix. The first factor is an $N' \times N'$ array of $1 \times r$ submatrices.

The M matrix is an r-fold repetition of (15). Additional rows can be added to the first factor of (16) until each row of the M matrix is formed in place. The $N \times N'$ array of $1 \times r$ submatrices produced in this manner is the first factor of the algorithm termed F_1.

$$F_1 = \begin{bmatrix}
1 & 1 & \cdots & 1 & 0 & 0 & \cdots & 0 & 0 & 0 & \cdots & 0 & \cdots & 0 & 0 & \cdots & 0 \\
0 & 0 & \cdots & 0 & 1 & W^1 & \cdots & W^{r-1} & 0 & 0 & \cdots & 0 & \cdots & 0 & 0 & \cdots & 0 \\
0 & 0 & \cdots & 0 & 0 & 0 & \cdots & 0 & 1 & W^2 & \cdots & W^{2(r-1)} & \cdots & 0 & 0 & \cdots & 0 \\
 & & & & & & & & \vdots & & & & & & & & \\
0 & 0 & \cdots & 0 & 0 & 0 & \cdots & 0 & 0 & 0 & \cdots & 0 & \cdots & 1 & W^{N'-1} & \cdots & W^{(N'-1)(r-1)} \\
1 & W^{N'} & \cdots & W^{N'(r-1)} & 0 & 0 & \cdots & 0 & 0 & 0 & \cdots & 0 & \cdots & 0 & 0 & \cdots & 0 \\
0 & 0 & \cdots & 0 & 1 & W^{N'+1} & \cdots & W^{(N'+1)(r-1)} & 0 & 0 & \cdots & 0 & \cdots & 0 & 0 & \cdots & 0 \\
 & & & & & & & & \vdots & & & & & & & & \\
0 & 0 & \cdots & 0 & 0 & 0 & \cdots & 0 & 0 & 0 & \cdots & 0 & \cdots & 1 & W^{N'r-1} & \cdots & W^{(N'r-1)(r-1)}
\end{bmatrix} \cdot \quad (17)$$

Note that the first factor contains nonzero entries identical to the first r columns of the M matrix, but staggered in a cyclic fashion through the $N \times N$ array.

The second factor of (16) is of the same form as the M matrix except that each entry is a matrix rather than a single element, and each entry involves a power of W^r rather than W. The derivation of the first factoring equally applies to an array of matrices, so that if N' is not prime, then the second factor of (16) may be similarly factored. Therefore the second factor is denoted by M' to emphasize the recursive relation of the factoring. Continuing the notation, the factoring is

$$M = F_1 \cdot M^1 \quad (18)$$

$$M = F_1 \cdot F_2 \cdot M'' \quad (19)$$

$$\vdots$$

$$M = F_1 \cdot F_2 \cdot F_3 \cdots F_k \quad (20)$$

where k is the number of factors of N. Each factor F_i is formed by a cyclic staggering of the first r_i rows of submatrices from M^{i-1} through an $N \times N$ array where r_i is the factor of N separated at the ith factoring.

If each factor is a prime factor of N, and the factoring is complete, a minimum number of multiplications results for this algorithm. For convenience, the procedure employed in this paper is to order the factors by magnitude, postmultiplying in decreasing order. The procedure may be verified by partitioning F_1 and M' into $r \times r$ submatrices and multiplying.

Factoring Examples

Several sample factorings serve to illustrate the equivalence of the product of the factors to the original matrix. For $N = 6$, the M matrix and its two factors are

$$M = \left[\begin{array}{ccc|ccc}
1 & 1 & 1 & 1 & 1 & 1 \\
1 & W^1 & W^2 & W^3 & W^4 & W^5 \\
1 & W^2 & W^4 & W^0 & W^2 & W^4 \\
\hline
1 & W^3 & W^0 & W^3 & W^0 & W^3 \\
1 & W^4 & W^2 & W^0 & W^4 & W^2 \\
1 & W^5 & W^4 & W^3 & W^2 & W^1
\end{array}\right] = \left[\begin{array}{ccc|ccc}
1 & 1 & 1 & 0 & 0 & 0 \\
0 & 0 & 0 & 1 & W^1 & W^2 \\
1 & W^2 & W^4 & 0 & 0 & 0 \\
\hline
0 & 0 & 0 & 1 & W^3 & W^0 \\
1 & W^4 & W^2 & 0 & 0 & 0 \\
0 & 0 & 0 & 1 & W^5 & W^4
\end{array}\right] \cdot \left[\begin{array}{ccc|ccc}
1 & 0 & 0 & 1 & 0 & 0 \\
0 & 1 & 0 & 0 & 1 & 0 \\
0 & 0 & 1 & 0 & 0 & 1 \\
\hline
1 & 0 & 0 & W^3 & 0 & 0 \\
0 & 1 & 0 & 0 & W^3 & 0 \\
0 & 0 & 1 & 0 & 0 & W^3
\end{array}\right]$$

These matrices have been partitioned into 3×3 ($r \times r$) submatrices to emphasize the technique for verifying the factoring. Two factors are also found for $N = 15$, as follows.

$$F_1 = \left[\begin{array}{ccccc|ccccc|ccccc}
1 & 1 & 1 & 1 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 1 & W^1 & W^2 & W^3 & W^4 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & W^2 & W^4 & W^6 & W^8 \\
1 & W^3 & W^6 & W^9 & W^{12} & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 1 & W^4 & W^8 & W^{12} & W^1 & 0 & 0 & 0 & 0 & 0 \\
\hline
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & W^5 & W^{10} & W^0 & W^5 \\
1 & W^6 & W^{12} & W^3 & W^9 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 1 & W^7 & W^{14} & W^6 & W^{13} & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & W^8 & W^1 & W^9 & W^2 \\
1 & W^9 & W^3 & W^{12} & W^6 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
\hline
0 & 0 & 0 & 0 & 0 & 1 & W^{10} & W^5 & W^0 & W^{10} & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & W^{11} & W^7 & W^3 & W^{14} \\
1 & W^{12} & W^9 & W^6 & W^3 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 1 & W^{13} & W^{11} & W^9 & W^7 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & W^{14} & W^{13} & W^{12} & W^{11}
\end{array}\right]$$

$$F_2 = \left[\begin{array}{ccccc|ccccc|ccccc}
1 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 \\
0 & 1 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 \\
0 & 0 & 1 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 1 & 0 & 0 \\
0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 1 & 0 \\
0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 1 \\
\hline
1 & 0 & 0 & 0 & 0 & W^5 & 0 & 0 & 0 & 0 & W^{10} & 0 & 0 & 0 & 0 \\
0 & 1 & 0 & 0 & 0 & 0 & W^5 & 0 & 0 & 0 & 0 & W^{10} & 0 & 0 & 0 \\
0 & 0 & 1 & 0 & 0 & 0 & 0 & W^5 & 0 & 0 & 0 & 0 & W^{10} & 0 & 0 \\
0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & W^5 & 0 & 0 & 0 & 0 & W^{10} & 0 \\
0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & W^5 & 0 & 0 & 0 & 0 & W^{10} \\
\hline
1 & 0 & 0 & 0 & 0 & W^{10} & 0 & 0 & 0 & 0 & W^5 & 0 & 0 & 0 & 0 \\
0 & 1 & 0 & 0 & 0 & 0 & W^{10} & 0 & 0 & 0 & 0 & W^5 & 0 & 0 & 0 \\
0 & 0 & 1 & 0 & 0 & 0 & 0 & W^{10} & 0 & 0 & 0 & 0 & W^5 & 0 & 0 \\
0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & W^{10} & 0 & 0 & 0 & 0 & W^5 & 0 \\
0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & W^{10} & 0 & 0 & 0 & 0 & W^5
\end{array}\right]$$

For $N=16$, $M = F_1 \cdot F_2 \cdot F_3 \cdot F_4$. Omitting the zero entries and the partitioning, the four factors are derived as follows.

$$F_1 = \begin{bmatrix}
1 & 1 & & & & & & & & & & & & & & \\
 & & 1 & W^1 & & & & & & & & & & & & \\
 & & & & 1 & W^2 & & & & & & & & & & \\
 & & & & & & 1 & W^3 & & & & & & & & \\
 & & & & & & & & 1 & W^4 & & & & & & \\
 & & & & & & & & & & 1 & W^5 & & & & \\
 & & & & & & & & & & & & 1 & W^6 & & \\
 & & & & & & & & & & & & & & 1 & W^7 \\
1 & W^8 & & & & & & & & & & & & & & \\
 & & 1 & W^9 & & & & & & & & & & & & \\
 & & & & 1 & W^{10} & & & & & & & & & & \\
 & & & & & & 1 & W^{11} & & & & & & & & \\
 & & & & & & & & 1 & W^{12} & & & & & & \\
 & & & & & & & & & & 1 & W^{13} & & & & \\
 & & & & & & & & & & & & 1 & W^{14} & & \\
 & & & & & & & & & & & & & & 1 & W^{15}
\end{bmatrix}$$

$$M' = \begin{bmatrix}
1 & & 1 & & 1 & & 1 & & 1 & & 1 & & 1 & & 1 & \\
& 1 & & 1 & & 1 & & 1 & & 1 & & 1 & & 1 & & 1 \\
1 & & W^2 & & W^4 & & W^6 & & W^8 & & W^{10} & & W^{12} & & W^{14} & \\
& 1 & & W^2 & & W^4 & & W^6 & & W^8 & & W^{10} & & W^{12} & & W^{14} \\
1 & & W^4 & & W^8 & & W^{12} & & W^0 & & W^4 & & W^8 & & W^{12} & \\
& 1 & & W^4 & & W^8 & & W^{12} & & W^0 & & W^4 & & W^8 & & W^{12} \\
1 & & W^6 & & W^{12} & & W^2 & & W^8 & & W^{14} & & W^4 & & W^{10} & \\
& 1 & & W^6 & & W^{12} & & W^2 & & W^8 & & W^{14} & & W^4 & & W^{10} \\
1 & & W^8 & & W^0 & & W^8 & & W^0 & & W^8 & & W^0 & & W^8 & \\
& 1 & & W^8 & & W^0 & & W^8 & & W^0 & & W^8 & & W^0 & & W^8 \\
1 & & W^{10} & & W^4 & & W^{14} & & W^8 & & W^2 & & W^{12} & & W^6 & \\
& 1 & & W^{10} & & W^4 & & W^{14} & & W^8 & & W^2 & & W^{12} & & W^6 \\
1 & & W^{12} & & W^8 & & W^4 & & W^0 & & W^{12} & & W^8 & & W^4 & \\
& 1 & & W^{12} & & W^8 & & W^4 & & W^0 & & W^{12} & & W^8 & & W^4 \\
1 & & W^{14} & & W^{12} & & W^{10} & & W^8 & & W^6 & & W^4 & & W^2 & \\
& 1 & & W^{14} & & W^{12} & & W^{10} & & W^8 & & W^6 & & W^4 & & W^2
\end{bmatrix}$$

