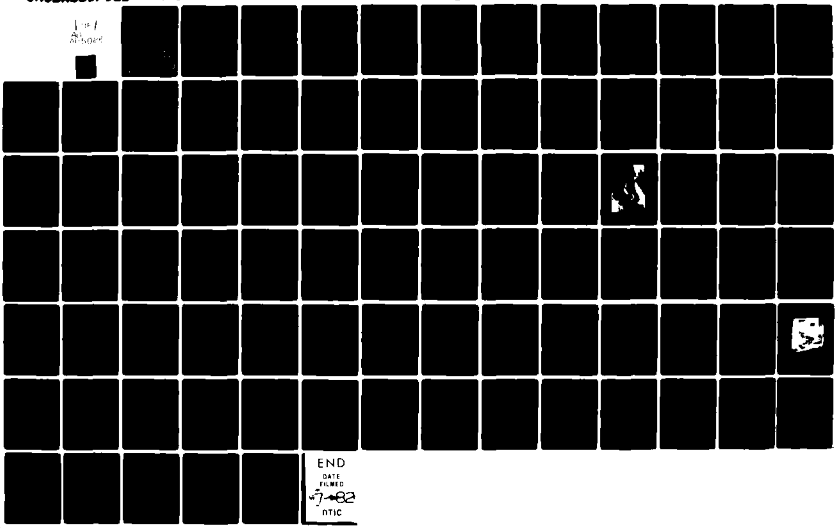
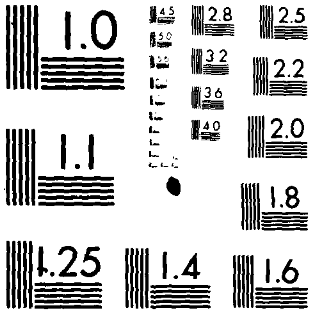
AD-A115 026   MASSACHUSETTS INST OF TECH  LEXINGTON LINCOLN LAB   F/G 9/2
THE PROGRAMMABLE MATCHED FILTER: A DESIGN STUDY.(U)
APR 82  A E FILIP, J D DRINAN, A H HUNTOON   F19628-80-C-0002
UNCLASSIFIED  TR-603   ESD-TR-82-005   NL

1 OF 1
AD
A115026



END
DATE
FILMED
7-82
DTIC



MICROCOPY RESOLUTION TEST CHART
NATIONAL BUREAU OF STANDARDS-1963-A

ESD-TR-82-005

12

# Technical Report 603

A.E. Filip
J.D. Drinan
A.H. Huntoon
J.D. Kurtze
D. Malpass

# The Programmable Matched Filter: A Design Study

1 April 1982

Prepared for the Department of the Air Force
under Electronic Systems Division Contract F19628-80-C-0002 by

**Lincoln Laboratory**
MASSACHUSETTS INSTITUTE OF TECHNOLOGY
*LEXINGTON, MASSACHUSETTS*

Approved for public release; distribution unlimited.

The work reported in this document was performed at Lincoln Laboratory, a center for research operated by Massachusetts Institute of Technology, with the support of the Department of the Air Force under Contract F19628-80-C-0002. A part of this support was provided by the Rome Air Development Center.

The views and conclusions contained in this document are those of the contractor and should not be interpreted as necessarily representing the official policies, either expressed or implied, of the United States Government.

The Public Affairs Office has reviewed this report, and it is releasable to the National Technical Information Service, where it will be available to the general public, including foreign nationals.

This technical report has been reviewed and is approved for publication.

FOR THE COMMANDER

Raymond L. Loiselle

Raymond L. Loiselle, Lt.Col., USAF
Chief, ESD Lincoln Laboratory Project Office

MASSACHUSETTS INSTITUTE OF TECHNOLOGY
LINCOLN LABORATORY

# THE PROGRAMMABLE MATCHED FILTER: A DESIGN STUDY

*A.E. FILIP*
*J.D. DRINAN*
*A.H. HUNTOON*
*Group 27*

*J.D. KURTZE*
*Group 21*

*D. MALPASS*
*RCA*

**TECHNICAL REPORT 603**

**1 APRIL 1982**

***Approved for public release; distribution unlimited.***

LEXINGTON MASSACHUSETTS

## ABSTRACT

The Programmable Matched Filter was envisioned as a flexible approach toward real-time, multi-dimensional matched filtering problems. Its design features multiple, parallel arithmetic elements communicating with multiple memories via a crossbar switch at a clock rate of 16.7 MHz. The machine will sustain a throughput rate of more than 500 million real operations per second, or more than six Cray-1 computers. The extensive use of low-dissipation CMOS technology in large scale integrated circuits yields an estimated total of 5200 integrated circuits, dissipating less than one kilowatt, occupying 0.2 cubic meters and weighing approximately fifty kilograms.

# CONTENTS

Appendix

## LIST OF TABLES

## LIST OF FIGURES

# Chapter 1

# INTRODUCTION

## 1.1 OVERVIEW

The Programmable Matched Filter (PMF) was originally conceived as a solution to real-time, multi-dimensional matched filtering problems requiring a degree of flexibility not readily available in dedicated hardwired systems. Moreover, the PMF was to be used in an operational setting which required that size, weight, power, and reliability had to be considered in parallel with the logical design of the machine. Thus this design study addresses not only the architectural features of the PMF, but also digital technology trade-offs, use of semi-custom, large scale integrated circuits (LSI), and advanced packaging techniques which were becoming available in 1979 when the study was performed.

In the taxonomy of distributed processing systems, the PMF would be classified as a single-instruction-multiple-data (SIMD) type machine. This is because it contains multiple, parallel arithmetic elements (AE's) which respond to a single control stream. If additional flexibility were required, programmable processors could be attached to the AE ports, thereby converting the PMF to a multiple-instruction-multiple-data (MIMD) type machine.

The basic architectural features of the PMF include a pool of 32 high speed working memories, four basic AE's, four auxiliary AE's, a crossbar subsystem to interconnect the memories and AE's, and a control subsystem. (Figure 1). Interfaces to the outside world masquerade as AE's to simplify their connection to the working memories and to allow data transfers to overlap data processing. The clock rate for the PMF is 16.7 MHz (60 nsec period) which allows the use of LSI circuits in the implementation.

The combination of modest clock rate, multiple AE's and separate control and data processing hardware results in a peak com-

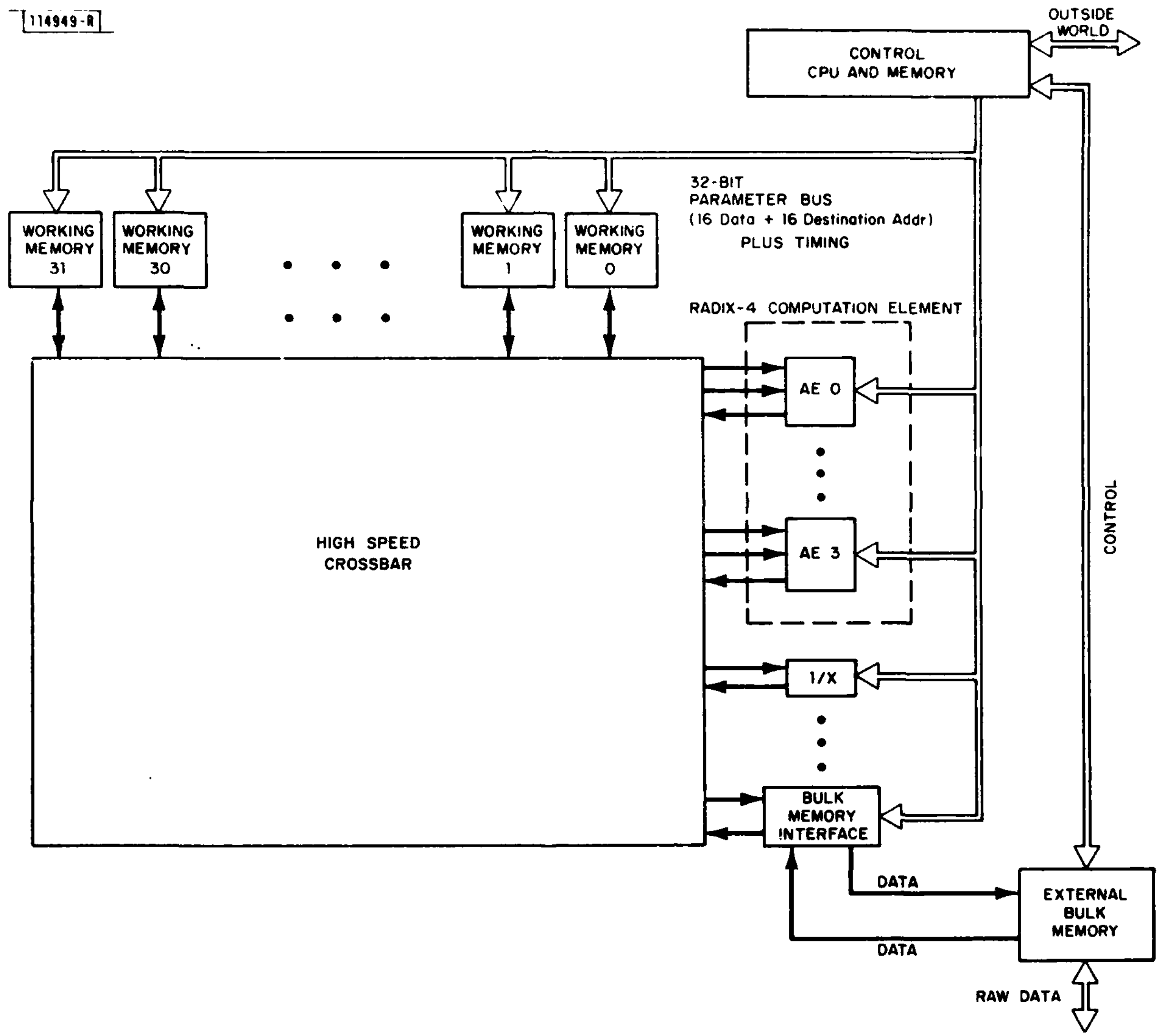


Fig. 1. PMF Block Diagram

putation rate of over 500 million real operations per second (MOPS). For comparison, the Cray-1 computer advertises a maximum of 80 MOPS. The 500 MOPS computation rate can be sustained for fast Fourier transform (FFT) calculations. Other algorithms may use less of the available pool of AE's yielding a proportionately lower throughput rate.

The projected physical parameters of the PMF are equally impressive due to extensive use of LSI, CMOS compunents. A total of 5200 integrated circuits (IC's) dissipate 750 watts, occupy a volume of 0.2 cubic meters, and weigh only 50 kilograms. This implies tht the computing power of half a dozen Cray-1 machines can be condensed into a package which is comparable to a large, desk-top computer.

Because matched filtering calculations are dominated by FFT computations, the PMF was designed primarily for a complex data type. The data is represented in full floating point notation with a sign bit plus a 16 bit mantissa, and a 7 bit exponent for a total of 48 bits per complex word. To improve system reliability, error detection and correction logic (EDAC) was included to provide single error correction and double error detection (SECDED). EDAC provides immunity to random errors in the memories and in the crossbar and, further, allows operation in a degraded mode with single point hardware failures. SECDED EDAC requires the addition of 7 bits to each complex word for a total word length of 55 bits.

## 1.2 ORGANIZATION

Chapter 2 reviews the classical, multi-dimensional matched filtering problem and derives the throughput requirements for the PMF. Chapter 3 discusses the architectural features of the PMF, and describes in turn the four major subsystems: (1) working memories, (2) arithmetic elements, (3) crossbar switch, and (4) control. Hardware considerations that influence the actual implementation of the PMF are described in Chapter 4. Chapter 5 discusses the software aspects of the PMF design including both support and application software.

Appendix A contains a summary of the mean-time-between-failure (MTBF) calculations for the PMF. Finally, Appendix B contains a sample PMF assembly language program.

# Chapter 2

# SYSTEM REQUIREMENTS

## 2.1 INTRODUCTION

The PMF was intended for those applications which require a real-time, multi-dimensional matched filtering capability. Examples of such systems include synthetic aperture radar systems (SAR), radar imaging systems, infra red systems, and standard television systems. Depending on the system, the data may be a function of azimuth and elevation angles, range and Doppler velocity, or simply x-y-z spatial coordinates.

Because the PMF is programmable, it isn't necessary to tailor its design to any particular application. Only the generic characteristics of such problems must be identified to establish the design goals for the PMF. To this end, the next section briefly reviews the theory of multi-dimensional matched filters, and identifies the functional requirements for the PMF. To establish approximate computation rate requirements, a real-time television imaging problem is examined in Section 2.3.

## 2.2 MATCHED FILTERS

For this design study, a matched filter is taken to be a linear system whose impulse response is chosen to optimize some characteristic of the output of the filter. In the classic detection problem, the matched filter maximizes the output signal-to-noise-ratio (SNR), or alternatively, it maximizes the detection probability for a fixed false alarm probability. In communications systems, the probability of miss and probability of false alarm are combined into a single figure of merit, the probability of error, which is minimized by use of a matched filter. In an estimation context, the matched filter minimizes the mean-squared error of the estimate.

### 2.2.1 Fast Convolution

The output of the matched filter is generated by convolving the input data with the impulse response of the filter. When the duration (in space and/or time) of the filter impulse response is small, then the most efficient algorithm is to compute the multi-dimensional convolution directly. In this case, the number of arithmetic operations needed to genearate each output point is proportional to the number of samples in the multi-dimensional impulse response.

For extended impulse responses, the most efficient realization is by means of "fast convolution" (Ref. 7). This algorithm first computes the Fourier transform of the input data, multiplies it by the transfer function of the matched filter, and finally, computes the inverse Fourier transform of the product. The number of operations needed per output point for fast convolution is proportional to the logarithm of the number of points in the multi-dimensional observation window.

The point at which fast convolution becomes more efficient than direct convolution depends upon the number of samples in the impulse response as compared to the size of the observation window. As the dimensionality of the problem increases, the crossover point in computational efficiency occurs for smaller impulse response durations. Consider a d-dimensional problem. Assume that the impulse response contains p**d points and the observation window contains N**d points. (The double asterisk denotes exponentiation.) Then the ratio of computational efficiency is proportional to (p**d)/d* log(N).

Since many applications involve extended impulse responses, a fast convolution capability was required for the PMF. This implies that the PMF must be very efficient in the execution of the fast Fourier transform (FFT) algorithm.

### 2.2.2 Adaptive Whitening

The second functional requirement is derived from the need to perform adaptive whitening in some applications. A whitening capability is required when the additive background noise has a non-uniform power spectral density function. The term "whitening" refers to multiplying the Fourier transform of the received data by the reciprocal of the power spectrum of the noise.

In the event that the noise power spectrum is not known a priori, or if it changes over space and/or time, then the matched filter must incorporate a means for adaptively estimating the power spectrum. The algorithm used to estimate the power spectrum is based on the classical smoothed periodogram approach. This algorithm merely convolves the magnitude squared of the Fourier transform of the input data with a smoothing function to control statistical fluctuations.

### 2.2.3 Windowing and Reference Function Multiplication

The next operation required in a multi-dimensional matched filter is to multiply the whitened data by the conjugate of the Fourier transform of the reference signal. The complex multiplier used to perform the multiplication can also be used to window or weight the input data prior to the forward Fourier transform. The ability to apply arbitrary, multi-dimensional windows to the data is used to improve spectral resolution and control the leakage of strong spectral components (e.g., jammers).

### 2.2.4 Threshold Comparison

One final requirement placed on the PMF design was the ability to perform the operations of threshold comparison and peak detection. For the detection and communication problems, these operations are invariably cascaded with the output of matched filters.

### 2.2.5 Summary

The above considerations lead to the matched filter functional block diagram shown in Figure 2. The functional requirements include (1) FFT's, (2) adaptive whitening, (3) windowing and reference function multiplication, and (4) threshold comparison.

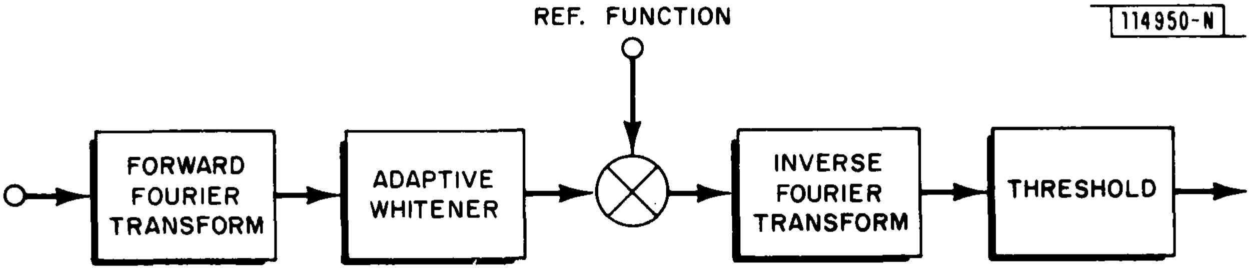


Fig. 2. Matched Filter Functional Block Diagram

## 2.3 COMPUTATIONAL REQUIREMENTS

The previous section established the functional requirements for the PMF. This section derives the computational requirements for the PMF. They are based on a hypothetical, real-time television imaging problem.

Assume that the television sensor generates 256 x 256 pixel frames at a rate of 30 Hertz for a system intended to detect the presence of spatially resolved vehicles (moving and stationary). Assume further that the power spectrum of the background clutter is unknown so that the adaptive whitening function must be used. The signal-to-noise ratio is assumed to be high enough so that the frames may be individually processed to determined whether or not a target is present. The last assumption is importanr because it avoids the need to integrate groups of frames prior to matched filtering.

The fast convolution algorithm applied to individual frames requires that forward and inverse Fourier transforms of size 256 x 256 be computed every 33 msec. Assuming a standard radix-2 implementation, the computational requirement for this portion of the algorithm is 157 million real operations per second (MOPS). A recursive implementation of the adaptive whitener results in the need for an additional 7 MOPS. Finally, windowing, thresholding, and reference function multiplication would each require approximately 2 MOPS. These requirements are summarized in Table 1. Thus, a reasonable computational goal for the PMF is a throughput rate on the order of a few hundred million real operations per second. In addition, the FFT calculation clearly dominates the computational load, accounting for over 90% in the above example. Thus, the PMF must be particularly efficient in performing FFT calculations.

TABLE 1

COMPUTATIONAL REQUIREMENTS FOR REAL-TIME TELEVISION SYSTEMS

| Function | Real Operations per Second (x10$^{6}$) |
|---|---|
| FFT | 157 |
| Whitening | 7 |
| Threshold | 2 |
| Window | 2 |
| Ref Func | 2 |
| | ----- |
| Total | 170 |

# Chapter 3

# PMF ARCHITECTURE

## 3.1 INTRODUCTION

The PMF design evolved from a desire to provide a more flexible approach toward real-time, multi-dimensional matched filtering while retaining as many of the desirable features (e.g., high throughput, small size, weight, power) of a hardwired machine as possible. The requirement for high throughput drives the design to use a 16.7 MHz clock rate, a moderate amount of parallelism and pipelining, and separate control and data processing hardware.

The PMF consists of four major subsystems: a set of 32, 4K by 55-bit working memories, a set of powerful arithmetic elements containing a total of 16 floating point multipliers and 24 floating point adders, a crossbar switch to connect memories and arithmetic elements, and a control unit. The following sections describe these subsystems in detail.

