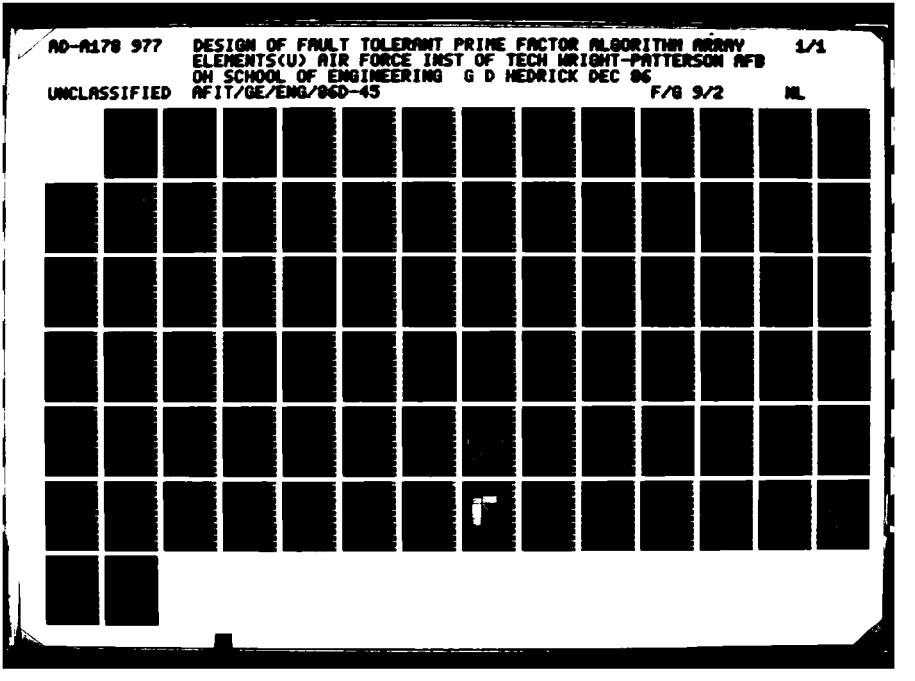
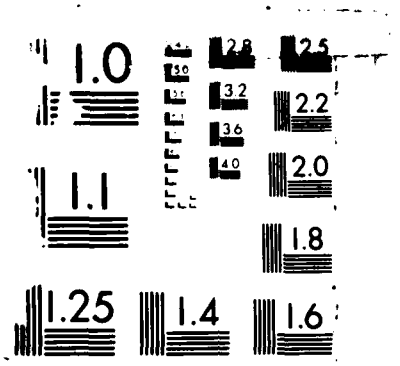
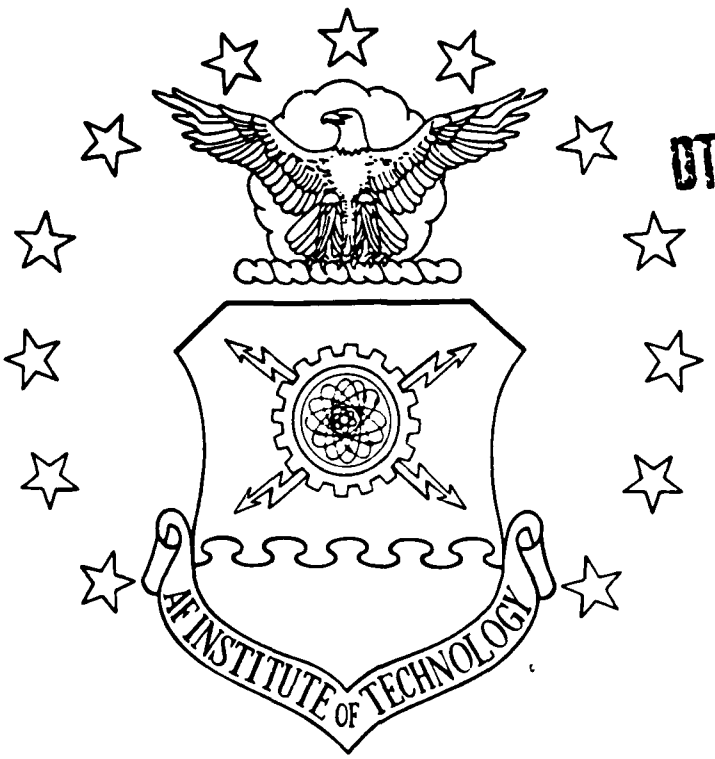

AD-A178 977 DESIGN OF FAULT TOLERANT PRIME FACTOR ALGORITHM ARRAY 1/1
ELEMENTS(U) AIR FORCE INST OF TECH WRIGHT-PATTERSON AFB
OH SCHOOL OF ENGINEERING G D HEDRICK DEC 86
UNCLASSIFIED AFIT/GE/ENG/86D-45 F/G 9/2 NL


1.0
1.1
1.25
1.4
1.6
1.8
2.0
2.2
2.5
2.8
3.2
3.6
4.0



DESIGN OF FAULT TOLERANT PRIME FACTOR ALGORITHM ARRAY ELEMENTS

THESIS

GARY D. HEDRICK B.S.E.E.
CAPTAIN, US AIR FORCE
AFIT/GE/ENG/86D-45

DEPARTMENT OF THE AIR FORCE
AIR UNIVERSITY
AIR FORCE INSTITUTE OF TECHNOLOGY

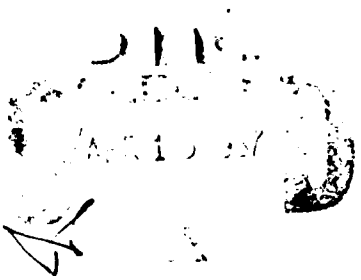
Wright-Patterson Air Force Base, Ohio

AFIT/GE/ENG/86D-45

DESIGN OF FAULT TOLERANT PRIME FACTOR ALGORITHM ARRAY ELEMENTS

THESIS

GARY D. HEDRICK B.S.E.E.
CAPTAIN, US AIR FORCE
AFIT/GE/ENG/86D-45



Approved for public release; distribution unlimited

AFIT/GE/ENG/86D-45

DESIGN OF FAULT TOLERANT PRIME FACTOR ALGORITHM ARRAY ELEMENTS

THESIS

Presented to the Faculty of the School of Engineering

of the Air Force Institute of Technology

Air University

In Partial Fulfillment of the

Requirement for the Degree of

Master of Science in Electrical Engineering

GARY D. HEDRICK

CAPTAIN, US AIR FORCE

December 1986

Acknowledgement

I would like to thank my loving wife, Sandra, whose introduction to married life came during the course of this effort. Without her support and understanding this effort would not have been completed.

I would also like to thank my thesis advisor, Captain Richard Linderman, for the guidance and timely remotivation needed to ensure the successful completion of my thesis.

TABLE OF CONTENTS

TABLE OF CONTENTS (continued)

TABLE OF CONTENTS (continued)

LIST OF FIGURES

Chapter 1

Chapter 2

Chapter 3

LIST OF FIGURES (continued)

Chapter 4

Chapter 5

LIST OF FIGURES (continued)

Abstract

The calculation of the discrete Fourier transform (DFT) has long been a significant bottleneck in many Digital Signal Processing applications. This ~~research effort~~ contributes to the goal of implementing a very large-scale integrated (VLSI) circuit which uses the Winograd and Good-Thomas algorithms for computing DFTs with composite blocklengths. Winograd processors use both the small and large Winograd algorithms to compute DFTs with blocklengths of 15, 16, and 17. Longer blocklength DFTs (240, 255, 272, and 4080 are computed using a pipeline of Winograd processors, dula-port memories, and a controller. The pipeline uses the Good-Thomas Prime Factor Algorithm (PFA). Fault tolerance was enhanced by the use of watchdog processors to provide concurrent fault detection and error control coding (ECC) to provide fault masking.

A pipeline controller was designed as a Reduced Instruction Set Computer (RISC) to control the pipeline and regulated the flow of data. Also, a dual ported, double buffered RAM memory was designed to provide intra-pipeline data storage.

DESIGN OF FAULT TOLERANT

PRIME FACTOR ALGORITHM

ARRAY ELEMENTS

CHAPTER 1

INTRODUCTION

1.1. Background

Many of the problems of digital signal processing involve computation of the Discrete Fourier Transform (DFT). These problems need spectral information, given by the DFT, on a large number of sample points to obtain the necessary spectral resolution for precise calculation of operations such as correlation and convolution. Current applications for rapid spectrum analysis include radar, sonar, pattern recognition, voice processing, target acquisition systems, and artificial intelligence.

With the introduction of large scale integration of digital systems, it has become practical to implement complex signal processing functions on a single chip. Future applications include not only enhancements of current implementations but also many potential applications not feasible due to speed and size limitations of today's technology. Advances in hardware technology must be matched with clever algorithmic designs to handle the computational tasks. Although new DFT algorithms and systems architectures are constantly being developed, the most advanced systems available today are still not able to satisfy all the military requirements for compact, high speed digital signal processing systems that utilize the DFT.

Therefore, the Air Force Office of Scientific Research, DARPA - Defense Advanced Research Projects Agency, and Air Force Space Division are sponsoring research for the development of a high speed DFT processor. To achieve a more advanced DFT processing system which meets the Air Force requirements, new transform algorithms and hardware architectures must be combined to simultaneously increase data processing throughput and decrease size and weight. One method of increasing the throughput is to reduce the number of operations required to compute the DFT. Winograd has shown a class of algorithms (Winograd Fourier Transform Algorithms, WFTA) to use the fewest number of multiplications in computing the DFT [Wi78]. Reducing the number of multiplications gives a larger increase in performance than reducing the number of additions because the multiplications are more complicated, requiring several additions to yield the product.

The Winograd DFT algorithm is of interest in VLSI because the matrix form of the algorithm maps efficiently, in terms of space and regularity of structure, into a signal processing architecture. Also, large data block lengths may be computed by combining various Winograd modules into a pipelined architecture in the manner specified by the Good-Thomas Prime Factor algorithm. The Good-Thomas Prime Factor Algorithm (PFA) uses relatively prime modules to compute true multi-dimensional DFTs. The 4080-point processor was chosen as a representative system. The 4080-point PFA processor uses Winograd modules of lengths 15,16, and 17. The layout of the 4080-point PFA processor is shown in Figure 1-1. There are separate chips for each Winograd processor, for each dual-port memory, and for the PFA controller. Systems for computing DFTs with block lengths of 240, 255, and 272 points will use one less Winograd processor and dual-port memory.

In the figure the memories are divided in two parts with hashing to illustrate the symmetrical operations of the memory accesses. At any point in time the pipeline is structured so that all reads are accomplished on memory banks with one hashing and

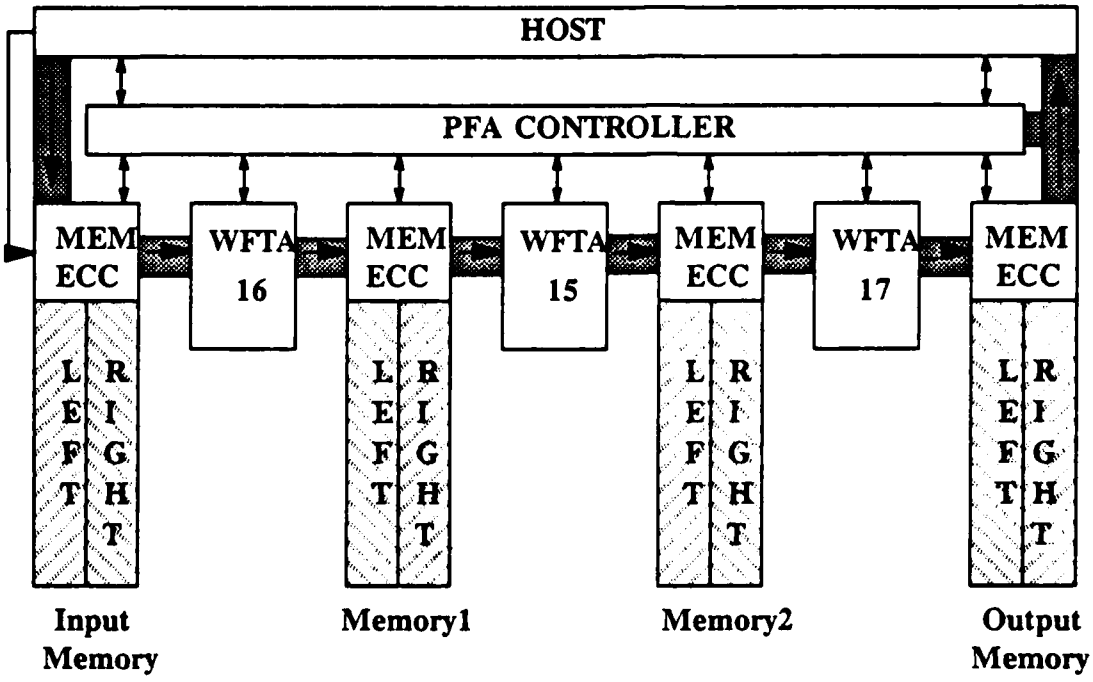


Figure 1-1 4080-POINT PFA PROCESSOR

all writes are accomplished on the memory banks with the other type of hashing.

