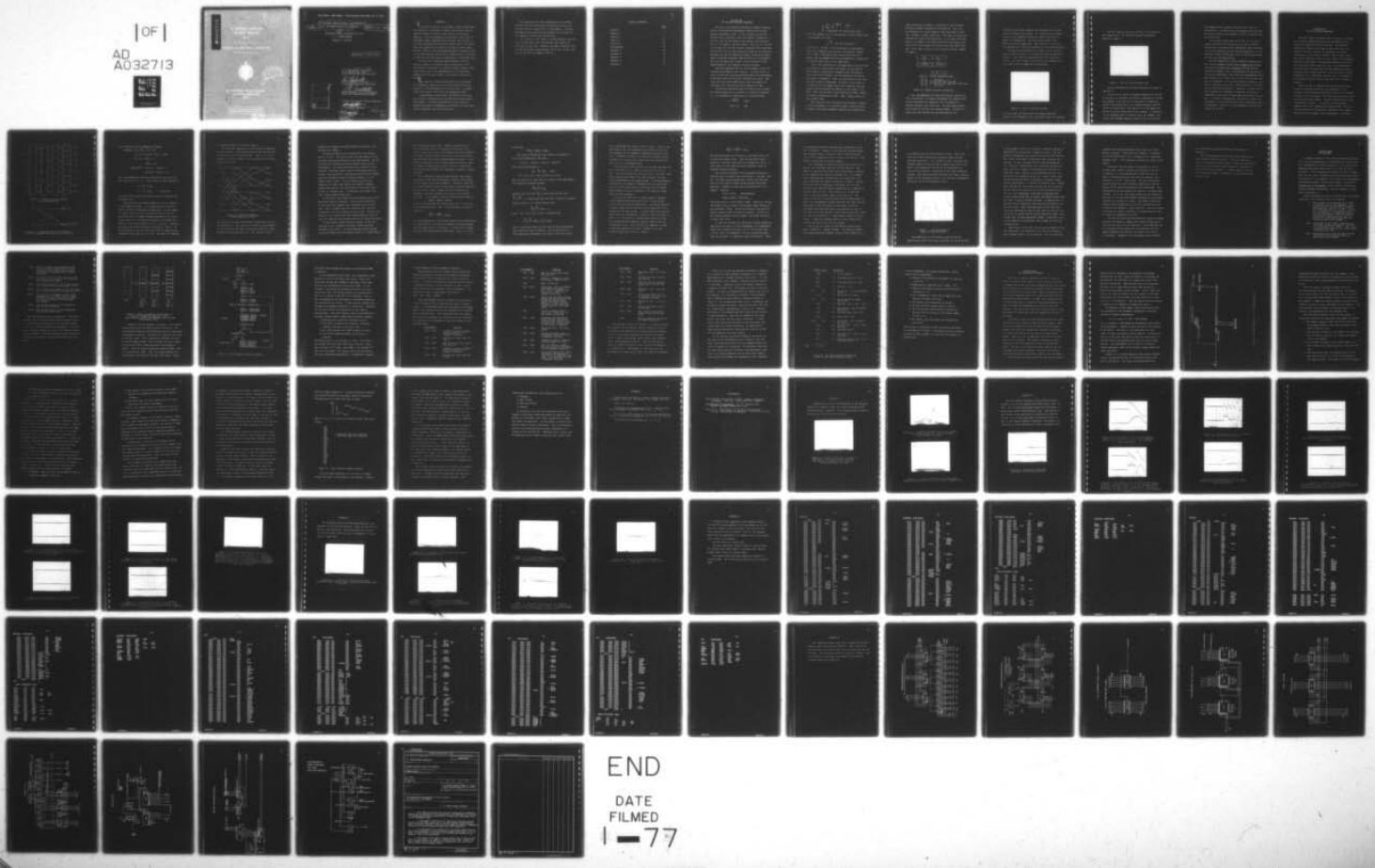
AD-A032 713 NAVAL ACADEMY ANNAPOLIS MD F/G 14/2
FREQUENCY ANALYSIS USING A MINICOMPUTER.(U)
MAY 76 D C BOCH
UNCLASSIFIED USNA-TSPR-75 NL



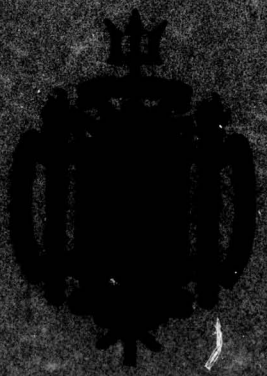
END

DATE FILMED
1-77

PROJECT REPORT

NO. 76

FREQUENCY ANALYSIS USING A MINICOMPUTE



UNITED STATES NAVAL ACADEMY
ANNAPOLIS, MARYLAND
1976

FREQUENCY ANALYSIS USING A MINICOMPUTER

(21) Report on A Trident Scholar Project Report (9) Research rept.,

by

(10) Midshipman 1/C D. C. Boch, Class 1976
U. S. Naval Academy
Annapolis, Maryland

David A Wright
D. A. WRIGHT, MAJ, CAE
Electrical Engineering Department

F. J. Eberhardt
F. J. EBERHARDT, Professor
Electrical Engineering Department

H. M. Neustadt
H. M. NEUSTADT, Associate Professor
Electrical Engineering Department

Accepted for Trident Scholar Committee

C. W. RECTOR, Chairman

(11) 17 MAY 76
Date

(12) 87 p.

245 600
by d

ABSTRACT

The spectral analysis of waveforms, whether these waves are acoustic or electrical in nature, has evolved into an important aspect of quite a wide variety of scientific endeavors. Utilized primarily by the Navy in the study of underwater sound, frequency analysis also finds utility in research on mechanical vibrations, speech, music, et cetera. Real-time capability is necessary for many of these applications. That is, the transformation must be completed within the time interval over which its sampled data is acquired so that spectral plots may be generated on a continuing basis.

The basis for a computer-aided frequency analysis scheme is known as a Discrete Fourier Transform, or DFT. The inherent flexibility of a general purpose computer lends itself quite well to the implementation of a high speed adaptation of the DFT, called a Fast Fourier Transform or FFT.

This study was initiated with the aim of determining the fundamental capabilities and limitations of one minicomputer in the development of a spectral plot. Included within this general topic were two fundamental goals:

1) Development of the fastest FFT possible utilizing only the basic capabilities of the PDP-8/E general purpose minicomputer. Successful completion of this objective brought the machine's fundamental limitations for real-time spectral analysis into clear perspective.

2) Investigation of the feasibility of an external hardware "add-on" which would considerably increase the spectral analysis capabilities of the computer. Proving worthwhile, the design of such hardware was launched but not completed, due to time restrictions.

As a result of this study, it became apparent that all three systems (DFT, FFT, FFT hardware) are capable of operation in real time. However, the upper frequency limit for the DFT would be approximately 50 Hz, for the FFT, 500 Hz, and for the FFT hardware, 2500 Hz.

TABLE OF CONTENTS

CHAPTER ONE:
THE DISCRETE FOURIER TRANSFORM

The first step towards developing a frequency analysis system on the PDP-8/E minicomputer involved the Discrete Fourier Transform, or DFT. As the ultimate goal of the research was to develop a real-time analysis capability, it was felt that the DFT would be the logical starting point. This is so primarily for two reasons. First of all, the DFT is the basis for the Fast Fourier Transform (FFT) algorithm. The FFT was to be the main course of study for this analysis. Secondly, the DFT is relatively simple, allowing the opportunity to become thoroughly familiar with the minicomputer, while at the same time, to gain a solid foothold in the understanding of the frequency analysis problem.

The basic approach used in the development of the DFT software was primarily very cautious. In other words, in order to avoid any overflow problems within the programming, triple precision (36 bit word) arithmetic was used. Likewise, as a full field (4096 points) was the simplest to implement, the first DFT was built around this idea.

The Fourier Transform itself is actually quite simple. The energy proportion, w, present at a specific frequency, ℓ, in a waveform of N data points is as given below.

$$w = \sqrt{a^2+b^2} \qquad \text{where}$$

$$a+jb = S_\ell \qquad \text{and}$$

$$S_\ell = \sum_{n=0}^{N-1} X_{n\ell}\, e^{j\frac{2\pi}{N} n\ell} \qquad \text{where}$$

X_n = "amplitude" of a given data point.

Since the computer can not distinguish a complex number from a real number, the term

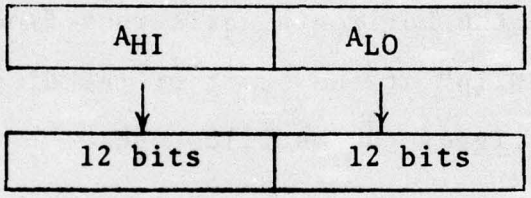
$$e^{j\frac{2\pi}{N}} \quad \text{must be represented}$$

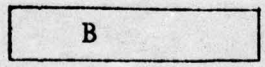
as $\cos \frac{2\pi}{N} + j \sin \frac{2\pi}{N}$. This necessitates a trigonometric table to be computed and stored in the PDP-8/E's memory. As N = 4096, a 4096 point SINE curve was written into memory, and the COSINE terms were determined by a simple 90° (1/4 field) phase shift of the SINE table.

Due to the uncertainty of the magnitude of each S_ℓ term which would be formed and of the degree of significance which should be maintained throughout each calculation, triple precision arithmetic was developed and utilized at all stages of summation. Thus, the possibility of any overflow was virtually eliminated. Overflow is the result of the addition of two numbers whose sum is greater than the largest word the machine can handle, in this case, 2^{12} or 4096. Hence the sum 3874 + 2521 would not be 6395. Rather, it would be represented as $6395 - 4096 = 2299_{10}$, which is 6395, modulo 4096.

When concerned with both positive and negative numbers in the minicomputer, the seemingly simple triple precision arithmetic can appear somewhat confusing upon application.

Double precision arithmetic is available in the Extended Arithmetic Element installed on the PDP-8/E. Thus, only the inclusion of a third, highest order word poses a problem. Consider a double order case. The same principles are involved for triple order precision. If a single precision word, B, is added to the double precision word $A_{HI}A_{LO}$, four cases exist which require an action on the high order word A_{HI}. These cases, as well as the necessary actions, appear in Figure 1.





A_{LO}	B	Action required on A_{HI}
>0	>0	no action
>0	<0	if Link set, Add 1 to A_{HI}
<0	>0	if Link not set, Subtract 1 from A_{HI}
<0	<0	no action

Figure 1: Triple Precision Operations.

After approximately one and a half months, the DFT was fully operational. In order to determine its effectiveness, several waveforms were sampled by the minicomputer's Analog-to-Digital (A/D) converter. The 4096 sample points were stored in core and the transform begun. In order to assure that the software was running properly, the

resultant S_ℓ for each frequency was displayed on an oscilloscope as that S_ℓ was formed. As a result, the transform could be seen "growing," which was rather impressive to watch, if nothing else. The DFT required nearly five minutes to run to completion. This run-time is based on the computation of only 512 frequencies of the 4096 possible.

The first test waveform was a simple 100 Hz, 10 volt peak-to-peak sine wave. The energy distribution appears in Figure 2. Note that the horizontal axis runs from zero to 512 Hz. The first "blip" at the start of the display is a sync pulse used to trigger the oscilloscope.

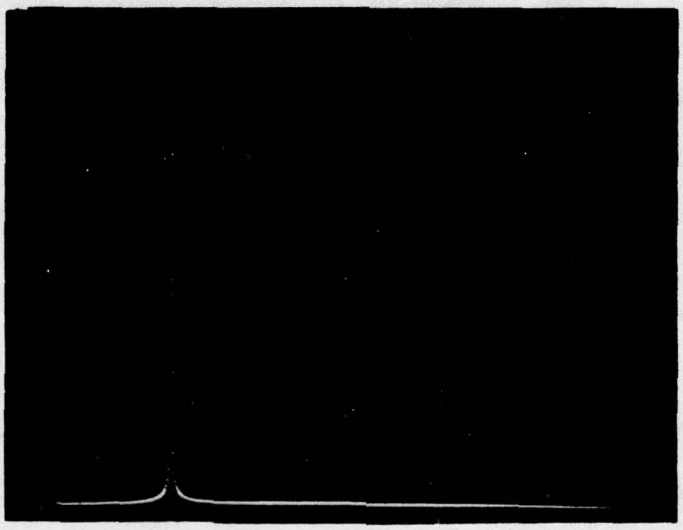


Figure 2: DFT of a 100 Hz Sine Wave

As can be seen, the point with the highest magnitude occurs at the frequency of the sinusoidal input, as expected.

The next logical test was a 100 Hz, 10 volt peak-to-peak square wave. The resultant energy distribution appears in Figure 3.

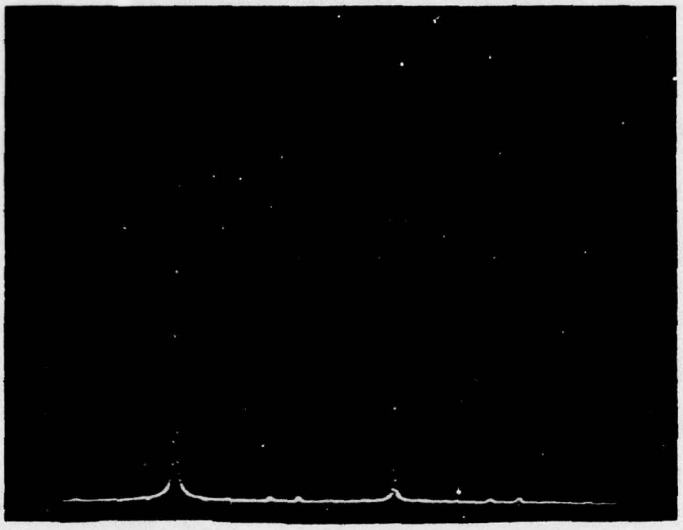


Figure 3: DFT of a 100 Hz Square Wave

Energy distributions for other waveforms are shown in Appendix A.

The ability to choose a minimum frequency for the DFT to operate on is probably the greatest advantage of the DFT. For example, if one were only interested in frequencies greater than twenty Hertz, the minimum frequency would be chosen as twenty Hertz, allowing 532 Hz as the upper frequency in one 512 point pass of the program. In addition, if the frequency band of interest were, for example, 125 - 185 Hz, the minimum frequency could be set at 125 Hz and

the program could be halted after the first sixty S_ℓ calculations. In other words, the program would only have to run for 60/512 X five minutes in order to analyze this particular band.

The primary disadvantage of the DFT is its lack of speed, especially when considering the entire spectrum. Considering a one second data acquisition time and perhaps a one second display time (at the very greatest), a transform which takes five minutes to run is wholly unsuitable for any real-time application.

The DFT proved to be a very effective learning tool. The extensive amount of time spent debugging the program later proved valuable when debugging the Fast Fourier Transform software. Also, the actual use of the machine's A/D and D/A converters, real-time clock and Extended Arithmetic Element was a better way to learn of their use than were the handbooks. As the system proved absolutely incapable of running in real-time, further pursuit of the DFT, was not considered worthwhile. However, it became obvious while running test waveforms through the system that a logarithmic scaler on the output data would prove very useful. On further investigation, the idea was found to be infeasible in that nearly all low order significance would be destroyed. Hence, it too was abandoned, and work was started on the FFT.

CHAPTER TWO:
THE FAST FOURIER TRANSFORM

The Fast Fourier Transform (FFT) was begun in the hope that it would significantly decrease the amount of time required to process any given waveform. The equations involved are basically the same as those used for calculation of the DFT. However, no transform is actually calculated, in the sense of the DFT. The FFT used (there are several different types) is quite simple on the surface. Basically, what occurs is successive reduction of one DFT into two smaller DFT's until the point is reached where a two point DFT is divided down into two single point DFT's. Actually, these single point DFT's are the relative energy proportions for a particular spectral line. An example of this reduction process for an eight point FFT is shown in Figure 4.

While the DFT is characterized as requiring N^2 operations to perform, the FFT has the advantage of only needing to perform $N\log_2 N$ operations, where N is the number of points in the transform. The DFT calculations were laid out in the previous chapter. The FFT calculations are best illustrated by Figure 5. $\tilde{Z}_a$ and $\tilde{Z}_b$ are both complex numbers; $e^{mj\theta}$ can be expressed as $\cos m\theta + j\sin m\theta$. The use of sine and cosine necessitates the compilation of a trigonometric table for storage in the minicomputer. The basic

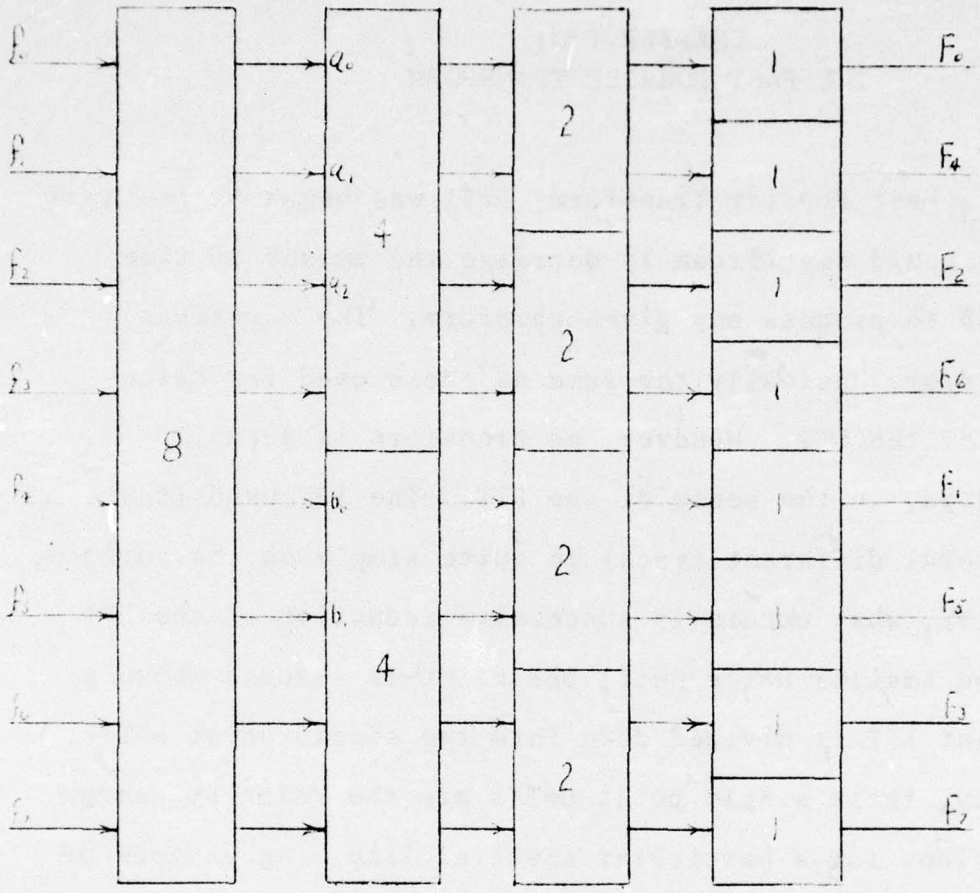


Figure 4: Reduction Process Involved in an Eight Point FFT.[1]

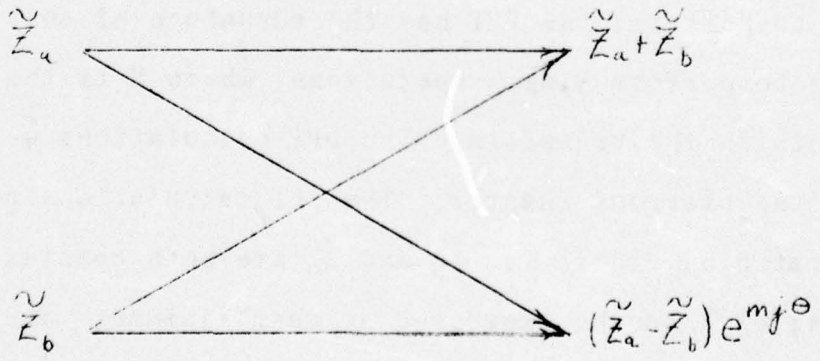


Figure 5: A Demonstration of the Formation of Two Points Required for the Division of One DFT.