## 3.2 WORKING MEMORIES

The working memories are not required to be random access because signal processing algorithms are generally structured to require orderly access to subarrays of their input data. Each working memory is provided with a block structured memory controller (Figure 3) which accesses data in a fashion similar to the way nested FORTRAN DO-loops can access a multidimensional array of data. This control structure provides a way for programmers to relate previous experience to the new architecture and should aid in reducing the cost of coding. The memories only function in either a read or a write mode during one block transfer. As described more fully in Section &xbarsect, this fact allows the crossbar to be split in two parts, the first serving the inputs of the AE ports and the second serving the AE outputs thereby reducing the size of the crossbar.

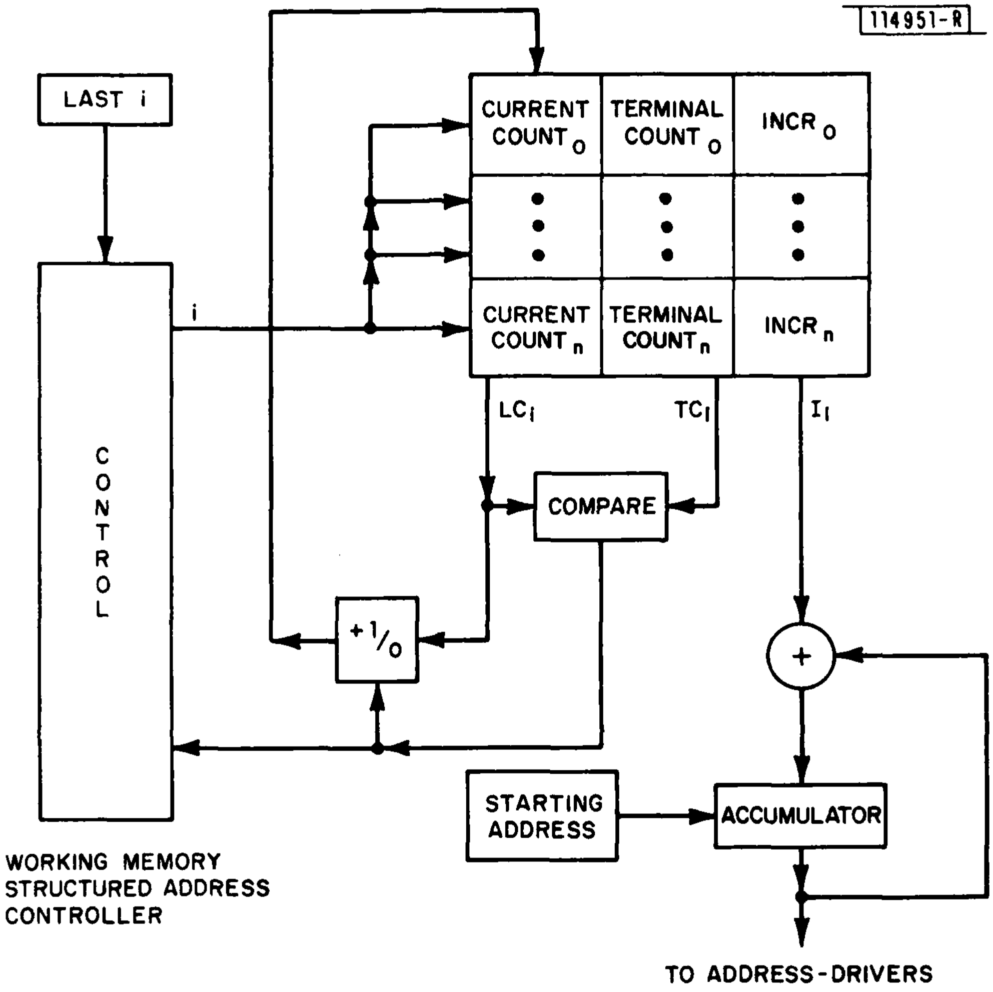


Fig. 3. Structured Memory Controller

Data transfers need not proceed as fast as the machine cycle time. Each memory controller can be interlocked with an enable such that transfers occur only when the controller is enabled. Thus slower external devices may be accommodated by the PMF with no buffering required in the interface, only a simple handshake protocol.

At any particular time some of the memories will be idle. The PMF architecture takes advantage of idle time by forcing an idle memory into a 'listener' state thus allowing chaining of AE's as described in Section &xbarsect on the crossbar.

The current design of the processor assumes 32 working memories each capable of storing 4K (4096) data words of 55 bits each. Each structured access controller (Figure 3) accepts certain parameters from the central control program. These parameters include:

1. A Starting Address for the block read or write.

2. Several loop parameters: Address Increments, Terminal Counts, and Current Counts. The lowest level increment is added to the current address and its associated current count is incremented until it equals the corresponding terminal count. The next higher level is then enabled for one cycle after which control returns to the lowest level. Current Counts are normally initialized to 0 prior to any block access. Four levels of loop control are known to be useful for matched filtering problems.

3. A Data Source for a block write into memory. The data source is the number of the AE port to which the memory must listen and is actually a crossbar configuration parameter, but is associated with the data sink. Data Source has no meaning for a read operation.

4. A Memory Group Assignment. Parameters may be distributed to working memories individually, or to a group of memories which are logically connected. Frequently data must be divided among a number of memories and read from those memories in parallel. In such a case the number of parameter transfers can be drastically reduced if elements can be grouped together. This grouping is principally of concern for reducing the amount of memory required to store the programs and in the processing of short data sets for which the parameter distribution time is longer than the data processing time. Certain parameters for a group may be different from member to member and those parameters are distributed to the memories individually.

5. The number of a Synchronizing Command Line and a Delay to insure that all elements that are cooperating for a processing subtask are synchronized. When the command is activated a delay counter is started (the delay may be

zero). At the expiration of the delay, the parameters are loaded from the holding registers into the active registers. If the controller is busy at the time, an error will be reported. The delay parameter and the next parameter will be covered in more detail in Section &contsect on control.

6. An Event Flag and the number of an Event Line to determine whether the controller should report the end of a block transfer and on which event line.

7. A Configuration parameter to specify miscellaneous items such as action on detection of a synchronization or EDAC error (whether to preserve status and stop (paranoid mode) or to proceed without reporting an error (reckless mode) or to proceed and report an error (normal mode)), and whether the enable is to be used to control storage or retrieval of data.

## 3.3 ARITHMETIC ELEMENTS (AE'S)

The AE ports are available to serve whatever special processing elements may be required. The PMP may therefore be tailored to the specific task by simply installing appropriate specialized processors. The general, multi-dimensional matched filtering problem described in Chapter 2 is adequately served by the units described below. As in the case of the working memories, configuration is controlled by parameter registers. Parameters are double buffered and reconfigurations are controlled by a synchronization command line and delay. AE's must be told to which working memory their inputs should connect, but have no concern about their output connection. Whatever working memories have been instructed to listen simply do so.

### 3.3.1 Basic AE's

There are four basic AE's (Figure 4) which may be operated independently or in concert. These four AE's are connected to AE ports 0 through 3 (AE0 through AE3) Each basic AE has two input connections (A and B) and one output connection to the crossbar subsystem. Each is comprised of an input section and one complex multiplier/adder followed by two more complex adders. The most general interconnection between the AE's is shown in Figure 4.

The description of the use of this interconnection is deferred until Section &r4sect.. Because of the concern for reliability, error checking may eventually be done in parallel with the arithmetic operations, perhaps using a scheme based on performing the same operations in a small ring (modulo-n arithmetic). The same operation performed on the modulo-n residue of the arguments yields the modulo-n residue of the result. Most errors in the arithmetic operations would produce a result with a different residue. Error correction cannot be easily performed with this technique, but real time error detection is straightforward. However, residue arithmetic is not part of the baseline design. In a less stressing application, one could instead run diagnostics in otherwise 'dead' time between processing tasks.

### 3.3.1.1 Independent Operation

In the independent mode (Figure 5), the input section can perform simple operations on the input data such as changing the sign on either or both the real and imaginary parts, and exchanging the real and imaginary parts. This capability corresponds to the ability to perform any combination of negation, conjugation, and multiplication by 'j'. In addition the 'B' input can latch its data to provide for operation by a constant thus reducing the need to tie up multiple memories with constants (one for each AE to be used simultaneously).

The complex multiplier/adder can perform either a multiplication, an addition (or subtraction), or a magnitude-squared operation on either or both of its input ports. The first adder following this section can be used either to pass the data unchanged (transparent mode), to accumulate, or to perform the thresholding function so common in detection processors. Thresholding may only be performed on real data and so the magnitude-squared of complex data would be calculated first. The threshold is set by the 'B' input and may, therefore, be latched. If it is complex it will be subjected to a magnitude-squared operation. Thresholding may proceed in two fashions. Either all of the data is passed to the output in the real part of the complex word with an indication of a threshold crossing in the imaginary part (a bit map), or only data which exceeds the threshold will be passed to the output with a sample number stored in the imaginary part (data compression). The sample number is produced by the imaginary part of the adder performing as a counter. In the latter mode, the enable input of a working memory controller would be used to store only data which passes muster.

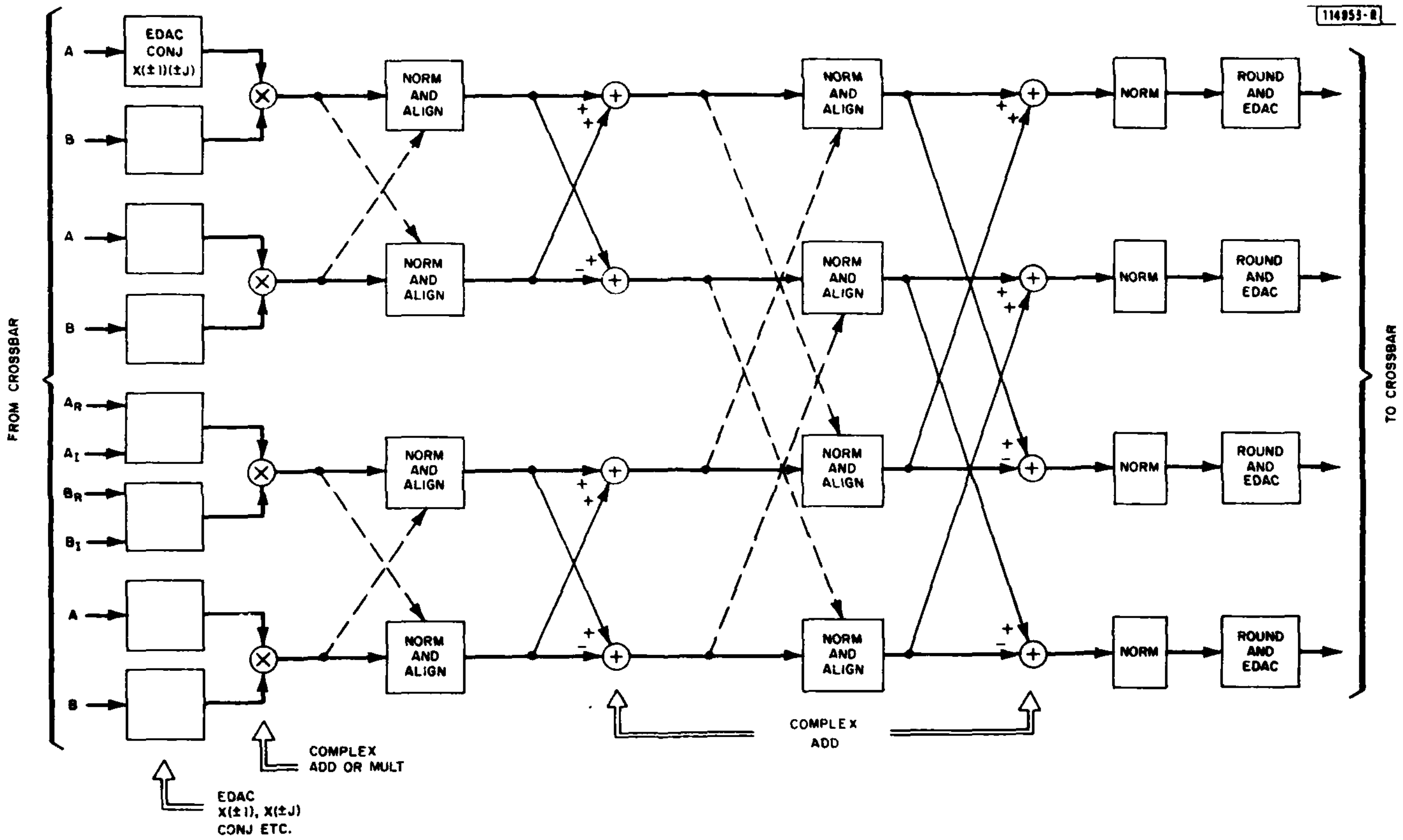


Fig. 4. Four Basic AE's

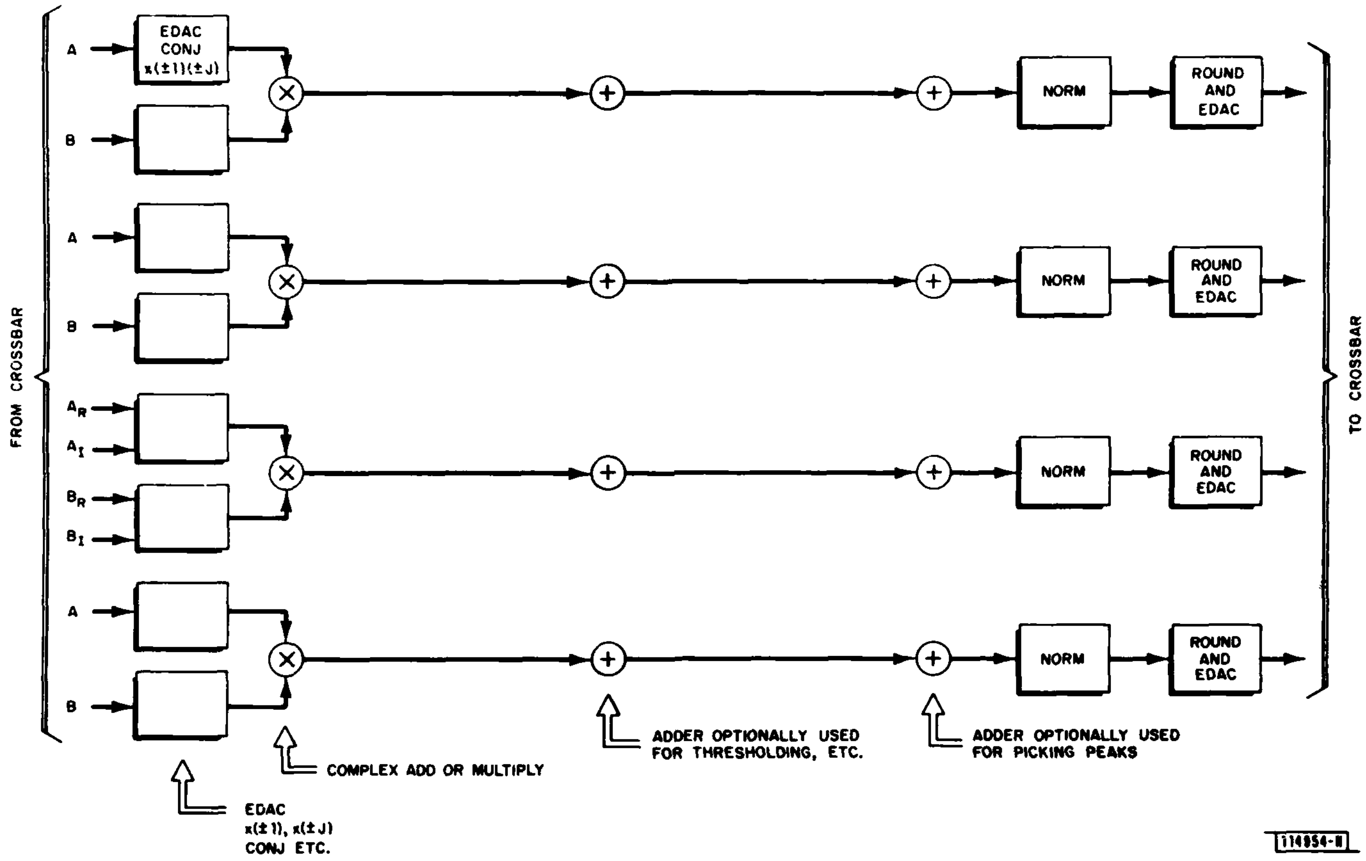


Fig. 5. Four Basic AE's - Independent Mode

The final adder is capable of operating in transparent mode or of reducing thresholded data even further by searching for local maxima. In the peak picking mode, this adder would release a datum which had been thresholded with compression only when both the preceding and the succeeding data had been smaller, i.e., when the sign of the difference between succeeding data changes from minus to plus.

If the threshold operations are ignored because of their infrequent use, the number of floating point operations performed each clock cycle by each basic AE is 2 for a complex add and 6 for a complex multiply (4 real multiplies and 2 real adds). This results in operation rates of between 133 and 400 MFLOPS for the basic AE's operating independently.

### 3.3.1.2 Radix 2 Operation

The basic AE's may also be configured in pairs. AE0 is paired with AE1 and AE2 with AE3. The pairing is independent in that, for example, if AE0 is operated in concert with AE1, then AE2 and AE3 may either be operated together or independently. In this mode the complex multiplier/adder operates as a multiplier and the uncommitted input of the adder which follows is connected to the output of the multiplier of the other AE (Figure 6). The lower adder performs a subtract operation rather than an add. The final adders operate in transparent mode. This configuration performs the basic radix 2 butterfly operation required for the FFT. In addition, the fact that there is a multiplier on each input allows weighting of the input data set on the first stage of the FFT with no time penalty. If the AE's had been restricted to the independent mode only, the FFT's could still be done, but data would have to be passed through the AE's twice, first for the multiplication, and second for the adds and subtracts. Since only half of the data are multiplied by a coefficient in a stage of an FFT, the independent mode would require 50% more clock cycles than the radix 2 mode.

The FFT requires an unusual distribution of the output data for efficient operation from stage to stage. The basic restriction placed on the PMF is that both outputs of a single radix-2 operation must be stored in a single working memory. The hardware to implement this is contained in the crossbar and its operation is described in Section &xbarsect..