$$F_2 = \begin{bmatrix}
1 & & 1 & & & & & & & & & & & & & \\
& 1 & & 1 & & & & & & & & & & & & \\
& & & & 1 & & W^2 & & & & & & & & & \\
& & & & & 1 & & W^2 & & & & & & & & \\
& & & & & & & & 1 & & W^4 & & & & & \\
& & & & & & & & & 1 & & W^4 & & & & \\
& & & & & & & & & & & & 1 & & W^6 & \\
& & & & & & & & & & & & & 1 & & W^6 \\
1 & & W^8 & & & & & & & & & & & & & \\
& 1 & & W^8 & & & & & & & & & & & & \\
& & & & 1 & & W^{10} & & & & & & & & & \\
& & & & & 1 & & W^{10} & & & & & & & & \\
& & & & & & & & 1 & & W^{12} & & & & & \\
& & & & & & & & & 1 & & W^{12} & & & & \\
& & & & & & & & & & & & 1 & & W^{14} & \\
& & & & & & & & & & & & & 1 & & W^{14}
\end{bmatrix}$$

$$M'' = \begin{bmatrix}
1 & & & & 1 & & & & 1 & & & & 1 & & & \\
& 1 & & & & 1 & & & & 1 & & & & 1 & & \\
& & 1 & & & & 1 & & & & 1 & & & & 1 & \\
& & & 1 & & & & 1 & & & & 1 & & & & 1 \\
1 & & & & W^4 & & & & W^8 & & & & W^{12} & & & \\
& 1 & & & & W^4 & & & & W^8 & & & & W^{12} & & \\
& & 1 & & & & W^4 & & & & W^8 & & & & W^{12} & \\
& & & 1 & & & & W^4 & & & & W^8 & & & & W^{12} \\
1 & & & & W^8 & & & & W^0 & & & & W^8 & & & \\
& 1 & & & & W^8 & & & & W^0 & & & & W^8 & & \\
& & 1 & & & & W^8 & & & & W^0 & & & & W^8 & \\
& & & 1 & & & & W^8 & & & & W^0 & & & & W^8 \\
1 & & & & W^{12} & & & & W^8 & & & & W^4 & & & \\
& 1 & & & & W^{12} & & & & W^8 & & & & W^4 & & \\
& & 1 & & & & W^{12} & & & & W^8 & & & & W^4 & \\
& & & 1 & & & & W^{12} & & & & W^8 & & & & W^4
\end{bmatrix}$$

The last two factors are the following.

$$F_3 = \begin{bmatrix}
1 & & & & 1 & & & & & & & & & & & \\
& 1 & & & & 1 & & & & & & & & & & \\
& & 1 & & & & 1 & & & & & & & & & \\
& & & 1 & & & & 1 & & & & & & & & \\
& & & & & & & & 1 & & & & W^4 & & & \\
& & & & & & & & & 1 & & & & W^4 & & \\
& & & & & & & & & & 1 & & & & W^4 & \\
& & & & & & & & & & & 1 & & & & W^4 \\
1 & & & & W^8 & & & & & & & & & & & \\
& 1 & & & & W^8 & & & & & & & & & & \\
& & 1 & & & & W^8 & & & & & & & & & \\
& & & 1 & & & & W^8 & & & & & & & & \\
& & & & & & & & 1 & & & & W^{12} & & & \\
& & & & & & & & & 1 & & & & W^{12} & & \\
& & & & & & & & & & 1 & & & & W^{12} & \\
& & & & & & & & & & & 1 & & & & W^{12}
\end{bmatrix}$$

$$F_4 = \begin{bmatrix}
1 & & & & & & & 1 & & & & & & & & \\
& 1 & & & & & & & 1 & & & & & & & \\
& & 1 & & & & & & & 1 & & & & & & \\
& & & 1 & & & & & & & 1 & & & & & \\
& & & & 1 & & & & & & & 1 & & & & \\
& & & & & 1 & & & & & & & 1 & & & \\
& & & & & & 1 & & & & & & & 1 & & \\
& & & & & & & 1 & & & & & & & 1 & \\
1 & & & & & & & & W^8 & & & & & & & \\
& 1 & & & & & & & & W^8 & & & & & & \\
& & 1 & & & & & & & & W^8 & & & & & \\
& & & 1 & & & & & & & & W^8 & & & & \\
& & & & 1 & & & & & & & & W^8 & & & \\
& & & & & 1 & & & & & & & & W^8 & & \\
& & & & & & 1 & & & & & & & & W^8 & \\
& & & & & & & 1 & & & & & & & & W^8
\end{bmatrix}.$$

Multiplications Required

The number of complex multiplications required after factoring is found from a recursive relation. The number of multiplications required after the first prime factor r_1 is separated is the number of nonunity entries in the first r_1 columns plus r_1 times the number of multiplies required for a matrix of order N/r_1. If $F(N)$ is the number of complex multiplications required for the operations in the M matrix, after one factoring the number then is

$$F(N) = (N - 1)(r_1 - 1) + r_1 F(N/r_1). \quad (21)$$

For $N = \prod_{i=1}^{n} r_i$, the number of multiplications is easily shown to be

$$F(N) = N \sum_{i=1}^{n} (r_i - 1) + 1 - N. \quad (22)$$

If $N = r^n$,

$$F(N) = Nn(r - 1) + 1 - N. \quad (23)$$

(These formulas are exact, except that $W^0 = 1$ and $W^{N/2} = -1$ are not used outside the first rows and first columns.) Equation (23) may also be written as

$$F(N) = N \ln (N)(r - 1)/\ln (r) + 1 - N. \quad (24)$$

Since $(r-1)/\ln (r)$ is monotone increasing for $r > 1$, the least number of multiplies for an integer r and $N = r^n$ occurs for $r = 2$.[1] However, a more important advantage than the formal efficiency accrues to the use of the base two since all multiplications occur in complex conjugate pairs and the number of multiplies may be further re-

[1] Cooley and Tukey [4] report that the number of multiples is equal to $Nr \log_r (N) = N \ln (N) r/\ln (r)$ which is a minimum for $r = e$, and on the integers, for $r = 3$.

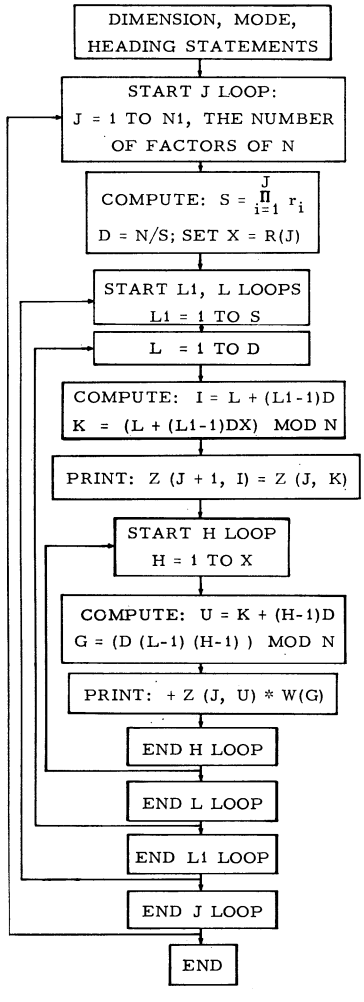

Fig. 1. Flow chart for the loops of the fast Fourier transform.

```
LIST

FTL00P    20:25   THURS. 07/11/68

100 DIMENSION Z(20,100),R(20)
110 INTEGER S,D,R,H,G,Z,U,V,X
120 4:READ,N,N1
130 PRINT,"EQUATIONS OF THE FAST FOURIER TRANSFORM,"
140 PRINT 2,N,
150 2:FORMAT(11X,8HFOR N = ,I2,3H = )
160 N2=N1-1
170 DO 1,J=1,N2
180 READ,R(J)
190 PRINT 1,R(J),
200 1:FORMAT(I2,2H *)
210 READ,R(N1)
220 PRINT 3,R(N1)
230 3:FORMAT(I2)
240 S=R(1)
250 DO 11,J=1,N1
260 X=R(J)
270 D=N/S
280 DO 31,L1=1,S
290 DO 31,L=1,D
300 61:FORMAT(2HZ(,I2,1H,,I2,6H) = Z(,I2,1H,,I2,2H) )
310 K=AMOD(L+(L1-1)*D*X,N)
320 I=L+(L1-1)*D
330 PRINT
340 PRINT 61,J+1,I,J,K,
350 DO 31, H=2,X
360 IF(AMOD(H-1,4)) 32,33,32
370 33:PRINT:PRINT"     ",
380 32:G=AMOD((L1-1)*D*(H-1),N)
390 U=K+(H-1)*D
400 PRINT 31,J,U,G,
410 31:FORMAT(4H+ Z(,I2,1H,,I2,4H)*W(,I2,2H) )
420 IF(N1-1-J) 11
430 S=S*R(J+1)
440 11:PRINT
450 PRINT;;;;GOTO 4:END
460 $DATA
470 6,2,2,3,
480 8,3,2,2,2,
490 15,2,3,5,
500 16,4,2,2,2,2,
510 27,3,3,3,3,
520 30,3,2,3,5,
530 64,6,2,2,2,2,2,2
540 STOP
```

Fig. 2. Program to write fast Fourier transform equations.

duced by half. Thus for $N=2^\gamma$, the required number of complex multiplications is

$$F(2^\gamma) = \frac{2^\gamma \gamma}{2} + 1 - 2^\gamma = 2^\gamma\left(\frac{\gamma}{2} - 1\right) + 1. \quad (25)$$

A program has been written in FORTRAN for the GE-265 time-sharing system which writes the equations for the fast Fourier transform. The program illustrates the simple set of four loops necessary to mechanize the transform for any N, that is, employing a mixed radix, without special addressing logic. (For N a power of 2, the H loop becomes a single execution, and only three loops are required.) The flow chart for the basic routine is shown in Fig. 1; the program and sample results are shown in Figs. 2 and 3.