FFT calculation can be expanded as follows.

Assuming, $\tilde{Z}_a = A+jB \quad \tilde{Z}_b = C+jD$

$$\cos m\theta + j \sin m\theta = X+jY \quad \text{then,}$$

$$\tilde{Z}_a + \tilde{Z}_b = (A+C) = A'$$
$$+$$
$$j(B+D) = jB'$$

$$(\tilde{Z}_a - \tilde{Z}_b)e^{mj\theta} = [(A-C)\ X\ -\ (B-D)Y] = C'$$
$$+$$
$$j[(A-C)Y + (B-D)X] = jD' \ .$$

Thus, the mathematics portion of the FFT program forms the four terms A', B', C', D'. These terms are then combined as

$$A' + jB' = \tilde{Z}_{a+1}$$

$$C' + jD' = \tilde{Z}_{b+1} \qquad \text{to form the}$$

new set of DFTs, which will then be ready for division into two other DFT's.

Choosing which two complex numbers are to be combined in the manner of Figure 5 is perhaps the most difficult portion of the FFT algorithm to understand. In the example of Figures 4 and 6, the first word in each of the two four point DFT's is formed through combination of f_0 and f_4. In this case, $f_0 = A+jB$ where $B=0$ and $f_4 = C+jD$ where $D=0$. Then, $a_o = A+C + j(o)$ and $b_o = (A-C)\ x + j(A-C)Y$. Likewise, a_1 and b_1 are formed by properly combining f_1 and f_5. The first position in the first two point DFT is formed through combination of a_o and a_2, and so on. The manner of indexing

is explained further in the next chapter.

Note that the subscripts on the frequency components are out of order. Careful observation shows that the subscripts are in a bit-reversed order. That is, F_4 is in position 1 ($4=100_2$, $1=001_2$). F_6 is in position 3 ($6=110_2$, $3=011_2$), and so on.

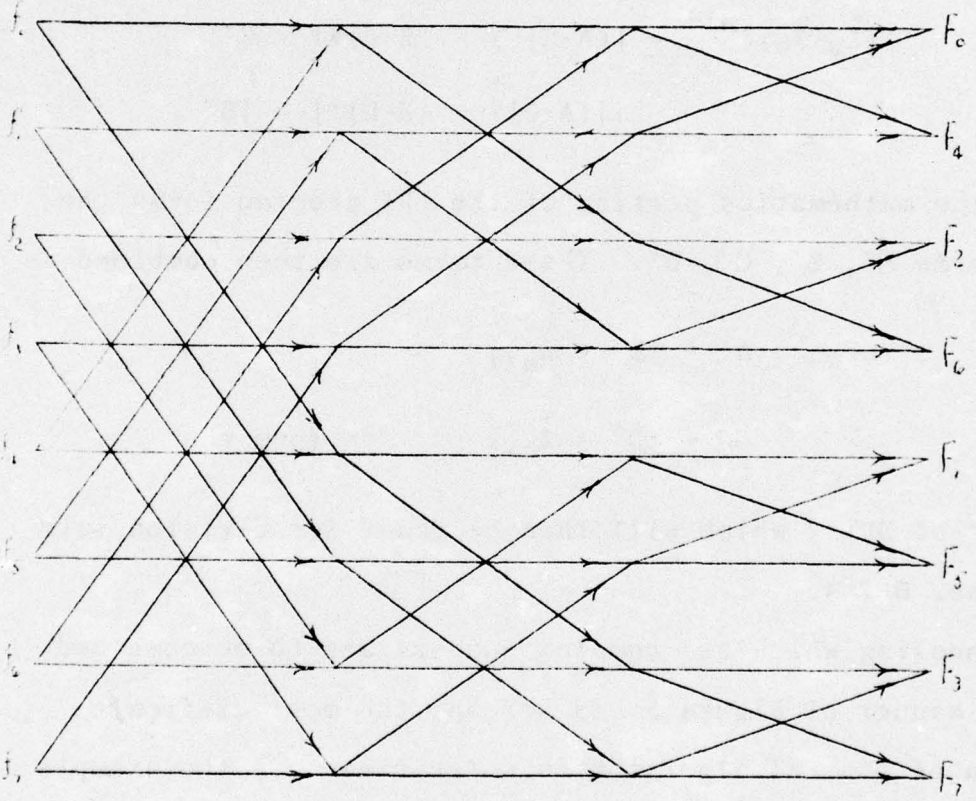


Figure 6: Method of Combining Expressions for the FFT.[2]

Bit reversal is easily and uniquely accomplished by use of the buffered I/O. The subscript (actually a core address) is put out on the I/O lines, reversed by means of a cable and put back in on the I/O line. The contents at the

original core address are then placed in the correct, bit-reversed core address.

Two major obstacles were encountered in the development of the FFT. The first concern was what degree of significance should be maintained in the calculations. The DFT indicated that double precision arithmetic would be most advantageous. By using double precision, round-off errors (actually truncation errors) would be eliminated when performing a multiplication. Multiplying two twelve bit numbers using the Extended Arithmetic Element produces a 24 bit (double order) result. If only single precision were required, the lower order twelve bits would be dropped. Through this process, the significance of the lower order 12 bits is obviously lost. In addition, any product less than $1\ 0000_8$ (4096) becomes zero.

On the other hand, double precision addition and store operations require, obviously, twice as many instruction cycles within the computer and, therefore, twice as much time. Also, a double precision store will not clear the accumulator, as will a single precision store. Further, the amount of core required for data storage would be twice as great for double precision operations as for single precision.

As the primary concern of this research is the time required to run a complete transform, single precision arithmetic was chosen. The time advantage gained through the use of single precision greatly offsets the significance

lost through multiplication. Likewise, the extra core space available allows a transform of 512 points to be located in one data field. With double precision arithmetic, the largest transform which can be run in one data field is only half the size of the aforementioned transform. As the range of frequencies to be analyzed depends primarily upon the number of sampled data points, the use of single precision operations affords the capability to examine a larger spectrum.

The second major problem became apparent when considering how to avoid overflow problems when not using double, or even triple, precision arithmetic. Obviously, some sort of scaling scheme had to be devised where overflow could not possibly occur. With a simple addition, dividing each operand by two before adding will result in no overflow occurring. As an example, consider (with unsigned numbers)

$$6000_8 + 6000_8 = 14000_8 .$$

In this case, overflow does indeed occur (the 1 in the sum is the thirteenth bit). Now, if each operand is scaled as suggested,

$$\frac{6000_8}{2} + \frac{6000_8}{2} = 6000_8 .$$

Since all operands are scaled before addition (or subtraction) the 6000_8 which is the sum has a somewhat different meaning than the 6000_8 which is an operand. In other words, before addition, the operands have become 3000_8. Hence it is easy

to see that

$$3000_8 + 3000_8 = 6000_8 .$$

This same divide-by-two scale factor also applies itself in the formation of the term

$$[(A-C)X - (B-D)Y] + j[(A-C)Y + (B-D)X] ,$$

which can be represented as

$$|\tilde{Z}_a - \tilde{Z}_b| \; |\tilde{w}_m| \quad \text{where}$$

$\tilde{Z}_a = A+jB$, $\tilde{Z}_b = C+jD$, and $\tilde{w}_m = X_m + jY_m$.

When the terms (A-C) and (B-D) are scaled in the same manner, the relation essentially becomes

$$\left|\frac{\tilde{Z}_a - \tilde{Z}_b}{2}\right| \; |\tilde{w}_m| .$$

Based on the fact that $|\tilde{w}_m|$ is unity and the fact that $\left|\frac{\tilde{Z}_a - \tilde{Z}_b}{2}\right| \leq 1$ (considering $|\tilde{Z}_a|$ and $|\tilde{Z}_b|$ to always be bounded above by unity), it is quite obvious that

$$\left|\frac{\tilde{Z}_a - \tilde{Z}_b}{2}\right| \; |\tilde{w}_m| \leq 1 .$$

Since $|\tilde{Z}_a + \tilde{Z}_b| \leq |\tilde{Z}_a| + |\tilde{Z}_b|$, it follows that

$$\left|\frac{\tilde{Z}_a - \tilde{Z}_b}{2}\right| \; |\tilde{w}_m| \leq |\tilde{Z}_a| \text{ or } |\tilde{Z}_b| .$$

Thus, a scale down factor of two cannot cause overflow when going from one vector to another. By the nature of the data acquisition techniques, the terms in the first vector

have a magnitude less than or equal to unity. That is, the magnitude of the real components are limited to unity and the magnitude of the imaginary components, naturally, are zero. Therefore as there is no overflow in going from the zeroeth vector to the first, there can be no overflow anywhere else, as just demonstrated.

The data acquisition software is such that the data are not scaled from 0 to 1, rather a scale of 0 to 3777_8 is used. The A/D converter was set up so that a ten volt signal would generate the number 3777_8 and a negative ten volt signal would generate the number 4000_8. In the minicomputer architecture, the numbers 0 through 3777_8 are considered positive (3777_8 being the most positive), and the numbers 4000_8 through 7777_8 are considered to be negative (4000_8 being the most negative).[4]

It turns out that such a scaling scheme is optimum. As the foregoing discussion points out, a scale factor greater than two would be unnecessary. The result would be a steady, almost drastic decrease in the magnitude of the contents of the data vectors. A scale factor less than two would surely be insufficient. Consider a worst-case situation. If a ten volt D.C. waveform were read in as data, the transform should produce a spectral line at zero H_z, of magnitude 3777_8. Since the zero H_z component is found by repeated additions of A and C, scaled by two, the eventual result will be 3777_8. That is

$$\frac{3777_8}{2} + \frac{3777_8}{2} = 3777_8 \cdot$$

This would continue for all points in the data vector (all of which have value 3777_8). Thus the end result would indeed be a spectral line at zero H_z of magnitude 3777_8. A scale factor of less than two would therefore result in repeated overflow problems.

The evidence presented in the foregoing discussions indicate that a divide-by-two scale factor is optimum. However, one other scaling difficulty results from the truncation of the product formed through the MUY (multiply) operation. Perhaps the best way to describe this is by example. Consider

$3777_8 \times 3777_8$ Approximating,

$4000_8 \times 4000_8 = 2000\ 0000_8$.

Truncating results in the product 2000_8. Therefore, scaling the product up by 2 results in the proper answer 4000_8, or more precisely, 3776_8. The multiplication of two positive numbers should result in a positive product. As 3777_8 is the largest possible positive number, the result can be no "larger" than 3777_8.

In order to overcome this difficulty without adding any additional time delays in the programming, the trigonometric tables were scaled on the basis of "$\pm$" 7777_8 rather than $\pm$ 3777_8. Thus, by permanently scaling the multiplicands, there was no need to repeatedly scale the products. There

is an apparent deviation from the sign conventions used by the minicomputer. However, as the multiply operation works with unsigned numbers, another method of determining the sign of the trigonometric values needs to be utilized. This method is described in the next chapter.

The trigonometric tables discussed above differ in yet another way from those used with the DFT software. The number of spectral lines which can be calculated is limited by the sampling rate and the transform size. For example, with a sampling rate of 4096 points per second and a 512 point transform, the Nyquist frequency is 2048 Hz. Because of the way the FFT algorithm is designed, the first 256 spectral lines show the energy proportions up to 2048 Hz in discrete steps of 8 Hz. The second set of 256 spectral lines is the exact mirror image of the first set. The nature of the FFT is such that the first referenced point in one of the trigonometric tables after the 180° point has the same value as the first point in the table, and so on. Therefore, only half-wave tables are required for the FFT. The same is true for the DFT, except that with the DFT, there is a gain in transform speed when separate tables are used. Figure 7 shows the combined effects of the "forced full-scaling" and the use of half-wave tables.

Due to the very nature of the PDP-8/E, there exists what is known as a "paging" problem. The method of memory addressing within the computer results in the capability to

only address directly 128 locations in core. Those locations not within a particular 128 point "page" must be addressed indirectly.[3] As a result of this, the entire FFT program cannot be stored so that a simple loop will be run. Rather, the mathematics portion of the program must be indirectly accessed as a subroutine by the index controller program. The indexing program maintains the proper indirect address (to be explained later) of the data and trigonometric tables for use by the mathematics loop.

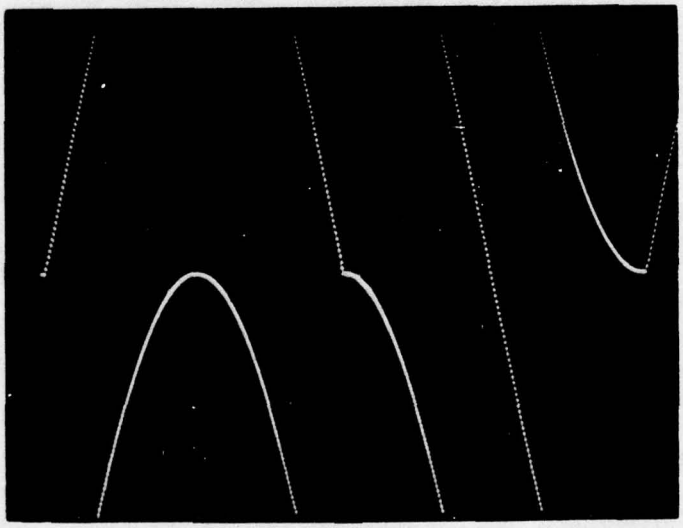


Figure 7: FFT Trigonometric Tables as Stored in Core.

The combination of the inherent speed of the FFT algorithm and those time-saving techniques discussed earlier

in this chapter result in a 512 point transform running to completion in 0.58 seconds. Though not quite a fair comparison (considering the relative size of the DFT) with the DFT, there is obviously a tremendous time advantage with the FFT. In fact, the goal of real-time processing has been realized. Considering a one second data acquisition time, the FFT is capable of performing the transform in sufficient time. However, as was discussed earlier, the results of the transform are not in a readily usable form. The operations that must be performed, including display, require approximately .3 seconds of processor time. Adding this time and the .58 seconds required for the actual FFT, and the sum easily nears the one second that would allow for real-time processing. Real-time as discussed here actually means being able to analyze and display data within the time it takes to acquire new data. A sampling rate of 4096 points per second will take exactly one second to fill one data field. Thus, the performance of the FFT could easily be completed in time for real-time for FFT's of size 512 and below (given the one second acquisition time), wide enough range for most underwater sounds. Large FFT's could not be completed in sufficient time to apply to real-time considerations.

With respect to the DFT, the one major drawback of the FFT algorithm is its inability to be selective insofar as what frequency band is to be analyzed. The transform must

calculate the energy proportions from zero Hz up to the Nyquist frequency. There does exist however, a variation on the FFT which allows for a "zooming in" on a specified frequency band. This variation is beyond the scope of this report.

Associated with the actual FFT program is a set of software which serves to organize the output of the FFT. In addition, it also acquires data, allows the user to select a desired portion of that data for analysis, and transfers it to FFT's data vector. The data acquisition routine allows for a user variable sampling rate and for display of the acquired waveform on an oscilloscope. Also incorporated into the acquisition routine is a movable data "window". As set up, the window allows for the viewing of the data 512 (or any other number, depending on the desired transform size) points at a time. The speed at which the window slides through the data field is controlled by the setting on the computer's switch register. Once the desired 512 points are chosen, the window is stopped and that data is loaded for operation by the FFT.

A second portion of the aforementioned software package performs the squaring, summing, and square-rooting operations on the results of the transform. Once completed, the bit-reversing routine takes over and unshuffles the frequency components and displays the final result on the oscilloscope. Examples of the transform results and both

the shuffled and unshuffled spectral lines appear in Appendix B.

The FFT performed quite well with the exception of a small amount of processing noise. There are numerous causes of noise of this type, most of which are unavoidable. Volumes have been written concerning noise, therefore it shall not be discussed here.[5] Several examples of transformed waveforms appear in Appendix C, along with descriptions of each.

CHAPTER THREE:
FFT SOFTWARE

A complete analysis of the actual FFT program is essential for the understanding of the difficulties involved and for a demonstration of how necessary it is for careful programming techniques (which result in as fast an FFT as possible). As was mentioned previously, the FFT is broken down into two areas, arithmetic operations and index control. The purpose of this chapter is not to teach the minicomputer programming language, therefore the reader is directed to Introduction to Programming, fourth edition, published by Digital Equipment Corporation. The FFT software is reproduced in Appendix D.

Before proceeding to discuss the operation of the index controller, the variables used will be defined.