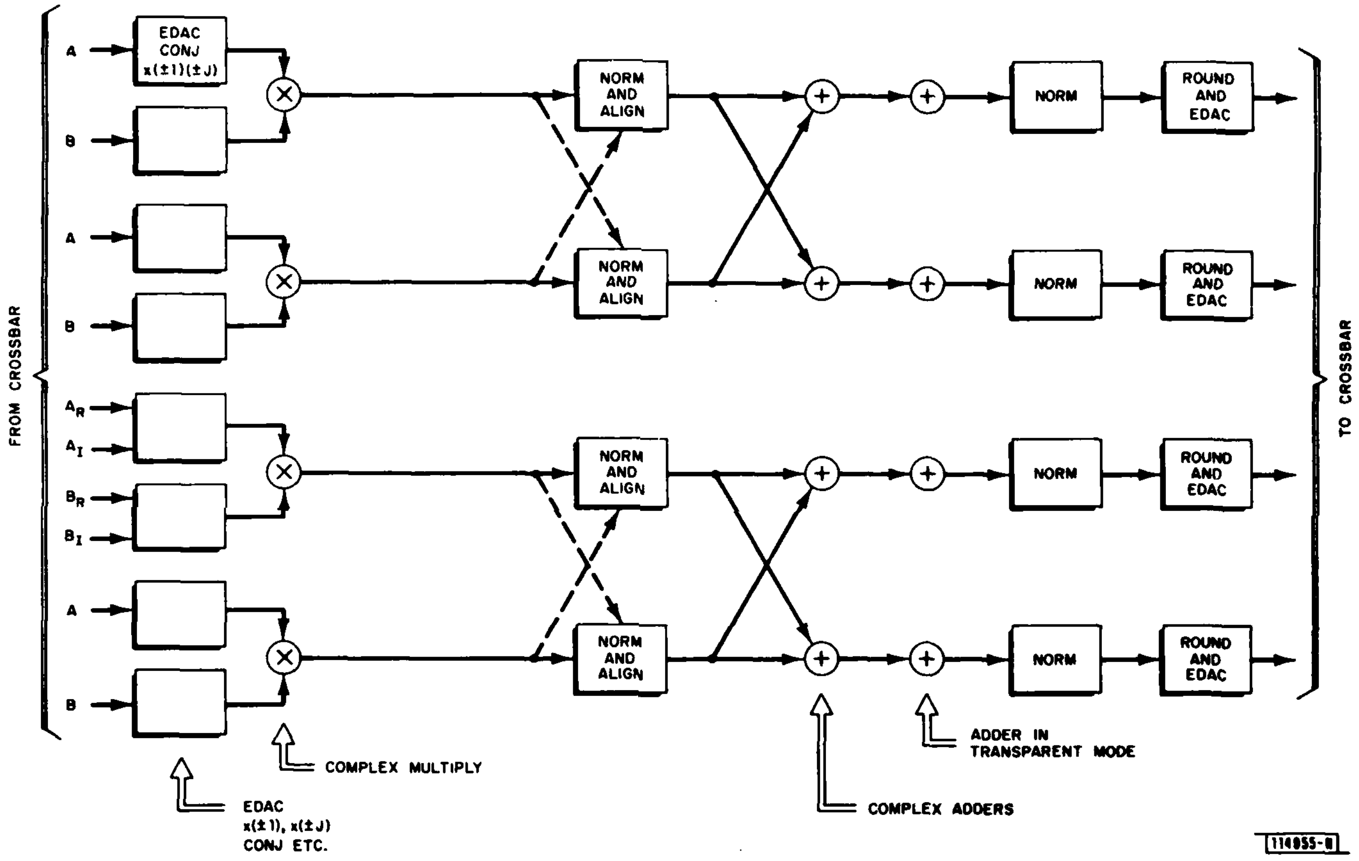


Fig. 6. Four Basic AE's - Radix 2 Mode

If both pairs of basic AE's operate in the radix 2 mode, the useful operation rate is 333 MFLOPS without weighting, or 533 MFLOPS with the input data set weighting.

### 3.3.1.3 Radix 4 Operation

The pairs of AE's may be interconnected to produce a radix 4 computational element. The only change is to cross-connect the inputs of the final adders with the corresponding element on the other radix 2 butterfly (Figure 4). An additional multiplication by '-j' must take place between the adders of AE3, but this is simply an exchange of the imaginary part and the negated real part and can be hard wired directly. All inputs except the input to AE0 must be multiplied by a coefficient. Since there are 4 multipliers available, input ata may also be weighted with no penalty in the radix 4 mode. The radix 4 octopus performs the equivalent of 4 radix 2 butterflies in one operation. Thus the radix 2 mode would require 100% more clock cycles than the radix 4 mode for a given FFT. The radix 4 FFT, however, requires data set lengths which are a power of 4 (an even power of 2). By permitting operation in the radix 2 mode for the final stage of an FFT, data set lengths which are an odd power of 2 are allowed.

As in the case of radix-2 operations, the radix-4 FFT requires an unusual distribution of the output data for efficient operation from stage to stage. The restriction in this case is that all four of the outputs of a single radix-4 operation must be stored in a single working memory. Again, the hardware to implement this is contained in the crossbar and its operation is described in Section &xbarsect..

The useful operation rate in the radix 4 mode is 567 MFLOPS without input data weighting and 667 MFLOPS with weighting.

### 3.3.1.4 Future Generalization

The basic AE, consisting of a complex multiplier and two complex adders could be decomposed into a real channel and an imaginary channel. Each channel would have half of the complex multiplier (two real multipliers and a real adder) and half of the complex adders (two real adders). This form of the basic AE could, by attaching the inputs of the multipliers and the adders to the output of a local crossbar, be configured in any of the above mentioned modes. This implementation would be more general in that other internal interconnections (not necessarily all useful) could be made with only a change of the local crossbar configuration. This could ease the job of processing real data and could reduce the design complexity by replicating a half-sized design twice as many times.

### 3.3.2 Auxiliary AE's

Any special functions are handled by a set of auxiliary AE's. These AE's would be tailored to the details of the processing task to be done. The units described below each have a single input and a single output.

#### 3.3.2.1 Complex Reciprocal Estimator

The matched filtering problem requires division only for the adaptive whitening function. For this reason, only one auxiliary AE is included to implement a reciprocal function. A true divider was not chosen for reasons of speed, size, and power considerations. The reciprocal estimator allows the multipliers to perform division by multiplying one number by the reciprocal of another. A complex reciprocal is simply the complex conjugate of the input divided by its magnitude-squared. The magnitude-squared is calculated and a table lookup, probably in two stages, is performed to find the reciprocal. (Ref 8) The real and imaginary parts of the input are multiplied by this result to produce the complex reciprocal. The output of this AE can be chained through the crossbar to the input of a multiplier without the time overhead of the result having to be stored in and retrieved from a working memory. Chaining is described in Section &xbarsect.

Applications which required higher division rates could implement multiple copies of this AE. A divider AE could also be built which multiplied a second input by the complex reciprocal.

#### 3.3.2.2 Programmable Delays

Programmable delays are devices which allow operations such as sliding window averages or simple finite impulse response

(FIR) digital filters to be performed without the necessity for data to be stored in two separate working memories. Delays could be programmed from 0 to the length of a working memory (currently 4096 words). In such cases, the data would be read from a memory and sent both to the delay memory and to one input of an AE. The output of the delay memory would be chained to the appropriate AE input to allow a weighted sum of the data set and the delayed data set to be produced. Without this memory, data to be processed in this fashion would have to be stored in two different working memories to be read in a staggered manner. The overhead of these extra memories for data waiting to be processed could be a significant fraction of the pool of working memories.

#### 3.3.2.3 Input/Output Ports

Data enters and exits the PMF through 'pseudo' AE's. Several of these ports may be active at one time which facilitates high speed loading and unloading of the working memories. Input and output ports are independent and may be used simultaneously. As described in Section 3.2, a simple handshake protocol can be used to match the working memory data rates to the desired I/O rates. One of the I/O ports will allow the control computer, a general purpose machine, to access the crossbar. This allows initialization of any memory constants, access to data for conditional branching in the software, and several maintenance functions on the data paths, memories, and AE's.

## 3.4 <u>CROSSBAR</u>

The minimum requirement for the crossbar is that any working memory input may be connected to the output of any AE, and that any AE input may be connected to any working memory output. Although a given interconnection 'map' is apt to remain unchanged for many system clock pulses (e.g., the duration of one stage of an FFT), the transition to the next configuration must take place rapidly - ideally within one clock period - if the PMF utilization is to remain high. The functional specification and resultant characteristics of the crossbar subsystem are summarized in Table 2. Figure 7 pictures the minimum functional requirement, and by indicating the signal count (assuming a 48 bit complex word plus 7 EDAC bits) implicitly eliminates a data bus structure from serious consideration. A bus permitting simultaneous flow between all AE ports and 32 working memories would have to contain over 2000 signals. Even at 40 signals to the inch, this represents a physical need for a four foot wide bus structure. This, and the complication and bottleneck problems inherent in bus structured systems, caused busses to be rejected due to the high speed nature of this application. Where appropriate, busses will be used outside the high speed data paths to distribute control information throughout the machine.

TABLE 2

CROSSBAR CHARACTERISTICS AND SPECIFICATIONS

1. Permits any working memory input to connect to any AE output.

2. Permits any AE input to connect to any working memory output.

3. Permits chaining data from one AE output directly to another AE input without necessarily writing into a working memory.

4. Provides a commutation function without additional logic by processing data-source addresses at working memory inputs.

5. 32 bidirectional columns (55 bits wide) available for working memory connections.

6. 12 rows for Basic AE's (8 inputs, 4 outputs).

7. 16 rows (8 inputs, 8 outputs) for auxiliary AE's.

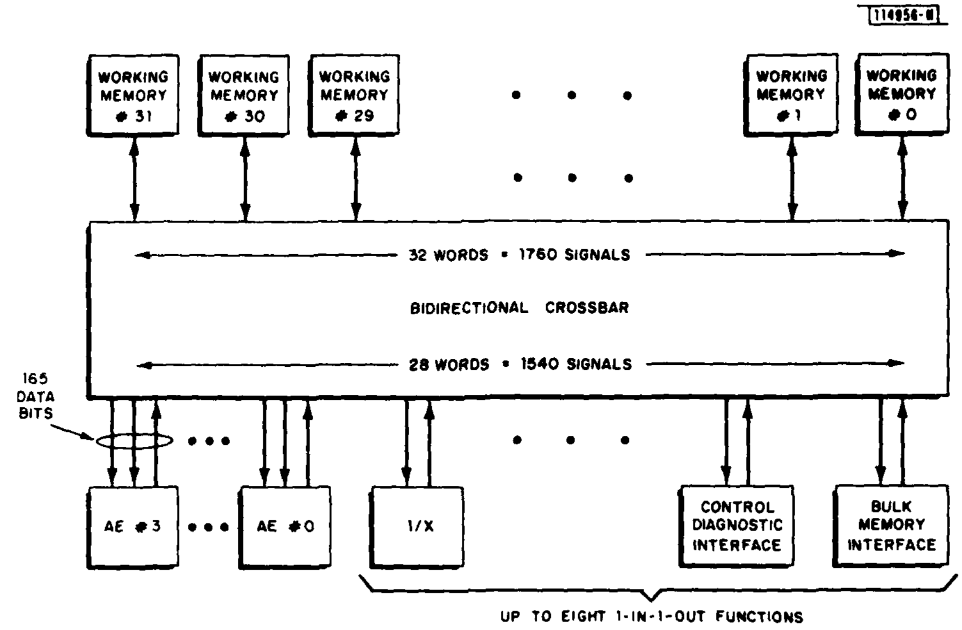


Fig. 7. PMF Interconnection Requirements

### 3.4.1 Basic Crossbar Operation

Conceptually, the PMF is realized as an L-shaped device with the 32 working memories connected to the upper portion of the crossbar and the AE's and other computation-like devices, including the interface to external Bulk Memory, connected to the rightmost side. As described in Section &pkgsect, the solution to the interconnection and packaging problems results in a physical implementation remarkably similar to the conceptual one of Figure 8.

The data paths are best illustrated by examining a one-bit slice through the crossbar, as in Figure 9. The fact that working memories cannot read and write simultaneously permits tri-state devices to be used at the memory interface boundary, thus halving the number of interconnections. The vertical columns in the crossbar are therefore bidirectional and both the working memory and crossbar IC connections to the columns will use tri-state logic.

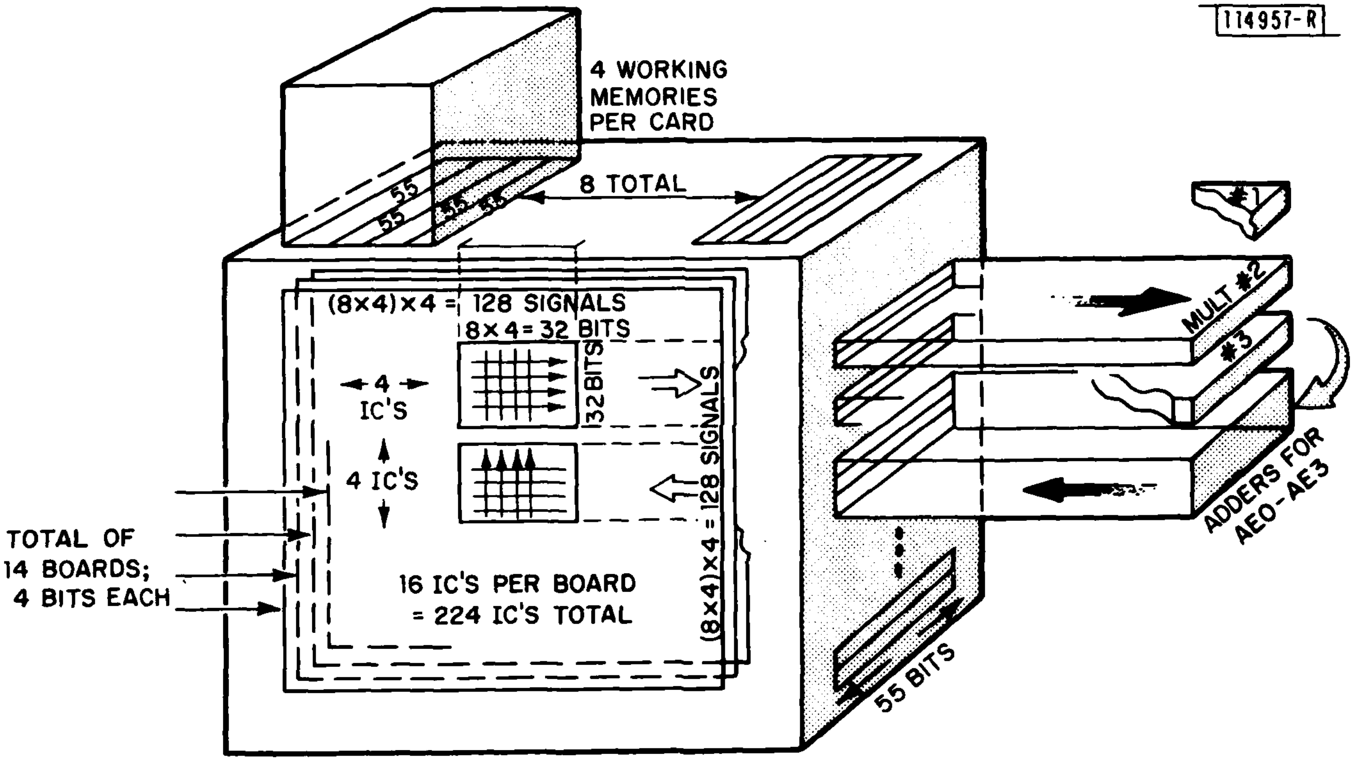


Fig. 8. PMF Conceptual Realization

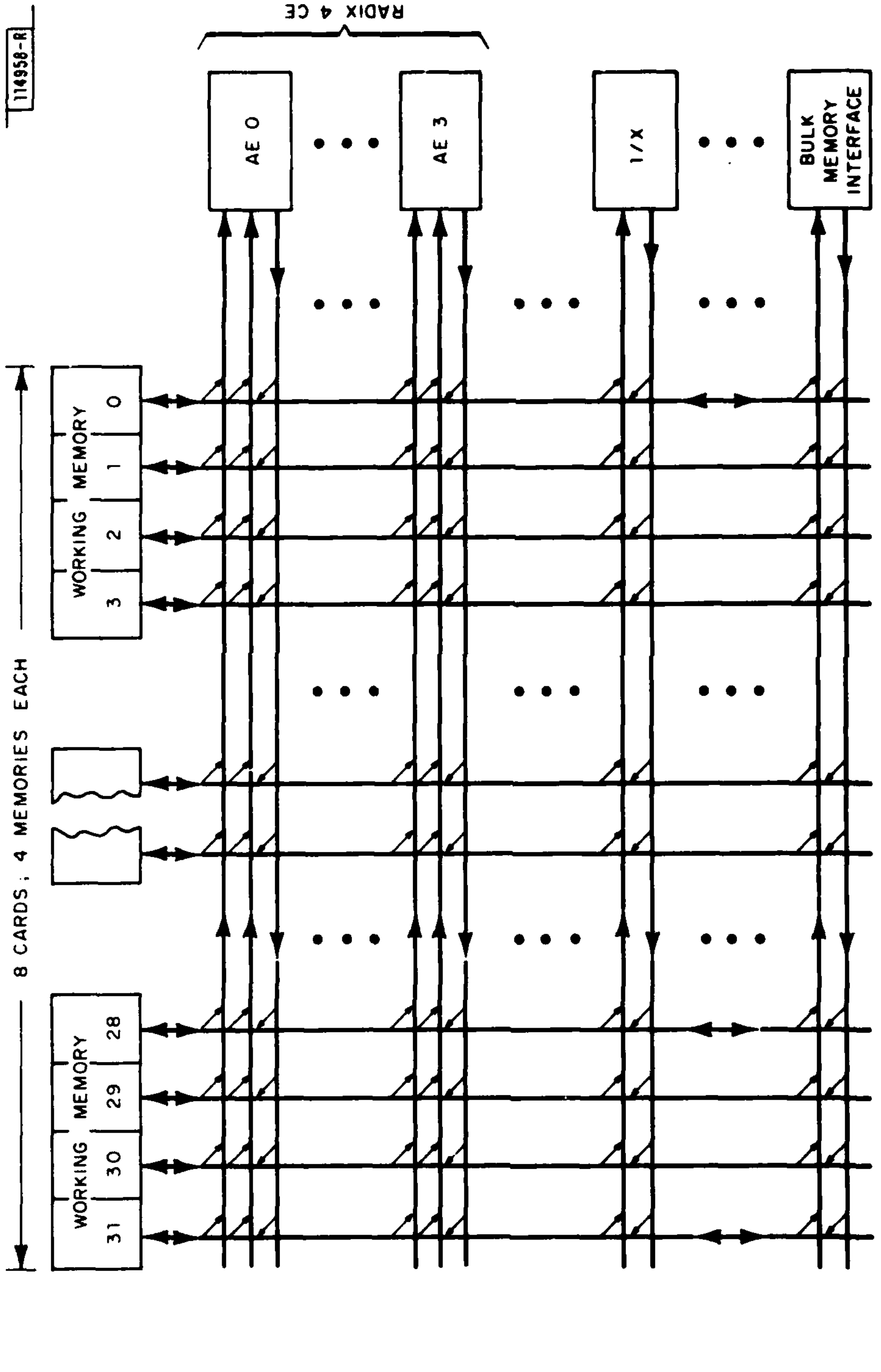


Fig. 9. One Bit Slice Through Crossbar

The arithmetic devices at the right side of Figure 9 simultaneously require one or two inputs and produce outputs. Therefore horizontal crossbar rows are unidirectional, as shown. In the plane of one bit, there are 32 columns (one for each working memory), 12 rows pointing left (one for each AE-like output), and 16 rows pointing right (four basic AE's with two inputs, plus eight auxiliary AE's). The small diagonal arrows in the figure show the possible 'switch closures' between rows and columns.

An important concept in the operation of the crossbar is that these 'switch closures' establishing data flow are specified from the point of view of the destinations (or data sinks). By implementing crossbar control in this way, a number of things are accomplished:

1. The potential conflict of two sources being assigned to the same destination has been eliminated. Each AE input designates a five bit (one of 32) column as its data source, and hence only one column may be connected to any right-pointing row. Similarly, each working memory commanded to write data supplies a four-bit row-source-number to the crossbar, so only one upward-pointing 'arrow' can be connected to any column at any instant. (If a working memory is commanded to read, rather than write, a fifth bit commands the crossbar output tri-state drivers to disconnect all upward-pointing connections to that column.)

2. Two or more working memories storing data may specify the same row (AE output) as their data source, thus providing multiple copies of data where appropriate (e.g., for diagnostics).