1.2. Problem

To implement a 4080-point PFA pipeline certain aspects must be considered:

1. WFTA processors
2. A Reduced Instruction Set Computer (RISC) for PFA pipeline control
3. High Speed RAM with Error Correction Circuitry for the PFA pipeline
4. Clock Generation
5. Hybrid Circuit/Board Design

This thesis will be concerned with the design of a pipeline controller to control the pipeline and regulate the flow of data. Also, an intermediate data storage medium.

Random Access Memories, will be implemented to facilitate the control of the data flow. This thesis will promote fault tolerance in the structure of the pipeline and its management. Also, fault tolerance will be enhanced by memory error detection and correction circuitry.

1.3. Scope

1.3.1. Past Theses.

Four past theses addressed other aspects of the PFA pipeline design. Taylor presented the PFA and WFTA theory, overall signal processing systems architecture, and numerical precision simulation results [Ta85]. Coutee presented the arithmetic circuitry for the 16 point WFT processor (WFTA16) [Co85]. Rossbach presented the control for the high speed Winograd DFT processor [Ro85]. Collins presented a validation program for WFTA16 processor operations, and described the WFTA16 modules in VHSIC (Very High Speed Integrated Circuit) Hardware Description Language (VHDL) [Co85.....].

1.3.2. Parallel Theses.

Shephard presents a design for testability of the WFTA 16 processor, integrates the macro cells designed by Coutee and Rossbach and completes the design of the WFTA16 processer [Sh86]. Cooper presents a description of WFTA modules in VHDL [Co86].

1.3.3. **Scope of This Thesis.** The scope of this thesis is to complete the development of a fault tolerant PFA pipeline processor by specifying the architecture of the PFA controller; designing a high speed RAM memory with error correction circuitry (ECC) for the PFA pipeline; and prototyping the ECC chip in a three-micron CMOS technology.

1.3.4. Future Theses.

Clock generation and Hybrid Circuit/Board Design will be addressed in future theses.

1.4. Summary of Current Knowledge

1.4.1. CMOS Technology.

CMOS (Complementary Metal Oxide Silicon) technology is recognized as a leading contender for existing and future VLSI systems. CMOS provides an inherently low power static circuit technology that has the capability of providing a lower power-delay product than NMOS or PMOS technologies using comparable design-rules [We85]. The PFA pipeline components are being implemented in CMOS technology. When discussing the performance characteristics of CMOS circuits, the main points to observe are that a CMOS gate consumes no DC power when the output is a "one" or "zero" level and the output logic levels of CMOS are fully restored.

Two basic circuits used are the transmission gate and the inverter. The transmission gate consists of a PMOS transistor in parallel with an NMOS transistor. The NMOS transistor passes a weak "one" and a strong "zero", while the PMOS transistor passes a weak "zero" and a strong "one". Since the PMOS transistor is in parallel with the NMOS transistor, the transmission gate passes a strong "one" and a strong "zero". Therefore the transmission gate closely approximates an ideal switch. The inverter is also built to ensure that a strong signal is always output. The inverter outputs Vdd (5 volts) when "zero" is input and outputs Vss (GND) when a "one" is input. Only one MOS transistor is turned on when the inverter is in either state. Therefore, there is no direct current flow when the inverter is not changing state and no static power dissipation except for a small amount due to junction leakage. During switching both transistors are on and current is drawn only while the input voltage passes from the threshold voltage for the NMOS transistor to Vdd minus the threshold voltage for the PMOS

transistor.

CMOS circuits are susceptible to latch-up because of the presence of a p-n-p-n structure. The circuit designs and I/O pad designs of the WFTA chips are designed to protect against latch-up [Li86].

1.4.2. PFA and WFTA.

1.4.2.1. Small/Large WFTA's PFA.

Constructing a DFT using Winograd's small DFT algorithm requires knowledge of Rader's prime algorithm [Ra68] and Winograd's short convolution algorithms [Wi78]. Algorithms are presented in great detail by Blahut [Bl85] and McClellan and Rader [Mc79]. Winograd's Large DFT algorithm produces a longer block length by combining relatively prime small Winograd modules to yield a block length equal to the product of the block lengths of the small Winograd modules. The Winograd modules provide an efficient means of computing short and medium length DFTs. One of the drawbacks of using the large Winograd modules is the matrices become large and unwieldy, imposing serious memory constraints for DFT computation (e.g., a 255-point transform computed using the large Winograd algorithm requires a multiplication matrix of size 1280). The Good-Thomas prime factor algorithm requires more overall multiplications than the large Winograd algorithm. But, since the DFT is broken into several component parts, the total number of multiplications is resolved into several subproblems (e.g., a 255-point DFT computed using the Good-Thomas PFA requires multiplication matrices of size 18 and 36 for the 15-point module and the 17-point module, respectively; for a total number of 1900 multiplications [Ta85].

1.4.2.2. Fault Tolerance.

There are two fundamentally different approaches that can be taken to increase the reliability of computing systems. The first approach is called fault prevention and

the second approach is called fault tolerance. In the traditional fault prevention approach, the objective is to increase the reliability by a prior elimination of all faults. Although semiconductor manufacturers try to ensure that their products are reliable, it is almost impossible to eliminate any chance faults somewhere in a system. The reliability of a system can be increased by employing the method of worst case design, using high quality components and imposing strict quality control procedures during the assembly phase. However, such measures can increase the cost of a system significantly. An alternative approach to reliable system design is to incorporate redundancy into a system with the aim of masking the effects of faults. This approach does not necessitate the use of high-quality components, instead standard components can be used in a redundant and reconfigurable architecture. In view of the decreasing cost of hardware components it is certainly less expensive to use the second approach to design reliable systems. This second approach is called fault tolerance.

Fault tolerance is the ability of a system to operate in the presence of errors. Fault tolerance includes fault avoidance, fault detection, and system recovery. Fault avoidance is an attempt to reduce the number of faults by careful component design [Si84].

Digital systems, even when designed with highly reliable components, do not operate forever without developing some faults. When a system ultimately does develop a fault, the fault has to be detected and located so that the effect of the fault can be removed. Fault detection means the discovery of something wrong in a digital systems or circuit. A fault may or may not cause a failure. A failure is said to have occurred if it deviates from it's specified behavior A fault may be permanent or temporary. A major portion of digital system malfunctions are caused by temporary faults Transient faults are non-recurring temporary faults. They are usually caused by alpha particle radiation or power supply fluctuation, and they are not repairable because there is no physical damage to the hardware. They are the major source of problems in

semiconductor memory chips.

Intermittent faults are recurring faults that reappear on an irregular basis. Such faults can occur due to loose connections, partially defective components, poor designs, or environmental conditions such as temperature, humidity, vibration, etc. The likelihood of intermittents due to environmental conditions depends on how well the system is protected from its physical environment through cooling, shielding, filtering, etc. Since intermittent faults are random, they can be modeled only by using probabilistic methods [Br73].

Two types of techniques are used in order to prevent temporary faults from causing system failures: fault masking and concurrent fault detection.

Fault masking, also known as static redundancy, uses extra components so that the effect of a faulty component is masked instantaneously. Two major techniques employed to obtain fault masking are triple modular redundancy (TMR) and the use of error control coding (ECC). TMR was originally suggested by von Neumann [Ne56].

Essentially, a voting element accepts the outputs of three sources and delivers the majority vote as its output. It has been suggested that the TMR scheme could also be a practical technique for designing fault tolerant VLSI chips [Se78]. The use of error control coding will be discussed in section 1.4.1.4.

Concurrent fault detection, also known as standby redundancy, uses extra components to generate error signals. The presence of a fault is detected, and a subsequent recovery action either eliminates the fault or corrects the error. The principle techniques are: replacement of the faulty element or system by a standby spare; reorganization of the system into a different configuration. These two methods presuppose the existence of a diagnostic procedure which will recognize the symptoms [Mo56], and of a switch which implements the replacement or reconfiguration [Fl68]. In general, dynamic redundant systems can be divided into two categories: cold-standby system and hot-standby system. In a cold-standby system one module is powered up and

operational, the rest are not powered, i.e., they are cold spares. Replacement of a faulty module by a spare is effected by turning off its power and powering a spare. In a hot-standby system all the modules are the same, the output of any arbitrarily selected module can be taken as the system output. When a fault is detected in a module the system is reconfigured so that the system output comes from one of the remaining modules [La85].

1.4.2.3. **Memory.**

Random access memory (RAM) has an access time independent of the physical location of the data. There are various bipolar and MOS technologies used to implement random access memories. The bipolar technologies, which include ECL, TTL, and I^2L, are employed to construct RAM circuits with resistors, diodes, and bipolar transistors. The primary characteristics of bipolar RAMs are high speed, high power consumption, and low density. MOS RAM ICs are constructed with N-channel and P-channel transistors. The two technologies widely used are NMOS and CMOS. These technologies yield RAMs with improved density and power characteristics compared to bipolar technologies. CMOS RAMs offer another advantage over bipolar RAMs, low power consumption.

The first MOS technology used to manufacture memory devices was PMOS technology. P-channel MOSFETs are used as the memory cell. The result was high density, low-cost memories, but with slow operation and high power consumption.

The next technology, NMOS, results in faster operating speeds because the mobility of electrons is higher for NMOS than PMOS. N-channel MOSFETs are used as the memory cell. Not only are high operating speeds achieved with NMOS, but NMOS also yields higher-density circuitry than does PMOS. These devices are still much slower than bipolar memories and do not overcome the high power limitation (at least for SRAM, not for DRAM).

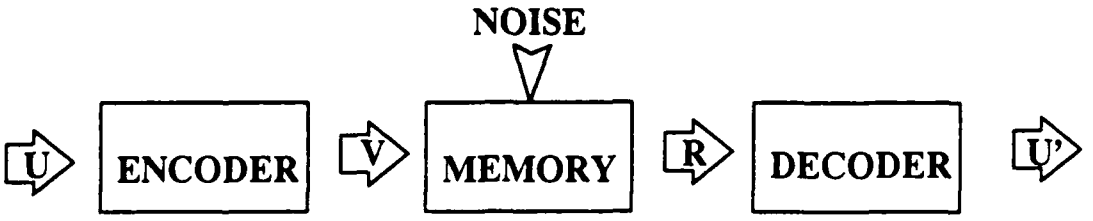
The problem of high power consumption is overcome with CMOS technology. Both P-channel and N-channel MOSFETs are used together to make a low-power storage cell, one MOSFET acts as a switch and the other as a load resistance. Since both P-channel and N-channel devices are used, the cells are less dense than NMOS RAM cells. The resulting memory cell has very good power characteristics. Also, memories made with CMOS technology enjoy higher noise immunity than memories made with any other technology [Tr82].

The memory cells used in RAMs can be divided into two basic categories: static RAM (SRAM) and dynamic RAM (DRAM). SRAMs use some form of latched storage, while DRAMs use dynamic storage by charging a capacitor.

Increasing demands for lower cost, higher bit density, and better performance have caused a proliferation of new semiconductor memories. Currently, off-the-shelf 256k-bit memories have access times of approximately 40 nanoseconds [Ya84]

1.4.2.4. **Error Detection and Correction Code.** In 1948, Shannon demonstrated that, by proper encoding of the information, errors induced by a noisy channel or storage medium can be reduced to any desired level without sacrificing the rate of information transmission or storage. A large amount of effort has been expended on the problem of devising efficient encoding and decoding methods for error control in a noisy environment. The use of coding for error control has become an integral part in the design of modern communication systems and digital computers. The transmission and storage of digital information have many similarities. Both transfer data from an information source to a destination. A typical storage (transmission) system may be represented by Figure 1-2.