M - This variable is initialized at 2^L, which is also the size of the transform. M is halved with each pass through the program. With each pass, the DFTs of size M are all forced into DFTs of size M/2. M serves as 1) a "counter" which guarantees that the transform goes to completion (DFTs of size one are formed), 2) an "incrementor" for the FFT's running indices, and 3) a definitive variable which indicates the size of the DFTs being formed.

IMAX - Initialized at one, IMAX is doubled with every pass through the program. It serves 1) to initialize ITAL (ITAL = - IMAX) 2) as a pointer within the sine and cosine tables to determine θ in $e^{jm\theta}$.

ITAL - Essentially ITAL counts the number of DFT pairs that must be operated upon with every pass of the program.

JTAL - JTAL is a counter which guarantees that each DFT block in Figure 4 gets filled. An example of how these four variables change for an eight point FFT is shown in Figure 8.

RFIND - This pointer for the first operand in the real vector is the same as "A" mentioned in the preceeding chapter.

RBIND - This variable points to the second operand in the real vector and corresponds to "C".

IMFIND - This variable corresponds to "B" and points to the first operand in the imaginary vector.

IMBIND - This variable corresponds to "D" in the imaginary vector. IMBIND, IMFIND, RBIND, and RFIND are known as running variables and are initialized by M for each DFT run through.

SWIND - This variable points to the appropriate word in the Sine table.

CWIND - This variable points to the appropriate word in the Cosine table.

The index controller has two functions. The first is to ensure that the correct data words are applied to the basic FFT calculation discussed in the previous chapter. As there is a definite pattern to the manner in which the indices progress through the vector, the indexing problem is eased somewhat. The second function of the controller is to halt the FFT once all single point DFTs have been formed.

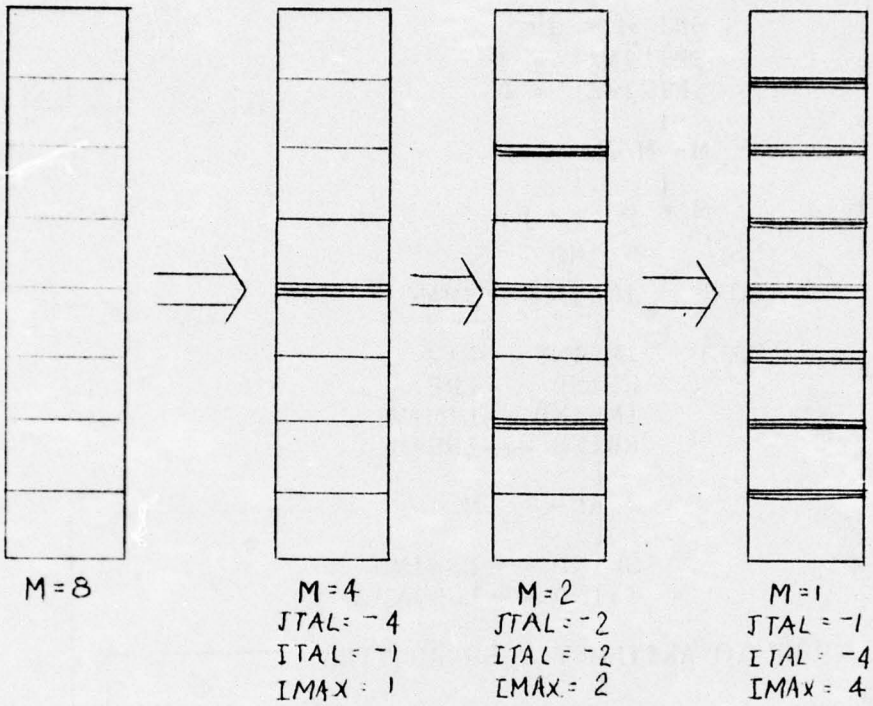


Figure 8: Values of counters at the start of each pass through the program. Those which start at values less than zero are run from that value to zero.

Examination of the flowchart in Figure 9 will clearly explain exactly how the index controller operates. The initialization shown at the beginning occurs only once when the program is started. Note that M is halved at the start of the first pass. This is because M indicates the size of the DFT being formed. Once M reaches zero (as a result of successive rotate right operations) the program is halted. With every pass through the program (LPASS), ITAL is initialized at -IMAX. Also, the running indices are initialized to the start of the DFTs last formed. Again,

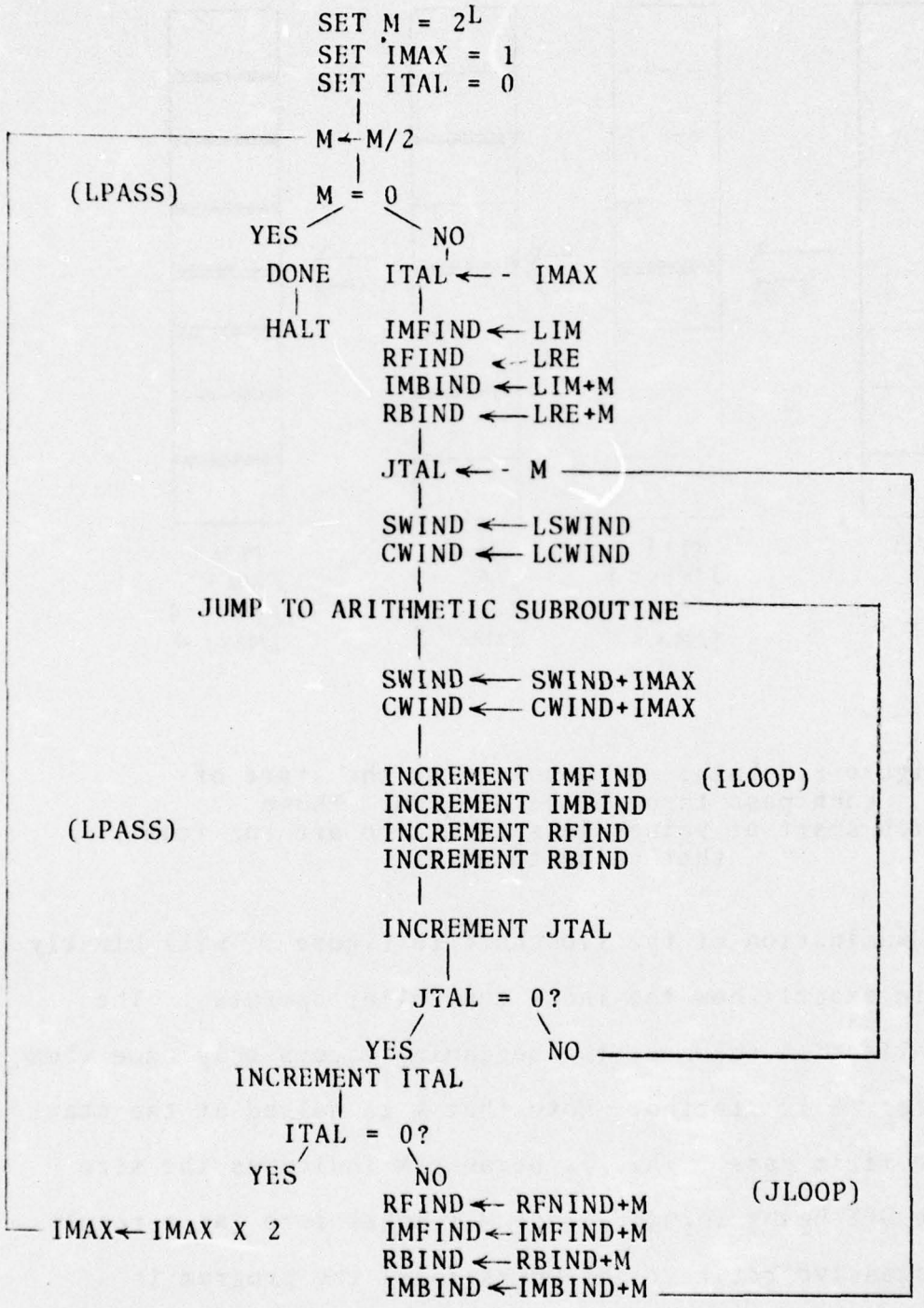


Figure 9: Flow Diagram for Index Controller

with every pass through the program, M is halved and IMAX is doubled.

At the beginning of the JLOOP, the trigonometric table pointers are set to the top of the tables. Then, with every pass through the arithmetic subroutine, these same pointers are incremented by the value of IMAX. As the arithmetic subroutine only performs one basic FFT calculation each time, the running indices are incremented by one so as to point to the next set of operands. ITAL is then incremented by one and checked to determine whether or not each new DFT are filled. If they are not, another pass through the arithmetic subroutine is called for. Otherwise, the next pair of DFTs are started on as the JLOOP is entered again. Once each complete vector transformation is finished, another LPASS is begun and the process repeats.

A timing analysis yields the following results.

1) The ILOOP, excluding the arithmetic subroutine, requires 22.8 μsec of processor time.
2) The JLOOP (without the ILOOP) requires 53.0μsec.
3) The initial setup and LPASS take 54.0μsec to complete.

The LPASS routine is run through L-1 times. The JLOOP is run through first one time, then twice, then four times, and so on up to 2^{L-1} times. The ILOOP is run through 2^{L-1} times for each LPASS. All totaled, the time required for the index controlling operations is insignificant compared

to that required for the arithmetic subroutine.

That portion of the FFT software which performs the actual calculations is called the arithmetic subroutine. A description of these calculations has already been presented. A discussion of how they are done does not require a flowchart, however, it may be helpful to refer to the proper portion of the software listed in Appendix D. The subroutine starts at line number 1030 which is

0000 0400 1030 FFTMATH 0.

Lines 1040 through 1110 are concerned with setting up a series of "flags" used to check for the sign of a given number. As the multiply operation only works with unsigned numbers, the absolute value of all operands must be formed before multiplication. After multiplication, the product must be given the proper sign before any further operation. The functions of the remainder of the lines appear in the following table.

Line Numbers	Function
1120 - 1150	Fetches A, scales it down by two and stores A/2 in a temporary location.
1160 - 1180	Fetches and scales down (by two) C.
1190 - 1200	Adds A/2 and C/2 and stores the sum as A´.
1210 - 1240	Fetches B, scales it down by two and stores B/2 in a temporary location.
1250 - 1270	Fetches and scales down (by two) D.

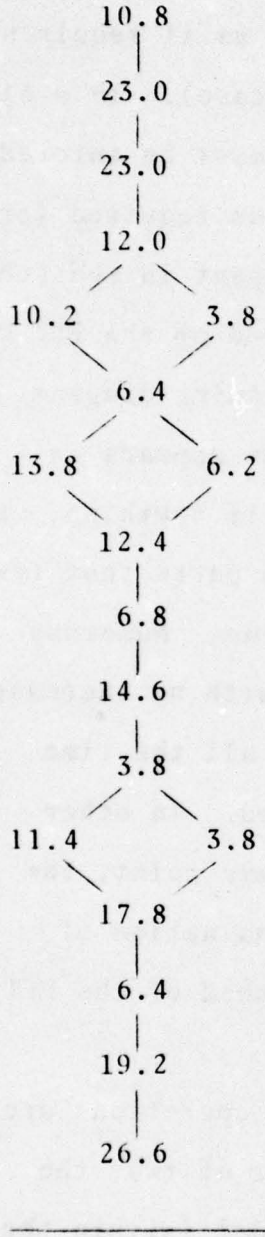
Line Numbers	Function
1280 - 1290	Adds B/2 and D/2 and stores the sum as B'.
1300 - 1330	Fetches C, changes its sign, and scales it down by two.
1340	Adds -C/2 and A/2.
1350 - 1410	Forms |A/2 - C/2|, increments the appropriate flags if A/2 - C/2 is negative, and stores |A/2 - C/2| in a temporary location.
1420 - 1440	Checks sign of cosine by comparing the addressed location within the cosine table against the table's middle location. Recall that the cosine table is a half-wave table.
1450	Causes a program jump to line 1475. Occurs only if the cosine is positive.
1460 - 1500	Increments the appropriate flags, fetches the cosine and forms the absolute value of the cosine. Occurs only if the cosine is negative.
1530 - 1570	Multiplies |A/2 - C/2| and |COS|.
1580 - 1610	Assigns the proper sign to the product and stores it in a temporary location.
1620 - 1650	Fetches D, scales it down by two and changes its sign.
1660 - 1680	Adds -D/2 and B/2, changes the sign of the sum and stores it in a temporary location.
1690 - 1770	Checks the sign of B/2 - D/2, increments the appropriate flags if it is negative, forms |B/2 - D/2| and stores it in a temporary location.

Line Numbers	Function
1780 - 1810	Multiplies |B/2 - D/2| and SIN.
1820 - 1830	Assigns the proper sign to the product.
1840 - 1850	Adds the last two products formed and stores the sum as C´.
1860 - 1890	Multiplies |B/2 - D/2| and |COS|.
1900 - 1920	Assigns the proper sign to the product and stores it in a temporary location.
1930 - 1960	Multiplies |A/2 - C/2| and SIN.
1970 - 1980	Assigns the proper sign to the product.
1990 - 2000	Adds the last two products formed and stores the sum as D´.
2010	Returns processor control to the index controller.

The remainder of the FFT software is a display routine which outputs the real vector and the imaginary vector on two different D/A channels for display on an oscilloscope. Line 180 in the index controller portion of the program causes a jump to the display routine after every complete vector transformation by the FFT. Such an instruction allows an interesting look at every step in the formation of a complete FFT. Photographs of each step of an FFT performed on a simple one Hz square wave appear in Appendix B.

Nearly all of the time required to generate a complete FFT is used up by the arithmetic subroutine as it requires approximately 217 μseconds per pass (worst case). In a 512 point transform, the arithmetic subroutine must be entered 9 x 256 times. Therefore, of the .58 seconds required for a complete 512 point FFT, .50 seconds are spent in the subroutine. In order to determine how the speed of the FFT can be improved, it is necessary to develop a timing diagram. Such a diagram for the arithmetic subroutine appears in Figure 10. The diagram shows that little, if anything, can be done to improve the speed. There are no parts that take any significantly long periods of time to run. Numerous variations of the program have been tried with no increase in speed whatsoever. Note that in summing all the time delays, the worst case was always considered. In other words, where there were two paths to the same point, the slowest path was always considered. An examination of other possible methods of increasing the speed of the FFT will be examined in the next chapter.

As the indexing system and arithmetic operations are the same for all FFTs whose size is a power of two, the basic system presented herein can be expanded (within the limits of memory) or contracted. The procedures necessary for such alterations are relatively straightforward. However, as a system expansion requires data field changes, and as data field changes are inherently tricky, expansion



Timing (μsec)	Operation
10.8	1. Set up indices.
23.0	2. Form and store A/2 + C/2.
23.0	3. Form and store B/2 + D/2.
12.0	4. Form A/2 - C/2.
10.2 3.8	5. Set up A/2 - C/2 for multiplication.
6.4	6. Check sign of cosine.
13.8 6.2	7. Set up cosine for multiplication.
12.4	8. Multiply |cos| x |A/2 - C/2|.
6.8	9. Apply proper sign to product and store.
14.6	10. Form and store -B/2 + D/2.
3.8	11. Fetch -B/2 + D/2.
11.4 3.8	12. Set up D/2 - B/2 for multiplication.
17.8	13. Multiply |D/2 - B/2| x |SIN| and sign.
6.4	14. Form and store C´.
19.2	15. Multiply |D/2 - B/2| x |cos| and sign.
26.6	16. Multiply |SIN| x |A/2 - C/2|, sign, and form D´.
Total 217 μsec	

Figure 10: FFT Timing diagram showing the steps and the time required for each.

is not recommended. For system contraction, several actions must be undertaken.

1) M must be initialized to the proper 2^L, the size of the FFT.
2) MCNEG must be redefined to be -3200_8 - 2^{L-2}.
3) MSIZE in the display routine of the FFT must be redefined as -2^L.
4) The trigonometric tables must be made half-wave based on 2^L points full wave.
5) The movable data window must be resized.
6) The data transfer program must be resized.
7) The bit-reversal routine must be resized and adjusted for the reversal of the proper number of bits (L-1).
8) The square, sum, and square root program must be resized.

The resizing is necessary so that the desired operations are performed on 2^L words. A discussion of how to properly perform the above changes is in the FFT Users Manual (in preparation).

CHAPTER FOUR:
THE "BLACK BOX" APPROACH

Since the arithmetic subroutine of the FFT requires such a relatively great amount of time to run (.50 seconds in comparison to the .58 seconds required for the entire FFT), the feasibility of utilizing an external device to perform the calculations was investigated. What is required is an extra set of hardware which will perform the basic FFT calculation described earlier. In order to be effective, the external device must be capable of high-speed operation so that the added time advantage will allow for real-time analysis. The device must also have a direct access to the minicomputer's memory for both the reading and writing of the data.

In order to access the required data quickly the device will have to do so directly from the PDP-8's memory. However, as the addresses of the desired data within memory change with every pass through the arithmetic subroutine, the device will have to access data in a pseudo-direct manner. That is, a set of addresses will need to be hard-wired into the device. The contents of memory at these locations will be the addresses of the six desired data words. The device will then store these addresses in its own memory for future reference. The data will then be acquired and stored by the device, again in its own memory.

Once the data has been acquired by the device, program

control will be returned to the processor for maximum effective use of time. While the device is performing the calculations, the processor will update the indices from the next calculation. When finished with all calculation, the device will signal the processor that the device is ready to accept control of the program. At this time, the device will load into memory the four results of the basic calculation into their proper locations (which were stored by the device originally). Thus the device need only operate as quickly as the computer can update the indices.

By operating in this manner, those calculations which were performed by the arithmetic subroutine in one half second will require approximately

$$16 \times 1.2\ \mu\text{sec} \times 9 \times 256 - .044 \text{ seconds.}$$

The 1.2 μseconds is the length of one machine cycle within the minicomputer. Sixteen of these cycles will be required by the device. The term 9 x 256 represents the total number of times the device would be used by a 512 point FFT. Thus it can be seen that what was once the slowest part of the FFT analysis has become nearly twice as fast as that which was once considered to be so fast as to have an almost negligible effect on the overall timing, namely the index controller.