3. Two or more AE inputs may select the same column (working memory output) as their data sources. This can avoid the need for duplicate tables of coefficients and can also facilitate duplicate calculations for redundancy or diagnostic purposes.

4. Data chaining may be implemented (i.e., the data stream out of one AE may be fed directly into the input of another AE, without the necessity of storing the entire intermediate data in a working memory). The mechanism for carrying this out is to have an idle working memory specify the desired data output (left-pointing row) as its source and have the destination AE input designate that idle working memory as its data source. If desired, the memory may store the data. In this way, a data path is opened up

from the output of the source AE, left to the column serving the otherwise unused memory, down that column and through the closure to the row corresponding to the input of the destination AE.

5. Similarly, a path for copying one working memory into another can be created by providing a 'null AE' (of the eight available one-input, one-output devices) which merely loops its input directly to its output. Hence, its input specifies the column of the source memory, and the destination memory specifies the output row of the 'null AE', completing a working memory to working memory path.

### 3.4.2 Commutation and Delay

As was mentioned in Section 3.3, the crossbar provides the commutation and delay functions necessary to implement the FFT algorithm. An ordinary pipelined implementation of the FFT uses an interstage delay memory (IDM) and commutator, where the IDM is variable in length from stage to stage. (See Ref. 7, Chapter 10.) The PMF is able to use fixed length IDM's with a delay of zero, one, two, or three. Figure 10 is used to illustrate this function of the crossbar in the radix-4 mode. The purpose of this hardware is to allow storing all four outputs of a single radix-4 butterfly operation in a single memory. The commutator changes the connection between the working memory and the AE outputs (after the IDM) every clock cycle.

Consider the data to be stored in the working memory numbered 31 during the first four clock pulses. The commutator is shown in its initial state, so, at the first clock pulse, working memory 31 stores the first output of AE0. The commutator then switches automatically to the output of row 1, but due to the one stage delay on AE1's output, the word stored in working memory 31 at the second clock pulse was generated by AE1 on the first clock pulse. Similarly, the third word stored is AE2's first output (delayed by two clock pulses), and the fourth word stored is AE3's first output (delayed by three clock pulses). The following, fifth, clock pulse finds the commutator back at row 0, so the word stored is AE0's fifth output. The working memory arbitrarily numbered 28 begins to store data at clock pulse 2 and will store the outputs generated by all four AE's on clock pulses 2, 6, 10, etc. This arrangement of delay and commutation allows outputs of one FFT stage to be stored in memory as required for the following stage.

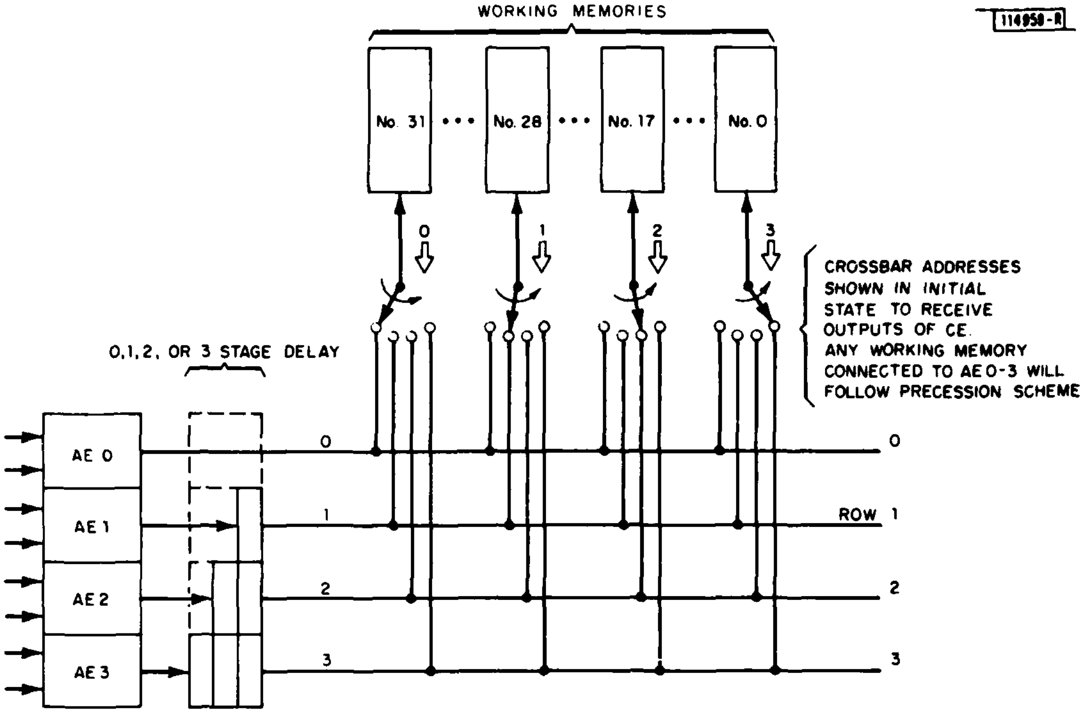

114958-R
WORKING MEMORIES
No. 31
No. 28
No. 17
No. 0
0
1
2
3
CROSSBAR ADDRESSES SHOWN IN INITIAL STATE TO RECEIVE OUTPUTS OF CE.
ANY WORKING MEMORY CONNECTED TO AE 0-3 WILL FOLLOW PRECESSION SCHEME
0,1,2, OR 3 STAGE DELAY
AE 0
AE 1
AE 2
AE 3
0
1
2
3
0
ROW 1
2
3

Fig. 10. Interstage Delay Memories

Operation in the Radix-2 mode is much simpler as only two working memories are involved and only a single unit delay is needed in the bottom output of the Radix-2 CE (either AE1, or AE3). Delays of two or three units are used only for operation in the Radix-4 mode. In such a case, each working memory alternately takes an output from each AE output. The single unit delay ensures that the two sequential words stored in a single working memory are the result of the same CE operation, as in the Radix-4 case.

### 3.4.3 Crossbar Partitioning

As suggested in the discussion of bus structures at the beginning of this section, the interconnection and packaging problems are formidable, even with the crossbar. The solution, as shown in Figure 8, lies in partitioning the crossbar along bit rather than word boundaries, tentative:y four bits to a card. The word axis is therefore perpendicular to the crossbar cards, and as Figure 8 shows, the working memory and AE cards are perpendicular to the cards in the crossbar assembly. The interconnection at the edge of the crossbar must make the transition, and the scheme for doing this results in what have come to be known as 'Corner Turning Cables', shown in Figures 11 and &mockup.. Further details of the packaging scheme are presented in the next chapter.

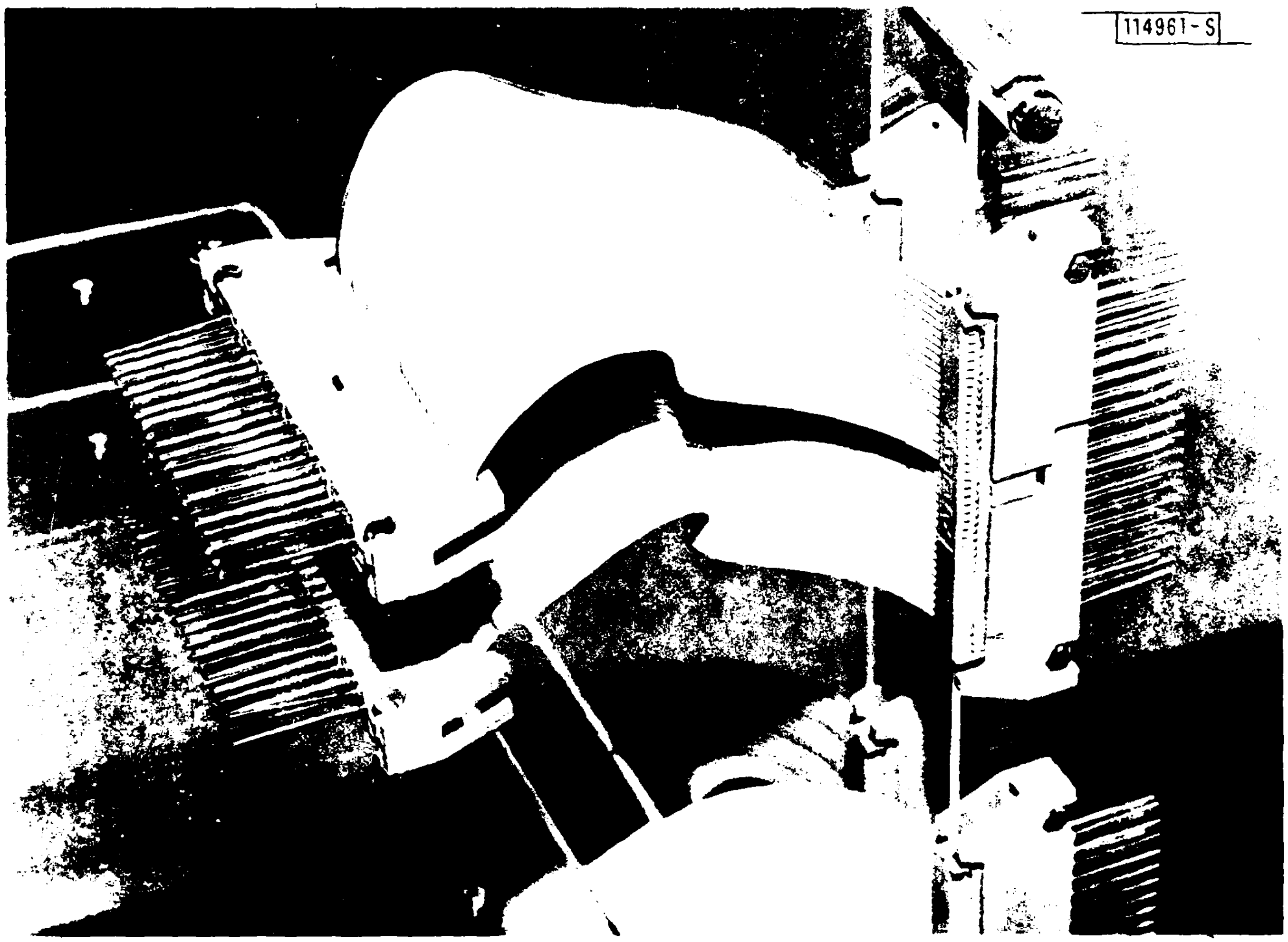


Fig. 11. Corner Turning Cable

## 3.5 CONTROL

Control of the PMF (Figure 12) is hierarchical in nature. On the lowest level are the distributed controllers, such as the working memory's structured address controllers, which are dedicated, hardwired devices. Above these is a parameter memory which simply distributes configuration and synchronization parameters. This function does not require any intelligence and is simply handled by a Direct Memory Access (DMA) controller. The master control of the PMF is the control computer which handles any decision making for the PMF. Most of the distributed control functions have already been described in Sections 3.3 and 3.2 The method of synchronizing the PMF will be described below as will the centralized portion of the control which is separated into two major blocks: the parameter memory and its controller on the one hand and the control computer on the other.

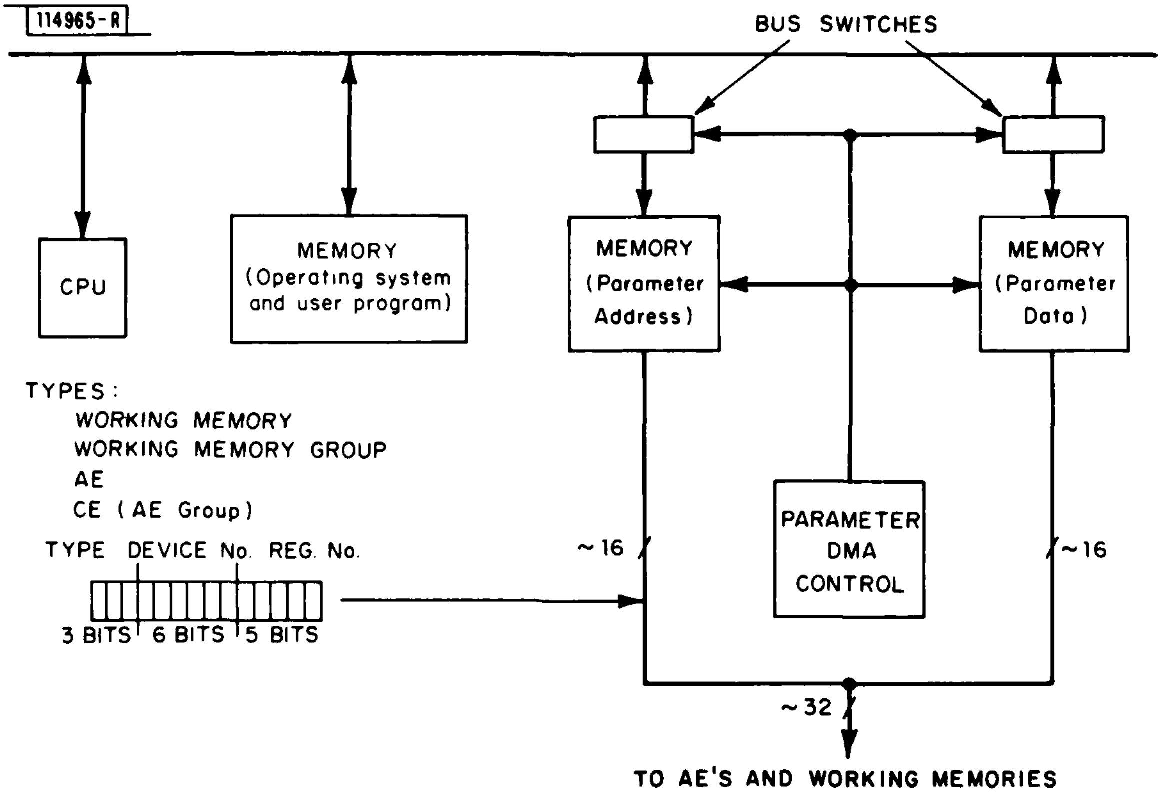


Fig. 12. PMF Control Structure

### 3.5.1 Synchronization Subsystem

Each distributed controller contains a holding register for each of its parameter registers. The number of a command line is distributed to each controller to allow synchronization. When the command line is activated the controller starts a delay count (which may be 0). At the expiration of the delay, the parameters from the holding registers are dropped into the active registers and, in the case of the working memory controllers, data transfers start immediately.

Command lines are activated by an event in the distributed controllers, or by the control computer. An event is primarily envisioned as a report by a working memory that it has completed a block transfer. Such e.ents are not, by nature, transient. They are a report of a 'state' of the PMF and so remain available if they are not used immediately. Connection of event lines to command lines may be made by a small crossbar which is only 1 bit wide. Interconnection is defined by a parameter register for each of the command lines. Parameters for all elements of the PMF participating in an upcoming processing stage are distributed before the interconnection parameter. Thus a conjunction of both the interconnection and the event must occur to initiate a processing stage. It is likely that the PMF design will require about 8 command lines, and 8 separate shared event lines.

Since event lines are shared, a wired-OR or wired-AND function could be allowed. A logical conjunction or disjunction of events could be used to enable a command line by adding to the parameters distributed to the working memory controllers a specification of whether an event is positive or negative logic (active high, or active low) and a similar specification for the command line.

One command line should be available for the important maintenance function of suspending current PMF operations. This could be connected to the error event lines under program control to allow the control computer to diagnose malfunctions and to examine the state of the PMF at the time of the error. For this reason it is highly desirable that the distributed parameter registers be designed to allow the control computer to read at least the bottom rank of the registers. This same command line could also be used to allow the control computer to single step the PMF as part of a diagnostic sequence. In addition, error events should be available to the control computer as interrupt lines so that appropriate remedial action can be taken. If all data must be processed without error, as in certain operational systems, the programmer could specify restart points in the program. The control computer could then retry the operation in case the error was transient (a 'soft' error). Certain types of 'hard' errors could be corrected by reassigning resources (new output working memories or new paths through the crossbar).

### 3.5.2 Parameter Memory

The parameter memory functions as a program memory for the distributed controllers. It simply contains a list of parameters and their associated destination addresses. Each parameter and address is assumed to be 16 bits long (32 bits total) and the pair can be accessed simultaneously. The destination address is tentatively partitioned into three fields: Type (AE, working memory, working memory group, and miscellaneous), Device (which AE, working memory, etc.), and Register (which parameter register within the device). The field lengths must be at least 2, 5, and 4 bits, respectively, leaving 5 uncommitted bits for expansion.

The controller is envisioned as a simple direct memory access (DMA) controller which has two parameter registers: a starting address and a word count. The controller reads a parameter and associated destination address from the memory and places them on a 32 bit wide parameter bus. The bus connects to every distributed controller and is intended to operate at the system clock rate of 60 nsec. The distributed controllers examine the bus, and the appropriate distributed controller(s) capture the parameter. The DMA controller continues placing parameters and destination addresses on the bus until the number of transfers specified by the word count register have been performed. After the parameters for the working memories and the AE's have been distributed, the parameter(s) for the synchronization system would be distributed enabling the reconfiguration of the PMF for the next processing stage upon the occurrence of the proper event. The upper rank of the parameter registers retains its value until reset, so any parameter which is not distributed is assumed to remain unchanged.

The final parameters to be distributed could be the new starting address and word count for the parameter memory controller itself, providing a linked list capability. Nonetheless these parameters will usually be distributed by the control computer itself upon an interrupt from the controller. The interrupt will indicate that the controller has finished a transfer and is available for a new starting address and word count.

Although not required for the baseline application, dynamic reallocation of resources, particularly for working memories, could be achieved without changing the parameter list by augmenting this part of the control system. One level of indirection could be provided on the address portion of the parameter list such that the destination address stored in the parameter memory is actually a logical address which is converted to a physical address by a lookup table. The table would be loaded by the control computer. When reallocation became necessary, e.g., when a working memory ceased to function, one of the other memories from the pool could be substituted by changing only one entry in the lookup table. Without this augmentation, reallocation would still be possible, but each reference to the failed memory would have to be changed in the parameter memory itself.

If it were important to conserve parameter memory space, a second lookup table could be used for parameter substitution. In this case the parameter memory would contain a template of parameters for setting up a processing stage, e.g., a stage of an FFT, and the actual parameters, which differ from stage to stage, would be stored in the parameter memory. A pointer to the beginning of the list of actual parameters would have to be set before the substitution could take place. Again, it is not anticipated that this would be required by the baseline application.

### 3.5.3 Control Computer

The control computer is the only 'intelligent' portion of the control subsystem and it is likely that it would be implemented with a standard 16 bit general purpose computer. It performs housekeeping functions for the PMF and handles the interface to the outside world. Among its jobs would be the loading of the parameter memory after system initialization or when the processing task changes. Access to the parameter memory is possible whenever the parameter memory controller is not accessing it. During such times the parameter memory occupies an extension of the control computer's memory space. Access may be fully random or by means of a window which is mapped to various sections of the parameter memory depending on how much memory the control computer must have for its own functions.

The control computer will also perform any data dependent decision making. For this reason one of the AE ports on the crossbar will be connected to the control computer. In addition this allows the control computer to load any necessary constants into the working memories. Since there is a mismatch in data transfer rates between the computer and the data sections of the PMF, the same handshake protocol as is used with the I/O interfaces is used here.

The housekeeping functions performed by the control computer include the real-time logging and reporting of errors and the determination and execution of remedial actions. If errors are detected, diagnostics may be run while other sections of the PMF continue processing, or the entire machine may be brought down for maintenance testing. The control computer also negotiates the synchronization of the PMF with the outside world. It accepts information on the availability of input data and output devices and sets up and initiates the data transfers. When the appropriate data is available in the PMF the computer initiates data processing.