One of several programs written in FORTRAN employing the algorithm for radix 2 is listed in Fig. 4. This sample program stores the output complex coefficients in the input file and when the program is recalled, the inverse transform is taken as a verification. The output is the magnitude of the Fourier coefficients and is written into a file for computer plotting. A sample run for a square pulse input is shown in Figs. 5 and 6 and the restored input function in Figs. 7 and 8. The program also includes folding of the input data as required by (5). Without loss of generality, the program outputs are computed for the regions of $-NT/2 \leq t < NT/2$ and $-1/2T \leq f < 1/2T$ to accommodate the symmetry available in the sample problem. The execution time for the time-sharing system appears to be dependent primarily upon the number of accesses to the disc for file data or for READ WRITE operations, and the program performs a transformation in approximately one minute for 128 complex coefficients and 128 input samples.

The fast Fourier transform shown in Fig. 4 employs a full buffer storage, the $A(\ )$ array, of length $2N$. This permits storing the full vector of complex numbers computed at each matrix multiplication, and may be a luxury in some applications. The basic algorithm requires buffer storage at each matrix multiplication, the extent depending upon the factor r being extracted and the position of the matrix in the sequence of matrix factors. The minimum buffer length has not been assessed; however, the maximum is clearly $2N$ words.

```
FTL00P    20:27   THURS. 07/11/68

EQUATIONS OF THE FAST FOURIER TRANSFORM.
          FOR N =  6 =  2 * 3

Z( 2, 1) = Z( 1, 1) + Z( 1, 4)*W( 0)
Z( 2, 2) = Z( 1, 2) + Z( 1, 5)*W( 0)
Z( 2, 3) = Z( 1, 3) + Z( 1, 6)*W( 0)
Z( 2, 4) = Z( 1, 1) + Z( 1, 4)*W( 3)
Z( 2, 5) = Z( 1, 2) + Z( 1, 5)*W( 3)
Z( 2, 6) = Z( 1, 3) + Z( 1, 6)*W( 3)


Z( 3, 1) = Z( 2, 1) + Z( 2, 2)*W( 0) + Z( 2, 3)*W( 0)
Z( 3, 2) = Z( 2, 4) + Z( 2, 5)*W( 1) + Z( 2, 6)*W( 2)
Z( 3, 3) = Z( 2, 1) + Z( 2, 2)*W( 2) + Z( 2, 3)*W( 4)
Z( 3, 4) = Z( 2, 4) + Z( 2, 5)*W( 3) + Z( 2, 6)*W( 0)
Z( 3, 5) = Z( 2, 1) + Z( 2, 2)*W( 4) + Z( 2, 3)*W( 2)
Z( 3, 6) = Z( 2, 4) + Z( 2, 5)*W( 5) + Z( 2, 6)*W( 4)



EQUATIONS OF THE FAST FOURIER TRANSFORM.
          FOR N =  8 =  2 * 2 * 2

Z( 2, 1) = Z( 1, 1) + Z( 1, 5)*W( 0)
Z( 2, 2) = Z( 1, 2) + Z( 1, 6)*W( 0)
Z( 2, 3) = Z( 1, 3) + Z( 1, 7)*W( 0)
Z( 2, 4) = Z( 1, 4) + Z( 1, 8)*W( 0)
Z( 2, 5) = Z( 1, 1) + Z( 1, 5)*W( 4)
Z( 2, 6) = Z( 1, 2) + Z( 1, 6)*W( 4)
Z( 2, 7) = Z( 1, 3) + Z( 1, 7)*W( 4)
Z( 2, 8) = Z( 1, 4) + Z( 1, 8)*W( 4)


Z( 3, 1) = Z( 2, 1) + Z( 2, 3)*W( 0)
Z( 3, 2) = Z( 2, 2) + Z( 2, 4)*W( 0)
Z( 3, 3) = Z( 2, 5) + Z( 2, 7)*W( 2)
Z( 3, 4) = Z( 2, 6) + Z( 2, 8)*W( 2)
Z( 3, 5) = Z( 2, 1) + Z( 2, 3)*W( 4)
Z( 3, 6) = Z( 2, 2) + Z( 2, 4)*W( 4)
Z( 3, 7) = Z( 2, 5) + Z( 2, 7)*W( 6)
Z( 3, 8) = Z( 2, 6) + Z( 2, 8)*W( 6)


Z( 4, 1) = Z( 3, 1) + Z( 3, 2)*W( 0)
Z( 4, 2) = Z( 3, 3) + Z( 3, 4)*W( 1)
Z( 4, 3) = Z( 3, 5) + Z( 3, 6)*W( 2)
Z( 4, 4) = Z( 3, 7) + Z( 3, 8)*W( 3)
Z( 4, 5) = Z( 3, 1) + Z( 3, 2)*W( 4)
Z( 4, 6) = Z( 3, 3) + Z( 3, 4)*W( 5)
Z( 4, 7) = Z( 3, 5) + Z( 3, 6)*W( 6)
Z( 4, 8) = Z( 3, 7) + Z( 3, 8)*W( 7)
```

(a)

```
EQUATIONS OF THE FAST FOURIER TRANSFORM.
          FOR N = 15 =  3 * 5

Z( 2, 1) = Z( 1, 1) + Z( 1, 6)*W( 0) + Z( 1,11)*W( 0)
Z( 2, 2) = Z( 1, 2) + Z( 1, 7)*W( 0) + Z( 1,12)*W( 0)
Z( 2, 3) = Z( 1, 3) + Z( 1, 8)*W( 0) + Z( 1,13)*W( 0)
Z( 2, 4) = Z( 1, 4) + Z( 1, 9)*W( 0) + Z( 1,14)*W( 0)
Z( 2, 5) = Z( 1, 5) + Z( 1,10)*W( 0) + Z( 1,15)*W( 0)
Z( 2, 6) = Z( 1, 1) + Z( 1, 6)*W( 5) + Z( 1,11)*W(10)
Z( 2, 7) = Z( 1, 2) + Z( 1, 7)*W( 5) + Z( 1,12)*W(10)
Z( 2, 8) = Z( 1, 3) + Z( 1, 8)*W( 5) + Z( 1,13)*W(10)
Z( 2, 9) = Z( 1, 4) + Z( 1, 9)*W( 5) + Z( 1,14)*W(10)
Z( 2,10) = Z( 1, 5) + Z( 1,10)*W( 5) + Z( 1,15)*W(10)
Z( 2,11) = Z( 1, 1) + Z( 1, 6)*W(10) + Z( 1,11)*W( 5)
Z( 2,12) = Z( 1, 2) + Z( 1, 7)*W(10) + Z( 1,12)*W( 5)
Z( 2,13) = Z( 1, 3) + Z( 1, 8)*W(10) + Z( 1,13)*W( 5)
Z( 2,14) = Z( 1, 4) + Z( 1, 9)*W(10) + Z( 1,14)*W( 5)
Z( 2,15) = Z( 1, 5) + Z( 1,10)*W(10) + Z( 1,15)*W( 5)


Z( 3, 1) = Z( 2, 1) + Z( 2, 2)*W( 0) + Z( 2, 3)*W( 0) + Z( 2, 4)*W( 0)
    + Z( 2, 5)*W( 0)
Z( 3, 2) = Z( 2, 6) + Z( 2, 7)*W( 1) + Z( 2, 8)*W( 2) + Z( 2, 9)*W( 3)
    + Z( 2,10)*W( 4)
Z( 3, 3) = Z( 2,11) + Z( 2,12)*W( 2) + Z( 2,13)*W( 4) + Z( 2,14)*W( 6)
    + Z( 2,15)*W( 8)
Z( 3, 4) = Z( 2, 1) + Z( 2, 2)*W( 3) + Z( 2, 3)*W( 6) + Z( 2, 4)*W( 9)
    + Z( 2, 5)*W(12)
Z( 3, 5) = Z( 2, 6) + Z( 2, 7)*W( 4) + Z( 2, 8)*W( 8) + Z( 2, 9)*W(12)
    + Z( 2,10)*W( 1)
Z( 3, 6) = Z( 2,11) + Z( 2,12)*W( 5) + Z( 2,13)*W(10) + Z( 2,14)*W( 0)
    + Z( 2,15)*W( 5)
Z( 3, 7) = Z( 2, 1) + Z( 2, 2)*W( 6) + Z( 2, 3)*W(12) + Z( 2, 4)*W( 3)
    + Z( 2, 5)*W( 9)
Z( 3, 8) = Z( 2, 6) + Z( 2, 7)*W( 7) + Z( 2, 8)*W(14) + Z( 2, 9)*W( 6)
    + Z( 2,10)*W(13)
Z( 3, 9) = Z( 2,11) + Z( 2,12)*W( 8) + Z( 2,13)*W( 1) + Z( 2,14)*W( 9)
    + Z( 2,15)*W( 2)
Z( 3,10) = Z( 2, 1) + Z( 2, 2)*W( 9) + Z( 2, 3)*W( 3) + Z( 2, 4)*W(12)
    + Z( 2, 5)*W( 6)
Z( 3,11) = Z( 2, 6) + Z( 2, 7)*W(10) + Z( 2, 8)*W( 5) + Z( 2, 9)*W( 0)
    + Z( 2,10)*W(10)
Z( 3,12) = Z( 2,11) + Z( 2,12)*W(11) + Z( 2,13)*W( 7) + Z( 2,14)*W( 3)
    + Z( 2,15)*W(14)
Z( 3,13) = Z( 2, 1) + Z( 2, 2)*W(12) + Z( 2, 3)*W( 9) + Z( 2, 4)*W( 6)
    + Z( 2, 5)*W( 3)
Z( 3,14) = Z( 2, 6) + Z( 2, 7)*W(13) + Z( 2, 8)*W(11) + Z( 2, 9)*W( 9)
    + Z( 2,10)*W( 7)
Z( 3,15) = Z( 2,11) + Z( 2,12)*W(14) + Z( 2,13)*W(13) + Z( 2,14)*W(12)
    + Z( 2,15)*W(11)
```