The encoder transforms the information sequence U into a discrete encoded sequence V called a code word. This waveform enters the storage medium and is corrupted by noise. The decoder processes the received sequence R. The decoder transforms the received sequence R into a binary sequence U' called the estimated



DECODER CONSISTS OF :

SYNDROME CALCULATION CIRCUIT
ERROR PATTERN DETECTING CIRCUIT

Figure 1-2 TYPICAL STORAGE SYSTEM

sequence. The decoding strategy is based on the rules of channel encoding and the noise characteristics of the storage medium. Ideally, U' will be a replica of the information sequence U, although the noise may cause some decoding errors.

There are two different types of codes in common use today, block codes and convolution codes. The encoder for a convolution code accepts k-bit blocks of the information sequence U and produces an encoded sequence V of n-symbol blocks. In convolutional coding, the symbol U and V are used to denote sequences of blocks rather than a single block. However, each encoded block depends not only on the corresponding k-bit message block at the same time unit, but also on m previous message blocks. Therefore, the encoder has a memory order of m.

The encoder for a block code divides the information sequence into message blocks of k information bits each. In block coding, the symbol U is used to denote a k-bit

message rather than the entire information sequence. Since the n-symbol output code word depends only on the corresponding k-bit input message, the encoder is memoryless.

1.5. Approach

The responsibilities for further refinement and development of a fault tolerant PFA pipeline were distributed between the three thesis efforts presented previously. The macro cells of the WFTA16 were to be integrated with testability incorporated into the design. The development of a fault tolerant PFA pipeline processor was to be completed by: specifying the architecture of the PFA controller; designing a pipeline memory bank with ECC; and prototyping the ECC memory chip in three micrometer CMOS. The WFT modules were to be described in VHDL.

1.6. Sequence of the Presentation

Chapter 2 examines the requirements for control of the PFA pipeline, memory capacity and control, and suggests alternative approaches to satisfy those requirements.

Chapter 3 presents an architecture and algorithm to best satisfy the requirements presented in chapter 2.

Chapter 4 presents aspects of the VLSI implementation of the circuits described in chapter 3, block diagrams and some unique cells.

Chapter 5 offers recommendations and conclusions

CHAPTER 2

DETAILED ANALYSIS OF THE PROBLEM

2.1. Introduction

As stated previously, this thesis effort encompasses the design of a controller for the Prime Factor Algorithm (PFA) pipeline and the design of dual ported, double buffered high speed RAM memory with error detection and correction circuitry for the pipeline.

2.2. RISC PFA Controller

The control tasks for the PFA pipeline can be divided into two major sections. One section is concerned with the sequencing of the elements of the pipeline, this section is called data flow control. The other section is concerned with the fault tolerance operations, this section is called configuration control.

2.2.1. Requirements for Data Flow Control.

The 4080 point PFA system uses independent Winograd processors of length 16, 15, and 17. DFTs with block lengths of 240, 255, and 272 points can be implemented with the Winograd modules of length 15 and 16, 15 and 17, and 16 and 17, respectively. Since the Winograd processors operate autonomously they can be pipelined. The layout of the 4080-point PFA pipeline was presented in Figure 1-1. There are eight separate chips: three Winograd processors, four dual-port memories, and the PFA controller. The dual-port memories must allow one processor (or host) to write to half of the memory while another processor reads from the other half of the memory. The PFA controller provides overall control of the PFA pipeline. Asynchronous control signals to the memories, to and from the Winograd processors, and to and from the host

must flow through the PFA controller. The only exception to this rule is the write (read) strobe generated by the Winograd processors or by the host which is sent directly to the memories.

The PFA controller operates autonomously, communicating with the host only at the beginning and end of each DFT. The interface between the PFA controller and each of the Winograd processors is similar. The Winograd processors also operate autonomously and communicate with the PFA controller only at the beginning and end of each 16, 15, 17-point DFT. The only control the PFA controller exerts on the memories is to switch the status of the left and right halves of the memories.

The operation of the PFA pipeline can be explained by stating that even though data flows serially through the pipeline there is a small sequence of operations that are performed in parallel for similar modules.

There are four PFAC-memory interfaces. Also, there are five PFAC-processor interfaces. Three of the PFAC-processor interfaces are the interfaces to the Winograd processors. The other two PFAC-processor interfaces are the input host to the PFAC and the output host to the PFAC. The input host and the output host can be the same entity but this is not required. As a matter of fact, the two functions probably won't be accomplished by the same entity. The input data will probably be input by a host with limited analytical capabilities. Whereas, the output data will be accepted by a host with the ability to analyze the data and then make decisions concerning the validity of the data and the health of the PFA pipeline.

The PFAC-memory interface consists of one signal, RIGHT LEFT (R L). When R L is high, data is read from the right side of memory and written to the left side of memory.

All of the PFAC-processor interfaces have OPERATE (OPR) and DONE signals. The PFAC-processor interfaces for the Winograd processors have additional signals for processing data. These signals are used to pass the block floating point exponent to the

Winograd processors (SF0, SF1, SF2) and to receive the block floating point exponent from the Winograd processors (SO0, SO1, SO2). The PFAC must accumulate the total block floating point exponent for each transform computed.

All four of the PFAC-memory interfaces will operate in a parallel lock- step fashion as will all five of the PFAC-processor interfaces.

Initially, the R/L signal is high. All memories will receive the same R/L signal simultaneously. Next, the PFAC must pass the magnitude of the input transform data (SF0, SF1, SF2). Then, a master OPERATE (OPR) will be transmitted to all PFAC-processor interfaces simultaneously. The PFAC will monitor the DONE lines for all five PFAC-processor interfaces. The PFAC waits until all five DONE signals are high, then the PFAC latches the magnitude of the scaling exponent output by each Winograd processors. The PFAC then lowers OPERATE. While OPERATE is low, the PFAC accumulates the magnitude of the output transform data for each problem in the pipeline and switches the R/L signal.

2.2.2. **Requirements for Configuration Control.**

The configuration control section of the PFAC is concerned with fault detection, fault correction, and system recovery. The PFAC must keep track of the number of errors reported by each element of the PFA pipeline and report these errors to the host. The memories have on-chip error-correcting codes to provide fault detection capability. Errors that are detected and corrected are reported to the PFAC using the ERROR CONTROL CODE CORRECTED (ECCC) signal. Errors that are detected and uncorrected are reported to the PfAC using the ERROR CONTROL CODE UNCORRECTED (ECCU) signal.

If a memory chip should suffer catastrophic failure, the on-chip ECC would be useless. However, parity checking on the Winograd chips will detect this condition and report to the PFAC using the signal PARITY ERROR (PE). Odd parity is used so

the "stuck at zero" or "stuck at one" states (caused by memory power failure) may be detected.

The pipeline can be configured as shown in Figure 2-1. In the absence of faults the PFA pipeline can operate correctly as shown in Figure 1-1. However, digital systems do not operate forever without developing some faults. The configuration shown in Figure 2-1 employs concurrent fault detection. The extra components generate error signals. The Winograd processors can operate normally (active mode) or in the watchdog mode. The PFAC programs each active and watchdog by setting or resetting a watchdog mode flip-flop on each Winograd chip.

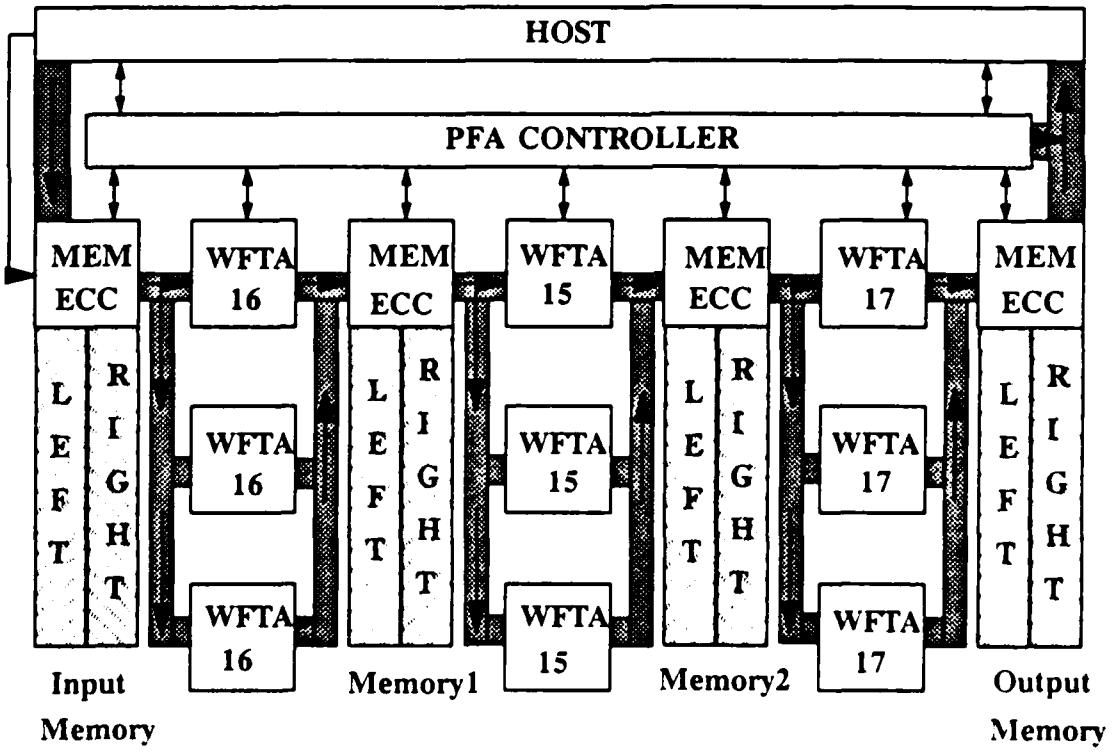


Figure 2-1 4080-POINT PFA PROCESSOR WITH WATCHDOGS

The particular configuration shown is three active processors in series to form the basic PFA pipeline, with two watchdog processors in parallel with each active processor. In theory as many watchdogs as desired can be placed in parallel. The present PFA controller only allows two Watchdogs due to pin limitations. Each watchdog processor computes the same DFT as its corresponding active processor. If the DFT results of the watchdog processor do not agree with those computed by the active processor, the watchdog processor reports a WATCHDOG ERROR (WE) to the PFAC.

The PFAC must decide if the active processor is in error or if a watchdog processor is in error. Since there are two watchdog processors, in this case, the PFAC can use voting to assign errors. If the error is assigned to the watchdog processor then the PFAC simply keeps track of the error and reports the error to the host. If the error is assigned to an active processor then the PFAC must still keep track of the error and report to the host but must also report to the host that the data may be erroneous, by turning off the DATA VALID bit in the problem status word.

Whenever an active processor commits an error the PFAC must reconfigure that element of the pipeline (i.e., decide which of the watchdog processors to make the active processor). If the number of errors assigned to the different processors are not equal then the current watchdog processor with the smallest number of errors is programmed as the new active processor. After an error the active must become a watchdog unless the other two are marked as "dead" in the health register. The status of the PFA elements must be placed on the data lines of the output by the PFAC whenever the host inquires about status. The host can control the configuration of the pipeline by sending in a configuration word. The host can interrupt operations to check status of the pipeline and to input limited instructions regarding the configuration of the pipeline and error reporting.