Chapter 4

# HARDWARE CONSIDERATIONS

## 4.1 DIGITAL TECHNOLOGY COMPARISONS

The successful development of any digital signal processor is a function of both architectural design and hardware implementation. The choice of digital technology is particularly significant for the PMF, whose complexity is sufficient to provide computational rates of 567 MFLOPS, yet physically must be of tractable size, weight, and power. The following sections offer brief summaries of the several technologies that were reviewed for possible use in the PMF. The reader wanting to concentrate on PMF design issues should proceed to Section &cmossect where the preferred ISO-CMOS gate arrays are described.

### 4.1.1 Overview

The choice of PMF technology involves the assessment of existing LSI/VLSI technologies in terms of density, interconnection capability, speed, power dissipation, maturity, and other factors. Previous sizings of systems like the PMF have shown that mixed SSI/MSI/LSI implementations can imply integrated circuit counts of 40,000 devices or more. In terms of the sheer number of interconnections alone and impact upon system reliability, there is ample incentive to identify IC process technologies which offer high logic density (>1000 gates/chip) while also being amenable to contemporary leadless carrier packaging techniques. Accordingly, the availability of custom and/or semi-custom circuits having more than 1000 gates/chip (on average) and upwards of 100 contacts per leadless package were taken as requirements for the PMF.

The PMF achieves its high data rates through the combined effects of parallelism, pipelining, and a nominal clock period of 60 nsec. For most instructions, the 60 nsec epoch represents a good match between memory and arithmetic speeds, tentatively assuming the use of one or more variants of well-known TTL technology. However, while TTL speeds are acceptable, the accompanying power dissipation leads to high power densities when custom VLSI devices having over 1000 gates per chip are envisioned. In fact, much of the system size and weight reduction attributable to the use of leadless custom and semi-custom IC's might be offset by cooling system and power conditioning needs. The crossover point between simple forced air cooling and the need for elaborate and unwieldy liquid systems occurs at surface power densities of about 1.5 watts/cm$^2$, a figure easily exceeded if the usual TTL process is employed for VLSI. Moreover, the resultant D.C. system power budget has a direct bearing on power supply volume and weight allocations. The current state-of-the-art for switching power supplies is approximately 0.1 watts/cm$^3$ and 25 watts/kg. Therefore, a suitable choice for PMF technology would be one which retains the speed of TTL (74,74LS,74S), yet achieves a significant reduction in dissipated power per gate.

Figure 13 provides a first-order measure of the merits of several logic technologies in terms of propagation delay and power per gate. Both standard and custom technologies are shown. The entries in Figure 13 are average numbers, and do not precisely delineate the behavior of some logic families throughout their operating frequency range. For example, while ECL circuits exhibit a relatively flat dissipation characteristic with frequency, low-power Schottky TTL has a quiescent dissipation plus a rate-dependent component at higher frequencies. Complementary technologies (ISOCMOS, CMOS/SOS) dissipate power in direct proportion to operating frequency, with no quiescent component.

The following is a brief discussion of candidate groups of digital technologies in terms of their applicability for the PMF. The eventual technology chosen was semi-custom CMOS, augmented by bipolar clock drivers where necessary. Exceptions to this choice were made for (1) memory devices, an area of design specialization where suitable high-performance devices were available within the marketplace, and (2) control logic, an area where designs were to become final very late in the project, and for which minimal device customization payoffs were expected.

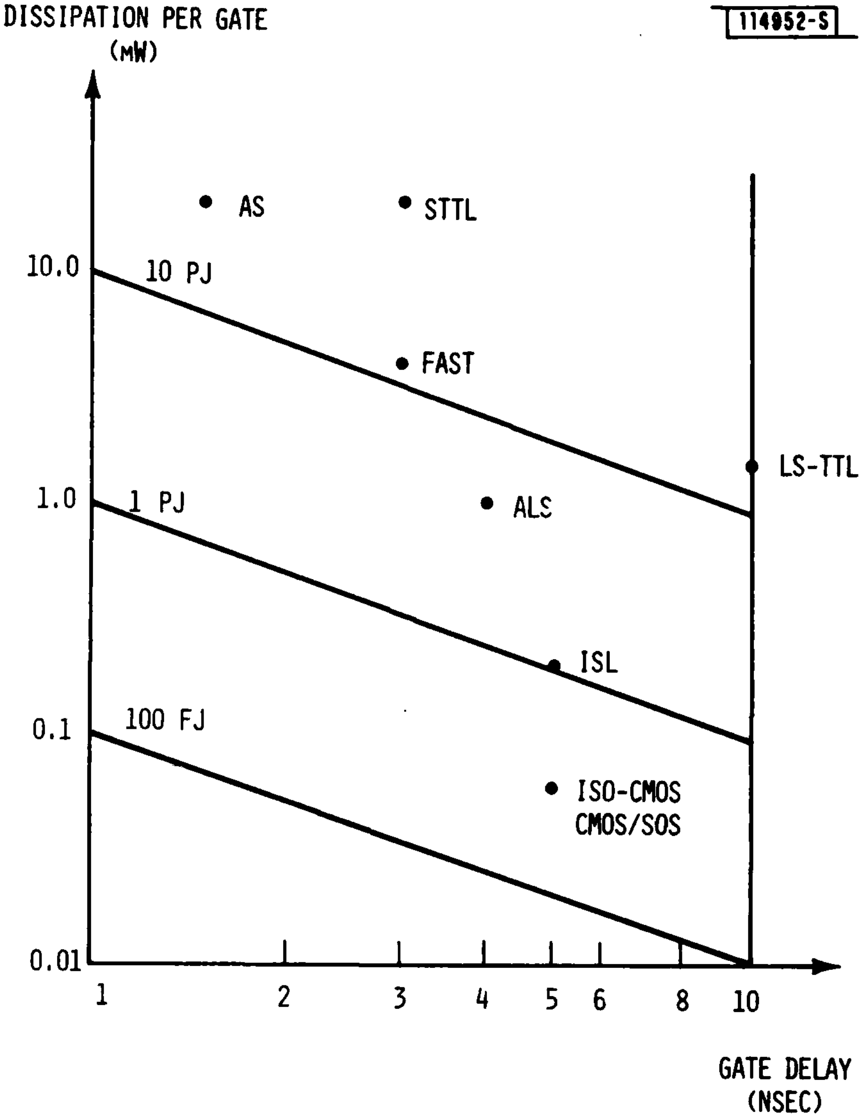


Fig. 13. Speed Power Comparison

### 4.1.2 74,74LS,74S

As a group, the standard series of TTL logic was considered inappropriate due to the unavailability of LSI. None of these standard families offered the prospect of high density packaging needed for the PMF, even if chip carriers were employed instead of DIP's. In particular, the nearly-obsolete 74 series is being supplemented by 74LS, which together with 74S has adequate speed but unacceptable dissipation and density.

### 4.1.3 ALS, AS

New third-generation TTL families were announced by TI and Raytheon in 1978. TI's advanced low-power Schottky (ALS) provides the usual SSI and MSI circuits, but operates at twice the speed and half the power of the LS series. Both TI and Raytheon offer advanced Schottky (AS), an MSI series which operates at twice the speed of 74S with no additional power penalty. Again, in the PMF context, inadequate density was the primary obstacle. Consideration was given to the possibility of assembling thick-film hybrids using ALS chips in an attempt to increase density, but this was dropped for reasons of cost and schedule, and because an alternative was available. *(Section &cmossect)*

### 4.1.4 Macrocell, F200

The use of semi-custom ECL arrays was considered during the early PMF definition phase. In fact, initial system sizings indicated that a Macrocell-based ECL PMF having a 30 nsec clock period could be implemented and would outperform the eventual baseline 60 nsec CMOS PMF by a factor of two. In terms of equivalent computational capability, however, the ECL realization imposed a power requirement of more than six times that of the CMOS version.

The Macrocell product by Motorola is an ECL 10K VLSI chip containing 106 cells (750 gates) which may be customer-configured via metallization into one of 85 logic functions called 'macro's. The resultant semi-custom devices are compatible with standard ECL 10K devices, but power dissipation per chip is typically 4 watts. The F200 gate array by Fairchild uses the Isoplanar II process, and is an ECL 100K compatible, mask-programmable array of 144 internal switches. This F200 LSI product is compatible with standard ECL 100K parts, and dissipates 4.5 watts when the total array is used. Both ECL candidate technologies can be more appropriately used in smaller systems than the PMF, where raw processing speed is mandatory, and where penalties imposed by more specialized cooling, power distribution, and controlled-impedance interconnections are acceptable.

### 4.1.5 $I^2L$, FAST

Integrated-injection logic ($I^2L$) is a relatively new bipolar technology (1975) now being found in ROM's, RAM's, and micro-processors, as well as semi-custom arrays. Historically, the process has been used more for linear IC's than for digital IC's, and the maturity outlook for this technology is uncertain. $I^2L$ devices normally employ a 5 volt power supply to retain TTL compatibility, but may be operated over a range of voltages and depend on current sourcing for switching. In terms of potential density, the $I^2L$ process employs a minimal area emitter/collector region in proximity to an injector rail, but single-level metal interconnects are consumptive of chip area. Gate arrays are known to be available from Exar in sizes up to 500 gates, and from Signetics, whose 8A2000 array offers 2000 gates rated over the commercial temperature range (0-70°C) with typical performance at 15 nsec and 260 microwatts per gate. Fixed injector current is assumed at the recommended +5 volts. As illustrated in Figure 13, $I^2L$ arrays offer reasonable delay-power products (3.0 pJ), but gate delays of 15 nsec are unacceptable for PMF.

Fairchild Advanced Schottky TTL (FAST) is an MSI logic family which, although outwardly TTL, is internally $I^2L$. It offers slightly higher speed than Schottky logic with the power-per-gate only 25% of Schottky, but does not offer the needed PMF densities.

### 4.1.6 ISL

Integrated Schottky Logic (ISL) is a 1200 gate array by Signetics (Philips) which combines low-power Schottky performance with the packing density of $I^2L$. Gate propagation delays are typically 5 nsec for the 8A1200 device. Thus, in terms of gate delay, power dissipation, and array size, ISL offers an attractive fall-back position as compared to the preferred ISO-CMOS gate arrays described in the next section.

### 4.1.7 CMOS/SOS, ISO-CMOS

As the semiconductor industry enters the era of Very Large Scale Integration (VLSI), concern over on-chip power density has led to a renewed interest in CMOS technology. Today, more than a dozen companies offer semi-custom gate array products in CMOS. In this section, two gate array products which offer high potential for the PMF application are described. They are the ISO-CMOS (Isolated CMOS) Masterslice arrays by International Microcircuits, Inc., and the Gate Universal Array (GUA) series in CMOS/SOS available from RCA. An extensive amount of background information has been gathered on these two families, and as of this writing, some combination of devices from one or both vendors has been adopted as the baseline PMF technology. In general, CMOS devices offer high packing densities by virtue of very small p-n junction areas, high noise immunity, and consume virtually no quiescent power. These properties combined with dielectric isolation or silicon-on-sapphire (SOS) substrates offer the genuine prospect of low-power Schottky performance in VLSI (>1000 gate) semi-custom devices, wherein conventional air cooling and power distribution techniques may be employed. Because these technologies normally have limited capability to drive capacitive loads, most systems utilize ceramic hybrid substrates for chip interconnections within leadless hermetic packages. Ceramic circuit cards provide the next higher level of interconnection, and are matched to the substrates in terms of thermal expansion. In addition to its low-capacitance property, ceramic dielectric permits narrow line widths and close spacing via silk screening. Finally, because layer-to-layer vias are possible with this process, multilayer circuits can be used more effectively.

As indicated in Figure 13, CMOS/SOS technology is particularly attractive in terms of delay-power product (0.5 pJ) and its potential for high-performance VLSI as channel widths are progressively reduced. Significant work using this process is occurring at major firms such as RCA, GE, Rockwell and Raytheon. In addition to previously noted power and density advantages, SOS is virtually impervious to radiation because photocurrents cannot be generated in the substrate. In general, the CMOS process permits some latitude in gate propagation delay as a function of supply voltage. However, the price paid for this flexibility is the need for logic level translation when interfacing with standard (e.g., TTL) devices. For PMF purposes, where a significant portion of the machine resquire special logic design, the CMOS/SOS universal array series (GUA) by RCA has appeal and has been fabricated as 632 gate structures having 5-6 micron channels. Four micron devices will soon be offered. The manufacturing costs of SOS devices should also be improving, due in large measure to improved ribbon-pulling technology for sapphire.

Another candidate technology which offers still higher density in CMOS gate arrays is the ISO-CMOS Masterslice line available from International Microcircuits, Inc. (IMI). Using oxide-isolated silicon gate circuits, chips may be designed having up to 2000 gates, each having three N and three P channel devices. Gate delays of 5 nsec at 5V bias have been demonstrated. Quick turnaround and moderate cost make this product attractive for the PMF.

If implementation of the PMF design were to proceed at this time, the technology of choice would be ISO-CMOS, CMOS/SOS, or a mix of both.

## 4.2 GATE ARRAYS

Gate arrays offer the logic designer most of the advantages of Large Scale Integration (LSI) while avoiding the two major disadvantages--high cost, and lengthy design times. While a fully custom LSI design would offer greater density and somewhat better performance, for modest quantities gate arrays are clearly preferred when density and performance goals can be met.

The array itself is an integrated circuit which has been fabricated up to the final metallization layer(s) on which the logic interconnections are made. A conventional SSI designer typically supplies wirewrap interconnect information for blank wiring cards containing integrated circuit sockets and card-edge connectors. Similarly, a given Universal Array contains unconnected logic gates and in/out (I/O) pads in a predetermined geometry. The logic designer, usually with a Computer Aided Design (CAD) system, specifies the interconnections, which follow a fairly simple set of rules furnished by the array manufacturer. These interconnections are then used to produce the mask layout for the final metallization step of IC fabrication. Typically the cost is 30% of a full-custom design, and the time to delivery of the first part may be as little as three months versus a year or more for full custom.

Gate arrays are commercially available from various manufacturers in a variety of logic families. For reference, a partial tabulation follows. Several of the entries have been the subject of individual project memos.

1. ECL

   a) Motorola "Macrocell"; approximately 1000 gates, 60 I/O pins.

   b) Fairchild; approximately 1000 gates, 48 I/O pins.

   c) Plessey; (specs not known).

   d) International Computers Ltd. (U.K.); 400 gates.

   e) Siemens; 450 gates.

2. ISL

   a) Signetics; gate count unknown, 36 I/O pins.

3. IIL

   a) Exar Integrated Systems; 500 gates, 42 pins, 50 nsec gate delays.

4. NMOS

   a) Interdesign; 200 gates, 25 nsec gate delays.

5. LSTTL

   a) Interdesign; 200 gates, 10 nsec gate delays.

6. Standard CMOS

   a) Interdesign; 200 gates, 200 nsec gate delays.

   b) California Devices; 800 gates, 80 pins, 25 nsec gate delays.

7. High Speed CMOS

   a) RCA (CMOS/SOS); 630 gates, 62 I/O pins. Three smaller sizes also available.

   b) International Microcircuits, Inc. (Isolated CMOS); 1960 gates, 116 I/O pins. Five smaller sizes also available.

For a variety of reasons detailed in Section 4.1 high-speed CMOS was chosen for the PMF. Discussions were held with both RCA and IMI to help evaluate their products. The results of the comparative evaluation are shown in Table 3. The decision ultimately came down to the fact that the largest RCA array currently available provides only 64 I/O pads. In the PMF application, this is a serious limitation, and as a result, the first array design will use IMI's 1440 gate array, containing 100 pins, mounted in a 96-pin chip carrier (See Section &pkgsect).

Although IMI offers a mask layout service in which we need supply only timing constraints and TTL logic drawings, there are a number of advantages in doing the layout work ourselves. Although reduced cost and possibly reduced turnaround time are among them, the chief benefit suggested by IMI is the likelihood of better performance, since the logic designer is the person best able to evaluate the relative merits of alternate logic implementation methods. As we would be demanding performance closer to the limits of the device than other IMI customers have required, this could be an important benefit. The obvious disadvantage is the necessity for a Laboratory designer to climb the learning curve necessary to merge the Laboratory's modest CAD facility and IMI's design rules. Consequently, the first circuit would probably take longer than if IMI were to design it, but subsequent circuits would almost certainly be designed faster, and should have improved performance.

TABLE 3

HIGH-SPEED CMOS GATE-ARRAYS

RCA

64 I/O Pads (maximum)
Maximum gate count of 576
Maximum I/O Cells: 32
Maximum low-Z Cells: 8
Questionable Calma Compatibility
Simulation system not accessible from LL
More solid company, believable track record
In-house systems experience
East Coast location
Complete documentation
Complete CAD support
Appropriate test parts available
Higher cost
Slightly longer delivery time

IMI

116 I/O pads (maximum), including 100 & 84
Maximum gate count > 2500 (1960 cells)
Output buffers = (pads 4)/2
Output buffers provide adequate drive
Calma compatibility
Simulation available on commercial timesharing system
Short track record, but believed reliable
No systems experience
West Coast location
Sketchy Documentation
We must do CAD
Test parts not representative of present product
Lower Cost
Slightly shorter delivery time

### 4.2.1 Crossbar Gate Array

The crossbar array was chosen as the first design to implement because it is the simplest of the six gate array designs that have been identified for use in the PMF. The crossbar part also has a modest gate count, is highly repetitive, and is expected to present fewer timing stresses than the other required arrays. The functions to be carried out by the crossbar have been described in Section 3.4. In addition, the commutation and variable data delay functions described in Section 3.2 and 3.3 are to be physically contained in the crossbar circuits in order to implement them in LSI rather than in more space-consuming SSI.

The data path logic is shown symbolically in Figure 14. Temporarily ignoring the effects of the Variable Data Delay, it may be seen that the eight Data Inputs are latched each clock cycle in the lower right portion of the drawing. Each of eight columns selects ("3-bit address") one of these rows using an 8:1 multiplexer, and the multiplexer outputs are also latched. The latch outputs are brought to the IC pads through tri-state drivers, allowing expansion of the multiplexer by paralleling the corresponding outputs of two crossbar IC's. As will be described, the control logic for a given bit enables the tristate driver of only one IC depending on whether the source-row selected is 0-through-7 or 8-through-15.

Returning to the Variable Data Delay, Section 3.3 outlined the several operating modes of the four basic AE's. As described in Secton 3.4, this established the need for delaying the output of AE1 by zero or one clock pulse, AE2 by zero or two, and AE3 by zero, one, or three clock pulses. A three-bit Commutate Mode selects one of these options, which typically remains in effect for the duration of an FFT stage. The data path implementation merely requires multiplexers to select 'taps' off of shift registers (of length zero, one, two, and three) whose inputs are received from AE0 through AE3. Data delays of zero are called for where AE devices do not perform FFT-related functions.

Figure 15 shows the control path logic. The "3-bit multiplexer address" and "commutate mode" required by the data path logic described above are derived from parameters distributed by the control logic over the parameter bus. By double-buffering the holding registers in the crossbar IC, this distribution may take place during one stage of an FFT while the previous stage is being carried out. Then in only a single clock interval any or all multiplexers in the crossbar may be reconfigured by strobing the first rank of parameters into the second rank.