(b)

```
EQUATIONS OF THE FAST FOURIER TRANSFORM.
          FOR N = 16 =  2 * 2 * 2 * 2

Z( 2, 1) = Z( 1, 1) + Z( 1, 9)*W( 0)
Z( 2, 2) = Z( 1, 2) + Z( 1,10)*W( 0)
Z( 2, 3) = Z( 1, 3) + Z( 1,11)*W( 0)
Z( 2, 4) = Z( 1, 4) + Z( 1,12)*W( 0)
Z( 2, 5) = Z( 1, 5) + Z( 1,13)*W( 0)
Z( 2, 6) = Z( 1, 6) + Z( 1,14)*W( 0)
Z( 2, 7) = Z( 1, 7) + Z( 1,15)*W( 0)
Z( 2, 8) = Z( 1, 8) + Z( 1,16)*W( 0)
Z( 2, 9) = Z( 1, 1) + Z( 1, 9)*W( 8)
Z( 2,10) = Z( 1, 2) + Z( 1,10)*W( 8)
Z( 2,11) = Z( 1, 3) + Z( 1,11)*W( 8)
Z( 2,12) = Z( 1, 4) + Z( 1,12)*W( 8)
Z( 2,13) = Z( 1, 5) + Z( 1,13)*W( 8)
Z( 2,14) = Z( 1, 6) + Z( 1,14)*W( 8)
Z( 2,15) = Z( 1, 7) + Z( 1,15)*W( 8)
Z( 2,16) = Z( 1, 8) + Z( 1,16)*W( 8)


Z( 3, 1) = Z( 2, 1) + Z( 2, 5)*W( 0)
Z( 3, 2) = Z( 2, 2) + Z( 2, 6)*W( 0)
Z( 3, 3) = Z( 2, 3) + Z( 2, 7)*W( 0)
Z( 3, 4) = Z( 2, 4) + Z( 2, 8)*W( 0)
Z( 3, 5) = Z( 2, 9) + Z( 2,13)*W( 4)
Z( 3, 6) = Z( 2,10) + Z( 2,14)*W( 4)
Z( 3, 7) = Z( 2,11) + Z( 2,15)*W( 4)
Z( 3, 8) = Z( 2,12) + Z( 2,16)*W( 4)
Z( 3, 9) = Z( 2, 1) + Z( 2, 5)*W( 8)
Z( 3,10) = Z( 2, 2) + Z( 2, 6)*W( 8)
Z( 3,11) = Z( 2, 3) + Z( 2, 7)*W( 8)
Z( 3,12) = Z( 2, 4) + Z( 2, 8)*W( 8)
Z( 3,13) = Z( 2, 9) + Z( 2,13)*W(12)
Z( 3,14) = Z( 2,10) + Z( 2,14)*W(12)
Z( 3,15) = Z( 2,11) + Z( 2,15)*W(12)
Z( 3,16) = Z( 2,12) + Z( 2,16)*W(12)


Z( 4, 1) = Z( 3, 1) + Z( 3, 3)*W( 0)
Z( 4, 2) = Z( 3, 2) + Z( 3, 4)*W( 0)
Z( 4, 3) = Z( 3, 5) + Z( 3, 7)*W( 2)
Z( 4, 4) = Z( 3, 6) + Z( 3, 8)*W( 2)
Z( 4, 5) = Z( 3, 9) + Z( 3,11)*W( 4)
Z( 4, 6) = Z( 3,10) + Z( 3,12)*W( 4)
Z( 4, 7) = Z( 3,13) + Z( 3,15)*W( 6)
Z( 4, 8) = Z( 3,14) + Z( 3,16)*W( 6)
Z( 4, 9) = Z( 3, 1) + Z( 3, 3)*W( 8)
Z( 4,10) = Z( 3, 2) + Z( 3, 4)*W( 8)
Z( 4,11) = Z( 3, 5) + Z( 3, 7)*W(10)
Z( 4,12) = Z( 3, 6) + Z( 3, 8)*W(10)
Z( 4,13) = Z( 3, 9) + Z( 3,11)*W(12)
Z( 4,14) = Z( 3,10) + Z( 3,12)*W(12)
Z( 4,15) = Z( 3,13) + Z( 3,15)*W(14)
Z( 4,16) = Z( 3,14) + Z( 3,16)*W(14)
```

(c)

```
Z( 5, 1) = Z( 4, 1) + Z( 4, 2)*W( 0)
Z( 5, 2) = Z( 4, 3) + Z( 4, 4)*W( 1)
Z( 5, 3) = Z( 4, 5) + Z( 4, 6)*W( 2)
Z( 5, 4) = Z( 4, 7) + Z( 4, 8)*W( 3)
Z( 5, 5) = Z( 4, 9) + Z( 4,10)*W( 4)
Z( 5, 6) = Z( 4,11) + Z( 4,12)*W( 5)
Z( 5, 7) = Z( 4,13) + Z( 4,14)*W( 6)
Z( 5, 8) = Z( 4,15) + Z( 4,16)*W( 7)
Z( 5, 9) = Z( 4, 1) + Z( 4, 2)*W( 8)
Z( 5,10) = Z( 4, 3) + Z( 4, 4)*W( 9)
Z( 5,11) = Z( 4, 5) + Z( 4, 6)*W(10)
Z( 5,12) = Z( 4, 7) + Z( 4, 8)*W(11)
Z( 5,13) = Z( 4, 9) + Z( 4,10)*W(12)
Z( 5,14) = Z( 4,11) + Z( 4,12)*W(13)
Z( 5,15) = Z( 4,13) + Z( 4,14)*W(14)
Z( 5,16) = Z( 4,15) + Z( 4,16)*W(15)
```

```
EQUATIONS OF THE FAST FOURIER TRANSFORM.
          FOR N = 27 =  3 * 3 * 3

Z( 2, 1) = Z( 1, 1) + Z( 1,10)*W( 0) + Z( 1,19)*W( 0)
Z( 2, 2) = Z( 1, 2) + Z( 1,11)*W( 0) + Z( 1,20)*W( 0)
Z( 2, 3) = Z( 1, 3) + Z( 1,12)*W( 0) + Z( 1,21)*W( 0)
Z( 2, 4) = Z( 1, 4) + Z( 1,13)*W( 0) + Z( 1,22)*W( 0)
Z( 2, 5) = Z( 1, 5) + Z( 1,14)*W( 0) + Z( 1,23)*W( 0)
Z( 2, 6) = Z( 1, 6) + Z( 1,15)*W( 0) + Z( 1,24)*W( 0)
Z( 2, 7) = Z( 1, 7) + Z( 1,16)*W( 0) + Z( 1,25)*W( 0)
Z( 2, 8) = Z( 1, 8) + Z( 1,17)*W( 0) + Z( 1,26)*W( 0)
Z( 2, 9) = Z( 1, 9) + Z( 1,18)*W( 0) + Z( 1,27)*W( 0)
Z( 2,10) = Z( 1, 1) + Z( 1,10)*W( 9) + Z( 1,19)*W(18)
Z( 2,11) = Z( 1, 2) + Z( 1,11)*W( 9) + Z( 1,20)*W(18)
Z( 2,12) = Z( 1, 3) + Z( 1,12)*W( 9) + Z( 1,21)*W(18)
Z( 2,13) = Z( 1, 4) + Z( 1,13)*W( 9) + Z( 1,22)*W(18)
Z( 2,14) = Z( 1, 5) + Z( 1,14)*W( 9) + Z( 1,23)*W(18)
Z( 2,15) = Z( 1, 6) + Z( 1,15)*W( 9) + Z( 1,24)*W(18)
Z( 2,16) = Z( 1, 7) + Z( 1,16)*W( 9) + Z( 1,25)*W(18)
Z( 2,17) = Z( 1, 8) + Z( 1,17)*W( 9) + Z( 1,26)*W(18)
Z( 2,18) = Z( 1, 9) + Z( 1,18)*W( 9) + Z( 1,27)*W(18)
Z( 2,19) = Z( 1, 1) + Z( 1,10)*W(18) + Z( 1,19)*W( 9)
Z( 2,20) = Z( 1, 2) + Z( 1,11)*W(18) + Z( 1,20)*W( 9)
Z( 2,21) = Z( 1, 3) + Z( 1,12)*W(18) + Z( 1,21)*W( 9)
Z( 2,22) = Z( 1, 4) + Z( 1,13)*W(18) + Z( 1,22)*W( 9)
Z( 2,23) = Z( 1, 5) + Z( 1,14)*W(18) + Z( 1,23)*W( 9)
Z( 2,24) = Z( 1, 6) + Z( 1,15)*W(18) + Z( 1,24)*W( 9)
Z( 2,25) = Z( 1, 7) + Z( 1,16)*W(18) + Z( 1,25)*W( 9)
Z( 2,26) = Z( 1, 8) + Z( 1,17)*W(18) + Z( 1,26)*W( 9)
Z( 2,27) = Z( 1, 9) + Z( 1,18)*W(18) + Z( 1,27)*W( 9)
```

(d)