The sequence of operations necessary to control the flow of data

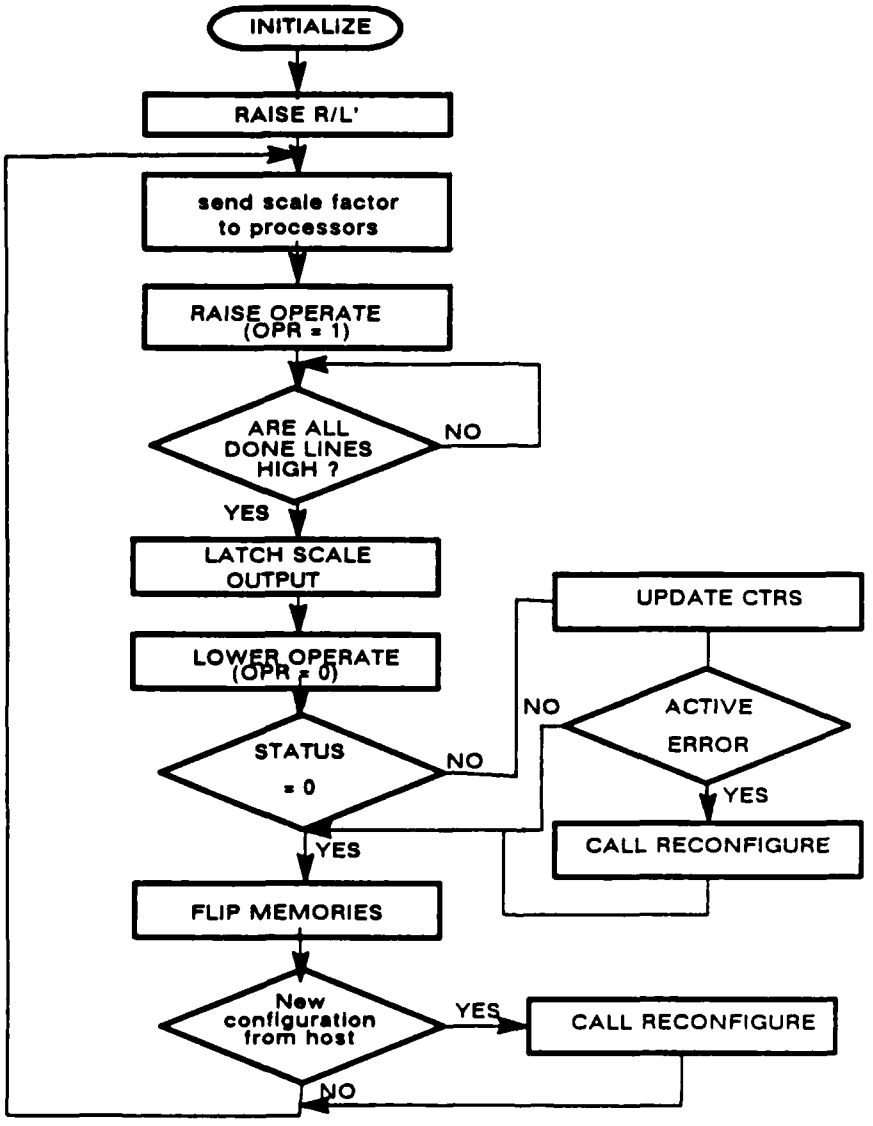


Figure 2-2 SEQUENCE OF OPERATIONS

2.2.3. **Approaches and Trade-Offs.**

The PFAC is a sequential state machine. A method for implementing control circuitry for a sequential state machine is needed. There are a number of design methods available to implement a sequential control circuit. The most popular of these control methods include custom logic implementations, gate array implementations, programmable logic arrays (PLA) or ROMS.

Custom logic implementations are an interconnection of flip-flops and gates. The gates by themselves constitute a combinational circuit but when included with the flip-flops, the overall circuit is classified as a sequential circuit. The design procedure consists of translating the circuit specification into state diagrams, state tables, and excitation tables [Ma79].

Gates arrays are fixed arrays of identical logic circuits. Designing with gate arrays consists of specifying the wiring connections between the given logic elements in the array [On84].

ROMs can be used in the design of sequential circuits. A ROM generates an input-output relation specified by a truth table. As such, a ROM can implement any combinational logic circuit with k inputs and n outputs.

A programmable logic array (PLA) is similar in concept to a ROM except that a PLA does not provide full decoding of the input lines.

ROMs can also be used to store coded information that represents the sequence of internal control variables needed for the operation of a sequential state machine. When a ROM is used to store binary control information, it is called microprogrammable control [Ma82]. The control unit initiates a series of sequential steps of micro operations. Changes can be made to a microprogrammed control without changing the hardware by just modifying the sequence of micro-operations.

The PFAC is divided in function between data flow control and configuration control. The PFAC data flow control section requires limited programmability. Whereas, the configuration control section needs to have a greater amount of programmability to store the detection/reconfiguration algorithms.

2.3. Memory Capacity and Control

2.3.1. Requirements.

With today's 1.2 micron CMOS technology, all the memory control and arithmetic circuitry for the WFT processor can be placed on the chip [Sh86]. However, the data generated by each WFT processor cannot be stored on the same chip Coutee 1985 Therefore, separate chips are required for data storage. For a 4080-point transform the memory must be organized into two sides of 4080 48-bit words (24 bits for real and 24 bits for imaginary data, total number of bits = 391,680). The memory must be dual-ported and double buffered. A double buffered memory enables 100% utilization of the memory. One port will always read and the other port will always write. Each port will be connected to only one side of the memory at a time. The memory must accept a signal RIGHT/LEFT (R/L) to switch the connection of the ports to one side of memory or the other side of memory. Each port has a WRITE (READ)-STROBE (WS) associated with its operation. The memory must perform a write (read) operation in response to this signal. The access time for the data memories must be less than 28 nanoseconds.

The parity checking of the Winograd processors can detect catastrophic failure of a memory chip. Error control code (ECC) circuitry must be placed on the memory chip to minimize the effect of transient faults on the data. The ECC should correct all single bit error patterns and detect all two bit error patterns. In addition, the code should correct as many double bit errors as possible.

2.3.2. Approaches and Trade-Offs.

2.3.2.1. Static RAM / Dynamic RAM .

As mentioned in chapter 1, the two categories of RAM memory cells are static and dynamic. A six transistor SRAM cell is shown in Figure 2-3.

A four transistor SRAM cell can be obtained by replacing the p transistors of the static cell with resistors, as shown in Figure 2-4. A DRAM can ge obtained by eliminating the pullups altogether. Although the basic DRAM cell is smaller than the SRAM cell, the DRAM cell has to be refreshed to retain the contents of the cell. However, when the pipeline is operating at normal operating speed the amount of time between the write and read operation for a memory location is less than the amount of time for a refresh cycle. The normal operation of the memory would refresh the memory cells,

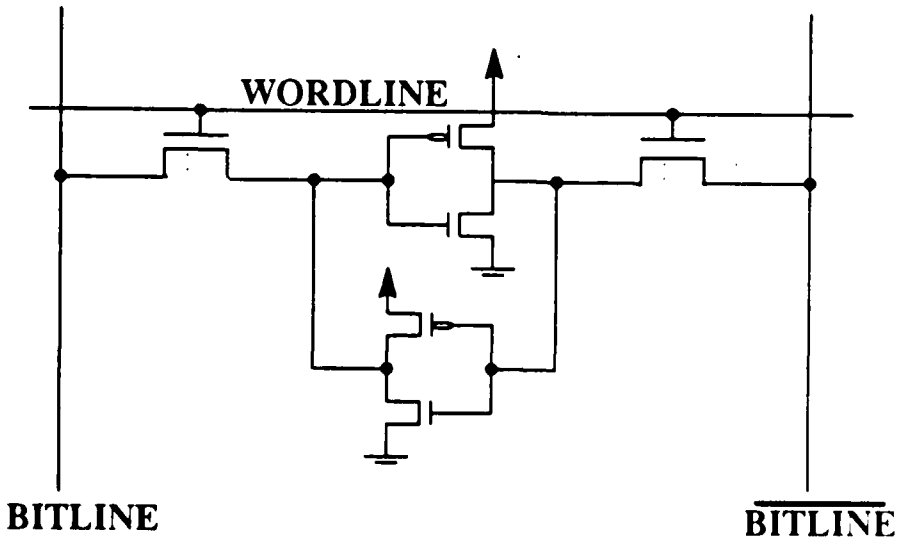


Figure 2-3 SIX TRANSISTOR SRAM

so that separate refresh circuitry would be unnecessary.

SRAMs and DRAMs can be designed with less transistors per memory cell. However, these denser designs require control or at least knowledge of the processes that will be used to fabricate the memories. SRAMs are easier to design and are potentially less troublesome than DRAMs. SRAMs tend to be faster (but much larger) than DRAMs (Weste and Eshragian, 1985).

2.3.2.2. **Memory Partioning Options.**

Each Winograd processor outputs two 24-bit data words in parallel. One 24-bit data word is for the real data and the other is for the imaginary data. The data is sent to the memory on two parallel 24-bit busses. Although identical parallel operations are performed on the real data and the imaginary data, the real data and the imaginary

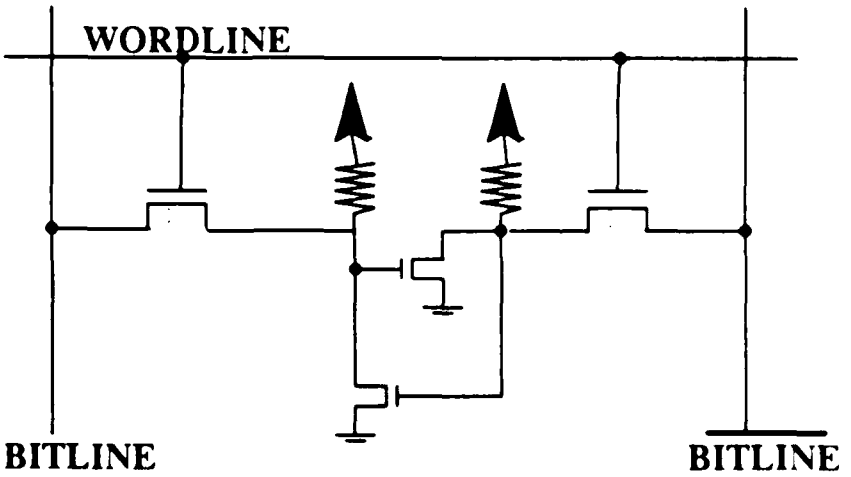


Figure 2-4 FOUR TRANSISTOR DRAM

data can be treated separately. Therefore, the memory can be partioned between real data and imaginary data as shown in Figure 2-5. Also, the memory read and write operations are mutually exclusive. Therefore, the memory can be partioned between read and write operations as shown in Figure 2-6.

When memory is partitioned between real and imaginary data, there are four address decoders and eight address busses the address busses are longer and the data busses are shorter. With this approach data can arrive before the address is selected.

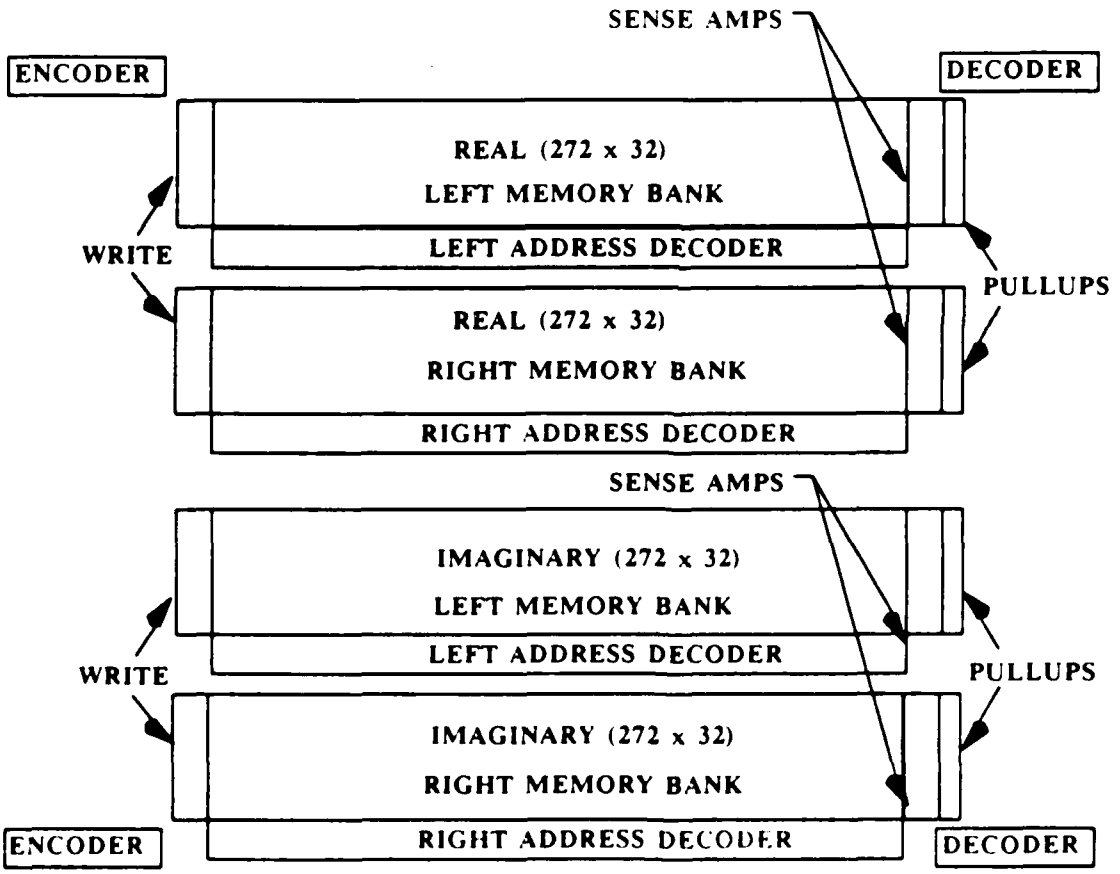


Figure 2-5 MEMORY PARTITIONED - REAL IMAGINARY

When memory is partitioned between read and write operations, there are only two address busses, the address busses are shorter and the data busses are longer. With this approach, the address can be selected before the data arrives.