Figure 11 is a block diagram of the external hardware device, indicating the flow of information between each part of the device. The figure also demonstrates the

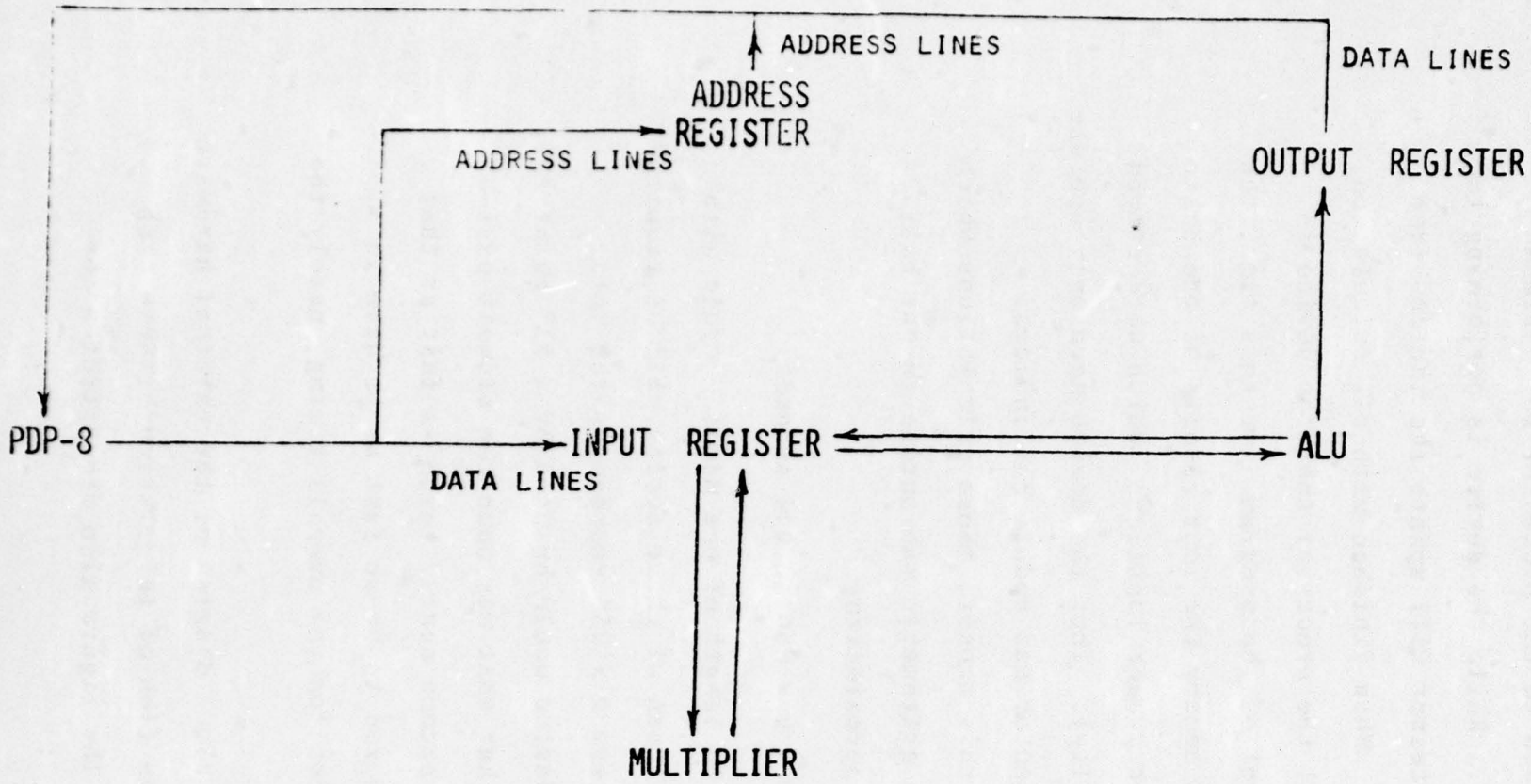


Figure 11: Block Diagram of an FFT Hardware Device

relationship between the device and the computer. The operations which occur within the hardware are best outlined as follows for a basic understanding of the function of the device.

The first step is initiated by means of an IOT instruction within the FFT index controller. The purpose of the instruction is to tell the device that it is to set up a so-called DMA (Direct Memory Access) state within the minicomputer. A DMA state allows a device external to the computer to directly access the memory of the computer. The time required for the transfer of one data word is 1.2 μ seconds, corresponding to one fast machine cycle.

Once the device acquires control of the memory, it fetches the necessary six data words from memory as follows:

1) The address register loads the first of the hard-wired memory addresses onto the minicomputer's MA (Memory Address) lines.
2) The contents of memory at that address are fed back into the address register for storage and readdressing of the memory.
3) The contents of memory at the latter address are then stored in the input register as the first data word.
4) The above three steps are repeated until all six data words (A, B, C, D, X, and Y) are stored in the input register. Note that, as the final output

from the device will be four words (A′, B′, C′, and D′) only the addresses of A, B, C, and D are kept by the address register.

With all of the data read into the device, control of the memory is relinquished and returned to the processor. As the processor updates the indices for the next pass through the external hardware, this same hardware performs the basic FFT calculation described in a previous chapter.

First the term (A+C), appropriately scaled, is formed and stored as A′ in the output register. Similarly, (B+D) is formed and stored as B′ in the output register. Then the terms (A-C) and (B-D) are formed and loaded into the input register for use by the multiplier. By use of the multiplier, the terms (A-C)X, (A-C)Y, (B-D)X, and (B-D)Y are formed and again stored in the input register. Finally, the Arithmetic Logic Unit (ALU) combines these last four terms into (A-C)X - (B-D)Y and (A-C)Y + (B-D)X and loads each into the output register as C′ and D′, respectively.

As soon as all data is loaded into the output register, the device sets up a break request, telling the processor that control of the memory is again desired. Upon completion of the current instruction, the processor will relinquish control of the memory. Then the device loads the minicomputer memory from the output register as follows:

1) The address loads the first stored address (that of A) onto the computer's MA lines.

2) The contents of the first location in the output register are loaded into the addressed location in memory.

3) The above steps are then repeated until all four data words are loaded into memory.

At this time, memory control is returned to the processor and the program continues from the point where the data break occurred.

The schematic diagrams of the address register, output register, input register and ALU can be found in Appendix E. Also included are diagrams of buffers for one set of input lines and output lines on both the ALU and multiplier, diagrams of the internal register addressing counters, of the device selection logic, and of the logic which activates the DMA state in the processor.

The multiplier is an array of individual 2x4 bit multiplier chips arranged to produce a 24 bit product from two twelve bit operands. As it operates with two's complement numbers there is no need to check for the sign of an operand before multiplication. Thus, the design of the hardware is simplified considerably.

The Arithmetic Logic Unit is a simple three chip device which must be capable of both addition and subtraction, the choice being made with one control line. It too is a twelve bit two's complement device. As both the ALU and multiplier devices are combinatorial logic devices,

no clocking is required for either. However, as there is no clock, there must exist a method of holding data on the outputs so a change on one of the inputs will not affect the desired answer. In order to read the outputs into the input register (as described earlier), the data at these outputs will necessarily be reapplied to the inputs. The consequences of not providing for an output latch could therefore be disastrous. Similarly, as both inputs to the ALU (as well as the multiplier) originate with the input register, a latch must be provided on one input line which will hold the proper data word while the other data word is being read from the input register.

The aforementioned latches must each be 12 bits wide. In addition, the clock should be edge triggerable, in order to avoid timing ambiguities which might otherwise arise as a result of too long (or too short) a clock pulse used with level triggering.

Likewise, the input, output, and addressing registers must all be twelve bits wide. As they are actually memories, however, and not simple latches, control signals are required for reading from, writing into, and addressing each of the registers. In order to read or write, the proper control line must be made low. On the other hand, the addressing is best performed by counters. Both the output and addressing registers are addressed sequentially from 0-3. The address sequence for the input register is not

quite so simple (Figure 12). The most convenient sequence yet contrived results in the input register holding the following words, in order from left to right.

Word						
0	A		(A-C)X			
1	B			(A-C)Y		
2	X				(B-D)X	
3	Y					(B-D)Y
4	C	(A-C)				
5	D		(B-D)			

where A, B, X, Y, C, and D are written in order from top to bottom.

```
000
001
010
011
100        * indicates that these addresses
101          occur two times in succession.
000
100*
001
101*
100
010
000
011
001
101
010*
011*
000
011
001
010
```

Figure 12: Input register address sequence.

As can be seen from Figure 12, the use of a simple counter utilizing J-K flip-flops is not feasible. However

a 5 bit counter can be used to address a ROM sequentially such that the ROM output at each address corresponds to the proper 3 bit code of Figure 12. The ROM address would be set to 00000 at the start of every pass through the device. The counter controlling the ROM address would be clocked every time a new location within the input register is to be accessed. The ROM (actually a PROM) would be by far the most economical approach, in terms of both money and hardware, as fewer components would be required with a ROM than without one.

It also appears that rather than suffer the design of a complicated hardware logic system to control the individual components within the FFT hardware, a ROM should be incorporated. Again, the ROM would be addressed sequentially by a counter which would be clocked every time another "instruction" was desired. In this instance, however, in order to ensure that all the outputs of the ROM change at the same time with a change in address, the outputs should be clocked into a latch. This clock would best be controlled by a time delay device (one-shot multivibrator) so that enough time is allowed for the outputs to change state as necessary.

Due to the limited time left for research, the control of external components will have to be investigated at a later date. It appears at this point that the idea of an external FFT hardware device is quite feasible. Once

constructed, the PDP-8/E's only functions will be:

1) indexing
2) data storage
3) data acquisition
4) display.

In the future, the last four functions of the minicomputer listed above could conceivably be taken over by a second set of hardware, most likely controlled by a super high-speed microprocessor. In this manner, an entire real-time FFT system could be constructed. This system would be entirely self-contained and relatively inexpensive (in comparison with the PDP-8/E). Hopefully such a design will be undertaken by this author within the next several years.

FOOTNOTES

[1]Bernard Gold and Charles M. Rader, Digital Processing of Signals, (New York: McGraw-Hill, Inc., 1969), p. 173.

[2]Ibid., pp. 186-188.

[3]Introduction to Programming, 4th ed. (Maynard, Mass: Digital Equipment Corporation, 1973), pp. 2-2 - 3-24.

[4]P.C.Y. Yip, "Some Aspects of the Zoom Transformation," I.E.E.E. Transactions of Computers, C-25 (1976), pp. 287-296.

[5]Introduction to Programming, pp. 1-1 - 1-34.

BIBLIOGRAPHY

Gold, Bernard, and Charles M. Rader. Digital Processing of Signals. New York: McGraw-Hill, Inc., 1969.

Introduction to Programming. 4th ed. Maynard, Mass.: Digital Equipment Corporation, 1973.

Yip, P.C.Y. "Some Aspects of the Zoom Transformation." I.E.E.E. Transactions of Computers. C-25 (1976), 287-296.

APPENDIX A

Following are a series of photographs of the spectral density of a group of input wave forms developed by the Discrete Fourier Transform. Each photograph has a width of 512 spectral lines representing 512 Hz.

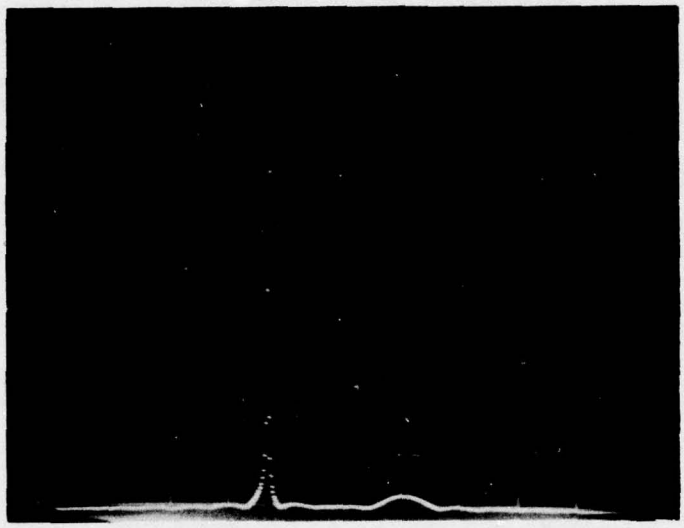


Figure A-1: DFT of a 31/32 of a second of a 180 Hertz 10 volt peak-to-peak sine wave with 1/32 of a second of a 300 Hertz 10 volt peak-to-peak sine wave.

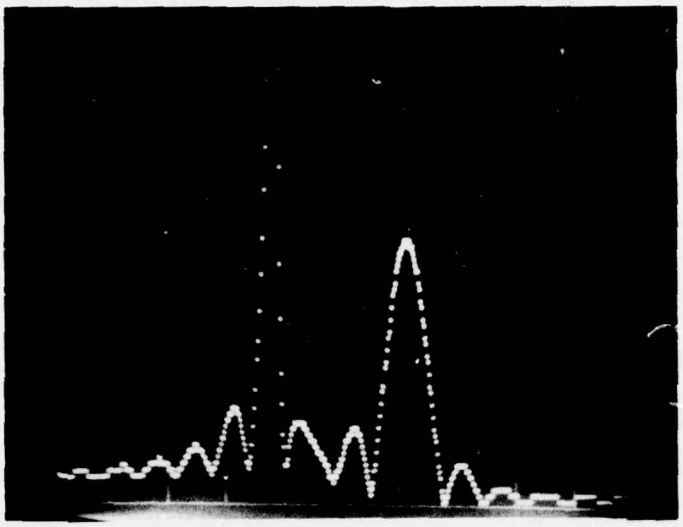


Figure A-2: Amplitude expanded view of the waveform described in Figure A-1. Note the effect on the 180 Hertz component.

Figure A-2: DFT of a 100 Hertz sinusoidal input waveform buried approximately 10 dB down in "white" noise.

APPENDIX B

The Fast Fourier Transform, as described previously, divides a set of time sampled data points into a series of DFTs. The following photographs demonstrate the manner in which these vector transformations occur. Each photograph has a width of 512 points and is scaled at five volts/cm. The top trace in each photograph corresponds to the real part of the complex frequency components, the bottom trace corresponds to the imaginary part (unless otherwise noted).

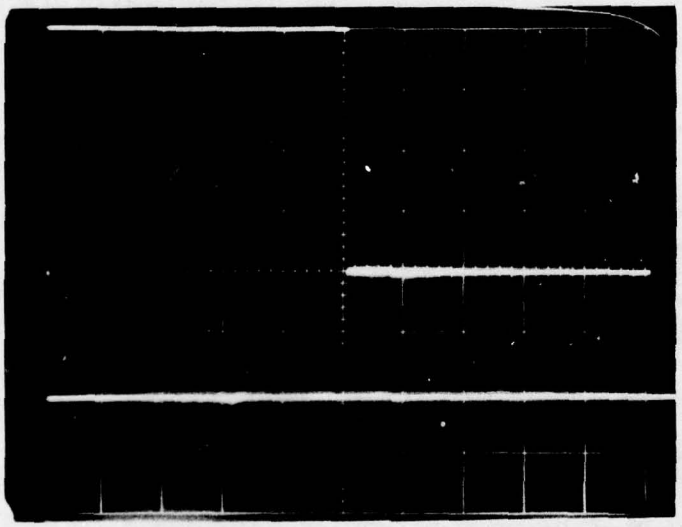


Figure B-1: A one-cycle square-wave upon which the FFT was performed.

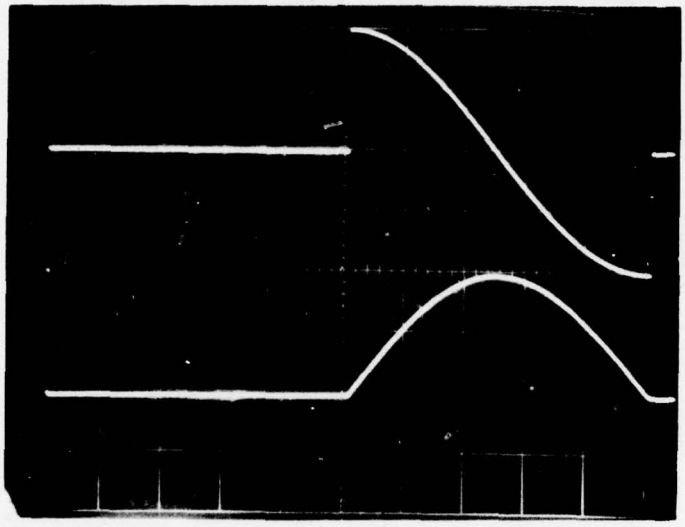


Figure B-2: The contents of the real and imaginary vectors after the first FFT vector transformation. Note the correlation with the COS (top) and with the SIN (bottom).

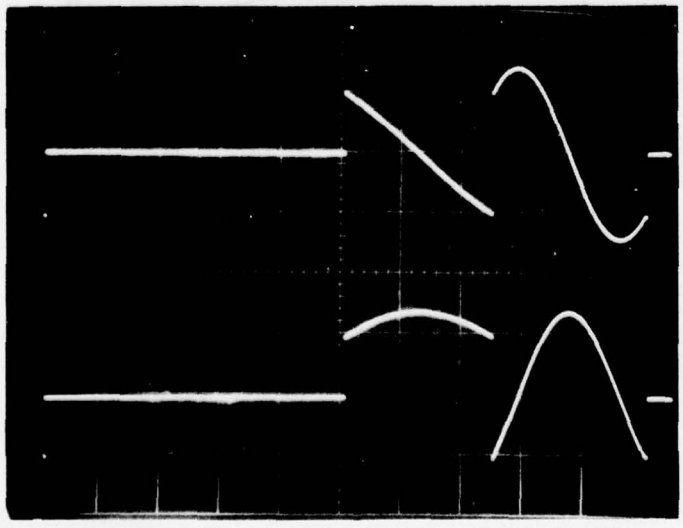


Figure B-3: The results of the second FFT vector transformation. Note the division of the two distinct forms in B-2 into four distinct forms. Also note the decrease in amplitude. The small points approximately 1 cm below the zero levels are images from the last point in the respective vectors.