Fig. 3. Sample equations of the fast Fourier transform.

```
Z( 3, 1) = Z( 2, 1) + Z( 2, 4)*W( 0) + Z( 2, 7)*W( 0)
Z( 3, 2) = Z( 2, 2) + Z( 2, 5)*W( 0) + Z( 2, 8)*W( 0)
Z( 3, 3) = Z( 2, 3) + Z( 2, 6)*W( 0) + Z( 2, 9)*W( 0)
Z( 3, 4) = Z( 2,10) + Z( 2,13)*W( 3) + Z( 2,16)*W( 6)
Z( 3, 5) = Z( 2,11) + Z( 2,14)*W( 3) + Z( 2,17)*W( 6)
Z( 3, 6) = Z( 2,12) + Z( 2,15)*W( 3) + Z( 2,18)*W( 6)
Z( 3, 7) = Z( 2,19) + Z( 2,22)*W( 6) + Z( 2,25)*W(12)
Z( 3, 8) = Z( 2,20) + Z( 2,23)*W( 6) + Z( 2,26)*W(12)
Z( 3, 9) = Z( 2,21) + Z( 2,24)*W( 6) + Z( 2,27)*W(12)
Z( 3,10) = Z( 2, 1) + Z( 2, 4)*W( 9) + Z( 2, 7)*W(18)
Z( 3,11) = Z( 2, 2) + Z( 2, 5)*W( 9) + Z( 2, 8)*W(18)
Z( 3,12) = Z( 2, 3) + Z( 2, 6)*W( 9) + Z( 2, 9)*W(18)
Z( 3,13) = Z( 2,10) + Z( 2,13)*W(12) + Z( 2,16)*W(24)
Z( 3,14) = Z( 2,11) + Z( 2,14)*W(12) + Z( 2,17)*W(24)
Z( 3,15) = Z( 2,12) + Z( 2,15)*W(12) + Z( 2,18)*W(24)
Z( 3,16) = Z( 2,19) + Z( 2,22)*W(15) + Z( 2,25)*W( 3)
Z( 3,17) = Z( 2,20) + Z( 2,23)*W(15) + Z( 2,26)*W( 3)
Z( 3,18) = Z( 2,21) + Z( 2,24)*W(15) + Z( 2,27)*W( 3)
Z( 3,19) = Z( 2, 1) + Z( 2, 4)*W(18) + Z( 2, 7)*W( 9)
Z( 3,20) = Z( 2, 2) + Z( 2, 5)*W(18) + Z( 2, 8)*W( 9)
Z( 3,21) = Z( 2, 3) + Z( 2, 6)*W(18) + Z( 2, 9)*W( 9)
Z( 3,22) = Z( 2,10) + Z( 2,13)*W(21) + Z( 2,16)*W(15)
Z( 3,23) = Z( 2,11) + Z( 2,14)*W(21) + Z( 2,17)*W(15)
Z( 3,24) = Z( 2,12) + Z( 2,15)*W(21) + Z( 2,18)*W(15)
Z( 3,25) = Z( 2,19) + Z( 2,22)*W(24) + Z( 2,25)*W(21)
Z( 3,26) = Z( 2,20) + Z( 2,23)*W(24) + Z( 2,26)*W(21)
Z( 3,27) = Z( 2,21) + Z( 2,24)*W(24) + Z( 2,27)*W(21)


Z( 4, 1) = Z( 3, 1) + Z( 3, 2)*W( 0) + Z( 3, 3)*W( 0)
Z( 4, 2) = Z( 3, 4) + Z( 3, 5)*W( 1) + Z( 3, 6)*W( 2)
Z( 4, 3) = Z( 3, 7) + Z( 3, 8)*W( 2) + Z( 3, 9)*W( 4)
Z( 4, 4) = Z( 3,10) + Z( 3,11)*W( 3) + Z( 3,12)*W( 6)
Z( 4, 5) = Z( 3,13) + Z( 3,14)*W( 4) + Z( 3,15)*W( 8)
Z( 4, 6) = Z( 3,16) + Z( 3,17)*W( 5) + Z( 3,18)*W(10)
Z( 4, 7) = Z( 3,19) + Z( 3,20)*W( 6) + Z( 3,21)*W(12)
Z( 4, 8) = Z( 3,22) + Z( 3,23)*W( 7) + Z( 3,24)*W(14)
Z( 4, 9) = Z( 3,25) + Z( 3,26)*W( 8) + Z( 3,27)*W(16)
Z( 4,10) = Z( 3, 1) + Z( 3, 2)*W( 9) + Z( 3, 3)*W(18)
Z( 4,11) = Z( 3, 4) + Z( 3, 5)*W(10) + Z( 3, 6)*W(20)
Z( 4,12) = Z( 3, 7) + Z( 3, 8)*W(11) + Z( 3, 9)*W(22)
Z( 4,13) = Z( 3,10) + Z( 3,11)*W(12) + Z( 3,12)*W(24)
Z( 4,14) = Z( 3,13) + Z( 3,14)*W(13) + Z( 3,15)*W(26)
Z( 4,15) = Z( 3,16) + Z( 3,17)*W(14) + Z( 3,18)*W( 1)
Z( 4,16) = Z( 3,19) + Z( 3,20)*W(15) + Z( 3,21)*W( 3)
Z( 4,17) = Z( 3,22) + Z( 3,23)*W(16) + Z( 3,24)*W( 5)
Z( 4,18) = Z( 3,25) + Z( 3,26)*W(17) + Z( 3,27)*W( 7)
Z( 4,19) = Z( 3, 1) + Z( 3, 2)*W(18) + Z( 3, 3)*W( 9)
Z( 4,20) = Z( 3, 4) + Z( 3, 5)*W(19) + Z( 3, 6)*W(11)
Z( 4,21) = Z( 3, 7) + Z( 3, 8)*W(20) + Z( 3, 9)*W(13)
Z( 4,22) = Z( 3,10) + Z( 3,11)*W(21) + Z( 3,12)*W(15)
Z( 4,23) = Z( 3,13) + Z( 3,14)*W(22) + Z( 3,15)*W(17)
Z( 4,24) = Z( 3,16) + Z( 3,17)*W(23) + Z( 3,18)*W(19)
Z( 4,25) = Z( 3,19) + Z( 3,20)*W(24) + Z( 3,21)*W(21)
Z( 4,26) = Z( 3,22) + Z( 3,23)*W(25) + Z( 3,24)*W(23)
Z( 4,27) = Z( 3,25) + Z( 3,26)*W(26) + Z( 3,27)*W(25)
```

(e)

```
EQUATIONS OF THE FAST FOURIER TRANSFORM.
          FOR N = 30 =  2 * 3 * 5

Z( 2, 1) = Z( 1, 1) + Z( 1,16)*W( 0)
Z( 2, 2) = Z( 1, 2) + Z( 1,17)*W( 0)
Z( 2, 3) = Z( 1, 3) + Z( 1,18)*W( 0)
Z( 2, 4) = Z( 1, 4) + Z( 1,19)*W( 0)
Z( 2, 5) = Z( 1, 5) + Z( 1,20)*W( 0)
Z( 2, 6) = Z( 1, 6) + Z( 1,21)*W( 0)
Z( 2, 7) = Z( 1, 7) + Z( 1,22)*W( 0)
Z( 2, 8) = Z( 1, 8) + Z( 1,23)*W( 0)
Z( 2, 9) = Z( 1, 9) + Z( 1,24)*W( 0)
Z( 2,10) = Z( 1,10) + Z( 1,25)*W( 0)
Z( 2,11) = Z( 1,11) + Z( 1,26)*W( 0)
Z( 2,12) = Z( 1,12) + Z( 1,27)*W( 0)
Z( 2,13) = Z( 1,13) + Z( 1,28)*W( 0)
Z( 2,14) = Z( 1,14) + Z( 1,29)*W( 0)
Z( 2,15) = Z( 1,15) + Z( 1,30)*W( 0)
Z( 2,16) = Z( 1, 1) + Z( 1,16)*W(15)
Z( 2,17) = Z( 1, 2) + Z( 1,17)*W(15)
Z( 2,18) = Z( 1, 3) + Z( 1,18)*W(15)
Z( 2,19) = Z( 1, 4) + Z( 1,19)*W(15)
Z( 2,20) = Z( 1, 5) + Z( 1,20)*W(15)
Z( 2,21) = Z( 1, 6) + Z( 1,21)*W(15)
Z( 2,22) = Z( 1, 7) + Z( 1,22)*W(15)
Z( 2,23) = Z( 1, 8) + Z( 1,23)*W(15)
Z( 2,24) = Z( 1, 9) + Z( 1,24)*W(15)
Z( 2,25) = Z( 1,10) + Z( 1,25)*W(15)
Z( 2,26) = Z( 1,11) + Z( 1,26)*W(15)
Z( 2,27) = Z( 1,12) + Z( 1,27)*W(15)
Z( 2,28) = Z( 1,13) + Z( 1,28)*W(15)
Z( 2,29) = Z( 1,14) + Z( 1,29)*W(15)
Z( 2,30) = Z( 1,15) + Z( 1,30)*W(15)


Z( 3, 1) = Z( 2, 1) + Z( 2, 6)*W( 0) + Z( 2,11)*W( 0)
Z( 3, 2) = Z( 2, 2) + Z( 2, 7)*W( 0) + Z( 2,12)*W( 0)
Z( 3, 3) = Z( 2, 3) + Z( 2, 8)*W( 0) + Z( 2,13)*W( 0)
Z( 3, 4) = Z( 2, 4) + Z( 2, 9)*W( 0) + Z( 2,14)*W( 0)
Z( 3, 5) = Z( 2, 5) + Z( 2,10)*W( 0) + Z( 2,15)*W( 0)
Z( 3, 6) = Z( 2,16) + Z( 2,21)*W( 5) + Z( 2,26)*W(10)
Z( 3, 7) = Z( 2,17) + Z( 2,22)*W( 5) + Z( 2,27)*W(10)
Z( 3, 8) = Z( 2,18) + Z( 2,23)*W( 5) + Z( 2,28)*W(10)
Z( 3, 9) = Z( 2,19) + Z( 2,24)*W( 5) + Z( 2,29)*W(10)
Z( 3,10) = Z( 2,20) + Z( 2,25)*W( 5) + Z( 2,30)*W(10)
Z( 3,11) = Z( 2, 1) + Z( 2, 6)*W(10) + Z( 2,11)*W(20)
Z( 3,12) = Z( 2, 2) + Z( 2, 7)*W(10) + Z( 2,12)*W(20)
Z( 3,13) = Z( 2, 3) + Z( 2, 8)*W(10) + Z( 2,13)*W(20)
Z( 3,14) = Z( 2, 4) + Z( 2, 9)*W(10) + Z( 2,14)*W(20)
Z( 3,15) = Z( 2, 5) + Z( 2,10)*W(10) + Z( 2,15)*W(20)
Z( 3,16) = Z( 2,16) + Z( 2,21)*W(15) + Z( 2,26)*W( 0)
Z( 3,17) = Z( 2,17) + Z( 2,22)*W(15) + Z( 2,27)*W( 0)
Z( 3,18) = Z( 2,18) + Z( 2,23)*W(15) + Z( 2,28)*W( 0)
Z( 3,19) = Z( 2,19) + Z( 2,24)*W(15) + Z( 2,29)*W( 0)
Z( 3,20) = Z( 2,20) + Z( 2,25)*W(15) + Z( 2,30)*W( 0)
Z( 3,21) = Z( 2, 1) + Z( 2, 6)*W(20) + Z( 2,11)*W(10)
Z( 3,22) = Z( 2, 2) + Z( 2, 7)*W(20) + Z( 2,12)*W(10)
Z( 3,23) = Z( 2, 3) + Z( 2, 8)*W(20) + Z( 2,13)*W(10)
Z( 3,24) = Z( 2, 4) + Z( 2, 9)*W(20) + Z( 2,14)*W(10)
Z( 3,25) = Z( 2, 5) + Z( 2,10)*W(20) + Z( 2,15)*W(10)
Z( 3,26) = Z( 2,16) + Z( 2,21)*W(25) + Z( 2,26)*W(20)
Z( 3,27) = Z( 2,17) + Z( 2,22)*W(25) + Z( 2,27)*W(20)
Z( 3,28) = Z( 2,18) + Z( 2,23)*W(25) + Z( 2,28)*W(20)
Z( 3,29) = Z( 2,19) + Z( 2,24)*W(25) + Z( 2,29)*W(20)
Z( 3,30) = Z( 2,20) + Z( 2,25)*W(25) + Z( 2,30)*W(20)
```