2.3.2.3. **Error Detection and Correction.**

As stated in chapter 1, by proper encoding of information, errors induced by a noisy storage medium can be reduced to any desired level without sacrificing the rate of information storage. There are two different types of codes in common use today, block codes and convolution codes. Since the encoder of a convolution code has a memory, sequential logic circuits are used to implement the convolution code encoder. However, the encoder for a block code does not have a memory. Combinational logic circuits are used to implement a block code encoder. Since, combinational logic circuits are, generally, easier to implement, convolution codes will not be discussed further.

The encoder for a block code divides the information sequence into message blocks of "k" information bits each. A message block is represented by the binary k-tuple U = (U1, U2, ..., Uk) called a message. In block coding, the symbol "U" is used to denote a k-bit message rather than the entire information sequence. There are a total of 2^k different possible messages. The encoder transforms each message "u" independently into an n-tuple "V" = (v1, v2,....,vn) of discrete symbols called a code word. "V" denotes an n-symbol block rather than the entire encode sequence. There are 2^k different possible code words at the encoder output corresponding to the 2^k different possible messages.

In a binary code, each code word "V" is also binary. Since for a binary code to be useful it should have a different code word assigned to each message, then "k" less than or equal to N or R less than or equal to 1. When k is less than or equal to n, n-k redundant bits can be added to each message to form a code word. These redundant bits provide the code with the capability of offsetting the effects of noise. For a fixed

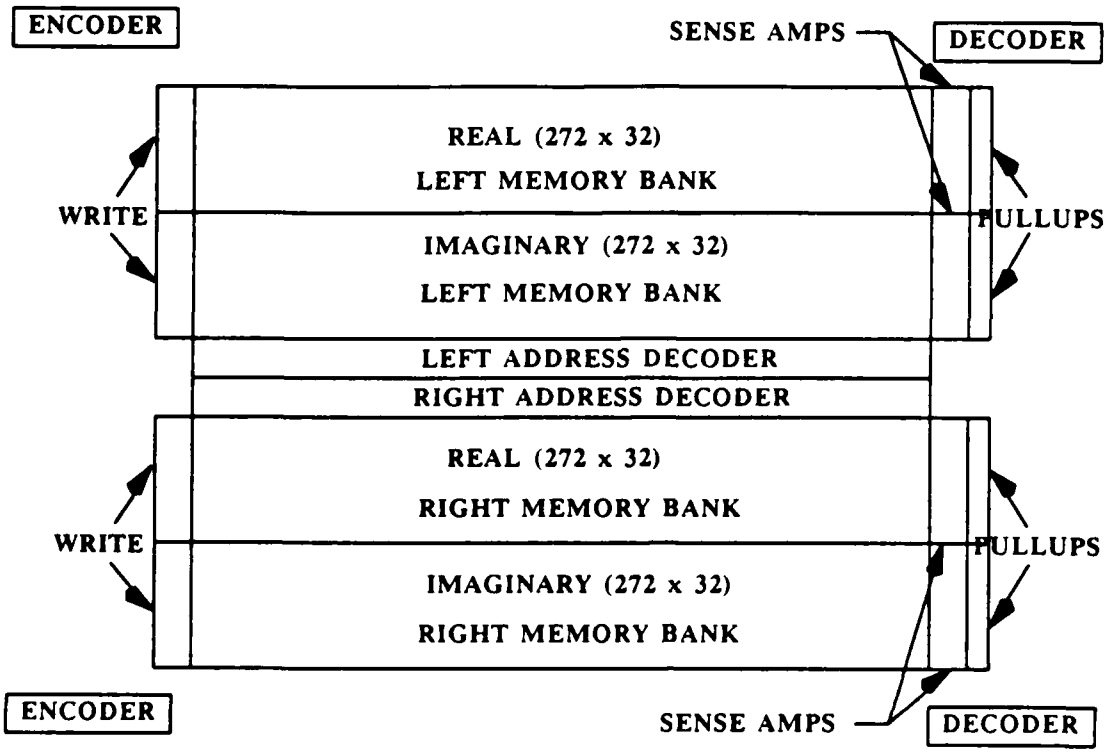


Figure 2-6 MEMORY PARTITIONED - READ WRITE

code rate R, more redundant bits can be added by increasing the block length n of the code while holding the ratio k/n constant.

In a block-coded system, the decoder output "U'" represents the k-bit estimate of the encoded message. The decoder must produce an estimate "U'" of the information sequence "U" based on the received sequence "R". For ease of code synthesis and implementation, attention should be restricted to a subclass of all the block codes, the linear block codes. Linear block codes are defined and described in terms of generator and parity-check matrices. By definition, a block code of length n and 2^k code words is

called a linear (n,k) code if and only if its 2^k code words form a k-dimensional subspace of the vector space of all the n-tuples over the Galois field GF(2).

Any k linearly independent code words of an (n,k) linear code can be used to form a generator matrix "G" for the code. Since an (n,k) linear code is completely specified by the k rows of a generator matrix "G", the encoder has only to store the k rows of "G" and to form a linear combination of these k rows based on the input message "U".

A desirable property for a linear block code to possess is a systematic structure, where a code word is divided into two parts, the message part and the redundant checking part. The message part consists of k unaltered information digits and the redundant checking part consists of n-k parity- check digits, which are linear sums of the information bits. A linear block code with this structure is referred to as a linear systematic code. A linear systematic (n,k) code is completely specified by a k x n matrix "G" shown in Figure 2-7, where Pij=0 or 1. "Ik" denotes the k x k identity matrix. Then "G" = ("P" * "Ik"). Furthermore, there exists an (n-k) x n matrix H (parity-check matrix) such that an n-tuple "V" is a code word if and only if "V" * "Ht"

$$G = \begin{bmatrix} g_0 \\ g_1 \\ g_2 \\ \vdots \\ g_{k-1} \end{bmatrix} = \left[\begin{array}{ccc|ccccc} p_{0,0} & p_{0,1} & \cdots & p_{0,n-k-1} & 1 & 0 & 0 & \cdots & 0 \\ p_{1,0} & p_{1,1} & \cdots & p_{1,n-k-1} & 0 & 1 & 0 & \cdots & 0 \\ p_{2,0} & p_{2,1} & \cdots & p_{2,n-k-1} & 0 & 0 & 1 & \cdots & 0 \\ & & & & & & & & \\ p_{k-1,0} & p_{k-1,1} & \cdots & p_{k-1,n-k-1} & 0 & 0 & 0 & \cdots & 1 \end{array}\right]$$

"P" Matrix — k x k identity matrix

Figure 2-7 G MATRIX

= 0. The parity-check matrix "H" is equal to ("In - k" * Pt) shown in Figure 2-8.

Let "R" be the received vector at the output of the memory. Because of noise, "R" may be different from "V". The vector sum "E" = "R" + "V" is an N-tuple where "E" = 1 for Ri not equal Ni and Ei=0 for Ri=Ni. This N-tuple is called the error vector. The ones in "E" are the transmission errors caused by the noise. Upon receiving "R", the decoder must determine whether "R" contains transmission errors. When "R" is received, the decoder computes the (n-k)-tuple S=R*Ht, which is called the syndrome of "R". The syndrome digits are as follows:

$$S0 = R0 + Rn\text{-}k\ P0,0 + Rn\text{-}k\text{+}1\ P1,0 + \ldots + Rn\text{-}1\ Pk\text{-}1,0$$

$$S1 = R1 + Rn\text{-}k\ P0,1 + Rn\text{-}k\text{+}1\ P1,1 + \ldots + Rn\text{-}1\ Pk\text{-}1,1$$

*

*

*

$$H = \left[\begin{array}{ccccc|ccc} 1 & 0 & 0 & \ldots & 0 & p0,0 & p1,0 & \ldots\ pk\text{-}1,0 \\ 0 & 1 & 0 & \ldots & 0 & p0,1 & p1,1 & \ldots\ pk\text{-}1,1 \\ 0 & 0 & 1 & \ldots & 0 & p0,2 & p1,2 & \ldots\ pk\text{-}1,2 \\ & & & & & & & \\ 0 & 0 & 0 & \ldots & 1 & p0,n\text{-}k\text{-}1 & p1,n\text{-}k\text{-}1 & \ldots\ pk\text{-}1,n\text{-}k\text{-}1 \end{array}\right]$$

n-k x n-k identity matrix — Transpose of "P" Matrix

Figure 2-8 H MATRIX

$$Sn\text{-}k\text{-}1 = Rn\text{-}k\text{-}1 + Rn\text{-}k\ P0,n\text{-}k\text{-}1 + Rn\text{-}k\text{-}1\ P1,n\text{-}k\text{-}1 + \ldots + Rn\text{-}1\ Pk\text{-}1,n\text{-}k\text{-}1$$

Then "S" = 0 if and only if "R" is a code word, and "S" not equal 0 if and only if "R" is not a code word. It is possible that the errors in certain error vectors are not detectable (i.e., "R" contains errors but "S" = "R" * "Ht" = 0). This happens when the error pattern "E" is identical to a nonzero code word. Erro. patterns of this kind are called undetectable error patterns. When an undetectable error pattern occurs, the decoder makes a decoding error. The syndrome "S" computed from the received vector "R" actually depends only on the error pattern "E" and not on the transmitted code word "V".

$$S = R * Ht = (V + E)\ Ht = V * Ht + E * Ht$$

$$V * Ht = 0$$

$$S = E * Ht$$

Determining the true error vector "E" is not a simple matter. There are 2^k error patterns that result in the same syndrome, and the true error pattern "E" is just one of them. To minimize the probability of a decoding error, the most probable error pattern is the one that has the smallest number of nonzero digits.

The decoding scheme is called syndrome decoding or table-lookup functions:

$$E0 = F0(S0,S1, \ldots ,Sn\text{-}k\text{-}1),$$

$$E1 = F1(S0,S1, \ldots ,Sn\text{-}k\text{-}1),$$

$$*$$

$$*$$

$$En\text{-}1 = Fn\text{-}1(S0,S1, \ldots ,Sn\text{-}k\text{-}1),$$

where S0, S1, ... , Sn-k-1 are the syndrome bits, which are regarded as switching variables, and E0, E1, ... , En-1 are the estimated error digits. When these n switching functions are derived and simplified, a combinational logic circuit with the n-k syn-