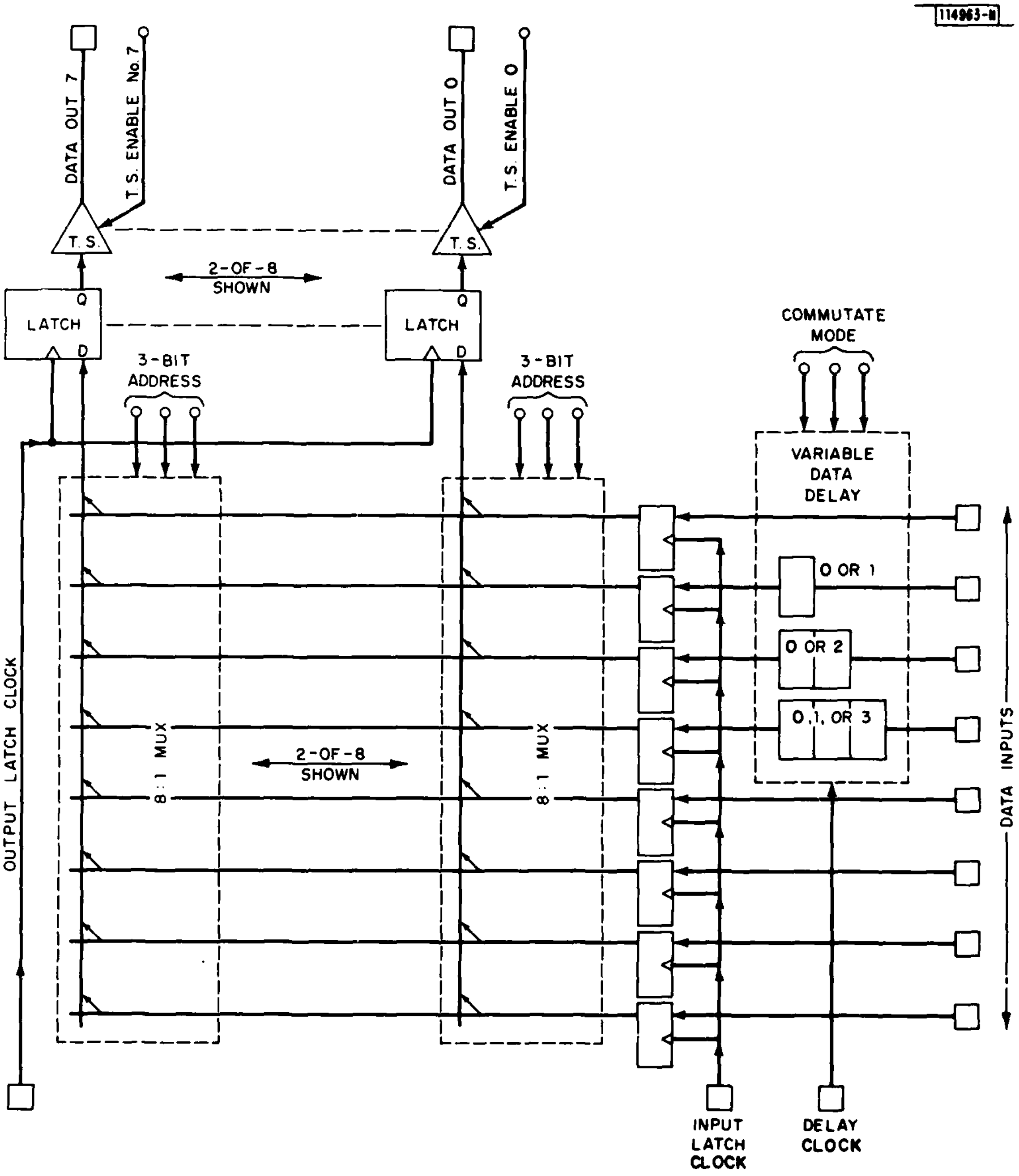


Fig. 14. Crossbar Data Path Logic

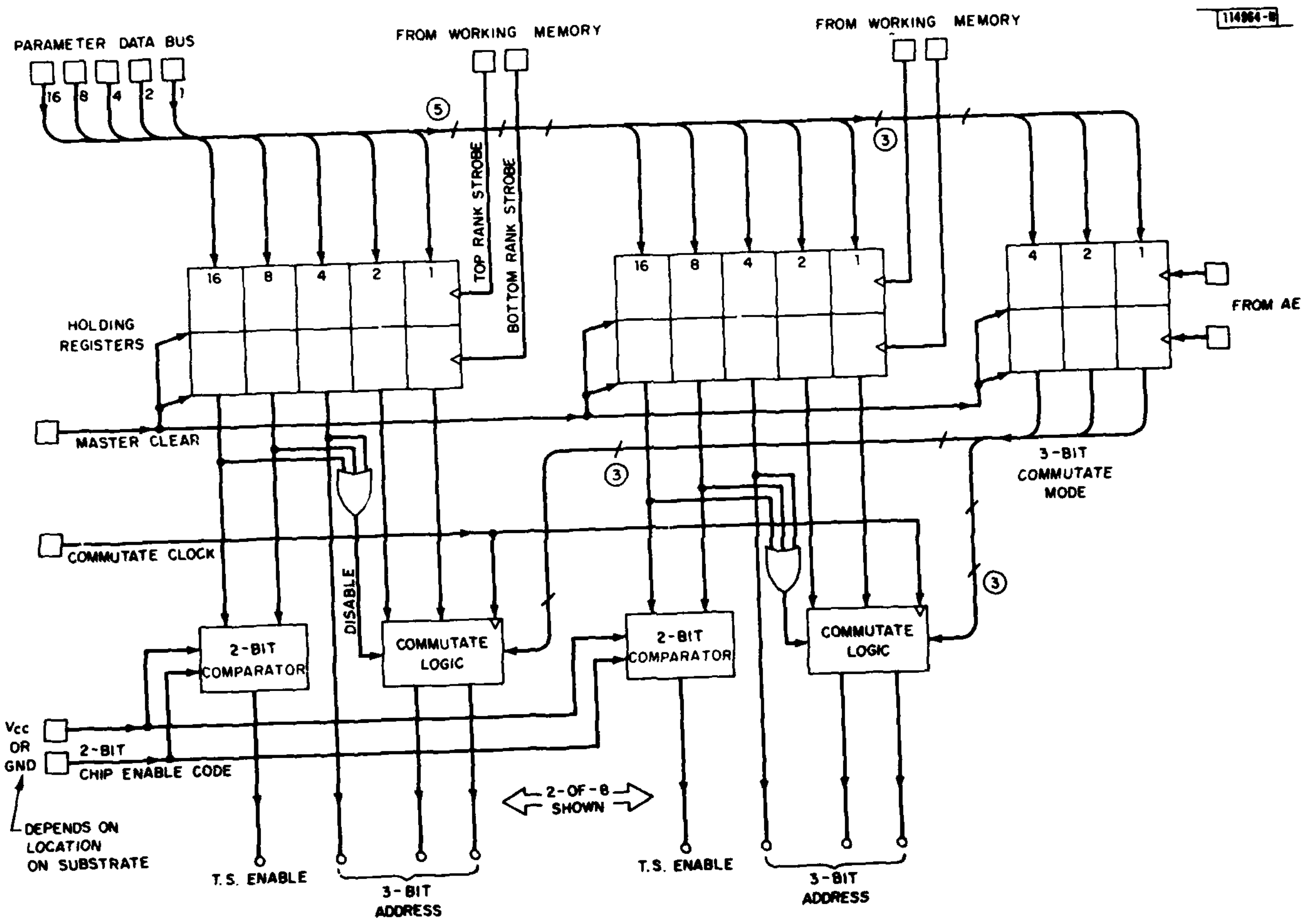


Fig. 15. Crossbar Control Path Logic

To effect a 32-to-1 multiplexer for the memory-to-AE-input paths, the outputs of four crossbar IC's are paralleled. For any one output bit, the tri-state output driver of only one IC is activated. This is accomplished in each IC by comparing the two most significant bits (MSB's) of the 5-bit parameter to a pair of levels providing a different "chip-select" code to each of the four IC's depending on where it is located on the printed mounting substrate. The three least significant bits (LSB's) of the parameter simply become the three-bit address input of the 8-to-1 multiplexer in each IC. (The commutate logic is unused in the memory-to-AE direction.)

In the AE-to-memory direction shown in Figure 14, the 16-to-1 multiplexer requirement implies two paralleled IC's. The "chip select" function therefore results from parameter bit "8". Bit "16" is, in a sense, a read/write bit, as it is used to disable the tri-state outputs from both IC's for those columns that correspond to working memories that are reading, rather than writing, data.

The remaining function implemented inside the crossbar IC is the commutation of data emerging from the AE's during Radix-4 or Radix-2 operations. The paths possible at any instant are shown in Figure 16. Note that only the two LSB's of the multiplexer addresses need to change when the commutate clock is pulsed. The resultant address is dependent upon a 3-bit commutate mode parameter (which remains unchanged throughout an FFT stage) and the two LSB's (of the five address parameters) which specified the AE output initially selected by that working memory. In Figure 15 the three high order bits also enter the commutate logic to disable commutation when the selected AE output number is greater than 3, and therefore not the CE.

Table 4 summarizes the signals required for the Crossbar IC. Control functions, including power, require 32 pins. The gate array mask would be designed and manufactured to handle a four-bit slice of the cross-bar, requiring a total of 96 pins. If a second source for 96-pin chip carriers is available, or if there is sufficient confidence in the maturity of the technique, the chips will be packaged in 96-pin carriers, taking advantage of its full potential. If for any reason 84- or 86-pin carriers must be chosen, not all of the chip pads will be bonded to the carrier, and the resultant part will implement a 3-bit slice of the crossbar, necessitating more cards in the full crossbar assembly.

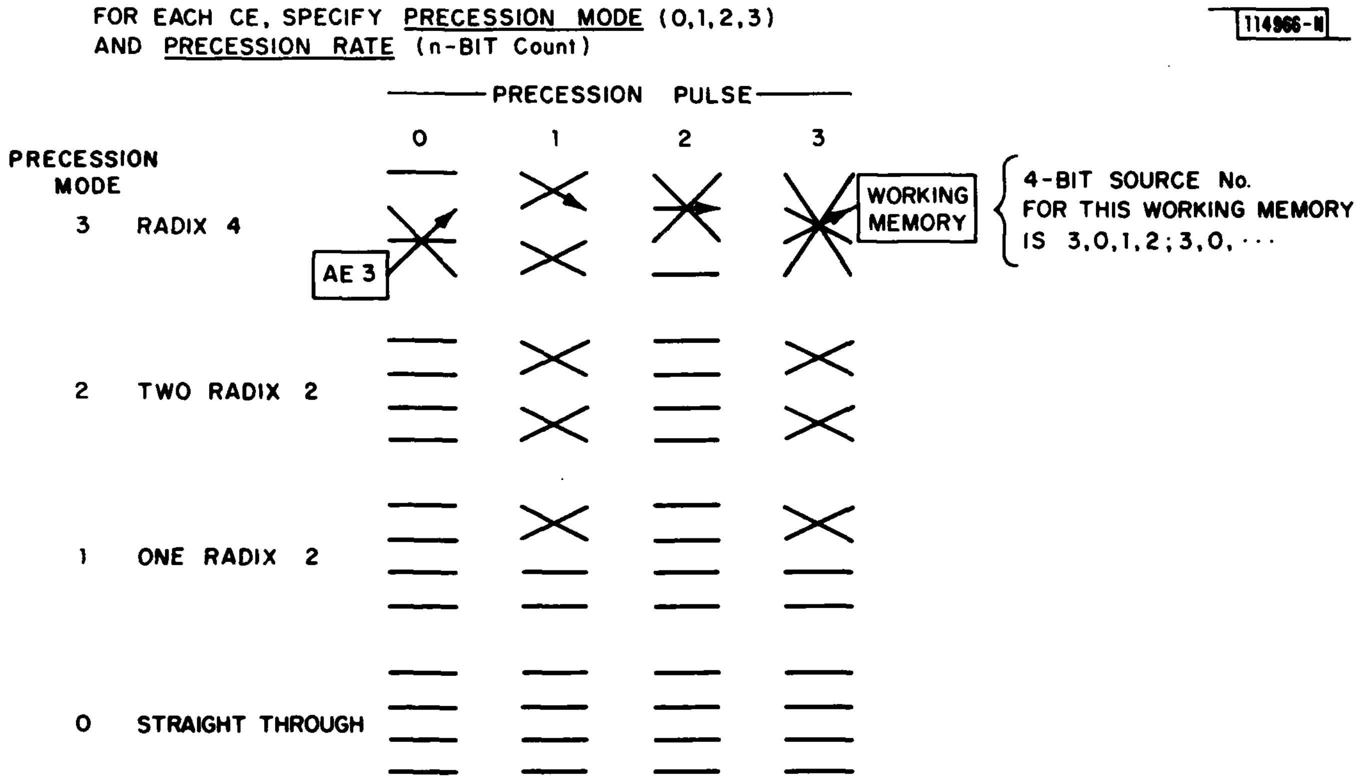


Fig. 16. Crossbar Commutation Paths

## TABLE 4

## PIN COUNT: CROSSBAR CIRCUIT

| | |
|---|---|
| 2 | Power |
| 16 | Holding Register Strobes From Working Memories |
| 2 | Holding Register Strobes From AE |
| 5 | Parameter Data |
| 1 | Input Latch Clock |
| 1 | Output Latch Clock |
| 1 | Delay Flip Flop Clock |
| 1 | Commutate Clock |
| 2 | Chip Enable Code |
| 1 | Master Clear |
| 32 | Control Pins, Subtotal |

IF CONNECTED AS 3-BITS DEEP PART

| | |
|---|---|
| 32 | Control Bits, Subtotal |
| 24 | Data In |
| 24 | Data Out |
| 80 | TOTAL, For 3-Bit Deep Design |

OR

If 4 Bits Deep

| | |
|---|---|
| 32 | Control Pins, Subtotal |
| 32 | Data In |
| 32 | Data Out |
| 96 | TOTAL, for 4-Bit Deep Design |

### 4.2.2 Working Memory Controller

#### 4.2.2.1 Functional Description

The prime function of the controller for the working memory is the structured generation of addresses. The secondary housekeeping functions, such as error reporting, synchronization, etc., have been described briefly in the architecture section.

The addresses are generated by a nested set of programmable loops. A nesting depth of 4 is required by the baseline processing task. Each level has several registers associated with it as well as a status bit indicating whether the level wants service. The lowest level (level 0) has the highest priority. Priority decreases with increasing level number.

The following PASCAL-like procedure illustrates the logical steps used in arbitrating the service requests and performing the address generation. Readers unfamiliar with PASCAL may refer back to Section 3.2 and Figure 3 for an alternative description. The procedure 'next_address' produces one address each time it is activated and simulates the hardware operation at each cycle. The function 'priority_encode' returns the number of the highest priority level requesting service (as designated in the variable 'status_bits'). The 'level' variable is used to index the 'current_count', 'terminal_count', and 'increment' arrays. The procedure 'bit_clear' clears the service request flag for the current level if the count has expired, and the loop containing the procedure 'bit_set' sets all the service requests for lower level (higher priority) loops.

```
PROCEDURE next_address;
BEGIN
level := priority_encode(status_bits);
address := address + increment(level);
current_count(level) := current_count(level) + 1;
IF current_count(level) = terminal_count(level)
        THEN bit_clear(level,status_bits);
FOR lower := level - 1 DOWNTO 0 DO bit_set(lower,status_bits);
END.
```

#### 4.2.2.2 I/O Pin Count

One of the limitations on LSI devices in this application is the number of I/O pins. The partitioning of the controller task depends on the number of pins needed. A list of the necessary pin connections is shown in Table 5. A total of 71 pins are required. This number of pins is available in the larger gate arrays.

TABLE 5

PIN COUNT: WORKING MEMORY CONTROLLER

| | |
|---|---|
| 2 | Power |
| 16 | Parameter bus - data |
| 16 | Parameter bus - address[1] |
| 16 | Working memory address[2] |
| 2 | EDAC error flags |
| 8 | Command Lines[3] including enable input |
| 8 | Event lines[3] including error output |
| 2 | Latch enables for crossbar switch |
| 1 | Clock input |
| ---- | |
| 71 | Pins Total |

[1] Can be reduced through precoding
[2] Only 12 needed for baseline application
[3] Can be reduced through multiplexing

#### 4.2.2.3 Gate Count

An approximate gate count is the second determinant for partitioning the controller function. Registers use the largest amount of space at about 3 gates per latch. The following registers are required:

1. One 16 bit register (master and slave latches) for the current address plus a holding register (a latch) using a total of 144 cells.

2. 12 double buffered 16 bit latches for the current and terminal counts and increments for 4 levels using a total of 1152 cells.

3. A 4 bit double buffered latch for the data source address using 24 cells.

4. A 4 bit memory group latch using 12 cells.

5. Two 4 bit double buffered latches for the command line and event line specifiers using 48 cells.

6. An 8 bit counter (master and slave latches) for the synchronizing delay using 48 cells.

7. A 16 bit double buffered miscellaneous configuration register using 96 cells.

Thus, a total of 1524 cells are needed just for the registers. This is nearly 80% of the largest gate array (1920 cells) and clearly indicates that the function must be partitioned into at least two semicustom gate arrays.

It would be desirable to keep the address generator sections together on one of the IC's, and, failing that, to separate the increment and current address registers from the loop control registers. The second alternative is less desirable since the information on which level is being serviced must be transmitted from one section to the other in much less than a clock cycle. Partitioning the address generator would increase the delay.

The latches for the address generator take up 1296 cells (68% of a 1920 cell array). It is, therefore, reasonable to expect that the entire address generator circuit could be contained in a large array although it would not require as many I/O pins as are available in such a device.

### 4.2.3 AE Gate Arrays

The parts of the PMF most sensitive to changes in the processing tasks are the AE's. For this reason less attention has been paid to refining their hardware implementation. Nonetheless, certain gate array requirements appear to be obvious from even a casual examination. While multipliers are commercially available, there are three distinct functions that must be performed in the basic AE's which are not yet available in LSI technology: (1) a wide (probably an expandable 8 or 16 bit) arithmetic logic unit (ALU) to perform the additions and subtractions required as well as thresholding and bypass operations; (2) an align/normalize function for floating point addition; (3) an error checking function to perform the same operations in residue arithmetic (modulo-n). The reciprocal estimator can also use the first two of these functions.

## 4.3 WORKING MEMORIES

The working memories constitute almost one-half of the total parts count in the PMF. As such, the selection of the memory chip used in the working memories is the largest single determinant of the size, weight, and power of a PMF module. The desired characteristics of the memory chip are: (1) high speed - less than 60 nsec access time; (2) low power dissipation; (3) high density; (4) static memory cell to simplify control; (5) maturity. The required 60 nsec access time quickly eliminated all but a few technologies (ECL, TTL, CMOS) and also limited the level of integration that could be obtained to four kilobit (4K bit) products. Note that while 16K bit static RAM's have recently been introduced, they are not mature enough to be considered for this design. They do, however, provide an easy means for later expansion. A representative, though not exhaustive, listing of such components is shown in Table 6. The remaining qualifier of power dissipation quickly leads one to the Hitachi CMOS component (HM6147LP-3). As shown in Table 6, this component dissipates less than one-tenth as much power as any of the other candidate components.

The Hitachi part is not without drawbacks, however. For example, Hitachi does not currently offer the part in a chip carrier package, which increases the volume of the working memories. In addition, like all CMOS components, its drive capability is quite limited. The solution to this problem is to add high-capacity TTL drivers to the working memory cards. This is described in Section &pkgsect on packaging considerations.