(f)

```
Z( 4, 1) = Z( 3, 1) + Z( 3, 2)*W( 0) + Z( 3, 3)*W( 0) + Z( 3, 4)*W( 0)
    + Z( 3, 5)*W( 0)
Z( 4, 2) = Z( 3, 6) + Z( 3, 7)*W( 1) + Z( 3, 8)*W( 2) + Z( 3, 9)*W( 3)
    + Z( 3,10)*W( 4)
Z( 4, 3) = Z( 3,11) + Z( 3,12)*W( 2) + Z( 3,13)*W( 4) + Z( 3,14)*W( 6)
    + Z( 3,15)*W( 8)
Z( 4, 4) = Z( 3,16) + Z( 3,17)*W( 3) + Z( 3,18)*W( 6) + Z( 3,19)*W( 9)
    + Z( 3,20)*W(12)
Z( 4, 5) = Z( 3,21) + Z( 3,22)*W( 4) + Z( 3,23)*W( 8) + Z( 3,24)*W(12)
    + Z( 3,25)*W(16)
Z( 4, 6) = Z( 3,26) + Z( 3,27)*W( 5) + Z( 3,28)*W(10) + Z( 3,29)*W(15)
    + Z( 3,30)*W(20)
Z( 4, 7) = Z( 3, 1) + Z( 3, 2)*W( 6) + Z( 3, 3)*W(12) + Z( 3, 4)*W(18)
    + Z( 3, 5)*W(24)
Z( 4, 8) = Z( 3, 6) + Z( 3, 7)*W( 7) + Z( 3, 8)*W(14) + Z( 3, 9)*W(21)
    + Z( 3,10)*W(28)
Z( 4, 9) = Z( 3,11) + Z( 3,12)*W( 8) + Z( 3,13)*W(16) + Z( 3,14)*W(24)
    + Z( 3,15)*W( 2)
Z( 4,10) = Z( 3,16) + Z( 3,17)*W( 9) + Z( 3,18)*W(18) + Z( 3,19)*W(27)
    + Z( 3,20)*W( 6)
Z( 4,11) = Z( 3,21) + Z( 3,22)*W(10) + Z( 3,23)*W(20) + Z( 3,24)*W( 0)
    + Z( 3,25)*W(10)
Z( 4,12) = Z( 3,26) + Z( 3,27)*W(11) + Z( 3,28)*W(22) + Z( 3,29)*W( 3)
    + Z( 3,30)*W(14)
Z( 4,13) = Z( 3, 1) + Z( 3, 2)*W(12) + Z( 3, 3)*W(24) + Z( 3, 4)*W( 6)
    + Z( 3, 5)*W(18)
Z( 4,14) = Z( 3, 6) + Z( 3, 7)*W(13) + Z( 3, 8)*W(26) + Z( 3, 9)*W( 9)
    + Z( 3,10)*W(22)
Z( 4,15) = Z( 3,11) + Z( 3,12)*W(14) + Z( 3,13)*W(28) + Z( 3,14)*W(12)
    + Z( 3,15)*W(26)
Z( 4,16) = Z( 3,16) + Z( 3,17)*W(15) + Z( 3,18)*W( 0) + Z( 3,19)*W(15)
    + Z( 3,20)*W( 0)
Z( 4,17) = Z( 3,21) + Z( 3,22)*W(16) + Z( 3,23)*W( 2) + Z( 3,24)*W(18)
    + Z( 3,25)*W( 4)
Z( 4,18) = Z( 3,26) + Z( 3,27)*W(17) + Z( 3,28)*W( 4) + Z( 3,29)*W(21)
    + Z( 3,30)*W( 8)
Z( 4,19) = Z( 3, 1) + Z( 3, 2)*W(18) + Z( 3, 3)*W( 6) + Z( 3, 4)*W(24)
    + Z( 3, 5)*W(12)
Z( 4,20) = Z( 3, 6) + Z( 3, 7)*W(19) + Z( 3, 8)*W( 8) + Z( 3, 9)*W(27)
    + Z( 3,10)*W(16)
Z( 4,21) = Z( 3,11) + Z( 3,12)*W(20) + Z( 3,13)*W(10) + Z( 3,14)*W( 0)
    + Z( 3,15)*W(20)
Z( 4,22) = Z( 3,16) + Z( 3,17)*W(21) + Z( 3,18)*W(12) + Z( 3,19)*W( 3)
    + Z( 3,20)*W(24)
Z( 4,23) = Z( 3,21) + Z( 3,22)*W(22) + Z( 3,23)*W(14) + Z( 3,24)*W( 6)
    + Z( 3,25)*W(28)
Z( 4,24) = Z( 3,26) + Z( 3,27)*W(23) + Z( 3,28)*W(16) + Z( 3,29)*W( 9)
    + Z( 3,30)*W( 2)
Z( 4,25) = Z( 3, 1) + Z( 3, 2)*W(24) + Z( 3, 3)*W(18) + Z( 3, 4)*W(12)
    + Z( 3, 5)*W( 6)
Z( 4,26) = Z( 3, 6) + Z( 3, 7)*W(25) + Z( 3, 8)*W(20) + Z( 3, 9)*W(15)
    + Z( 3,10)*W(10)
Z( 4,27) = Z( 3,11) + Z( 3,12)*W(26) + Z( 3,13)*W(22) + Z( 3,14)*W(18)
    + Z( 3,15)*W(14)
Z( 4,28) = Z( 3,16) + Z( 3,17)*W(27) + Z( 3,18)*W(24) + Z( 3,19)*W(21)
    + Z( 3,20)*W(18)
Z( 4,29) = Z( 3,21) + Z( 3,22)*W(28) + Z( 3,23)*W(26) + Z( 3,24)*W(24)
    + Z( 3,25)*W(22)
Z( 4,30) = Z( 3,26) + Z( 3,27)*W(29) + Z( 3,28)*W(28) + Z( 3,29)*W(27)
    + Z( 3,30)*W(26)
```