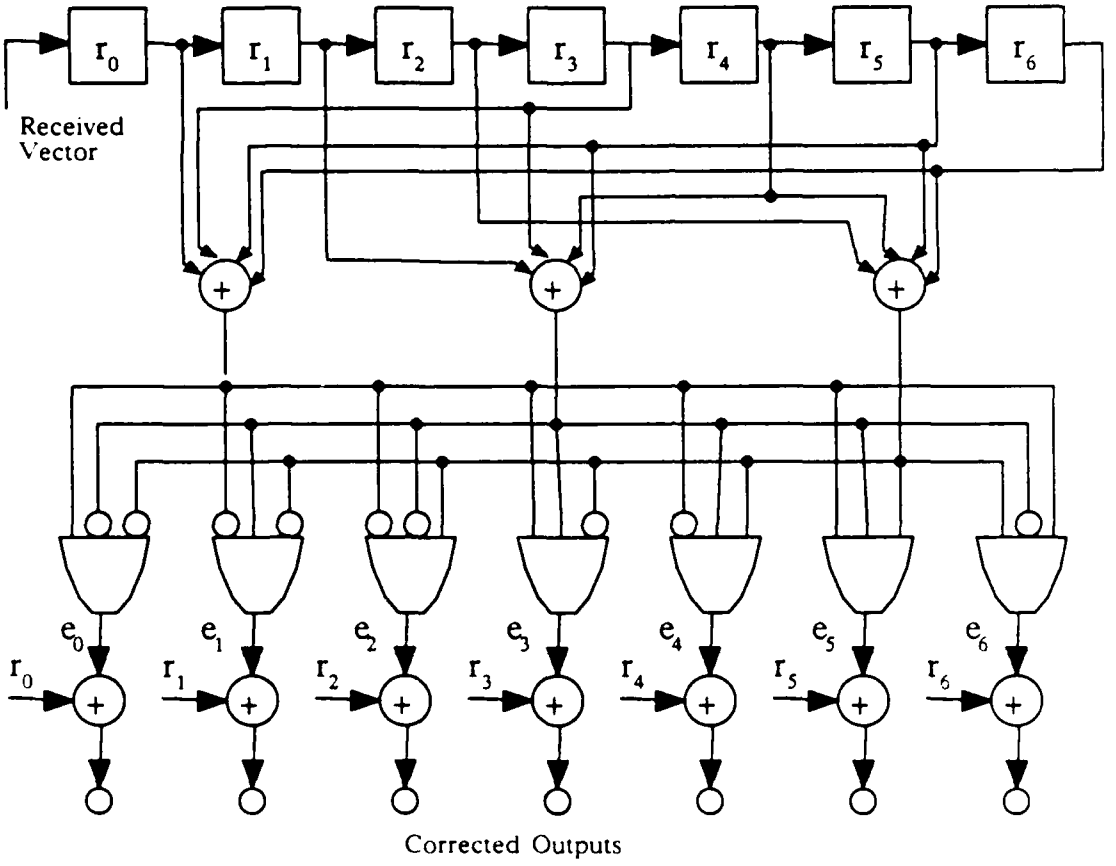


Figure 2-9 DECODER

drome digits as inputs and the estimated error digits as outputs can be obtained. The switching expression for the error digits is the ANDing of the syndrome digits (or the complement of the syndrome digits). A conceptual representation of the decoder is shown in Figure 2-9. Error control codes are discussed in detail by Lin and Costello [Li83].

CHAPTER 3

ARCHITECTURE AND ALGORITHMS

3.1. PFA Controller

In chapter 2, several design methods were presented for implementing control circuitry for the PFA controller (PFAC). These design methods included custom logic implementations, gate array implementations, and programmable logic arrays (PLA) or ROMs. Each method has its advantages and disadvantages. The goal is to use the implementation that will provide the PFAC with flexibility, extensibility of the design, and adequate speed of operation. The ROM implementation possesses an advantage over the other methods in flexibility and extensibility of the design. Although the PFA pipeline processors operate at clock rates of over 60 MHz, the PFAC can operate with clock rates of 20 MHz and still keep the pipeline operating with results every 120 microseconds. The PFAC design, as shown in Figure 3-1, is a Reduced Instruction Set Computer (RISC).

The RISC architecture is simple both in the instruction set and the hardware needed to implement that instruction set. The thrust of the RISC design is towards efficient implementation of a straightforward instruction set. The major components of the design are a processor unit and a microprogrammed control unit.

3.1.1. **Processor Unit.** The processor unit is shown in Figure 3-2. The processor unit consists of three busses, an arithmetic logic unit (ALU), twenty-eight processor registers, and bus selection circuitry. The three busses are 16 bits wide. The A and B busses are connected to the outputs of the registers. The B bus is also connected to 16 inputs of the branch-test multiplexer in the microprogrammed control unit. The A and B busses form the inputs to the ALU. The function selection lines select the arithmetic

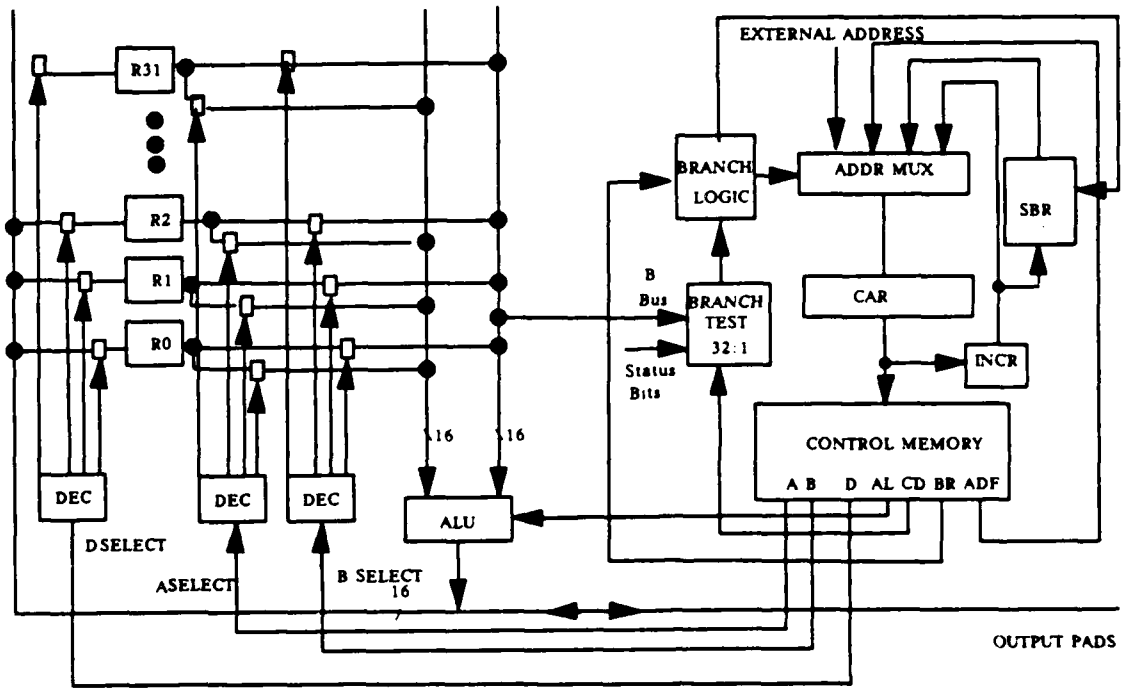


Figure 3-1 PFA CONTROLLER

or logic micro-operation that is to be performed by the ALU. The output of the ALU is connected to the D bus. The D bus is connected to the inputs of the registers and to output pads to output information. The bus selection circuitry consists of a 5 to 28 decoder for each bus. The processor registers vary in design according to the use of each register. Some registers simply store numbers, other registers have data input directly with various signals, and one register does a special five bit shift. These registers will be discussed, in detail, later in this chapter.

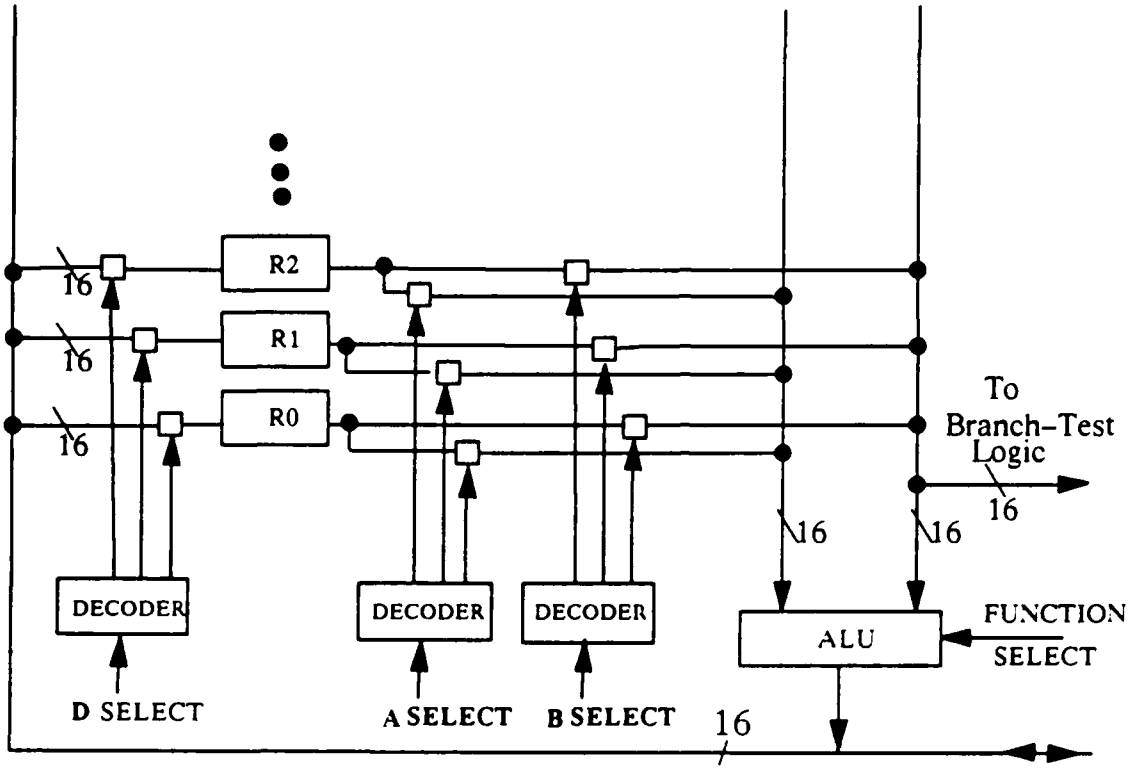


Figure 3-2 PROCESSOR UNIT

3.1.2. MICROPROGRAMMED CONTROL UNIT.

The control unit performs a series of sequential micro-operations.

The control memory operates as a combinational circuit with the address as the input and the corresponding word as the output. The word read from the control memory is the control word. The control word represents a microinstruction. The microinstruction has bits that specify one or more micro-operations and also has bits that determine the next address of the control memory. The bits of the microinstruction are divided into parts called fields. Each field defines a distinct, separable function.

The CPU is organized around a bus system. A bus organized processor can be controlled with vertical microinstructions. The decoders for the various fields are placed in the bus system. The microinstruction that controls this bus organized processor unit can be divided into four fields, with each field supplying the control bits to select the function in each component. The control word format for controlling the bus system is shown in Figure 3-3. Two fields A and B specify the source registers for the A and B input busses to the ALU. The ALU field specifies one of 12 ALU function. The D field designates the destination register for storing the results of the ALU operations. Since there are twenty-eight registers, the A, B, and D fields must contain five bits.

Therefore, the ALU field contains four bits, The SPEC field uses a horizontal format. Each bit of the field is used to generate a control signal. The specific names of these bits and the functions of the specific control signals will be discussed later in this chapter.

The rest of the fields of microinstruction are used by the address sequencer to determine the address of the next microinstruction to be executed. The CD field selects one of the status bits in the branch-test multiplexers as the condition bit. Since there are thirty-two inputs to the branch-test multiplexer, the CD field has five bits The BR field is three bits wide and specifies one of five operations for the sequencer. The ADF field is nine bits long since the control memory has 256 words.

The microprogrammed control unit consists of two parts: the control memory that stores the microinstructions and the associated circuits that control the generation of the next address. The address generation part is called a microprogram sequencer The block diagram of the microprogrammed control unit is shown in Figure 3-4 The purpose of a microprogram sequencer is to present an address to the control memory so that a microinstruction may be read and executed The next address logic of the sequencer determines the specific address to be loaded into the control address register

(CAR). The choice of the address source is guided by the next address information bits that the sequencer receives from the present microinstruction. The CAR sends the selected address to the control memory and to the incrementer. The output of the incrementer is applied to one of the address multiplexer inputs and to the subroutine return register (SBR). The SBR is a last-in, first-out stack. The output from SBR is applied to another of the address multiplexer inputs. The two other inputs to the address multiplexer come from the address field of the present microinstruction and from the external address circuit. The five condition field bits select one of 32 status bits in the branch-test multiplexer.

The branch-test multiplexer tests the value of the selected bit and the result is applied to the branch logic circuit. The branch logic circuit has four inputs and four outputs. The outputs of the branch logic select a sequencer operation and increment/decrement the stack pointer of the SBR. The truth table for the branch logic circuit is shown below.