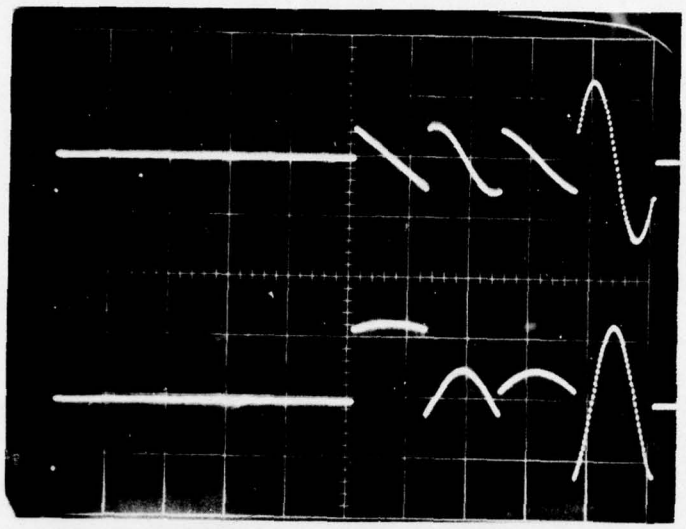


Figure B-4: The third FFT vector transformation. Again note the decreasing amplitude.

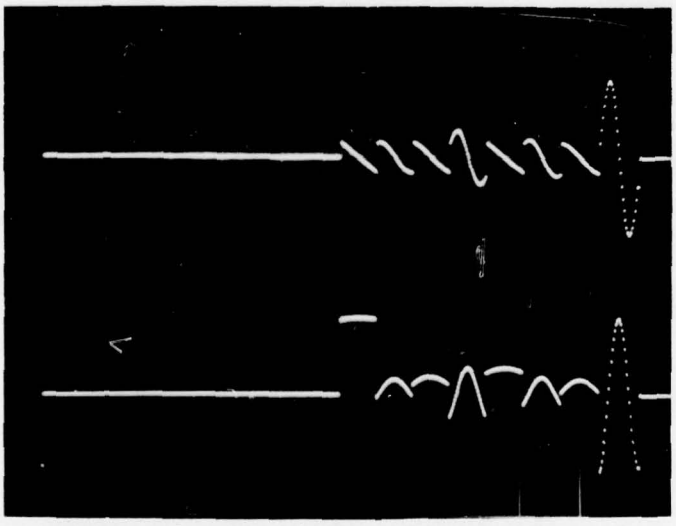


Figure B-5: Note how the top trace still shows correlation with the COS, and the bottom trace with the SIN.

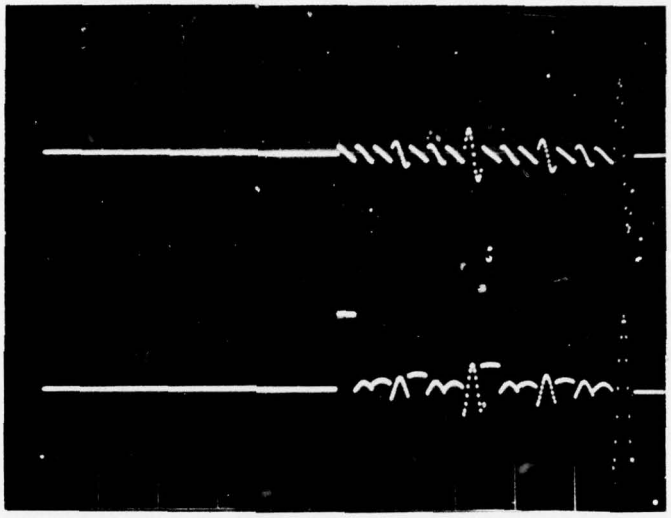


Figure B-6: Again note how drastically the amplitude has decreased since the first vector transformation.

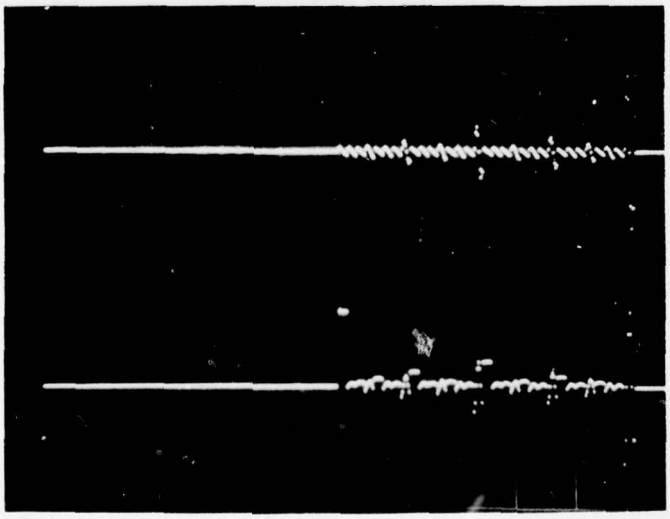


Figure B-7: After the sixth FFT vector transformation, the results become too complicated to see how well the transform is running, with the exception of the correlations mentioned in Figure B-5.

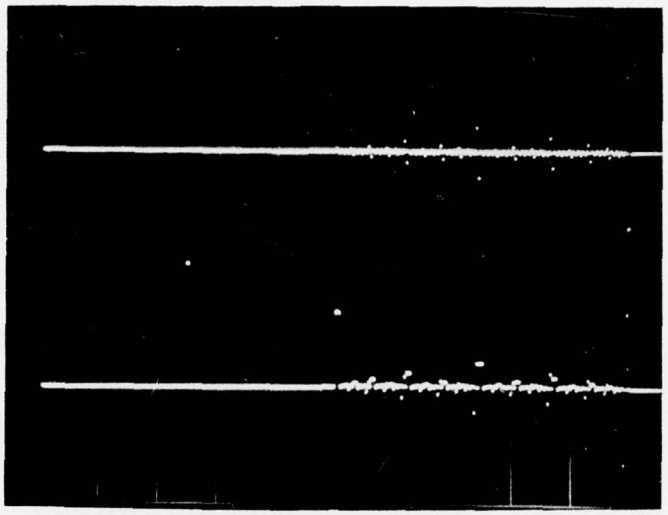


Figure B-8: After the seventh FFT vector transformation, the only distinguishable point is what corresponds to the first spectral line in the bottom trace 5 cm from the left edge.

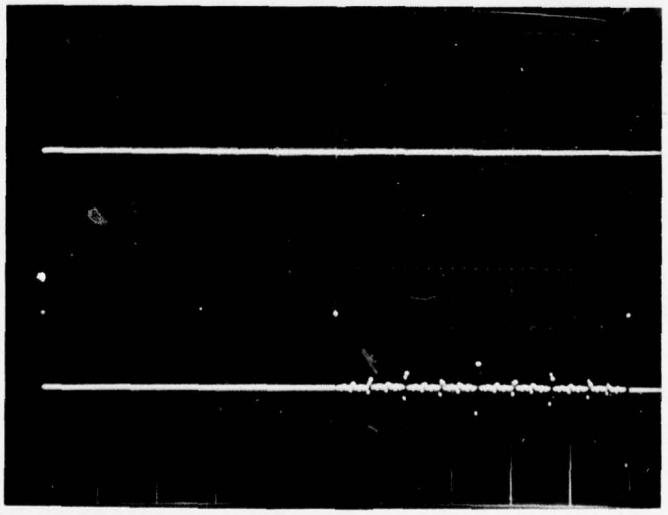


Figure B-9: The eighth vector transformation shows that the real terms are virtually equal to zero.

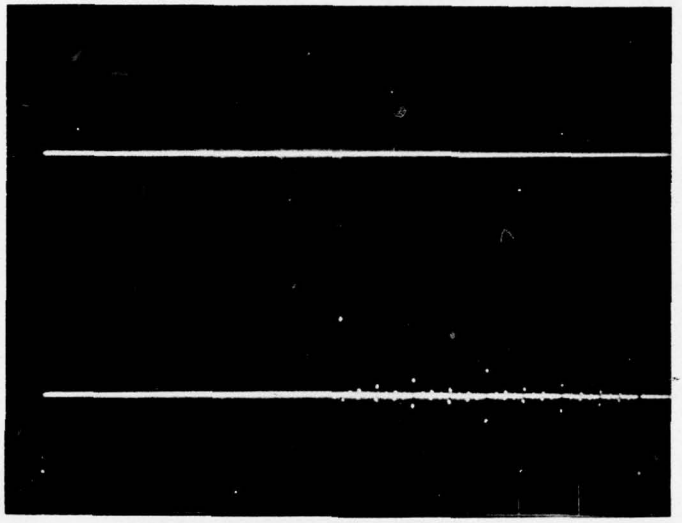


Figure B-10: The final step shows zero correlation with the COS and no even harmonics (the left half of the photograph).

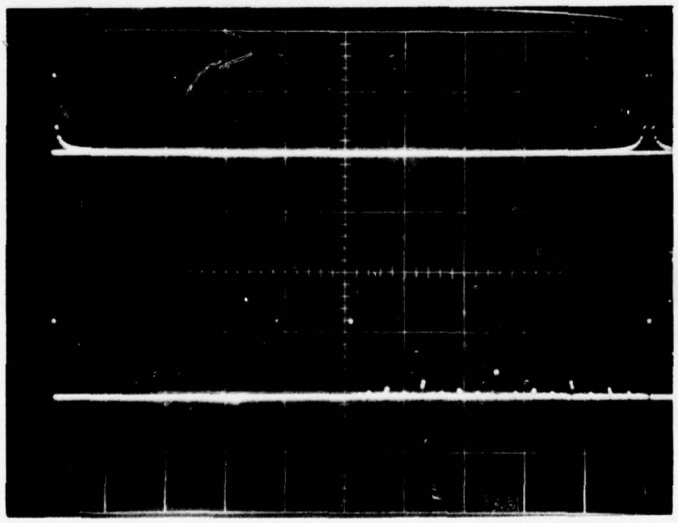


Figure B-11: This photograph shows, on the bottom, the magnitudes of the complex frequency components. The top trace shows the bit-reversed magnitudes of the components the bottom trace.

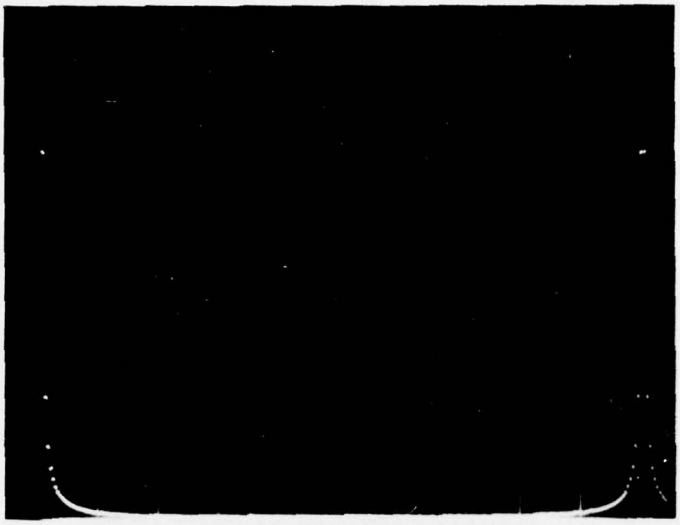


Figure B-12: An amplitude expanded view of the top trace in Figure B-11. Note the 1:1/3, 1:1/5, etc. magnitudes of the first, third, fifth, etc., harmonics. The even harmonics are zero. Also note the reflection of the first 256 spectral lines which appears on the right half of the photograph. If the photograph were folded on the 5 cm line, the two ends would match up perfectly due to this "mirror imaging."

APPENDIX C

The following series of photographs demonstrate the applicability of the FFT developed. Shown are the FFT's of several input waveforms. Each photograph has a width of 256 spectral lines, each line has a bandwidth, or resolution, of eight Hertz.

Figure C-1: A 100 Hertz, 10 volt peak-to-peak sinusoidal input results in this spectral distribution with the FFT.

Figure C-2: A 100 Hertz, 10 volt peak-to-peak square wave produced this spectral plot.

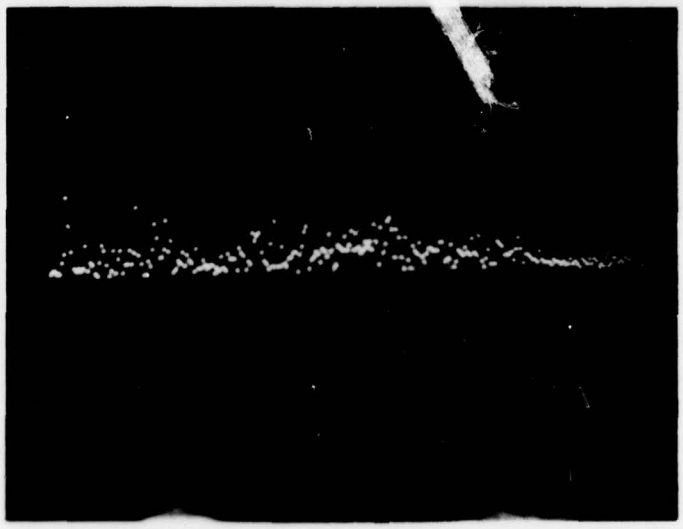


Figure C-3: A typical submarine running submerged with a screw rotation of 25 rpm yields this spectral plot. The input was taken from a Hewlett-Packard instrumentation recorder.

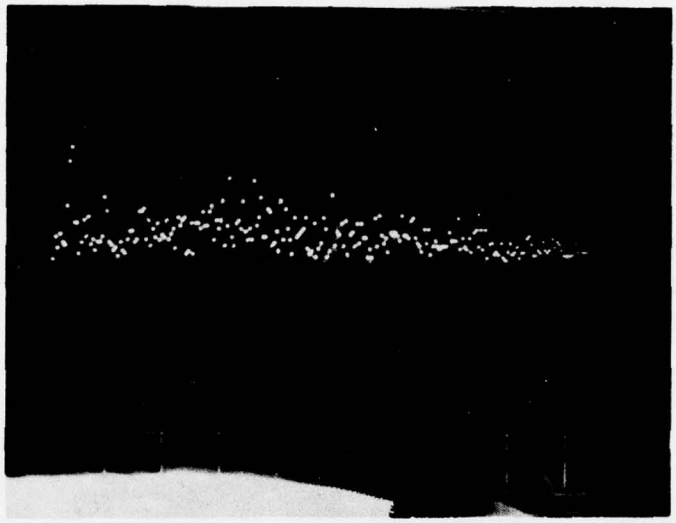


Figure C-4: The same submarine as in Figure C-3, recorded on audio quality tape recorder at 7 1/2 ips. Note the increase in noise level.

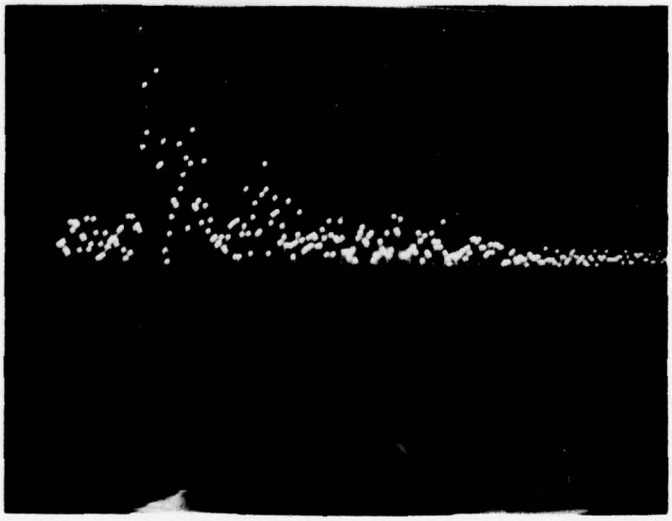


Figure C-5: Fish noise combined with the submarine in Figure C-3 from the HP tape recorder. The output level from the tape had to be reduced in order to avoid amplitude distortion. Thus the submarine cannot be seen.

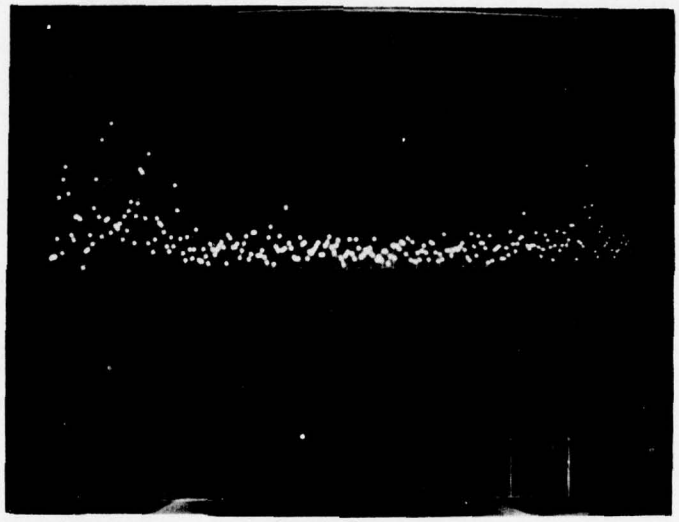


Figure C-6: The spectral plot of underwater background noise. Note strong peak at approximately 1900 Hertz.

APPENDIX D

Included in this appendix are the programs written for the PDP-8/E minicomputer in the development of the FFT. The first program is the FFT itself. The second is the data acquisition set of software. Finally, the programs which find the magnitude of the complex results and reorders these results is presented.

The FFT starts at location 200.

The data acquisition routine starts at location 6400. The sliding data window begins at location 3700; and the window loader starts at location 6600.

The program which calculates magnitudes begins at location 6000. The bit-reversal routine starts at location 3600.