TABLE 6

CANDIDATE WORKING MEMORY PARTS

| Part Name | Size | Access Time(nsec) | Power(mW) Act/Stnby | Type |
|---|---|---|---|---|
| AMI 2147-3 | 4Kx1 | 55 | 900/150 | VMOS |
| Fairchild 93474 | 1Kx4 | 45 | 750 | TTL |
| Intel 2147H-2 | 4Kx1 | 45 | 900/150 | HMOS |
| Hitachi HM6147LP-3 | 4Kx1 | 55 | 75/0.01 | CMOS |
| Fujitsu MB7067 | 1Kx4 | 35 | 500/300 | TTL |

## 4.4 CHIP CARRIERS AND CERAMIC SUBSTRATES

Industry's solution to the problem of high density IC packaging is the Leadless Hermetic Package (LHP), more commonly known as the Ceramic Chip Carrier (Refs 1-6). Its benefits are summarized in Table 7. The principal advantage from the standpoint of compact military electronic programs is a 5:1 reduction in IC area as compared to the conventional Dual In-Line Package (DIP). As shown in Table 7, the area of a 64-pin chip carrier is slightly less than one-half square inch. In addition, for CMOS LSI, chip carriers offer higher lead counts (presently 96, but JEDEC standards exist up to 156 leads) and reduced lead capacitance. The PMF prototype would use carriers no larger than 96 pins, partly because that is the size of the largest test-socket currently available.

One of the factors resulting in the high density achieved with chip carriers is the close spacing of the leads at the edge of the carrier. Lead spacing is typically 40 or 50 mils, but may

TABLE 7

CERAMIC CHIP CARRIERS

(Leadless Hermetic Packages)

Relative Density for 64 Pin IC

| | LEAD SPACING | FOOTPRINT | AREA SQ.IN. | RATIO |
|---|---|---|---|---|
| DIP | 0.1 | 3/4x3 | 2.25 | 5.11 |
| FLATPACK | 0.05 | 1x1 | 1 | 2.27 |
| CARRIER | 0.04 | 2/3x2/3 | 0.44 | 1 |

More Leads Possible: 64, 68, 84, 96 Pins Available

Several Attachment Options, Including Sockets (unlike hybrids)

Nearly Equal Lead Lengths From Chip

Reduced Pin Capacitance - 2 pF. versus 5 pF. for DIP

Good Dissipation Characteristics

be as close as 30 mils. Maintaining the required mounting tolerances, coupled with the necessity to reduce stray lead capacitance in CMOS systems, has caused industry to reject conventional glass-epoxy printed circuit cards in favor of ceramic substrates. Some of the features of ceramic substrates are listed in Table 8.

Multilayer ceramic substrates, by 3M Co. and others, have been used for years by suppliers of hybrid circuits. As 3M's 'Guidelines for Designing Multilayer Ceramic Substrates' describes, the various ceramic layers are silk screened with the conductor path layout including vias between layers, and the

TABLE 8

CERAMIC SUBSTRATES

- Being Used Extensively for SOS
- High Density Wiring
    - 4 mil Traces on 8 mil Centers
    - Lower Capacitance
    - Small Via Holes Between Layers
    - Shorter Conductor Paths
- Sophisticated CAD System Required. $8 - 15K per Design
- Silk Screening Limits Size to About 15 cm. x 20 cm.
- Production Cost   $500 Each
- Good Thermal Characteristics
    - Coefficient of Expansion Same as Carriers
        - - Permits Reflow Soldering
- Substrate Tested Completely in Pogo-Pin Tester w/o Carriers
- Conductive Silicone Allows Layout Test Before Bonding Carriers

whole structure is sintered to form a monolithic unit. Conductor widths of 4-mils are standard, which, together with the small diameter vias required, (compared to conventional PC boards) results in extremely high wiring densities. The shorter paths, along with the favorable characteristics of the dielectric, result in low capacitances well matched to the needs of the high speed CMOS circuits used.

Although 3M lists 12.5 cm. x 15 cm. as the largest size commonly sold, RCA's Missile and Surface Radar Division routinely makes larger sizes and feels that 15 cm. x 20 cm. is within the limits of the silk screen process. The bonding of the chip carriers to their substrates is done by depositing solder on the carrier pads, positioning the carriers on the substrates, and then, frequently by means of infrared heating, reflow-soldering the carriers in place. Precise manual placement of the IC's is not necessary because surface tension of the solder realigns each carrier on its substrate pads during reflow-soldering. Cards have even been made with IC carriers on both sides of a substrate. Again, surface tension holds the inverted IC's in place as the substrate is heated to mount the upper IC's.

Systems employing CMOS/SOS on ceramic substrates have been packaged in card cages containing a backplane assembly into which standard Navy ISEM cards are inserted. The card-edge connectors are soldered to pads on one edge of each ISEM substrate, allow 20 connections to the inch, and permit card spacings in the cage of 5-per-inch. As the following section explains, such card-cage packaging is inappropriate for the PMF, but individual portions (e.g., AE's and working memories as subsystems) will use similar techniques. The unique problems of crossbar edge connection are also described in Section &pkgsect..

## 4.5 OVERALL PACKAGING CONSIDERATIONS

As suggested in Section 3.4, the physical implementation of the PMF closely follows the conceptual logic architecture. Focusing on the crossbar as the first part to build, the functional specifications of the IC's and substrates have resulted from considerations of physical manufacturing limitations, numbers of connector pins per substrate card, IC spacing on the cards, word length, maximum numbers of working memories and AE-devices required, and methods of effecting control. Figure 17 shows a mockup demonstrating the packaging concepts, and suggests the size of the finished assembly. The principal features are enumerated below.

1. Fourteen identical crossbar cards, each approximately eight inches square and spaced on half inch centers, will make the crossbar assembly roughly an 8-inch cube. Each card will contain a 4-bit slice of the crossbar, allowing a word length up to 56 bits across the front-to-back dimension.

2. Above the crossbar will be a fairly conventional card cage providing vertical guides for 16 cards, each containing two 4K word by 55 bit Working Memories. Card spacing will

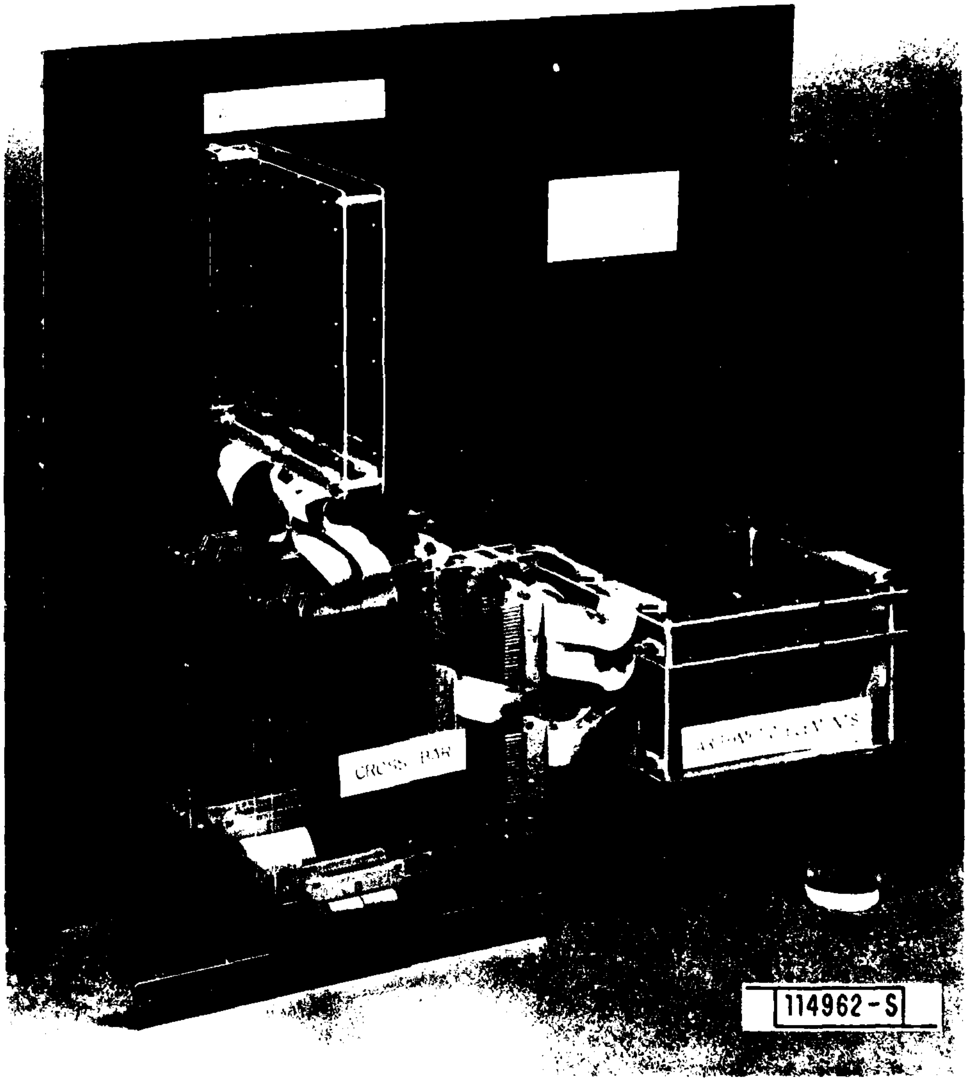


Fig. 17. PMF Mockup

again be 1/2 inch, and the bottom edge of each card will require 8 inches of length to connect to the crossbar. If the 110 memory IC's were available in chip-carriers rather than DIP's, the vertical card dimension would be less than 8 inches; with DIP's it will probably be greater than 10.

3. Between the backplane (bottom) of the memory cage and the top of the crossbar will be a rigid frame of ribbon cables constructed in such a way that the transition from the word-oriented memory cards to the bit-slice oriented crossbar cards is accomplished. Figure 11 illustrates this in simplified fashion, and Section &corncsect describes it in detail.

4. The right surface of the crossbar cube will similarly be a frame of "corner turning cables" terminating in the backplane of a second card cage containing horizontal slots for 16 AE cards. These too will be on 1/2 inch centers, and provide 8 inches of crossbar connectors along the left edge. Although Figure 17 shows square AE cards, some (e.g., interface to external bulk memory) would require less than 8 inch length, while the multipliers and adders that comprise the R4-CE would certainly require more area, and hence be longer than 8 inches.

5. Finally, cards required for control would be identical in size to the crossbar cards and mounted parallel to them -- in front and/or in back. As a result, the front-to-back dimension of the crossbar frame will be increased as required, and the corresponding dimension of memory and AE cards will probably become about 10 inches rather than the 8 used for illustrative purposes above. Connections from Control to AE's and Memories are done in the same way the crossbar communicates with them. Signals controlling the crossbar tend to require connections to all crossbar cards. For control, ribbon cables will daisy-chain across the bottom (or left edge) of the crossbar, bussing together all crossbar cards. Drive current to overcome the high capacitance of the resultant load is provided by high speed non-CMOS drivers on the control cards.

### 4.5.1 Working Memory Cards

The RAM circuit DIP's used in the working memories dictate a memory card size larger than ceramic substrate technology presently permits, but since the memory IC's themselves do not require low capacitance interconnections, this is acceptable. Current plans are to mount the RAM IC's on conventional glass epoxy PC cards and to provide space near the bottom edge of each card for a small substrate containing the CMOS LSI linear address generator circuits and, if required, EDAC circuits. Connections from these circuits to the crossbar and control cards will be accomplished using the corner turning cables of Figure 11. Memory cards will slide downward in their frame, and their edge-card connectors will engage fixed connectors at the top of the crossbar assembly. The Parameter Bus and control signals could be distributed along the same card edge, although a ribbon cable daisy-chained across the top edges of all memory cards would reduce the number of connections required at the crossbar.

### 4.5.2 "Corner Turning" Cables

The photograph illustrating the concept (Figure 11) shows a pair of 2 inch wide 40 conductor ribbons split in half at each end to join pairs of connectors mounted on one inch centers but rotated at 90 degrees from each other. In actual practice a card spacing of 1/2 inch will cause the cables to be split into quarters (10 conductors, rather than 20), thus providing the required paths for 8 bits and two control signals or grounds. The eight bits, at the crossbar top, are four bits each for the two independent memories on each working memory card. At the side of the crossbar, the 8 bits are four each for the two inputs of a multiplier, or four in and four out of a simple AE device. In the case of the R4-CE, the outputs of two adders would be paired on one AE board edge connector, and the eight bits would therefore contain four from each AE output.

### 4.5.3 AE Cards

Ceramic substrates will be required for the AE cards, which will make extensive use of high speed CMOS. As depicted in Figure 18, signal flow will be out of the crossbar into the eight inputs of four identical complex multiplier cards and through to their rightmost edges. Ribbon cables would then fold these intermediate stage outputs down to a single adder and butterfly-logic card whose signals would flow from right to left back into the two connectors that provide four crossbar inputs. The idea of building an assembly of two interconnected parallel cards (mating with the required two slots) for the adder/butterfly logic should be explored.

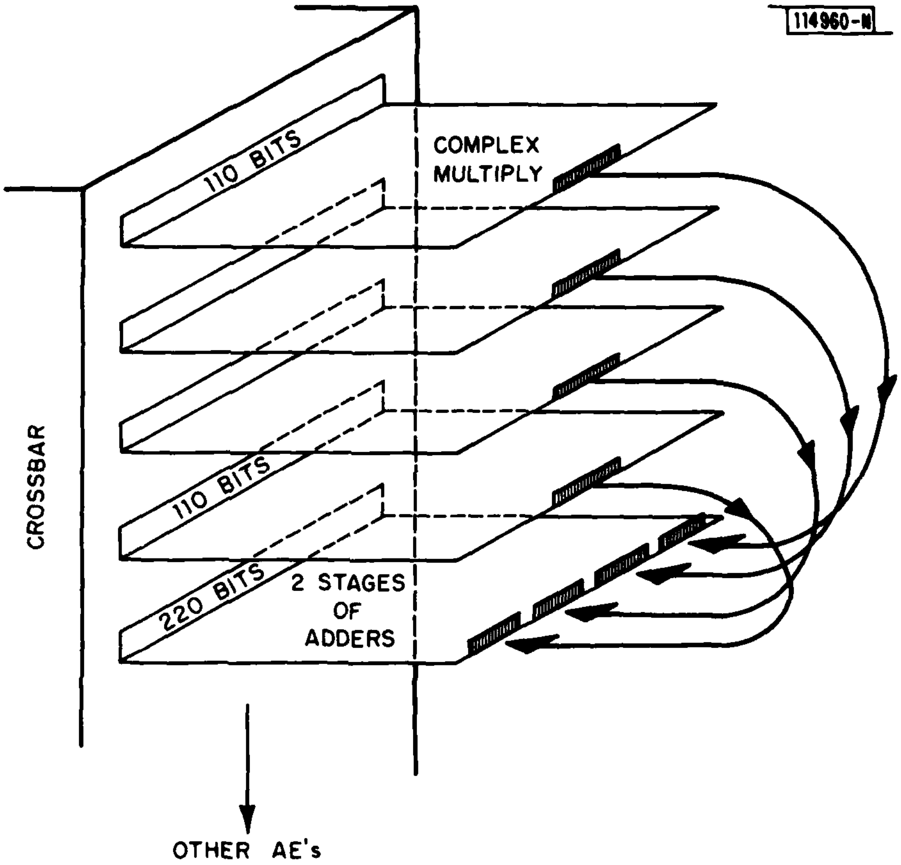


Fig. 18. AE Cards

Although the multiplier or adder card areas required may exceed the largest substrates presently obtainable, the logic can be partitioned in such a way that two or more substrates may be assembled end to end in a rigid frame; effectively doubling card area. The mounting of carriers on both sides of the substrates should also be evaluated as a possible solution of the real estate problem.

Because of the 2-in/1-out nature of the AE's in the CE, an alternative configuration using the unused rows 6 and 7 of the crossbar may be considered. The outputs of the multiplier cards (which may be commanded not to multiply) could be cabled not only to the adder/butterfly card in slots 4 and 5 but also daisy-chained to cards in slots in 6 and 7. This could provide opportunity to implement four channels of additional arithmetic or logical functions (e.g., peak picking) while removing that complication from the adder card design. Alternately, the space on the two additional cards might be used in implementing the butterfly logic if the card-to-card signal problem could be solved.

### 4.5.4 Crossbar Cards

The crossbar, consisting entirely of CMOS circuits, will clearly require ceramic substrates. The area necessary for the IC's themselves will be within the maximum substrate size available, but the top and right edges of each substrate must also provide approximately 160 connections to adjacent assemblies. The present manufacturing size limitation on substrates results from stability tolerances in the silkscreening process. As the tolerance requirements for card edge connector pads are extremely coarse, the feasibility of screening a larger substrate (e.g., 9"x9") with an "active logic" area smaller than 6"x8" should be investigated. See Section 4.4.

The other packaging consideration that complicates the crossbar design is card insertion. The proposed solution to insertion and removal of the crossbar substrates requiring interconnections at two adjacent edges is the use of Zero-Insertion-Force connectors (ZIF). AMP Inc., among others, offers a line of ZIF connectors that are open at one end, allowing cards to be inserted along the axis of the connector, and having a cam operated mechanism for applying contact pressure after the card has been seated. The use of these connectors in a rigid assembly containing the ribbon cables and the connectors for AE and memory cards permits a packaging design using today's technology. The 20-to-the-inch ribbon cables and connectors that the AE and Memory cards use have been industry standards for a decade. If future cable and connector density improvements are made, the generous half inch card pitch could be reduced if capacitance and card real estate requirements could still be met.

### 4.5.5 Control Signal Distribution

Both the parameter bus and the many timing signals required to control and synchronize the various subsystems require special attention. In general, both categories will have heavy capacitive loading caused by the cables used and the presence of multiple loads. Conductor length delays, even in such a small machine, must also be considered. The voltage swings will be large, because the CMOS arrays will have to be powered near their maximum voltages in order to attain the 17 MHz pipeline clock speeds required.

The proposed solution in the case of bus circuitry, which requires considerable switching current but moderate edge speeds and good delay repeatability through the drivers, is to use open collector schottky buffers (74S32) with pullup resistors strategically placed throughout the various loads. These devices, terminated properly, will provide both the speed and the voltage swings required.

Timing signals, while generally less heavily loaded than bus signals may be, have the additional requirement of precise switching times, also implying fast edge speeds. Where the 74S32 is inadequate to the task, one of several commercial pulse drivers will be selected. Present candidates include E-H International's Model 10900 100 MHz 10V driver, several TTL-to MOS drivers (Intel No. 3245, TI No. SN75365), and current boosters and operational amplifiers made by Optical Electronics, Inc. (Models 9963, 9740, and 9909). There is also the possibility that the growing community of high speed CMOS users will solve this recognized problem within the time span of the project.

When necessary, controlled impedance lines can be used, and compensation for lead length variations throughout the system will permit "fine tuning" the pipeline in the same way that ECL designs are adjusted.

# Chapter 5

# SOFTWARE CONSIDERATIONS

## 5.1 INTRODUCTION

The PMF is a special purpose processor designed to optimize the handling of signal processing algorithms. As such, it is not suitable for the task of program development. Consequently, a host computer facility is envisioned to support the development of PMF software.

Eventually, control of the PMF will reside in a small, fast, dedicated CPU which will be integral to the PMF. The main function of this local control computer will be the execution of the user's application program. In essence, control will perform this task by the dissemination of operating parameters among the working memories, AE's and external interfaces.