I2	I1	I0	T	S1	S0	OPERATION	NOTES
X	0	0	X	0	0	CAR<-EAD	BRANCH TO EXT ADDR
X	0	1	X	0	1	CAR<-SBR	RETURN FROM SUBR
X	1	1	X	1	0	CAR<-CAR+1	INCREMENT ADDR
0	1	1	0	1	0	CAR<-CAR+1	INCREMENT ADDR
0	1	1	1	1	1	CAR<-ADF	BRANCH TO ADF ADDR
1	1	1	0	1	0	CAR<-CAR+1	INCREMENT ADDR
1	1	1	1	1	1	CAR<-ADF,SBR<-CAR+1	BRANCH TO ADF ADDR

T and I2 are don't cares unless (I1 I0) = (1 1). For (I1 I0) = (0 0), an external address is selected. For (I1 I0) = (0 1), the address comes from the SBR (return from subroutine). For (I1 I0) = (1 0), the address comes from the incrementer. When (I2 I1 I0) = (0 1 1), the sequencer executes a conditional branch if T = 1, otherwise the CAR is incremented. When (I2 I1 I0) = (1 1 1), the sequencer executes a conditional subroutine call. A conditional branch and a conditional subroutine call both select the address from the ADF field of the microinstruction. However, the subroutine

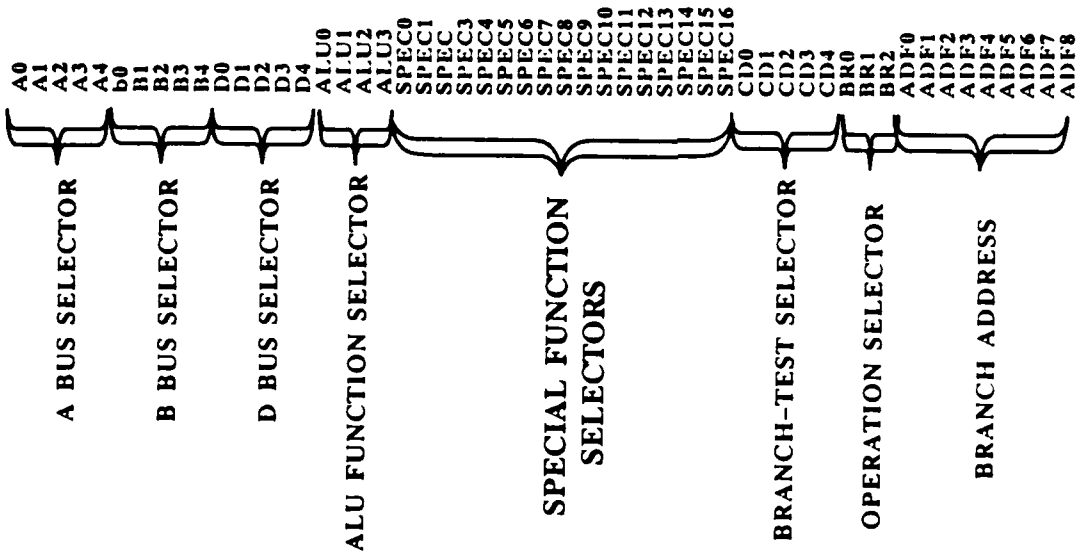


Figure 3-3 CONTROL WORD FORMAT

call stores the address from the incrementer in SBR. This is the address next in sequence, and it is called the return address. The return is transferred to the CAR with a subroutine return operation.

The simplified Boolean functions for the branch logic circuit are :
SI = I1
SO = I0 * (T + I1')
PUSH = I2 I1 I0 T
POP = I1' * IO

3.1.3. **Data Flow Control.**

In Chapter 2, the control functions of the PFAC were divided into data flow control and configuration control. The signals necessary for data flow control were specified in Chapter 2. The master OPERATE and RIGHT LEFT signals are gen-

erated as outputs of the control memory. The DONE signals are input to an NAND gate. The output of the NAND gate is applied to an input of the branch-test multiplexer (ALL DONE). The block floating point exponents for the three problems in the PFA pipeline are stored in the temporary scale register (Latch Scale Temporary, LST). The block floating point exponent from the WFTA processors for each problem in the PFA pipeline is accumulated by adding the contents of the temporary scale register to the contents of the permanent scale register. Next, the block floating point exponent of the leading problem is latched into the output problem status register, (Latch Scale Permanent, LSP). Then, the contents of the permanent scale register are shifted right five bits (Shift Register Mode, SRI, SRO). The block floating point exponent which was stored in the temporary scale register must be sent to the WFTA processors (STATE/SCALE, SSC, DRIVE LOAD SIGNALS, DLS).

3.1.4. **Configuration Control.**

The configuration control section, as stated in Chapter 2, is concerned with fault detection, fault correction, and system recovery. The flow chart for PFAC operation with configuration control is shown in Figure 3-8. All error signals from the PFA pipeline modules are latched into the pipeline status register, (Latch Pipeline Status, LPS). The pipeline status register's contents are checked for the presence of reported errors. The contents of the pipeline status register are applied to the inputs of a combinational logic voting circuit to determine which module should be assigned any errors reported. If no errors are reported then the PFAC continues operation of the PFA pipeline. If there were errors the results of the combinational logic voting circuit are latched into the error location register. The contents of the configuration register are AND'ed with the contents of the error location register. If the results are not equal to zero then the error was assigned to an active processor and the pipeline must be reconfigured. Otherwise, the errors are just added to the register for the modules that were assigned an error.

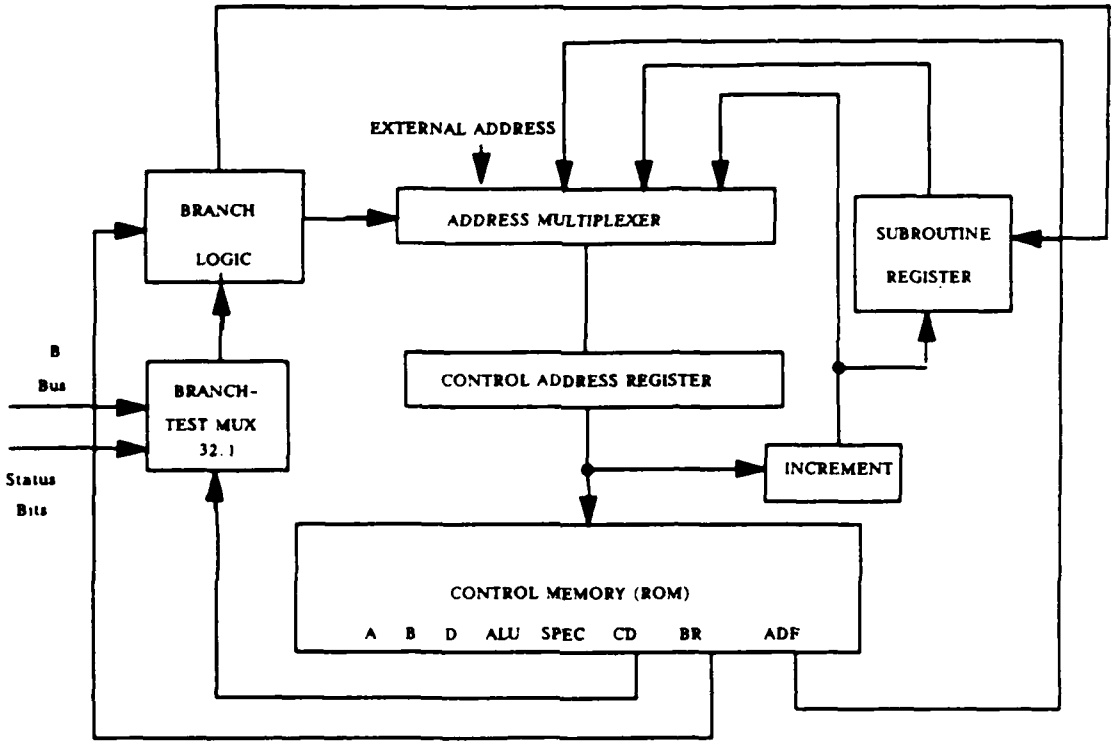


Figure 3-4 MICROPROGRAMMED CONTROL UNIT

Registers R0 thru R17 store the error count for each module. If the pipeline has to be reconfigured then errors are still added to the registers for the modules assigned an error. The reconfiguration process is specified in the microprogram as a sequence of examining each group of bits for each column and comparing the error count for the watchdog processors. The watchdog processor with the smallest number of errors is assigned the role of active. If both watchdog processors have the same number of errors then the watchdog processor with the lowest position designator (row 1 is lower than row 2) is assigned the role of active. The health register is used as a mask to determine if a processor is to be turned off. The health register is set by the host. The

configuration register determines which processors are active and which processors are watchdogs. The contents of the configuration is latched onto the scale bus. (STATE/SCALE, SSC; DRIVE LOAD SIGNALS, DLS; WATCHDOG/ACTIVE LOAD. WAL).

There are four registers that record any errors assigned to each problem in the pipeline. This is accomplished by latching signal from column 1 into Problem 1 Status Register, from column 2 into Problem 2 Status Register, and from column 3 into Problem 3 Status Register. The contents of the problem status registers are then transferred (Problem 3 Status Register) > Output problem Status register), (Problem 2 Status Register > Problem 3 Status Register), (Problem 1 Status Register > Problem 2 Status Register).

3.1.5. **Design for Testability.**

Because the complexity of circuits, boards, and systems is increasing, testing has become more expensive and more difficult. One approach to reducing the difficulty and cost of testing is to consider the testability of a circuit early in the design cycle and try to improve a circuit testability. Testability is defined as a measure of the ability to test easily or cost-effectively. Controllability and observability are two measures of testability. Controllability is a measure of how easily the internal logic of a circuit can be controlled from its inputs. Observability is a measure of how easily the internal logic of a circuit can be observed at it outputs. There are two design for testability techniques that can be implemented to test the PFA controller, partitioning, and the use of bus architecture systems. Both approaches use the divide and conquer approach.

Partitioning may be accomplished by using degating techniques. With degating individual components are isolated so that test patterns can be input and output to specific components.

The chip can be operated in normal mode or test mode. A TEST signal can control the mode of the chip. All the input pads drive a test input in addition to a functional

output and the output pads have a 2:1 multiplexer on their inputs. By controlling these multiplexers with the TEST signal most of the pads can perform a dual role in normal and test mode. Additional test circuitry can be isolated from the normal operation circuitry by the use of transmission gates. The registers can be converted to shift registers and also isolated from other components. When the test signal is high, test data can be propagated through the registers. The testing costs for this method include the area required for the signal routing and for the multiplexers.

The second design for testability technique is facilitated by the use of a bus-structured architecture. The host can access all the registers using the busses and isolate each register from the busses. The busses make it possible to test the registers by applying test patterns to each register separately. When the chip is in test mode the start address for the control memory can be incremented to select control words specifically designed to test CPU circuitry. The testing costs of this method include the additional memory and extra bits in the address sequencing circuits.

3.2. Memory

3.2.1. Introduction.

Chapter 1 examined the various technologies used to implement random access memories and presented error control codes as a method of error detection and control. In chapter 2, the requirements for memory capacity and control were presented. Approaches and trade-offs were also highlighted. Since we need high performance random access memories with low power consumption and high noise immunity, the memories are being implemented in CMOS technology using a six transistor SRAM cell with a linear block error control code for error detection and correction. A floor plan of

a PFA pipeline memory is shown in Figure 3-5. Its key sections are the RAM array, the read circuitry (pull ups and sense amplifiers), the write circuitry, the error detection and correction circuitry (encoder and decoder), the address selection circuitry, and the