DATAROUT

```
      3700        100    *3700
3700  6211        110           6211
3701  1343        120    TOP1   TAD        K7000
3702  3344        130           DCA        TALLY
3703  7604        140           LAS
3704  1345        150           TAD        POINT
3705  1343        160           TAD        K7000
3706  7421        170           MQL
3707  7701        180           ACL
3710  7440        190           SZA
3711  5324        200           JMP        OUT
3712  1346        210           TAD        K2000
3713  6551        220           6551
3714  7000        230           NOP
3715  7000                      NOP
3716  7000                      NOP
3717  7000                      NOP
3720  7000                      NOP
3721  7000                      NOP
3722  7200                      CLA
3723  6551        235           6551
3724  3345        240    OUT    DCA        POINT
3725  1346        250           TAD        K2000
3726  6553        260           6553
3727  7000        270           NOP
3730  7000        280           NOP
3731  7200        290           CLA
3732  6553        300           6553
3733  1745        310    LOOP2  TAD      I POINT
3734  6552        320           6552
3735  7200        330           CLA
3736  2345        340           ISZ        POINT
3[illegible]  7000        350           NOP
3740  2344        360           ISZ        TALLY
3741  5333        370           JMP        LOOP2
3742  5301        380           JMP        TOP1
3743  7000        390    K7000  7000
3744  0000        400    TALLY  0
3745  0000        410    POINT  0
3746  2000        420    K2000  2000
      6400        430    *6400
6400  6211        440           6211
6401  7200        450           CLA
6402  3247        460           DCA        INDEX1
6403  7240        470           STA
6404  7100        480           CLL
6405  6130        490           CLZE
6406  7200        500           CLA
6407  1245        510           TAD        K5600
6410  6132        520           CLOE
```

DATAROUT (CONTINUED)

```
6411    7200  530          CLA
6412    1246  540          TAD      M364
6413    6133  550          CLAB
6414    7200  560          CLA
6415    6535  570          ADSC
6416    6534  580          ADCV
6417    6131  590  LOOP    CLSK
6420    5217  600          JMP      .-1
6421    6135  610          CLSA
6422    6536  620          ADRC
6423    3647  630          DCA    I INDEX1
6424    2247  640          ISZ      INDEX1
6425    5217  650          JMP      LOOP
6426    1250  655  SYNC    TAD      K2002
6427    6553  656          6553
6430    7000  657          NOP
6431    7000                NOP
6432    7200                CLA
6433    6553  658          6553
6434    1647  660  DISP    TAD    I INDEX1
6435    6551  670          6551
6436    6131  680          CLSK
6437    5236  690          JMP      .-1
6440    6135  700          CLSA
6441    7200  710          CLA
6442    2247  720          ISZ      INDEX1
6443    5234  730          JMP      DISP
6444    5226  740          JMP      SYNC
6445    5600  750  K5600   5600
6446    7414  760  M364    -364
6447    0000  770  INDEX1  0
6450    2002  775  K2002   2002
   6600       780  *6600
6600    7200  790          CLA
6601    7604  800          LAS
6602    3246  810          DCA      BEGIN
6603    1247  820          TAD      M1000
6604    3250  830          DCA      TALLY1
6605    1251  840          TAD      K600
6606    3244  850          DCA      INDEX
6607    1252  860          TAD      K1600
6610    3253  870          DCA      ZDEX
6611    6211  880  READ    6211
6612    1646  890          TAD    I BEGIN
6613    6201  900          6201
6614    3644  910          DCA    I INDEX
6615    3653  920          DCA    I ZDEX
6616    2244  930          ISZ      INDEX
6617    2253  940          ISZ      ZDEX
6620    2246  950          ISZ      BEGIN
```

DATAROUT (CONTINUED)

```
     6621     7000  960              NOP
     6622     2250  970              ISZ       TALLY1
     6623     5211  980              JMP       READ
     6624     1245  990     TOP      TAD       K2001
     6625     6553  1000             6553
     6626     7000  1010             NOP
     6627     7000                   NOP
     6630     7200                   CLA
     6631     6553  1020             6553
     6632     3244  1030             DCA       INDEX
     6633     1247  1040             TAD       M1000
     6634     3250  1050             DCA       TALLY1
     6635     1644  1060     LOOP1   TAD     I INDEX
     6636     6551  1070             6551
     6637     7200  1080             CLA
     6640     2244  1090             ISZ       INDEX
     6641     2250  1100             ISZ       TALLY1
     6642     5235  1110             JMP       LOOP1
     6643     5224  1120             JMP       TOP
     6644     0000  1130     INDEX   0
     6645     2001  1140     K2001   2001
     6646     0000  1150     BEGIN   0
     6647     7000  1160     M1000   7000
     6650     0000  1170     TALLY1  0
     6651     0600  1180     K600    600
     6652     1600  1190     K1600   1600
     6653     0000  1200     ZDEX    0
        6654       1210     $
7200
```

CROSS REFERENCE TABLE

```
TOP1      120     380
K7000     120     160     390
TALLY     130     360     400
POINT     150     240     310     340
          410
OUT       200     240
K2000     210     250     420
LOOP2     310     370
INDEX1    460     630     640     660
          720     770
K5600     510     750
M364      540     760
LOOP      590     650
K2002     655     775
DISP      660     730     740
BEGIN     810     890     950     1150
M1000     820     1040    1160
TALLY1    830     970     1050    1100
```

DATAROUT (CONTINUED)

	1170			
MS00	840	1180		
INDEX	850	910	930	1030
	1060	1090	1130	
M1600	860	1190		
ZDEX	870	920	940	1200
READ	880	980		
TOP	990	1120		
M2001	990	1140		
LOOP1	1060	1110		

UNSCRAM

```
      3600        100    *3600
3600  7431        110           SWAB
3601  1242        120           TAD       M512
3602  3243        130           DCA       TALLY
3603  1244        140           TAD       C600
3604  3240        150           DCA       TGTX
3605  7001        160           IAC
3606  3241        170           DCA       LOCX
3607  7240        180    OUTP   CLA       CMA
3610  6503        190           DBCI
3611  6505        200           DBCO
3612  6500        210           DBDI
3613  7200        220           CLA
3614  1241        230           TAD       LOCX
3615  6506        240           DBSO
3616  6502        250           DBSK
3617  5216        260           JMP       .-1
3620  7200        270           CLA
3621  6504        280           DBRI
3622  7417        290           LSR
3623  0003        300           3
3624  1245        310           TAD       C1600
3625  3246        320           DCA       TEMP
3626  1646        330           TAD     I TEMP
3627  3640        340           DCA     I TGTX
3630  2243        350           ISZ       TALLY
3631  5233        360           JMP       .+2
3632  5236        370           JMP       FINIS
3633  2240        380           ISZ       TGTX
3634  2241        390           ISZ       LOCX
3635  5207        400           JMP       OUTP
3636  5637        410    FINIS  JMP     I L310
3637  0310        420    L310   310
3640  0000        430    TGTX   0
3641  0000        440    LOCX   0
3642  7000        450    M512   7000
3643  0000        460    TALLY  0
3644  0600        470    C600   600
3645  1600        480    C1600  1600
3646  0000        490    TEMP   0
      6000        500    *6000
6000  7431        510           SWAB
6001  1303        520           TAD       M1000
6002  3304        530           DCA       TALLY1
6003  1305        540           TAD       K600
6004  3306        550           DCA       RDEX
6005  1307        560           TAD       K1600
6006  3310        570           DCA       IDEX
6007  1706        580    TOP    TAD     I RDEX
6010  7510        590           SPA
```

UNSCRAM (CONTINUED)

```
6011    7041    600             CIA
6012    7421    610             MQL
6013    7701    620             ACL
6014    3311    630             DCA     MULT
6015    7405    640             MUY
6016    6111    650             MULT
6017    7445    660             DST
6020    6112    670             DTEMP
6021    7621    680             CAM
6022    1710    690             TAD     I IDEX
6023    7510    700             SPA
6024    7041    710             CIA
6025    7421    720             MQL
6026    7701    730             ACL
6027    3311    740             DCA     MULT
6030    7405    750             MUY
6031    6111    760             MULT
6032    7443    770             DAD
6033    6112    780             DTEMP
6034    7445    790             DST
6035    6112    800             DTEMP
6036    7621    810             CAM
6037    1314    820             TAD     M13
6040    3315    830             DCA     TALLY2
6041    1316    840             TAD     K4000
6042    3317    850             DCA     SQTRY
6043    1320    860             TAD     K2000
6044    3321    870             DCA     SQINC
6045    1317    880     UPHERE  TAD     SQTRY
6046    7421    890             MQL
6047    7405    900             MUY
6050    6117    910             SQTRY
6051    7575    920             DCM
6052    7443    930             DAD
6053    6112    940             DTEMP
6054    7620    950             SNL     CLA
6055    5262    960             JMP     DOWN
6056    1317    970     UP      TAD     SQTRY
6057    1321    980             TAD     SQINC
6060    3317    990             DCA     SQTRY
6061    5266    1000            JMP     THAR

6062    1321    1010    DOWN    TAD     SQINC
6063    7041    1020            CIA
6064    1317    1030            TAD     SQTRY
6065    3317    1040            DCA     SQTRY
6066    1321    1050    THAR    TAD     SQINC
6067    7110    1060            7110
6070    3321    1070            DCA     SQINC
```

UNSCRAM (CONTINUED)

```
    6071    2315  1080            ISZ       TALLY2
    6072    5245  1090            JMP       UPHERE
    6073    1317  1100            TAD       SQTRY
    6074    3710  1110            DCA     I IDEX
    6075    2310  1120            ISZ       IDEX
    6076    2306  1130            ISZ       RDEX
    6077    2304  1140            ISZ       TALLY1
    6100    5207  1150            JMP       TOP
    6101    5702  1160            JMP     I L3600
    6102    3600  1170  L3600     3600
    6103    7000  1180  M1000     -1000
    6104    0000  1190  TALLY1    0
    6105    0600  1200  K600      600
    6106    0000  1210  RDEX      0
    6107    1600  1220  K1600     1600
    6110    0000  1230  IDEX      0
    6111    0000  1240  MULT      0
    6112    0000  1250  DTEMP     0
    6113    0000  1260            0
    6114    7765  1270  M13       -13
    6115    0000  1280  TALLY2    0
    6116    4000  1290  K4000     4000
    6117    0000  1300  SQTRY     0
    6120    2000  1310  K2000     2000
    6121    0000  1320  SQINC     0
       6122       1330  $
INC
```

CROSS REFERENCE TABLE

```
M512      120     450
TALLY     130     350     460
0600      140     470
TGTX      150     340     380     430
LOCX      170     230     390     440
OUTP      180     400
01600     310     480
TEMP      320     330     490
FINIS     370     410
L310      410     420
M1000     520     1180
TALLY1    530     1140    1190
K600      540     1200
RDEX      550     580     1130    1210
K1600     560     1220
IDEX      570     690     1110    1120
          1230
TOP       580     1150
MULT      630     740     1240
M13       820     1270
```

UNSCRAM (CONTINUED)

TALLY2	830	1080	1280	
K4000	840	1290		
SQTRY	850	880	970	990
	1030	1040	1100	1300
K2000	860	1310		
SQINC	870	980	1010	1050
	1070	1320		
UPHERE	880	1090		
DOWN	960	1010		
UP	970			
THAR	1000	1050		
L3600	1160	1170		
DTEMP	1250			

FFT

```
     0200          100    *200
0200      7200     110    START   CLA
0201      7431     120            SWAB
0202      1020     130            TAD     KSIZE
0203      3023     140            DCA     M
0204      7001     150            IAC
0205      3024     160            DCA     IMAX
0206      3025     170            DCA     ITAL
0207      4310     180    LPASS   JMS     DISP
0210      1023     190            TAD     M
0211      7110     200            7110
0212      7450     210            SNA
0213      5302     220            JMP     FIN
0214      3023     230            DCA     M
0215      1024     240            TAD     IMAX
0216      7041     250            CIA
0217      3025     260            DCA     ITAL
0220      1040     270            TAD     LIM
0221      3027     280            DCA     IMFIND
0222      1037     290            TAD     LRE
0223      3030     300            DCA     RFIND
0224      1040     310            TAD     LIM
0225      1023     320            TAD     M
0226      3031     330            DCA     IMBIND
0227      1037     340            TAD     LRE
0230      1023     350            TAD     M
0231      3032     360            DCA     RBIND
0232      1023     370    JLOOP   TAD     M
0233      7041     380            CIA
0234      3026     390            DCA     JTAL
0235      1035     400            TAD     LSWIND
0236      3033     410            DCA     SWIND
0237      1036     420            TAD     LCWIND
0240      3034     430            DCA     CWIND
0241      4441     440    ILOOP   JMS   I LSUBR
0242      1033     450            TAD     SWIND
0243      1024     460            TAD     IMAX
0244      3033     470            DCA     SWIND
0245      1034     480            TAD     CWIND
0246      1024     490            TAD     IMAX
0247      3034     500            DCA     CWIND
0250      2027     510            ISZ     IMFIND
0251      2030     520            ISZ     RFIND
0252      2031     530            ISZ     IMBIND
0253      2032     540            ISZ     RBIND
0254      2026     550            ISZ     JTAL
0255      5241     560            JMP     ILOOP
0256      2025     570            ISZ     ITAL
0257      5261     580            JMP     .+2
0260      5276     590            JMP     CHANGE
```

FFT (CONTINUED)

```
0261  1030  600            TAD     RFIND
0262  1023  610            TAD     M
0263  3030  620            DCA     RFIND
0264  1027  630            TAD     IMFIND
0265  1023  640            TAD     M
0266  3027  650            DCA     IMFIND
0267  1032  660            TAD     RBIND
0270  1023  670            TAD     M
0271  3032  680            DCA     RBIND
0272  1031  690            TAD     IMBIND
0273  1023  700            TAD     M
0274  3031  710            DCA     IMBIND
0275  5232  720            JMP     JLOOP
0276  1024  730  CHANGE    TAD     IMAX
0277  7104  740            7104
0300  3024  750            DCA     IMAX
0301  5207  760            JMP     LPASS
0302  7402  770  FIN       HLT
    0020    780  *20
0020  1000  790  KSIZE     1000
0021  7000  800  MSIZE     -1000
0022  7400  810  MTIME     -400
0023  0000  820  M         0
0024  0000  830  IMAX      0
0025  0000  840  ITAL      0
0026  0000  850  JTAL      0
0027  0000  860  IMFIND    0
0030  0000  870  RFIND     0
0031  0000  880  IMBIND    0
0032  0000  890  RBIND     0
0033  0000  900  SWIND     0
0034  0000  910  CWIND     0
0035  2600  920  LSWIND    2600
0036  3200  930  LCWIND    3200
0037  0600  940  LRE       600
0040  1600  950  LIM       1600
0041  0400  960  LSUBR     400
0042  2000  970  KSYNC     2000

0043  0000  980  RINDX     0
0044  0000  990  IMINDX    0
0045  0000  1000 TALLY1    0
0046  0000  1010 COUNT     0
    0400    1020 *400
0400  0000  1030 FFTMATH                          0
0401  7344  1040           CLA     CLL     CMA     RAL
0402  3055  1050           DCA     CFLG1
0403  7340  1060           CLA     CLL     CMA
0404  3056  1070           DCA     CFLG2
0405  7344  1080           CLA     CLL     CMA     RAL
```

FFT (CONTINUED)

```
0406  3057  1090          DCA     DFLG1
0407  7340  1100          CLA     CLL     CMA
0410  3060  1110          DCA     DFLG2
0411  1430  1120  ACALC   TAD   I RFIND
0412  7415  1130          ASR
0413  0001  1140          1
0414  3050  1150          DCA     ATEMP
0415  1432  1160          TAD   I RBIND
0416  7415  1170          ASR
0417  0001  1180          1
0420  1050  1190          TAD     ATEMP
0421  3430  1200          DCA   I RFIND
0422  1427  1210  BCALC   TAD   I IMFIND
0423  7415  1220          ASR
0424  0001  1230          1
0425  3051  1240          DCA     BTEMP
0426  1431  1250          TAD   I IMBIND
0427  7415  1260          ASR
0430  0001  1270          1
0431  1051  1280          TAD     BTEMP
0432  3427  1290          DCA   I IMFIND
0433  1432  1300  CCALC   TAD   I RBIND
0434  7041  1310          CIA
0435  7415  1320          ASR
0436  0001  1330          1
0437  1050  1340          TAD     ATEMP
0440  7500  1350          SMA
0441  5246  1360          JMP     .+5
0442  2055  1370          ISZ     CFLG1
0443  2056  1380          ISZ     CFLG2
0444  7000  1390          NOP
0445  7041  1400          CIA
0446  3052  1410          DCA     CTEMP

MA    0447  1034  1420          TAD     CWIND
0450  1063  1430          TAD     MCNEG
0451  7750  1440          SPA     SNA     CLA
0452  5261  1450          JMP     NOCHNG
0453  2055  1460          ISZ     CFLG1
0454  7000  1470          NOP
0455  2057  1480          ISZ     DFLG1
0456  1434  1490          TAD   I CWIND
0457  7041  1500          CIA
0460  5262  1510          JMP     .+2
0461  1434  1520  NOCHNG  TAD   I CWIND
0462  7421  1530          MQL
0463  7701  1540          ACL
0464  3061  1550          DCA     COS
0465  7405  1560          MUY
0466  0052  1570          CTEMP
```

FFT (CONTINUED)

```
0467    2055    1580            ISZ       CFLG1
0470    5272    1590            JMP       .+2
0471    7575    1600            DCM
0472    3054    1610            DCA       HOLD
0473    1431    1620            TAD     I IMBIND
0474    7415    1630            ASR
0475    0001    1640            1
0476    7041    1650            CIA
0477    1051    1660            TAD       BTEMP
0500    7041    1670            CIA
0501    3053    1680            DCA       DTEMP
0502    1053    1690            TAD       DTEMP
0503    7500    1700            SMA
0504    5312    1710            JMP       .+6
0505    2060    1720            ISZ       DFLG2
0506    7000    1730            NOP
0507    2057    1740            ISZ       DFLG1
0510    7000    1750            NOP
0511    7041    1760            CIA
0512    3062    1770            DCA       HOLDI
0513    1433    1780            TAD     I SWIND
0514    7421    1790            MQL
0515    7405    1800            MUY
0516    0062    1810            HOLDI
0517    2060    1820            ISZ       DFLG2
0520    7575    1830            DCM
0521    1054    1840            TAD       HOLD
0522    3432    1850            DCA     I RBIND
0523    1062    1860  DCALC     TAD       HOLDI
0524    7421    1870            MQL
0525    7405    1880            MUY
0526    0061    1890            COS
0527    2057    1900            ISZ       DFLG1
0530    7575    1910            DCM
0531    3054    1920            DCA       HOLD
0532    1433    1930            TAD     I SWIND
0533    7421    1940            MQL
0534    7405    1950            MUY
0535    0052    1960            CTEMP
0536    2056    1970            ISZ       CFLG2
0537    7575    1980            DCM
0540    1054    1990            TAD       HOLD
0541    3431    2000            DCA     I IMBIND
0542    5600    2010            JMP     I FFTMATH
     0050       2020  *50
0050    0000    2030  ATEMP     0
0051    0000    2040  BTEMP     0
0052    0000    2050  CTEMP     0
0053    0000    2060  DTEMP     0
0054    0000    2070  HOLD      0
```