(g)

```
LIST

FFT       20:19   THURS. 07/11/68

100 INTEGER GAMMA,SPACE,S,V,G,U,D,F1,F2,F3,F4,F5,F6
110 DIMENSION Z(512),A(512),W(256),Y(256)
120 $FILE DAPLOT,FTD0/FTD1/FTD2/FTD3/FTD4/FTD5/FTD6/FTD7/FTD8/FTD9
130 READ (2,11) N1,T1,SW1;IF(SW1)41,31,41
140 31:PRINT 32,N1,T1;GOTO 42;41:PRINT 43,N1,T1;42:PRINT 51;PRINT 52,,
150 INPUT,N,SPACE;PHI=6.2831853/N;C=20/LOG(10);DELF=1/N/T1/SPACE
160 F1=N1/N/SPACE;F2=(N1+509)/510;F4=N/4+1
170 F5=(2*N1+509)/510;F6=(2*N+5)/6;M=N/2;K1=510
180 GAMMA=LOG(N)/LOG(2)+.1;N3=N*2;D=M;S=1
190 IF(-SW1) 71;D0 62,J=1,F2;IF(J-F2) 64;K1=N*SPACE*F1-(F2-1)*510
200 64:READ(2,21) (A(K),K=1,K1);D0 62,L=1,510
210 V=V+1;IF(V-N*SPACE*F1) 61,61;L=510;GOTO 62
220 61:IF(AMOD(V-1,SPACE)) 62,63,62-63:F3=AMOD(F3,N)+1
230 Z(2*F3-1)=Z(2*F3-1)+A(L);62:
240 GOTO 72;71:D0 72,J=1,F5;IF(J-F5) 77;K1=N*SPACE*F1*2-(F5-1)*510
250 77:READ(2,21) (A(K),K=1,K1);D0 72,L=1,510
260 V=V+1;IF(V-2*SPACE*N*F1) 76,76;L=510;GOTO 72:76:N2=AMOD(V-1,2*SPACE)
270 IF(N2) 74,73,74:73:F3=AMOD(F3,N)+1;Z(2*F3-1)=Z(2*F3-1)+A(L)
280 GOTO 72:74:IF(N2-1) 72,75,72:75:Z(2*F3)=Z(2*F3)+A(L); 72:
290 D0 91,J=1,M;W(J)=COS(PHI*(J-1))
300 91:W(N-M)=SIGN(SIN(PHI*(J-1)),.5-SW1)
310 D0 101,J1=1,GAMMA
320 D0 111,L1=1,S;D0 111,L=1,D
330 I=2*L+2*(L1-1)*D;K=2*AMOD(L+(L1-1)*D*2,N);U=K+2*D
340 G=(L1-1)*D+1
350 B1=Z(U-1)*W(G)-Z(U)*W(G+M);B2=Z(U-1)*W(G+M)+Z(U)*W(G)
360 A(I-1)=Z(K-1)+B1;A(I)=Z(K)+B2
370 A(I+N-1)=Z(K-1)-B1;111:A(I+N)=Z(K)-B2
380 D=D/2;S=S*2
390 D0 101,J=1,N3;101:Z(J)=A(J)
400 D0 141,J=1,N;141:Y(AMOD(J+M-1,N)+1)=SQRT(Z(2*J-1)↑2+Z(2*J)↑2)/F1/N
410 PRINT†††;PRINT 151;D0 194,J=1,N;194:W(J)=(J-M-1)*DELF;PRINT 152
420 PRINT††;D0 161,J=F4,M;IF(Y(2*J-1)) 171,173,171
430 171:PRINT 172,W(2*J-1),Y(2*J-1),C*LOG(Y(2*J-1)),;GOTO 182
440 173:PRINT 174,W(2*J-1),Y(2*J-1),;182:IF(Y(2*J)) 181,182,181
450 181:PRINT 172,W(2*J),Y(2*J),C*LOG(Y(2*J));GOTO 161
460 182:PRINT 174,W(2*J),Y(2*J);161:
470 WRITE(1,192) I;WRITE(1,193) N
480 WRITE(1,196) (W(J),J=1,N);WRITE(1,196) (Y(J),J=1,N)
490 WRITE(1,201);ENDFILE 1;REWIND 2;WRITE(2,11) N,DELF,1-SW1
500 WRITE(2,21) (Z(K),K=1,N3);ENDFILE 2
510 PRINT†††;PRINT 202,DELF
520 32:FORMAT(6HINPUT:,I5,22H REAL SAMPLES; PERIOD ,F8.4,6H UNITS)
530 43:FORMAT(6HINPUT:,I5,25H COMPLEX SAMPLES; PERIOD ,F8.4,6H UNITS)
540 51:FORMAT(68HHOW MANY COMPONENTS, N, ARE DESIRED AND WHAT SHOULD
550 +BE THE INTERVAL )
560 52:FORMAT(26H     BETWEEN SAMPLES, SPACE)
570 151:FORMAT(X,7HSECONDS,3X,9HAMPLITUDE,4X,5HPOWER,9X,
580 +8H SECONDS,3X,9HAMPLITUDE,4X,5HPOWER)
590 152:FORMAT(7H OR CPS,6X,5HX1000,9X,2HDB,11X,6HOR CPS,6X,5HX1000,
600 +8X,2HDB)
610 172:FORMAT(F8.3,X,3PF10.4,2X,0PF8.3,8X);81:FORMAT(6E11.5)
620 174:FORMAT(F8.3,X,3PF10.4,5X,2H--,11X)
630 196:FORMAT(4F10.4);201:FORMAT(30 SPECTRUM . . . . . .)
640 192:FORMAT(I1);193:FORMAT(I3);11:FORMAT(I4,F8.4,I1)
650 202:FORMAT(19H    THE INTERVAL IS ,F10.6,7H UNITS.)
660 END
```

Fig. 4 Fast Fourier transform program for radix two.

```
FFT       19:35   THURS. 07/11/68

INPUT: 1280 REAL SAMPLES; PERIOD     .0010 UNITS
HOW MANY COMPONENTS, N, ARE DESIRED AND WHAT SHOULD BE THE INTERVAL
    BETWEEN SAMPLES, SPACE? 128,10

 SECONDS   AMPLITUDE     POWER          SECONDS   AMPLITUDE     POWER
 OR CPS      X1000        DB            OR CPS      X1000        DB

    .000    62.5000   -24.082             .781    62.1054   -24.137
   1.563    60.9304   -24.303            2.344    58.0011   -24.583
   3.125    56.3603   -24.981            3.906    53.0661   -25.594
   4.688    49.1909   -26.162            5.469    44.8193   -26.971
   6.250    40.0456   -27.949            7.031    34.8719   -29.126
   7.813    29.7053   -30.543            8.594    24.3553   -32.268
   9.375    19.0305   -34.411           10.156    13.8369   -37.179
  10.937     8.8745   -41.037           11.719     4.2350   -47.163
  12.500      .0000  -202.378           13.281     3.7611   -48.494
  14.062     6.9926   -43.107           14.844     9.6537   -40.306
  15.625    11.7189   -38.622           16.406    13.1789   -37.602
  17.187    14.0396   -37.053           17.969    14.3223   -36.880
  18.750    14.0621   -37.039           19.531    13.3072   -37.518
  20.312    12.1165   -38.332           21.094    10.5584   -39.528
  21.875     8.7080   -41.202           22.656     6.6451   -43.550
  23.437     4.4519   -47.029           24.219     2.2104   -53.111
  25.000      .0000  -201.690           25.781     2.1045   -53.537
  26.563     4.0350   -47.893           27.344     5.7321   -44.834
  28.125     7.1464   -42.918           28.906     8.2399   -41.682
  29.688     9.9862   -40.928           30.469     9.3720   -40.563
  31.250     9.3960   -40.541           32.031     9.0695   -40.848
  32.813     8.4150   -41.499           33.594     7.4658   -42.539
  34.375     6.2639   -44.063           35.156     4.8592   -46.268
  35.938     3.3072   -49.611           36.719     1.6672   -55.560
  37.500      .0000  -210.033           38.281     1.6336   -55.737
  39.063     3.1753   -49.964           39.844     4.5711   -46.800
  40.625     5.7728   -44.772           41.406     6.7400   -43.427
  42.188     7.4408   -42.568           42.969     7.8532   -42.099
  43.750     7.9656   -41.976           44.531     7.7769   -42.184
  45.313     7.2968   -42.737           46.094     6.5451   -43.682
  46.875     5.5510   -45.113           47.656     4.3522   -47.226
  48.438     2.9933   -50.477           49.219     1.5246   -56.337


   THE INTERVAL IS     .781250 UNITS.


AT LINE NO. 660: STOP  END ,RAN 692/6 SEC.
```

Fig. 5. Listing of square pulse transform.

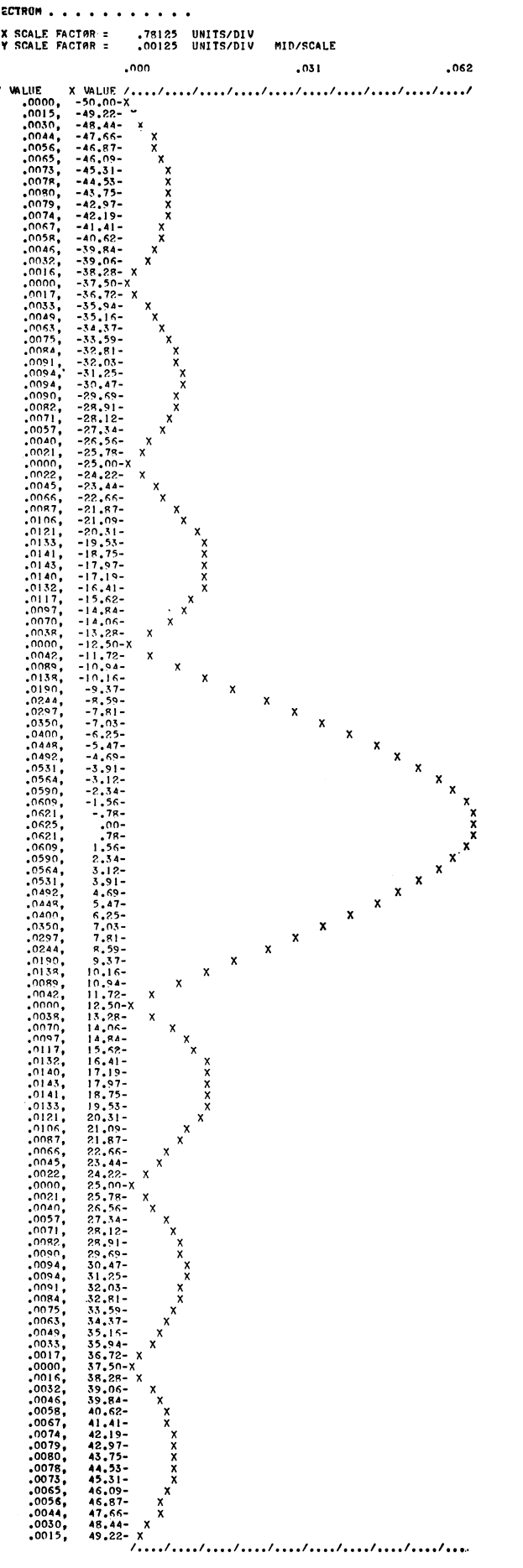

Fig. 6. Square pulse transform.

```
FFT       20:00   THURS. 07/11/68

INPUT:  128 COMPLEX SAMPLES; PERIOD    .7813 UNITS
HOW MANY COMPONENTS, N, ARE DESIRED AND WHAT SHOULD BE THE INTERVAL
    BETWEEN SAMPLES, SPACE? 128,1


 SECONDS   AMPLITUDE     POWER          SECONDS   AMPLITUDE     POWER
 OR CPS      X1000        DB           OR CPS      X1000        DB

    .000  1000.0002       .000            .010  1000.0005       .000
    .020  1000.0008       .000            .030  1000.0018       .000
    .040  1000.0020       .000            .050  1000.0015       .000
    .060  1000.0001       .000            .070   999.9978      -.000
    .080      .0022  -113.274             .090      .0002  -135.611
    .100      .0010  -120.101             .110      .0017  -115.461
    .120      .0023  -112.954             .130      .0002  -133.676
    .140      .0005  -125.615             .150      .0016  -115.805
    .160      .0006  -123.875             .170      .0015  -116.501
    .180      .0009  -120.818             .190      .0005  -126.789
    .200      .0015  -116.401             .210      .0006  -125.028
    .220      .0016  -116.005             .230      .0030  -110.482
    .240      .0025  -112.181             .250      .0000  -175.465
    .260      .0010  -120.319             .270      .0010  -119.875
    .280      .0007  -122.913             .290      .0006  -124.169
    .300      .0006  -124.306             .310      .0006  -124.569
    .320      .0019  -114.402             .330      .0055  -105.172
    .340      .0030  -110.356             .350      .0003  -131.277
    .360      .0012  -118.669             .370      .0019  -114.422
    .380      .0005  -126.754             .390      .0018  -114.943
    .400      .0023  -112.846             .410      .0003  -129.501
    .420      .0020  -114.124             .430      .0040  -107.914
    .440      .0013  -117.743             .450      .0013  -117.718
    .460      .0002  -132.407             .470      .0019  -114.637
    .480      .0001  -137.099             .490      .0024  -112.495
    .500      .0020  -114.010             .510      .0022  -113.012
    .520      .0030  -110.340             .530      .0005  -125.814
    .540      .0016  -116.180             .550      .0003  -131.633
    .560      .0011  -119.325             .570      .0014  -116.842
    .580      .0005  -126.457             .590      .0007  -122.598
    .600      .0034  -109.485             .610      .0040  -107.982
    .620      .0005  -125.575             .630      .0016  -115.675


  THE INTERVAL IS    .009999 UNITS.


AT LINE NO. 660: STOP  END ,RAN 433/6 SEC.
```