For the prototype version of the PMF, however, the control function will be assigned to a commercial minicomputer which will operate in concert with the PMF proper but from an outboard position. Thus, the host computer will also serve as the control component of the PMF configuration. This close coupling requirement of the host machine with the PMF effectively eliminates the Laboratory central computer facility as the host machine, although it could be used to simulate the PMF.

## 5.2 SOFTWARE TASKS

A preliminary set of programming tasks which have been identified for the mini-computer controller are listed below:

1. PMF Macro Assembler: The job of the PMF assembler is to translate a program written with PMF mnemonics and system-provided and/or user-defined macros that constitute a PMF source program, into an object module that can be recalled at any time for execution on the PMF. (See Section &alprogsect for an example of a PMF program.)

2. Librarian for System Macros: As PMF signal processing algorithms are written and checked out, a librarian program will build and maintain library modules that will be available to the programmer in designing new algorithms.

3. Loader Facility: Initially, PMF object programs will be memory modules which will permit simple one-stage loading by the loader program. Eventually, the loader may be required to link several object programs into a single runnable PMF module.

4. 'Host' Control Program: The concept of a control program is fundamental to the design of the initial version of the PMF. The control program can be thought of as common to all PMF application programs and as such will co-reside in the host computer memory with the object module produced by the assembler. The control program will interpret the object program pseudo-ops and parameters and through a Direct Memory Access interface will drive the various PMF components and external interfaces. In so doing, the control program will issue synchronizing signals, field interrupts, wait on event flags, etc. Additionally, the control program will interact with the user by reporting errors, providing status information and responding to commands.

5. PMF Diagnostic System: A comprehensive hardware diagnostic system will be developed for the PMF. There will be two modes of operation. The first mode is that of global diagnosis which will automatically test all of the PMF software modules. The second mode will allow for user-specified tests to be conducted only on certain modules. Anomalies will be reported in a user-prescribed fashion. The system will be designed to run as a background task so that interactive program development by several simultane-

ous users can proceed concurrently with the hardware testing.

6. Monitor/Debug System: As an aid to program and hardware development and checkout, a monitor/debug facility will be provided for the PMF. This facility will act in conjunction with the host control program through 'debug packet' inserts in the source code, to enable the program under test to stop at selected breakpoints, to single-step the PMF, to display values, and in general to interact with the programmer in the checkout process.

7. Signal Processing Algorithm Development: There is an obvious need to produce a repertoire of signal processing algorithms for the PMF. An extensive programming effort will be required to provide a comprehensive collection of commonly used programs such as Fast Fourier Transforms (FFT's). The intent will be to make these routines easily available (through the PMF Librarian) as building blocks with which complete signal processing systems may be constructed.

## 5.3 SIMULATION

A design goal of the PMF is to make the coding of signal processing algorithms on the PMF relatively straightforward. Working memory allocation and addressing are the principal items requiring attention in designing an algorithm. One set of simulation routines, implemented on the Laboratory's central computer, allows the algorithm designer to verify the logical consistency of PMF code as regards the shuffling of data between the working memories and the AE's for different algorithms.

Another set of simulation routines will permit running an actual data set through the simulated PMF. The results of this simulation can then be compared with the same data set being processed by the same algorithm implemented in a conventional fashion.

# Chapter 6

## SUMMARY

The PMF design study has shown that impressive performance capabilities are achievable with today's technology. While the architectural features of the machine provided a throughput rate far beyond today's supercomputers, careful attention to packaging considerations has yielded a very compact design which would be suitable for applications where size, weight, and power are limited. The principal features of the PMF design are summarized in Table 9.

## TABLE 9

## SUMMARY OF PMF FEATURES

| | |
|---|---|
| Throughput Rate<br>(Radix-4 FFT) | 567x$10^6$ Real Ops/Sec |
| No. IC's | 5206 |
| Power Dissipation | 738 W |
| Volume | 0.2 $m^3$ |
| Weight | 50 kg. |

Word length: 55 bits - 48 bit complex floating
point + 7 bits error detection.

Clock rate: 16.7 MHz

Working memories: 32 memories, each 4K words of 55 bits
with local address generation

Arithmetic elements: 4 primary AE's, each with
1 complex multiplier and 2 complex adders;
4 secondary AE's of various capabilities.

Crossbar in two sections:
One section with 32 inputs, 16 outputs
One section with 16 inputs, 32 outputs.

Control: double buffered parameter registers.

## Appendix A

## PMF RELIABILITY

This appendix summarizes the calculations used to predict the mean-time-between-failure (MTBF) for the PMF. The detailed calculations are contained in a 1979 memo by A. H. Huntoon.

For the most part, the calculations utilize component failure rates tabulated in MIL-HDBK-217C. In addition, a simple, series-connected reliability model is assumed. This results in a pessimistic estimate of the MTBF because it implies, for example, that a single failure in any of the 32 working memories would count as a system failure. In practice, a PMF module would continue to operate even if several working memories failed. Similar comments apply to the basic arithmetic elements.

Table 10 contains the failure rates for the individual PMF subsystems. By summing these rates, one obtains an MTBF estimate of over 1500 hours when error correction and control (ECC) is used on all of the memory modules.

TABLE 10

FAILURE RATES WITH ERROR CORRECTION

| Hardware Category | Working Mem's | Crossbar | AE's | Power Sup., Fans, etc. | Sub-total |
|---|---|---|---|---|---|
| Cable/PCB Connectors | 7.66 | 8.37 | 3.05 | 1.44 | 20.52 |
| IC Pins | 48.71 | 16.85 | 13.64 | 7.39 | 86.59 |
| Capacitors | 1.05 | 0.22 | 0.2 | 0.23 | 1.70 |
| MSI ICs | 137.4 | 28.15 | 65.77 | 63.2 | 294.52 |
| VLSI ICs | 70.75 | --- | 84.33 | --- | 155.08 |
| CMOS RAMs | 1.08 | --- | --- | --- | 1.08 |
| Misc. | --- | --- | --- | 99.87 | 99.87 |
| Sub-total | 266.65 | 53.59 | 166.99 | 172.13 | 659.36 |

Total Failures per Million Hours 659.36

MTBF = 1517 Hours

# Appendix B

# SAMPLE PROGRAM

## B.1 INTRODUCTION

This appendix contains samples of two levels of coding for the PMF. The first demonstrates the use of high-level PMF macros, and the second shows the assembly language equivalent. The sample task is the execution of a radix-4 FFT, 'constant geometry' algorithm on a 256 point data set. (See Ref. 7.)

It is assumed in these examples that before the FFT code is executed, previous sections of the overall program will have (1) set up the source (input) working memories with the digit-reversed data set and (2) set up the working memories with the appropriate coefficient values.

## B.2 ASSEMBLER MNEMONICS

The assembler mnemonics used in the two implementations of the algorithm are listed below.

| | |
|---|---|
| ALGOR | Algorithm to be used, e.g. Constant Geometry or Normally Ordered |
| CEx | Computation element x<br>SOURCEA A Input for an AE<br>SOURCEB B input for an AE |
| COEFF | Working memories to hold coefficient data |
| DIR | Direction of Transform: FORWARD=Forward INVERSE=inverse |
| DRI | Digit reversed input versus NOI (Normally ordered input) |
| GPx | Working memory group 'x' (see subfields for WMx) |
| GROUP | Collect the specified working memories into a group |
| PATTERN | Pattern to be used in the commutation scheme |
| PIPEDELAYx | A predefined constant equal to the delay before data reaches the element: |

| | | |
|---|---|---|
| | x=1 | for delay from working memory output to AE input |
| | x=2 | for total delay from working memory output, through AE, to working memory input |
| POINTS | Number of complex data points in the input data set | |
| SIGNAL | Activate new parameter set | |
| SINK | Working memories to hold stage 1 output | |
| WAIT EVENT | Control mechanism for synchronizing events | |
| WMx | Working memory 'x' | |
| | SOURCE | AE number from which to obtain data to store<br>if SOURCE=OUT then working memory is to be a data source |
| | SA | Starting address in the working memory |
| | Ix | 'x' level increment in the addressing scheme |
| | TCx | 'x' level terminal count in the addressing scheme |
| | SYNCH | Synchronizing signal |
| | DELAY | The number of clock cycles after the synch signal to start |
| | EVENT | The name of the event line to be asserted at the end of a working memory block transfer |
| | * | This is a comment line |

## B.3 MACRO INVOCATION

```
RADIX4      POINTS=256;ALGOR=CONSTANT GEOMETRY;INPUT=DRI;
            SOURCE=1,2,3,4;SINK=8,9,10,11;COEFF=5,6,7;DIR=FORWARD.
```

The two lines above illustrate how a user would invoke the macro RADIX4 by supplying the appropriate key word parameters. Most signal processing applications can be handled entirely by such macros contained in a macro library. When this is not the case, a PMF assembly language routine must be written which could eventually be added to the library. The next section includes a sample program to illustrate the assembly language structure of the PMF.

## B.4 ASSEMBLY LANGUAGE PROGRAM

This section contains a sample PMF assembly language program designed to execute the radix-4, FFT algorithm. The detailed derivation of various increments and terminal counts is contained in a 1979 memo by J. D. Kurtze.

```
* ASSEMBLER LEVEL CODE FOR PMF IMPLEMENTATION OF RADIX 4 FFT
* CONSTANT GEOMETRY - 256 POINTS
* FIRST, GROUP THE WORKING MEMORIES INTO THREE GROUPS
* THE SOURCE OF THE DATA FOR THE FIRST STAGE
  GROUP, 1, WM1, WM2, WM3, WM4
* THE DESTINATION OF THE FIRST STAGE RESULTS
  GROUP, 2, WM5, WM6, WM7, WM8
* THE COEFFICIENT MEMORIES
  GROUP, 3, WM9, WM10, WM11
* PARAMETERS FOR STAGE 1
* FIRST, ISSUE ORDER TO 'SOURCE' WORKING MEMORIES (WM)
  GP1, SOURCE=OUT, SA=0, I0=1, TC0=63, SYNCH=BEGIN_STAGE_1,
           DELAY=0
*
* NOW, ORDERS TO STAGE 1 'SINK' WORKING MEMORIES
  GP2, SA=0, I0=16, TC0=3, I1=-47, TC1=15, SYNCH=BEGIN_STAGE_1
  WM5, SOURCE=AE0, DELAY=PIPEDELAY2, EVENT=END_STAGE_1
  WM6, SOURCE=AE1, DELAY=PIPEDELAY2+1
  WM7, SOURCE=AE2, DELAY=PIPEDELAY2+2
  WM8, SOURCE=AE3, DELAY=PIPEDELAY2+3
*
* NOW, ORDERS TO STAGE 1 'COEFFICIENT' WORKING MEMORIES
  GP3, SOURCE=OUT, SA=0, I0=0, TC0=63, SYNCH=BEGIN_STAGE_1,
           DELAY=0
*
* NOW, SET UP AE DATA PATHS FOR COMPUTATION
  AE0, SOURCEA=WM1
  AE1, SOURCEA=WM2, SOURCEB=WM9
  AE2, SOURCEA=WM3, SOURCEB=WM10
  AE3, SOURCEA=WM4, SOURCEB=WM11
*
* NOW SET UP INTERNAL AE DATA PATH AND
*   COMMUTATION AND DELAY IN CROSSBAR
  CE0, RADIX=4, PATTERN=3, SYNCH=BEGIN_STAGE_1, DELAY=PIPEDELAY1
* STAGE 1 READY; START EXECUTION & SET UP FOR STAGE 2
  SIGNAL BEGIN_STAGE_1
* SWITCH SOURCE & SINK WORKING MEMORIES & ISSUE NEW
*   INSTRUCTIONS TO COEFFICIENT MEMORIES AND AE'S
  GP1, SA=0, I0=16, TC0=3, I1=-47, TC1=15, SYNCH=BEGIN_STAGE_2
  WM1, SOURCE=AE0, DELAY=PIPEDELAY2, EVENT=END_STAGE_2
  WM2, SOURCE=AE1, DELAY=PIPEDELAY2+1
  WM3, SOURCE=AE2, DELAY=PIPEDELAY2+2
  WM4, SOURCE=AE3, DELAY=PIPEDELAY2+3
  GP2, SOURCE=OUT, SA=0, I0=1, TC0=63, SYNCH=BEGIN_STAGE_2,
           DELAY=0
* NOW ORDERS FOR COEFFICIENT MEMORIES
  GP3, SA=0, I0=0, TC0=15, I1=16, TC1=3, SYNCH=BEGIN_STAGE_2,
           DELAY=0
* NEW AE ORDERS
  AE0, SOURCEA=WM5
  AE1, SOURCEA=WM6, SOURCEB=WM9
  AE2, SOURCEA=WM7, SOURCEB=WM10
  AE3, SOURCEA=WM8, SOURCEB=WM11
* CONTROL
  CE0, RADIX=4, PATTERN=3, SYNCH=BEGIN_STAGE_2, DELAY=PIPEDELAY1
  WAIT EVENT=END_STAGE_1
  SIGNAL BEGIN_STAGE_2
* STAGE 2 STARTED...SET UP STAGE 3
* PARAMETERS FOR STAGE 3
  GP1, SOURCE=OUT, SA=0, I0=1, TC0=63, SYNCH=BEGIN_STAGE_3,
           DELAY=0
  GP2, SA=0, I0=16, TC0=3, I1=-47, TC1=15, SYNCH=BEGIN_STAGE_3
  WM5, SOURCE=AE0, DELAY=PIPEDELAY2, EVENT=END_STAGE_3
  WM6, SOURCE=AE1, DELAY=PIPEDELAY2+1
  WM7, SOURCE=AE2, DELAY=PIPEDELAY2+2
  WM8, SOURCE=AE3, DELAY=PIPEDELAY2+3
  GP3, SOURCE=OUT, SA=0, I0=0, TC0=3, I1=4, TC1=15,
           SYNCH=BEGIN_STAGE_3, DELAY=0
  AE0, SOURCEA=WM1
  AE1, SOURCEA=WM2, SOURCEB=WM9
  AE2, SOURCEA=WM3, SOURCEB=WM10
  AE3, SOURCEA=WM4, SOURCEB=WM11
  CE0, RADIX=4, PATTERN=3, SYNCH=BEGIN_STAGE_3, DELAY=PIPEDELAY1
  WAIT EVENT=END_STAGE_2
  SIGNAL BEGIN_STAGE_3
* STAGE 3 EXECUTING, SET UP LAST STAGE
  GP1, SA=0, I0=16, TC0=3, I1=-47, TC1=15, SYNCH=BEGIN_STAGE_4
  WM1, SOURCE=AE0, DELAY=PIPEDELAY2, EVENT=END_STAGE_4
  WM2, SOURCE=AE1, DELAY=PIPEDELAY2+1
  WM3, SOURCE=AE2, DELAY=PIPEDELAY2+2
  WM4, SOURCE=AE3, DELAY=PIPEDELAY2+3
  GP2, SOURCE=OUT, SA=0, I0=1, TC0=63, SYNCH=BEGIN_STAGE_4,
           DELAY=0
  GP3, SOURCE=OUT, SA=0, I0=1, TC0=63, SYNCH=BEGIN_STAGE_4,
           DELAY=0
  AE0, SOURCEA=WM5
  AE1, SOURCEA=WM6, SOURCEB=WM9
  AE2, SOURCEA=WM7, SOURCEB=WM10
  AE3, SOURCEA=WM8, SOURCEB=WM11
  CE0, RADIX=4, PATTERN=3, SYNCH=BEGIN_STAGE_4, DELAY=PIPEDELAY1
  WAIT EVENT=EVENT=END_STAGE_3
  SIGNAL BEGIN_STAGE_4
* FINAL STAGE EXECUTING ... SET UP NEXT TASK HERE.
  .
  .
  .
  WAIT EVENT=END_STAGE_4
* HERE TRANSFORM IS COMPLETE...
* NORMALLY ORDERED IN WORKING MEMORIES 1, 2, 3 AND 4.
  END
```

## REFERENCES

1. "Using Chip Carriers for High Density Packaging," Electronic Packaging & Production, October 1977, p. 85.

2. "High-Density Packaging Goes Denser with QUIPS and Square-Chip Carriers," Electronic Design, March 15, 1979, p. 36.

3. "Semiconductor Packaging Trends," Semiconductor International, April 1979, p. 33.

4. "Chip Carrier Update," Electronic Packaging and Production, April 1979, p. 79.

5. "Ceramic Chip Carrier - The New Standard in Packaging," 3M Company reprint of paper presented at 1977 NEPCON, SEMICON, and WESCON conferences.

6. "State of the Art of Leadless Chip Carrier Applications for Avionics Packaging," IEEE Computer Packaging Symposium, Skytop, PA, May 1978.

7. B. Gold and L. R. Rabiner, *Theory and Applications of Digital Signal Processing,* (Prentice-Hall, New York, 1975).

8. A. H. Huntoon, "A Compact Hardware Realization for Approximate Division," Technical Note 1979-57, Lincoln Laboratory, M.I.T. (9 October 1979), DDC AD-A079370/3.

## REPORT DOCUMENTATION PAGE

READ INSTRUCTIONS BEFORE COMPLETING FORM

1. REPORT NUMBER
ESD-TR-82-005

2. GOVT ACCESSION NO.
AD-A115 026

3. RECIPIENT'S CATALOG NUMBER

4. TITLE *(and Subtitle)*
The Programmable Matched Filter:
A Design Study

5. TYPE OF REPORT & PERIOD COVERED
Technical Report

6. PERFORMING ORG. REPORT NUMBER
Technical Report 603

7. AUTHOR(s)
Anthony E. Filip
John D. Drinan
Albert H. Huntoon
Jeffrey D. Kurtze
Donald Malpass

8. CONTRACT OR GRANT NUMBER(s)
F19628-80-C-0002

9. PERFORMING ORGANIZATION NAME AND ADDRESS
Lincoln Laboratory, M.I.T.
P.O. Box 73
Lexington, MA 02173-0073

10. PROGRAM ELEMENT, PROJECT, TASK AREA & WORK UNIT NUMBERS
Program Element No.62702F
Project No.4594

11. CONTROLLING OFFICE NAME AND ADDRESS
Rome Air Development Center
Griffiss AFB, NY 13440

12. REPORT DATE
1 April 1982

13. NUMBER OF PAGES
88

14. MONITORING AGENCY NAME & ADDRESS *(if different from Controlling Office)*
Electronic Systems Division
Hanscom AFB, MA 01731

15. SECURITY CLASS. *(of this report)*
Unclassified

15a. DECLASSIFICATION DOWNGRADING SCHEDULE

16. DISTRIBUTION STATEMENT *(of this Report)*

Approved for public release; distribution unlimited.

17. DISTRIBUTION STATEMENT *(of the abstract entered in Block 20, if different from Report)*

18. SUPPLEMENTARY NOTES

None

19. KEY WORDS *(Continue on reverse side if necessary and identify by block number)*

Multi-processor
digital design
distributed processor
multi-dimensional
matched filter

20. ABSTRACT *(Continue on reverse side if necessary and identify by block number)*

The Programmable Matched Filter was envisioned as a flexible approach toward real-time, multi-dimensional matched filtering problems. Its design features multiple, parallel arithmetic elements communicating with multiple memories via a crossbar switch at a clock rate of 16.7 MHz. The machine will sustain a throughput rate of more than 500 million real operations per second, or more than six Cray-1 computers. The extensive use of low-dissipation CMOS technology in large scale integrated circuits yields an estimated total of 5200 integrated circuits, dissipating less than one kilowatt, occupying 0.2 cubic meters and weighing approximately fifty kilograms.