FFT (CONTINUED)

```
0055    0000  2080   CFLG1   0
0056    0000  2090   CFLG2   0
0057    0000  2100   DFLG1   0
0060    0000  2110   DFLG2   0
0061    0000  2120   COS     0
0062    0000  2130   HOLD1   0
0063    4400  2140   MCNEG   -3400
     0310     2150   *310
0310    0000  2160   DISP    0
0311    7200  2170   UP      CLA
0312    1022  2180           TAD     MTIME
0313    3046  2190           DCA     COUNT
0314    1037  2200   SYNC    TAD     LRE
0315    3043  2210           DCA     RINDX
0316    1040  2220           TAD     LIM
0317    3044  2230           DCA     IMINDX
0320    1021  2240           TAD     MSIZE
0321    3045  2250           DCA     TALLY1
0322    1042  2260           TAD     KSYNC
0323    6553  2270           6553
0324    7000  2280           NOP
0325    7000  2290           NOP
0326    7200  2300           CLA
0327    6553  2310           6553
0330    1443  2320   DISP1   TAD     I RINDX
0331    6551  2330           6551
0332    7200  2340           CLA
0333    1444  2350           TAD     I IMINDX
0334    6552  2360           6552
0335    7200  2370           CLA
0336    2043  2380           ISZ     RINDX
0337    2044  2390           ISZ     IMINDX
0340    2045  2400           ISZ     TALLY1
0341    5330  2410           JMP     DISP1
0342    2046  2420           ISZ     COUNT
0343    5314  2430           JMP     SYNC
0344    7604  2440           LAS
0345    7440  2450           SZA
0346    5311  2460           JMP     UP
0347    5710  2470           JMP     I DISP
     0350     2480   $
```

CROSS REFERENCE TABLE

```
START   110
KSIZE   130     790
M       140     190     230     320
        350     370     610     640
        670     700     820
```

FFT (CONTINUED)

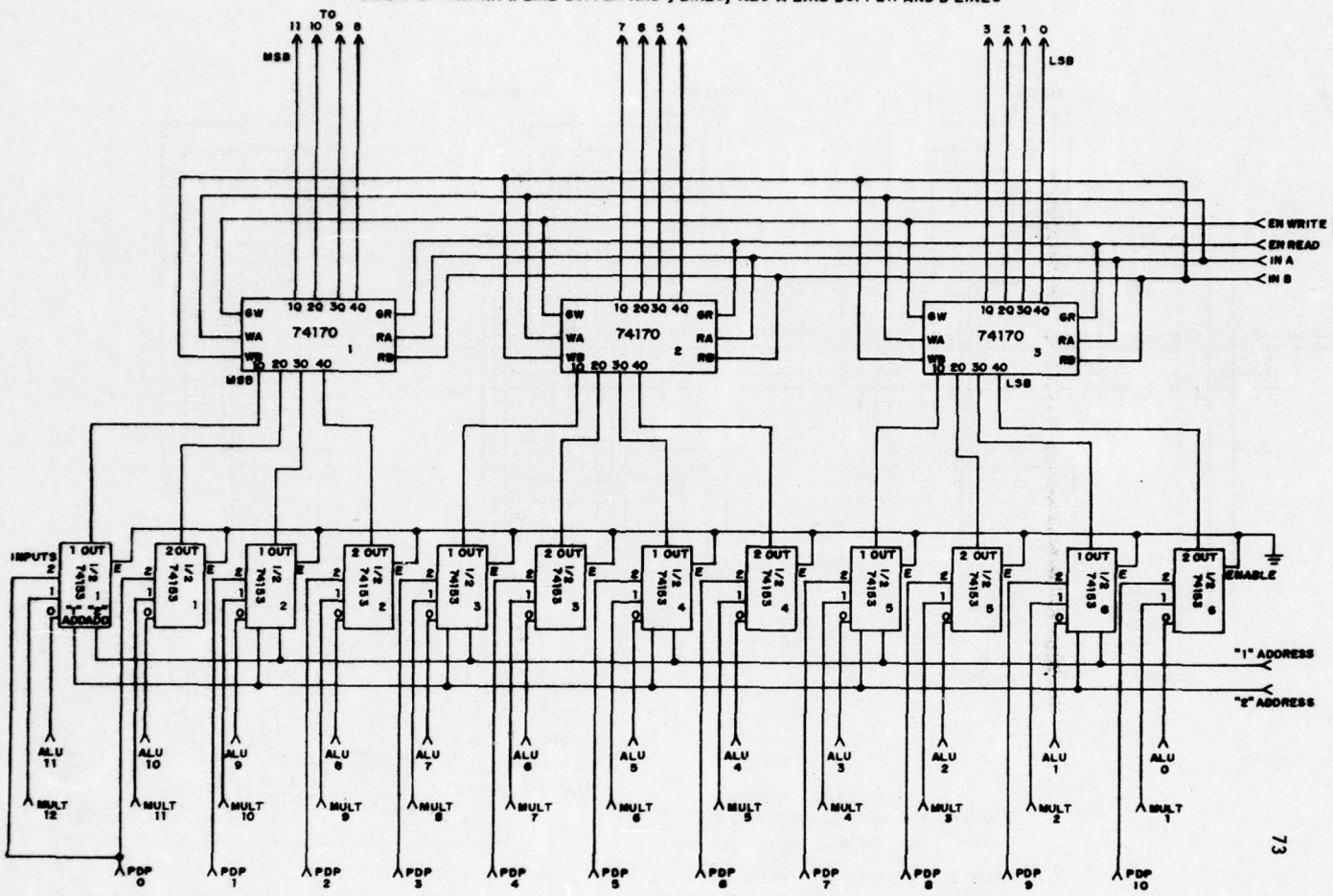
IMAX	160	240	460	490
	730	750	830	
ITAL	170	260	570	840
LPASS	180	760		
DISP	180	2160	2470	
FIN	220	770		
LIM	270	310	950	2220
IMFIND	280	510	630	650
	860	1210	1290	
LRE	290	340	940	2200
RFIND	300	520	600	620
	870	1120	1200	
IMBIND	1430	2140		

APPENDIX E

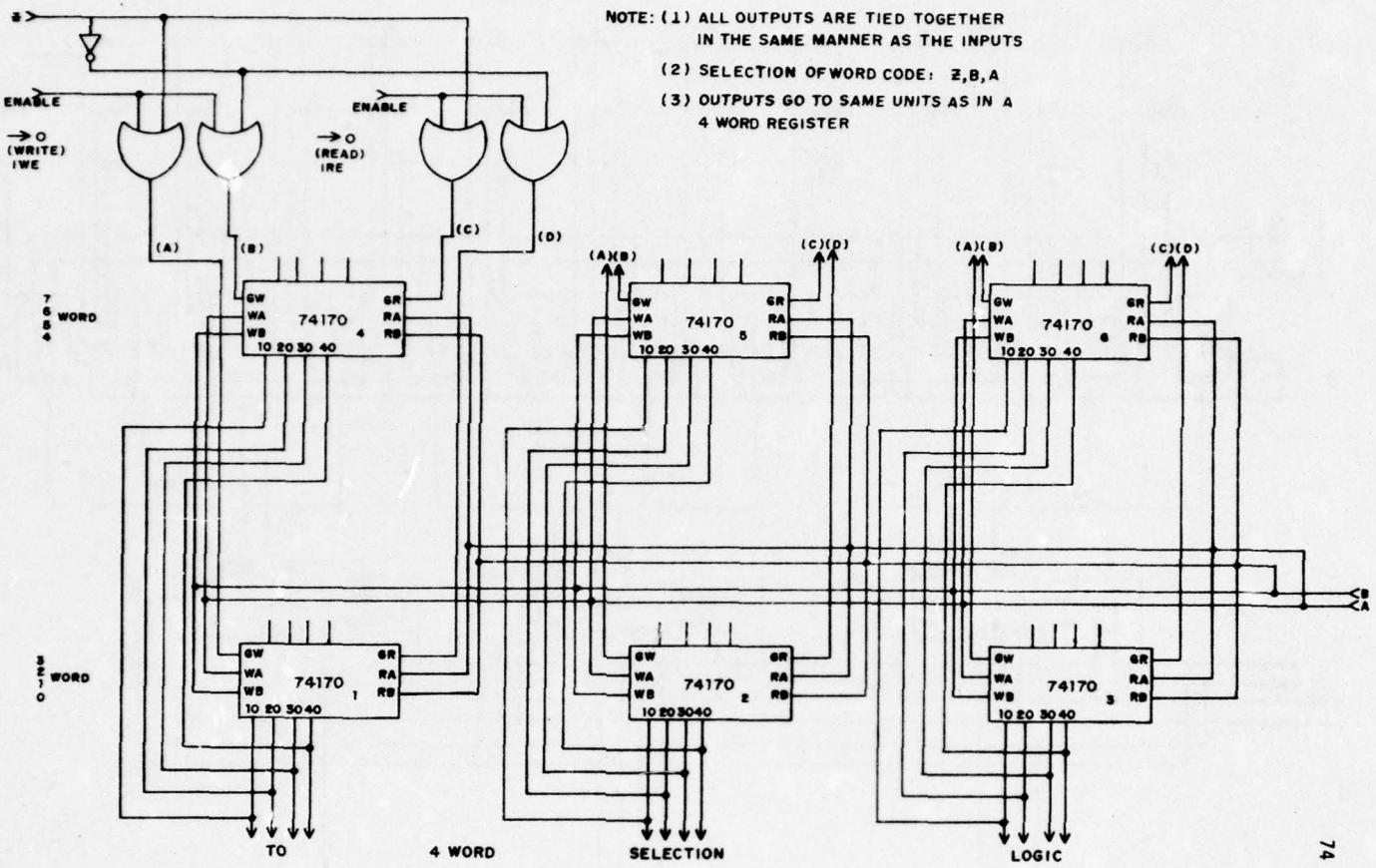
This appendix presents that logic designed for the FFT hardware device discussed in Chapter 4. Note that no controlling logic has been included, only the registers and the like have. The multiplier matrix is also not included. For a description of this matrix the reader is directed to Fairchild Application Note 329.

4 WORD INPUT REGISTER AND SELECTION LOGIC

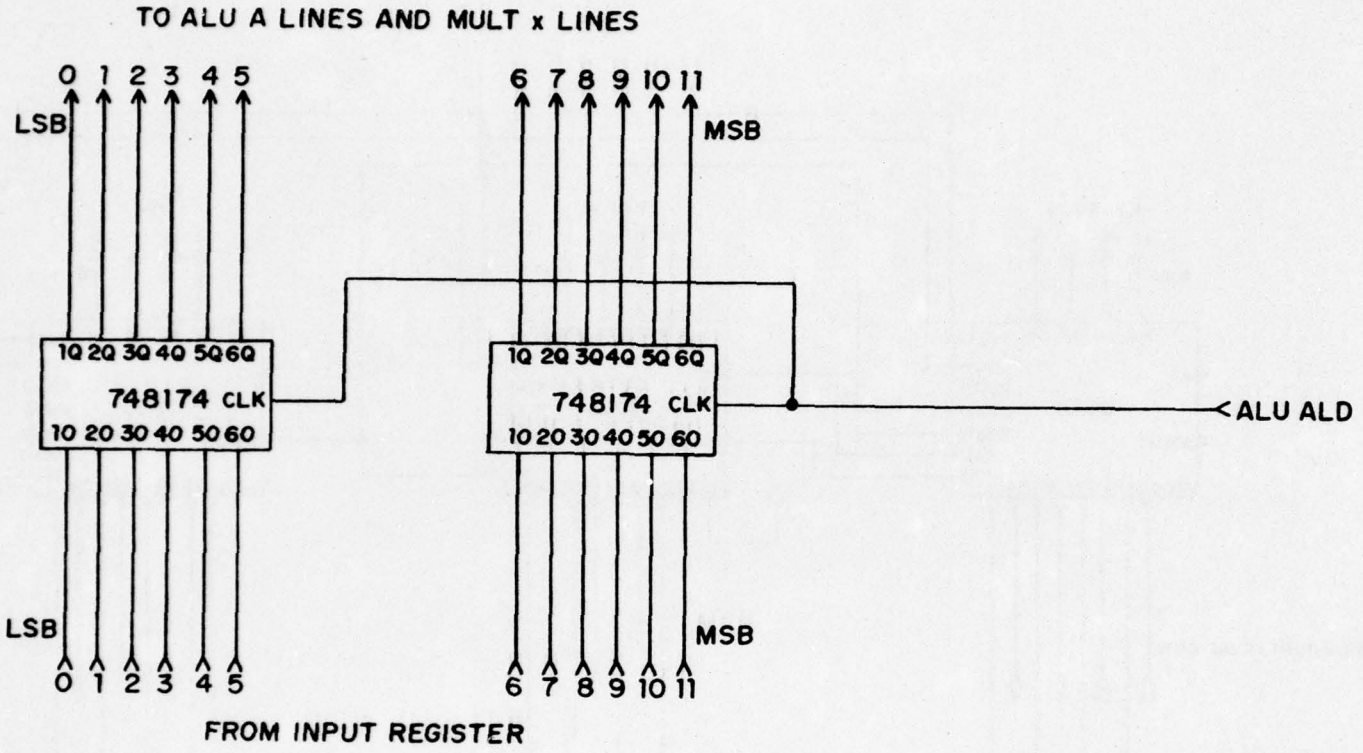
MULTIPLY MATRIX x LINE EUFFER AND y LINES, ALU A LINE BUFFER AND B LINES



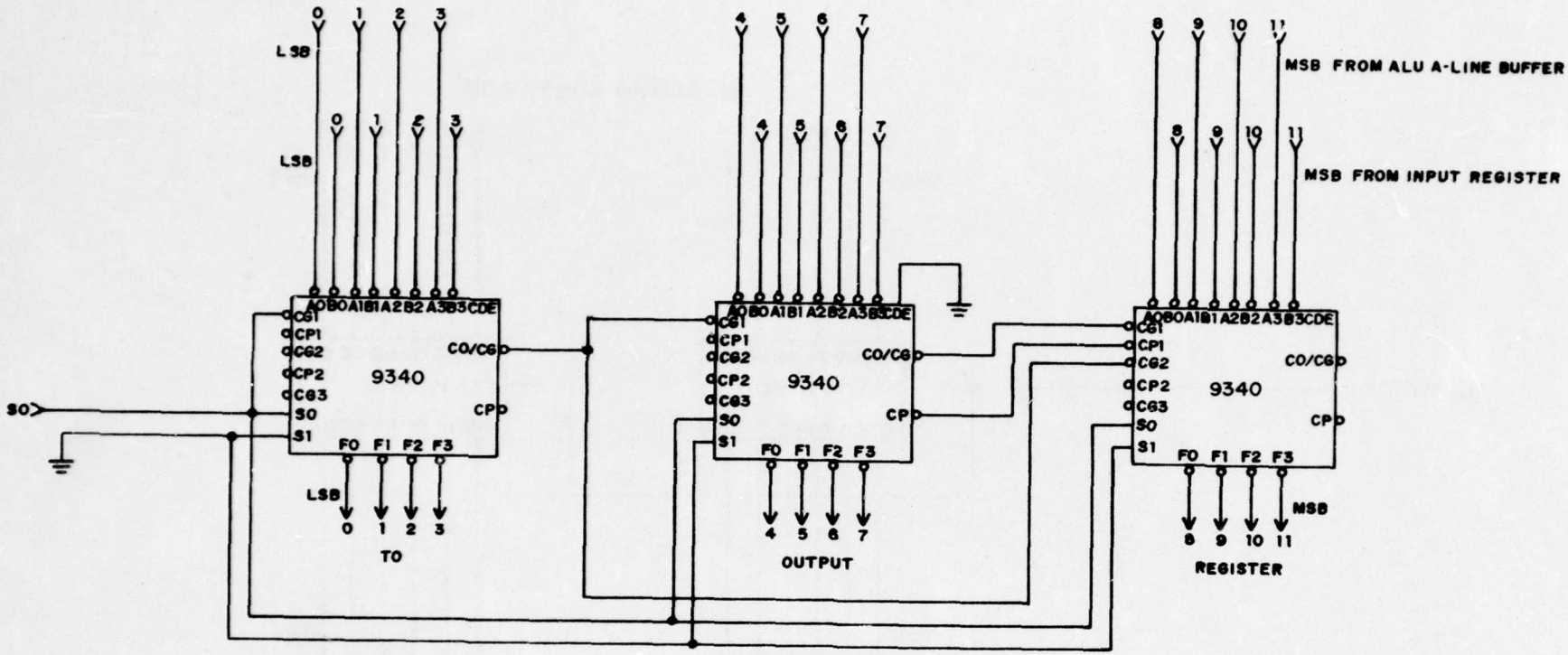
EXPANSION OF 4 WORD REGISTER TO 8 WORDS



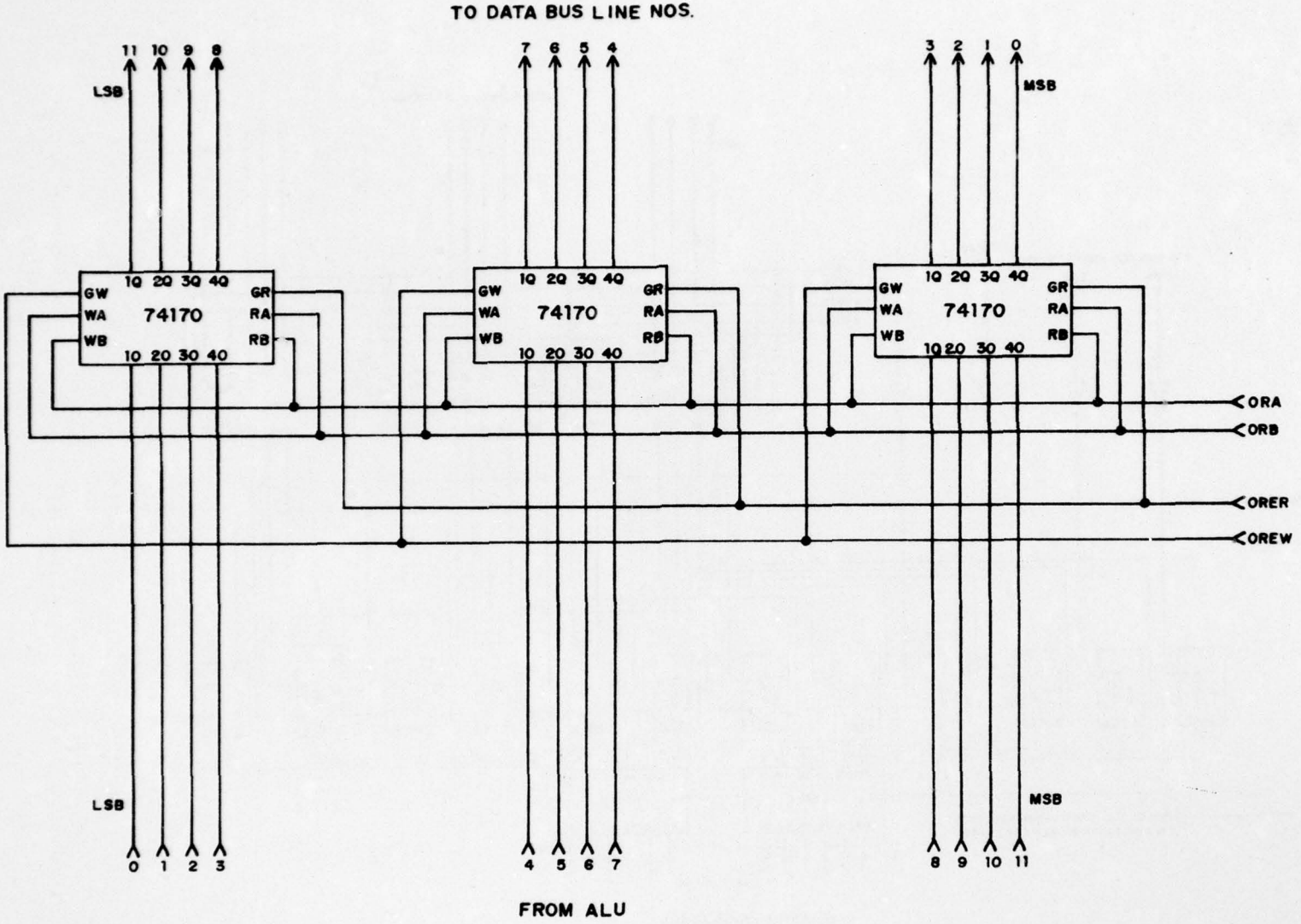
ALU A LINE BUFFER (+ MULT x LINE BUFF)