Fig. 7. Listing of restored pulse.

```
SPECTRUM . . . . . .

 X SCALE FACTOR =    .01000  UNITS/DIV
 Y SCALE FACTOR =    .02000  UNITS/DIV   MID/SCALE

                        .000                    .500                1.000

Y VALUE    X VALUE /..../..../..../..../..../..../..../..../..../
    .0000,    -.64-X
    .0000,    -.63-X
    .0000,    -.62-X
    .0000,    -.61-X
    .0000,    -.60-X
    .0000,    -.59-X
    .0000,    -.58-X
    .0000,    -.57-X
    .0000,    -.56-X
    .0000,    -.55-X
    .0000,    -.54-X
    .0000,    -.53-X
    .0000,    -.52-X
    .0000,    -.51-X
    .0000,    -.50-X
    .0000,    -.49-X
    .0000,    -.48-X
    .0000,    -.47-X
    .0000,    -.46-X
    .0000,    -.45-X
    .0000,    -.44-X
    .0000,    -.43-X
    .0000,    -.42-X
    .0000,    -.41-X
    .0000,    -.40-X
    .0000,    -.39-X
    .0000,    -.38-X
    .0000,    -.37-X
    .0000,    -.36-X
    .0000,    -.35-X
    .0000,    -.34-X
    .0000,    -.33-X
    .0000,    -.32-X
    .0000,    -.31-X
    .0000,    -.30-X
    .0000,    -.29-X
    .0000,    -.28-X
    .0000,    -.27-X
    .0000,    -.26-X
    .0000,    -.25-X
    .0000,    -.24-X
    .0000,    -.23-X
    .0000,    -.22-X
    .0000,    -.21-X
    .0000,    -.20-X
    .0000,    -.19-X
    .0000,    -.18-X
    .0000,    -.17-X
    .0000,    -.16-X
    .0000,    -.15-X
    .0000,    -.14-X
    .0000,    -.13-X
    .0000,    -.12-X
    .0000,    -.11-X
    .0000,    -.10-X
    .0000,    -.09-X
    .0000,    -.08-X
    .0000,    -.07-X
    .0000,    -.06-X
    .0000,    -.05-X
    .0000,    -.04-X
    .0000,    -.03-X
    .0000,    -.02-X
    .0000,    -.01-X
 1.0000,     .00-                                                    X
 1.0000,     .01-                                                    X
 1.0000,     .02-                                                    X
 1.0000,     .03-                                                    X
 1.0000,     .04-                                                    X
 1.0000,     .05-                                                    X
 1.0000,     .06-                                                    X
 1.0000,     .07-
    .0000,    .08-X
    .0000,    .09-X
    .0000,    .10-X
    .0000,    .11-X
    .0000,    .12-X
    .0000,    .13-X
    .0000,    .14-X
    .0000,    .15-X
    .0000,    .16-X
    .0000,    .17-X
    .0000,    .18-X
    .0000,    .19-X
    .0000,    .20-X
    .0000,    .21-X
    .0000,    .22-X
    .0000,    .23-X
    .0000,    .24-X
    .0000,    .25-X
    .0000,    .26-X
    .0000,    .27-X
    .0000,    .28-X
    .0000,    .29-X
    .0000,    .30-X
    .0000,    .31-X
    .0000,    .32-X
    .0000,    .33-X
    .0000,    .34-X
    .0000,    .35-X
    .0000,    .36-X
    .0000,    .37-X
    .0000,    .38-X
    .0000,    .39-X
    .0000,    .40-X
    .0000,    .41-X
    .0000,    .42-X
    .0000,    .43-X
    .0000,    .44-X
    .0000,    .45-X
    .0000,    .46-X
    .0000,    .47-X
    .0000,    .48-X
    .0000,    .49-X
    .0000,    .50-X
    .0000,    .51-X
    .0000,    .52-X
    .0000,    .53-X
    .0000,    .54-X
    .0000,    .55-X
    .0000,    .56-X
    .0000,    .57-X
    .0000,    .58-X
    .0000,    .59-X
    .0000,    .60-X
    .0000,    .61-X
    .0000,    .62-X
    .0000,    .63-X
              /..../..../..../..../..../..../..../..../..../
```

Fig. 8. Restored pulse.

References

[1] W. T. Cochran, J. W. Cooley, D. L. Favin, H. D. Helms, R. A. Kaenel, W. W. Lang, G. C. Maling, Jr., D. E. Nelson, C. M. Rader, and P. D. Welch, "What is the fast Fourier transform," *Proc. IEEE*, vol. 55, pp. 1664–1674, October 1967.

[2] V. W. Cooley, P. A. W. Lewis, and P. D. Welch, "Historical notes on the fast Fourier transform," *Proc. IEEE*, vol. 55, pp. 1675–1677, October 1967.

[3] E. O. Brigham and R. W. Morrow, "The fast Fourier transform," *IEEE Spectrum*, vol. 4, pp. 63–70, December 1967.

[4] V. W. Cooley and J. W. Tukey, "An algorithm for the machine calculation of complex Fourier series," *Math. Comp.*, vol. 19, pp. 297–301, 1965.

[5] J. L. Shanks and T. W. Cairns, "Use of a digital convolution device to perform recursive filtering and the Cooley–Tukey algorithm," *IEEE Trans. Computers*, vol. C-17, pp. 943–949, October 1968.

[6] R. C. Singleton, "ALGOL procedures for the fast Fourier transform," *Commun. ACM*, vol. 11, pp. 773–777, November 1968.

[7] M. C. Pease, "An adaptation of the fast Fourier transform for parallel processing," *J. ACM*, vol. 15, pp. 252–264, April 1968.

[8] R. C. Singleton, "An ALGOL procedure for the fast Fourier transform with arbitrary factors," *Commun. ACM*, vol. 11, pp. 777–779, November 1968.

[9] W. M. Gentleman and G. Sande, "Fast Fourier transforms—for fun and profit," *1966 Fall Joint Computer Conf., AFIPS Proc.*, vol. 29. Washington, D. C.: Spartan, 1966, pp. 563–578.

A Scheme for Synchronizing High-Speed Logic Part II

HERSCHEL H. LOOMIS, JR., MEMBER, IEEE

Abstract—**In this paper we concern ourselves with the problem of obtaining high sequence rate sequential machines, machines which are constructed from realistic devices to operate at an input sequence rate which is independent of the machine complexity. To accomplish this result we have only to show a construction to realize acceptably synchronous devices from badly timed, restricted fan-in and fan-out devices. Once a complete set of synchronous devices is obtained, the results of Arden and Arthurs [1] apply and we know that any finite state machine has a realization using these devices which accepts input sequence members at a rate which is characteristic of the set of devices, not of the machine. The technique we propose for achieving this result is to produce a lattice of interconnected clock pulse sources called clock pulse propagators (CPPs). These devices generate clock pulses which are acceptably synchronized with respect to the outputs of neighboring CPPs but are not required to be in synchronization with some machine-wide standard as in current practice. Once it is established that such a network is possible, techniques already known can be applied in the utilization of the clock-pulses to synchronize logic and signals. Part I[1] of the paper concerns the analysis of CPP networks, and Part II covers the synthesis of sequential machines using CPP networks as clocking sources.**

Index Terms—**Badly timed devices, clocked logic, completeness of synchronous logic, high-speed logic, maximum rate construction.**

Introduction

IN this part of the paper, we examine the use of a lattice of clock sources in conjunction with badly timed logic and storage devices (latches) to produce a complete set of synchronous logic devices for finite state machine synthesis purposes. The principal result of this part of the paper will be to provide a set of sufficient conditions for a set of badly timed logic devices to be complete in the sense of Loomis [1] and a design procedure for a clocked system. We will assume a lattice of clock pulse propagators (CPPs) as defined in Part I of this paper as our clock source, although the synchronizing method does not depend on the use of this source. Any source for which certain parameters can be bounded will suffice.

We first develop the constraints on the construction of the synchronous building blocks and on the physical spacing of the clock pulse lattice; then we present the theorem concerning completeness of badly timed devices. The paper is concluded with the presentation of a design procedure and an example.

General Discussion

The systems in which we shall use the CPP lattice as a clocking source shall consist of three elements:

1) the CPP lattice;
2) networks of gates which produce the desired switching functions;
3) latch circuits which synchronize the outputs of the gate networks with respect to the clock pulse sources.

Let us first consider the operation of the latch, and then we shall investigate the design constraints imposed by our maximum rate objective. The latch has been described in Maley and Earle [2]. It can be thought of

Manuscript received November 4, 1967; revised August 4, 1969. This paper is an extension of results reported in "A Theory of High-Speed Clocked Logic," presented at the 6th Annual Symposium on Switching Circuit Theory and Logical Design, October 1965. The work reported was supported by National Science Foundation Grant NSF GP3002.

H. H. Loomis, Jr., is with the Department of Electrical Engineering, University of California, Davis, Calif. 95616.

[1] H. H. Loomis, Jr., and M. R. McCoy, *IEEE Trans. Computers*, vol. C-19, pp. 39–47, January 1970.