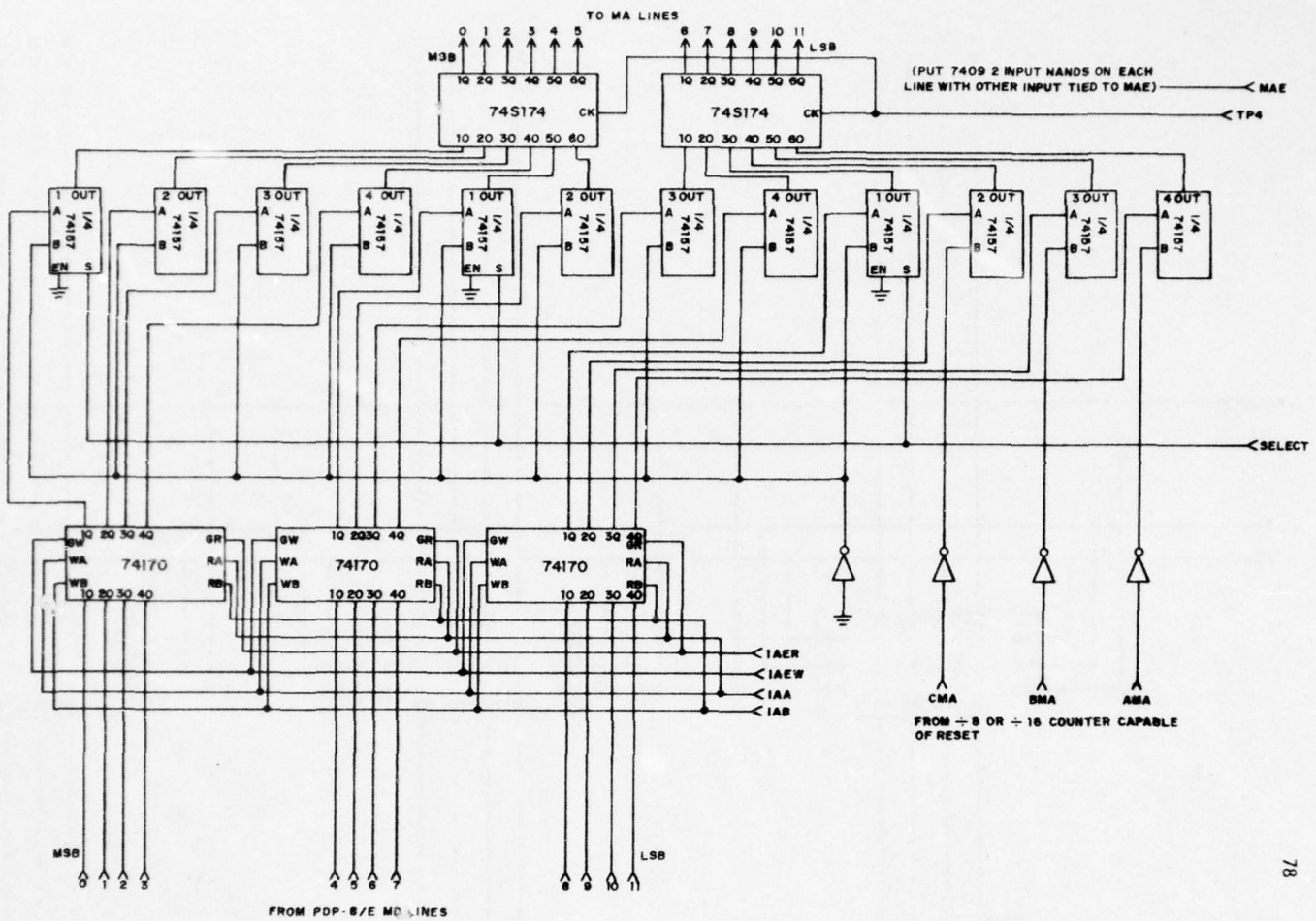
TWO'S COMPLEMENT ALU (ACTIVE-LOW)



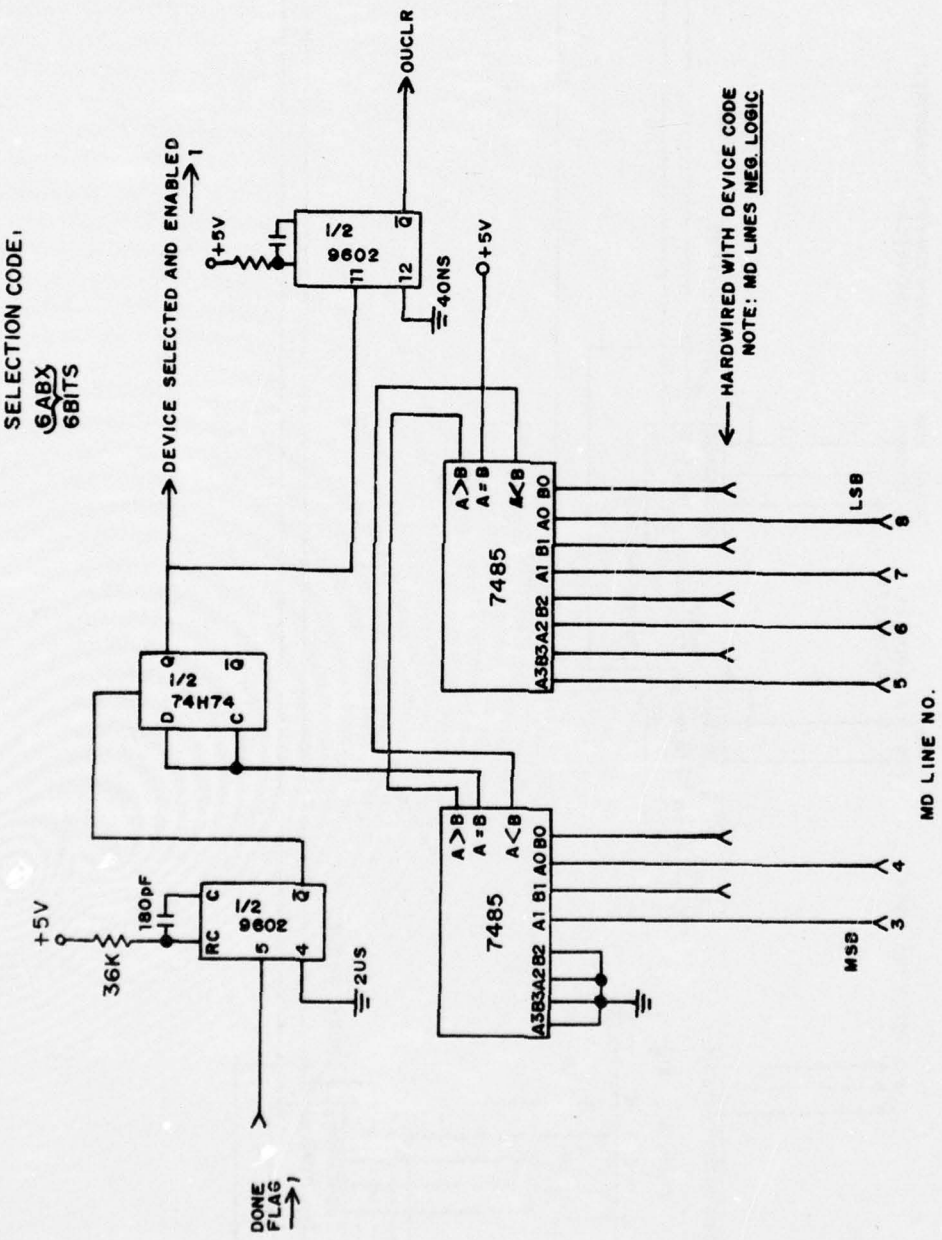
OUTPUT REGISTER



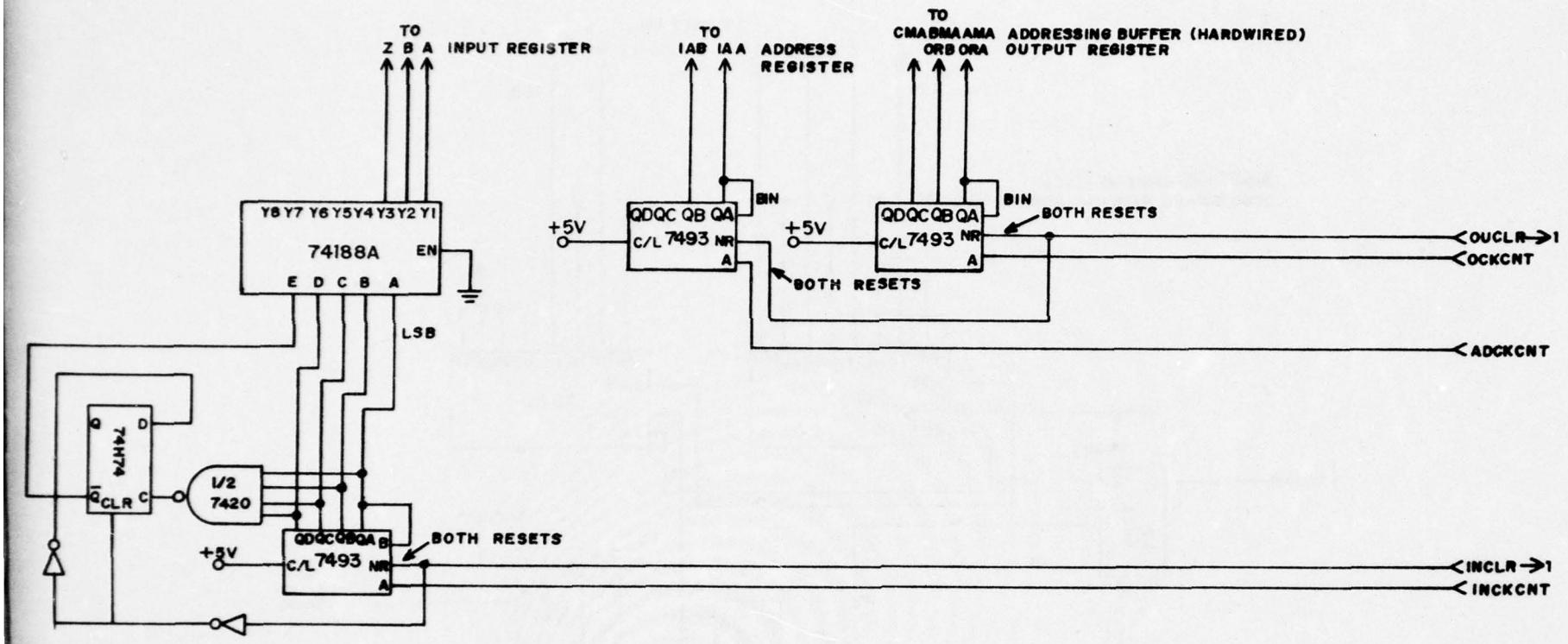
ADDRESSING BUFFER



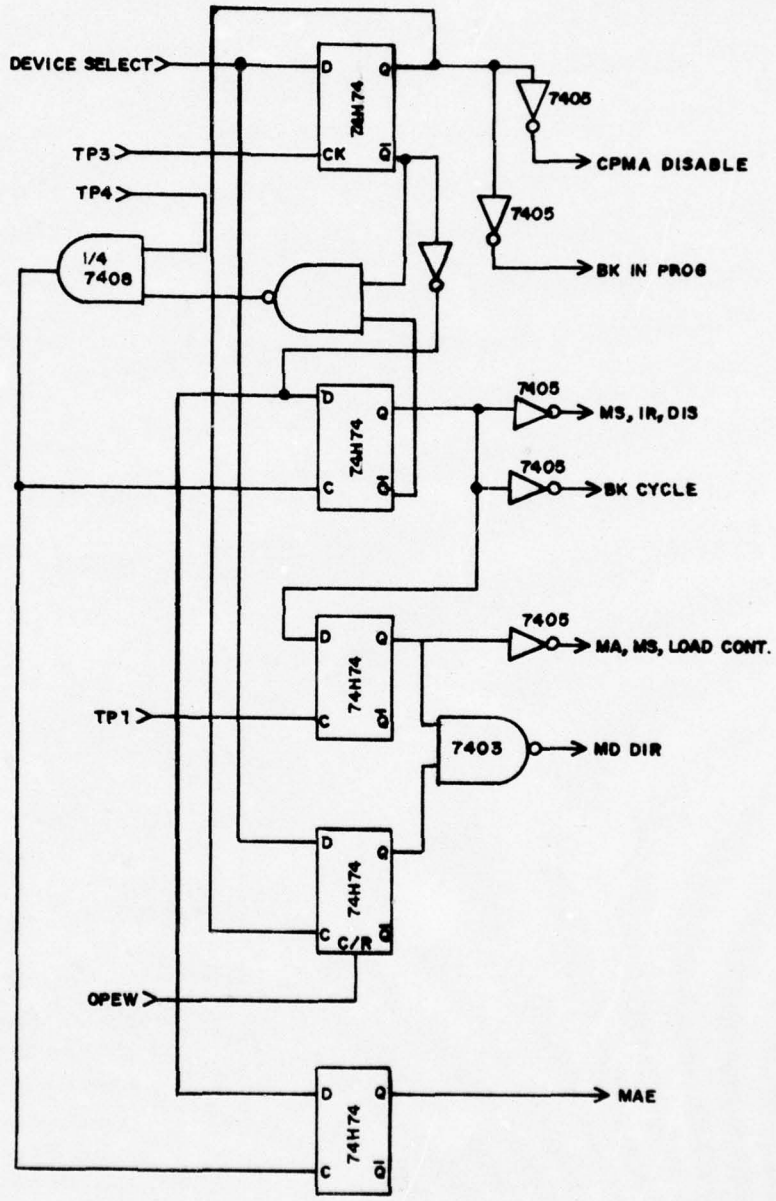
DEVICE SELECT



REGISTER ADDRESSING COUNTERS



LOGIC REQUIRED TO
CONTROL PROCESSOR
(INTO OMA)
(DATA XFER DIRECTION)



UNCLASSIFIED

Security Classification

DOCUMENT CONTROL DATA - R & D

(Security classification of title, body of abstract and indexing annotation must be entered when the overall report is classified)

1. ORIGINATING ACTIVITY *(Corporate author)*

U.S. Naval Academy, Annapolis.

2a. REPORT SECURITY CLASSIFICATION

UNCLASSIFIED

2b. GROUP

3. REPORT TITLE

Frequency analysis using a minicomputer.

4. DESCRIPTIVE NOTES *(Type of report and inclusive dates)*

Research report.

5. AUTHOR(S) *(First name, middle initial, last name)*

D.C. Boch.

6. REPORT DATE	7a. TOTAL NO. OF PAGES	7b. NO. OF REFS
17 May 1976.	81	3

8a. CONTRACT OR GRANT NO.

b. PROJECT NO.

c.

d.

9a. ORIGINATOR'S REPORT NUMBER(S)

U.S. Naval Academy, Annapolis - Trident Scholar project report, no. 75 (1976)

9b. OTHER REPORT NO(S) *(Any other numbers that may be assigned this report)*

10. DISTRIBUTION STATEMENT

This document has been approved for public release ; its distribution is UNLIMITED.

11. SUPPLEMENTARY NOTES

12. SPONSORING MILITARY ACTIVITY

U.S. Naval Academy, Annapolis.

13. ABSTRACT

This study was initiated with the aim of determining the fundamental capabilities and limitations of one minicomputer in the development of a spectral plot through Discrete Fourier Transform (DFT). Included within this general topic were two fundamental goals :

1/ Development of the fastest FFT (Fast Fourier Transform) possible utilizing only the basic capabilities of the PDP-8/E general purpose minicomputer. Successful completion of this objective brought the machnine's fundamental limitations for real-time spectral analysis into clear perspective.

2/ Investigation of the feasibility of an external hardware "add-on" which would considerable increase the spectral analysis capabilities of the computer. Proving worthwhile, the design of such hardware was launched but not completed - due to time restrictions.

As a result of this study, it became apparent that all three systems (DFT, FFT, FFT hardware) are capable of operation in real time. However, the upper frequency limit for the DFT would be approximately 50 Hz, for the FFT 500 Hz, and for the FFT hardware 2500 Hz.

DD FORM 1 NOV 65 1473 (PAGE 1)

S/N 0101-807-6801

UNCLASSIFIED.

Security Classification

14 KEY WORDS	LINK A		LINK B		LINK C	
	ROLE	WT	ROLE	WT	ROLE	WT