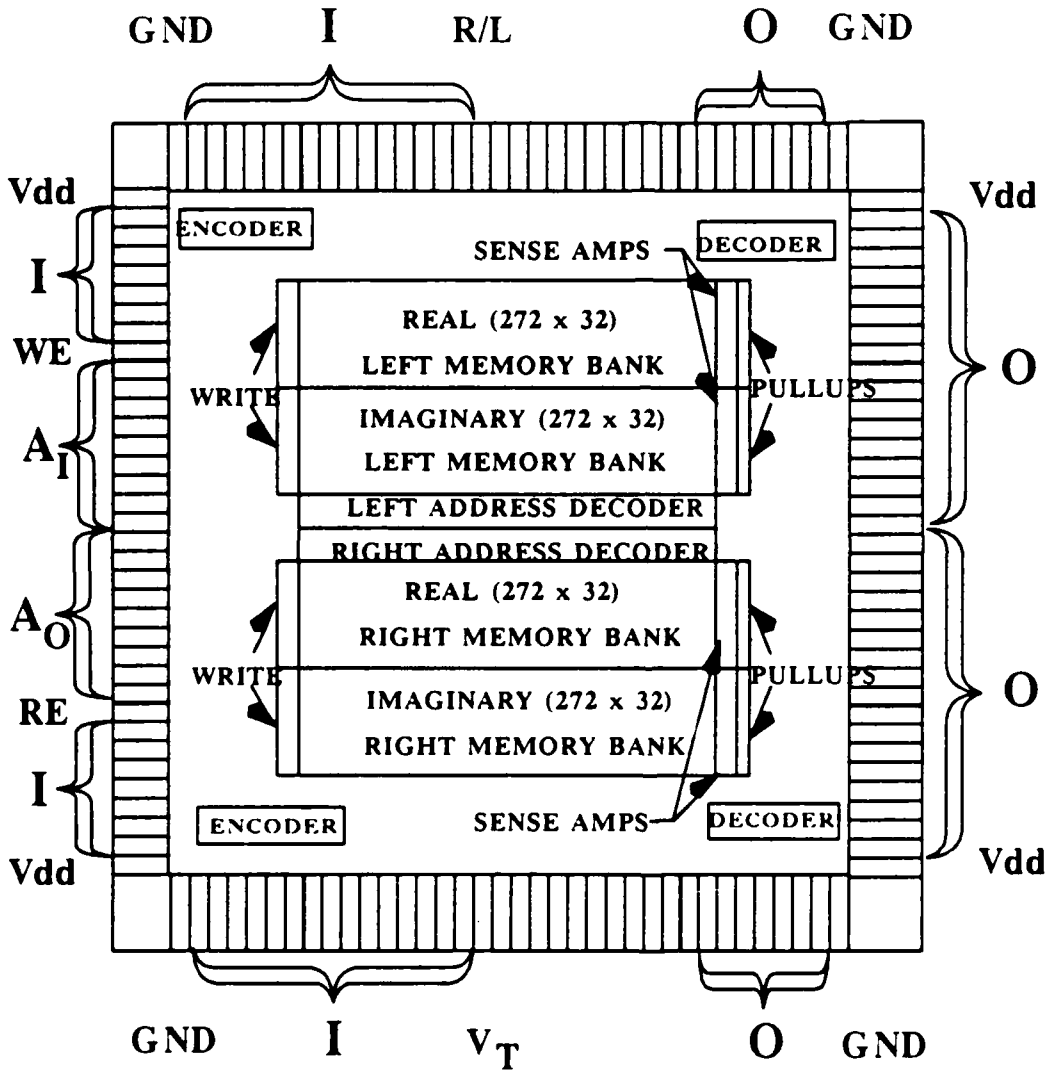


Figure 3-5 MEMORY FLOORPLAN

switching circuitry (the switching circuitry is embedded in various parts of the memory). A block diagram showing a RAM cell with associated read and write circuitry is shown in Figure 3-6.

3.2.2. **RAM**

A six-transistor CMOS static RAM, shown in Figure 2-3 is used for data storage for the PFA pipeline memory. The SRAM was selected because SRAMs tend to be faster than DRAMS, easier to design than DRAMS, and are potentially less troublesome than DRAMs [We85]. The design presented here is a "safe" design that is fairly tolerant to process variations but it is not the fastest that can be designed. However, high speeds can be achieved in large arrays because the output is a differential signal. A differential output signal eliminates a whole class of common mode noise sources which enables the use of a sense amplifier with lower noise margin. Lower noise margins permit higher speeds. The circuit consists of a pair of cross-coupled inverters connected by pass transistors to BITLINE and BITLINE'. A pair of cross-coupled inverters are the simplest form of static latch. In the absence of noise, a static latch will remember a state for as long as power is applied to the circuit. The static latch has elements with power gain that continually counteract the effect of charge leakage at the drains of the pull-down and pass transistors. These elements with power gain are the p-load transistors.

3.2.3. **Read Circuitry.**

To read from the memory cell, the pull-up circuitry is used with a sense amplifier Both Bitline and Bitline' are precharged to Vdd by turning transistors P4 and P5 on To ensure that Bitline and Bitline' are at the same potential, transistor P3 is turned on When P3 is turned on Bitline is shorted, through P3, to Bitline'. These three transistors are only turned on when PRE and R/W are true. Since the memory cell generates a differential output signal, a differential sense amplifier may be employed detect the

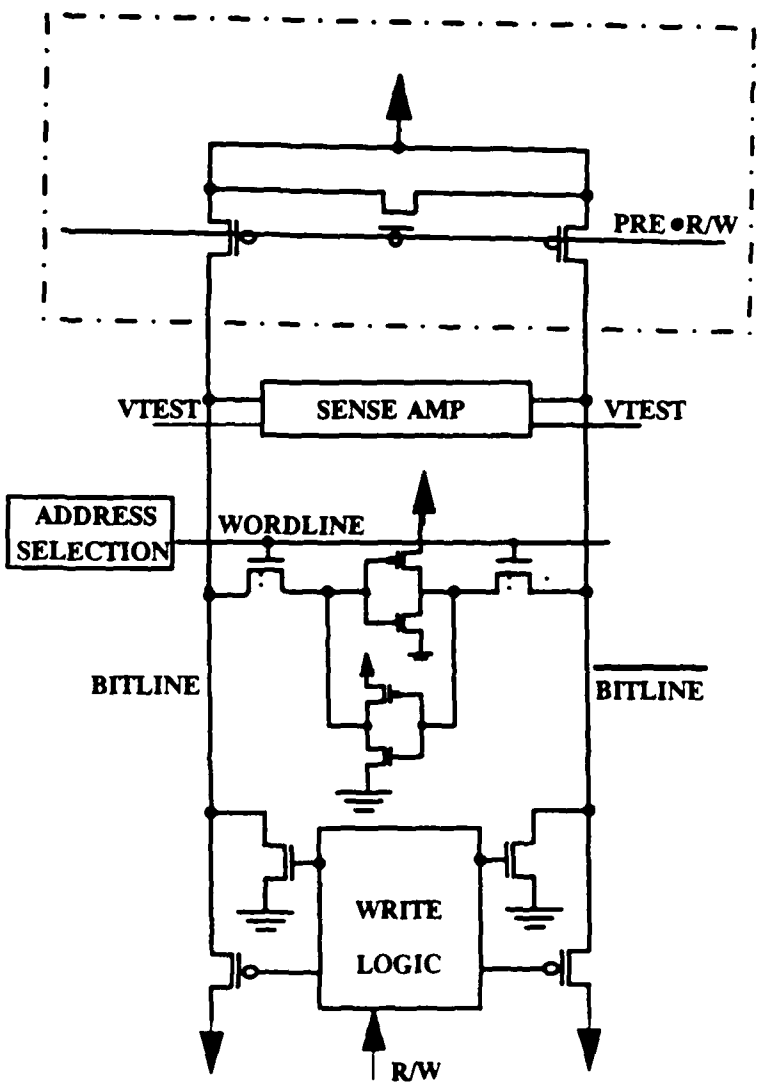


Figure 3-5 RAM CELL WITH READ AND WRITE CIRCUITRY

output signal. A differential sense amplifier senses a small difference between levels on the Bitline and Bitline' lines and amplifies this difference to provide very fast sensing

Thus the Bitline lines have to only change slightly in level to detect the state of the memory cell. A two stage amplifier gives a greater voltage gain, providing faster detection than a single stage differential sense amplifier. The theory of operation of the two

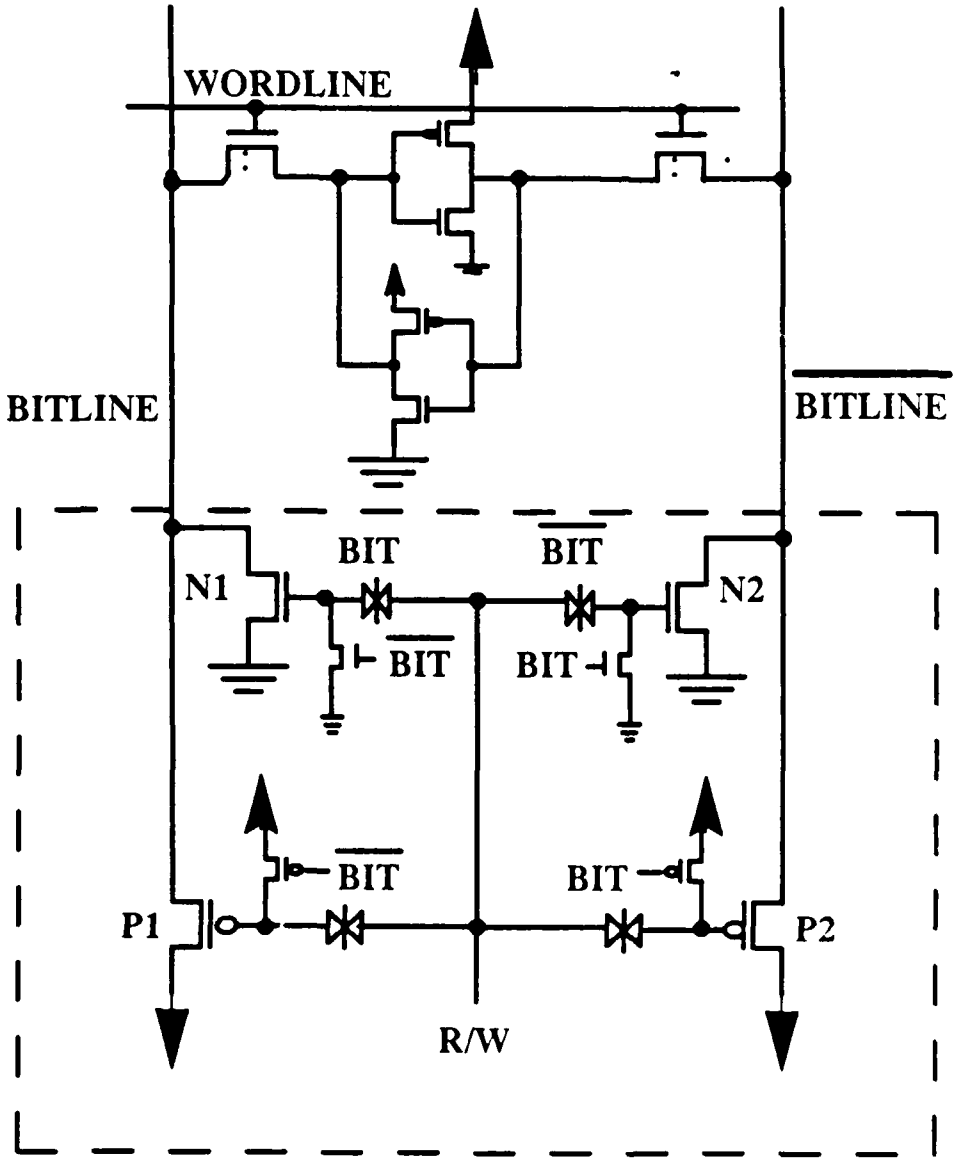


Figure 3-7 WRITE CIRCUITRY

stage sense amplifier is discussed in detail by Soclof [So85].

3.2.4. **Write Circuitry.**

Four transistors (P1,NI, P2, N2), controlled by simple logic circuitry, are used to write to the memory cell, as shown in Figure 3-7. Writing a "one" into the memory cell is accomplished by raising the word line after precharging Bitline to Vdd and discharging Bitline' to ground. Writing a "zero" into the memory cell is the same as writing a "one" except Bitline' is precharged to Vdd and Bitline is discharged to ground. The input signals gP1, gN1, gP2, and gN2 are the gate signals for transistors P1, N1, P2, and N2 respectively. During a read operation the write circuitry does not affect the bit lines.

The logic equations for the gate signals are :

gP1=(R/W'*BIT')+R/W
gN1=(R/W'*BIT')
gP2=(R/W'*BIT)+R/W
gN2=(R/W'*BIT)

3.2.5. **Address Selection Circuitry.**

The address selection circuitry includes a NAND PLA and wordline drivers. As shown in Figure 3-8, the NAND PLA decodes the 9-bit address input and a wordline driver raises the wordline that corresponds to that address. The address input is broadcast to every address in the PLA. The decoder's output to the wordline is pulled low when it is the selected wordline. The decoder's output is input to the RAM array through a two-input NOR gate, followed by two inverters. The wordline driver output to the wordline is pulled high it is the selected wordline. The PRE signal is input to the two-input NOR gate to disable wordline selection during precharge.

3.2.6. **Error Detection and Correction Circuitry.**

The error detection and correction circuitry consists of the encoder and the decoder. In chapter 2, we discussed error control codes were discussed. The linear

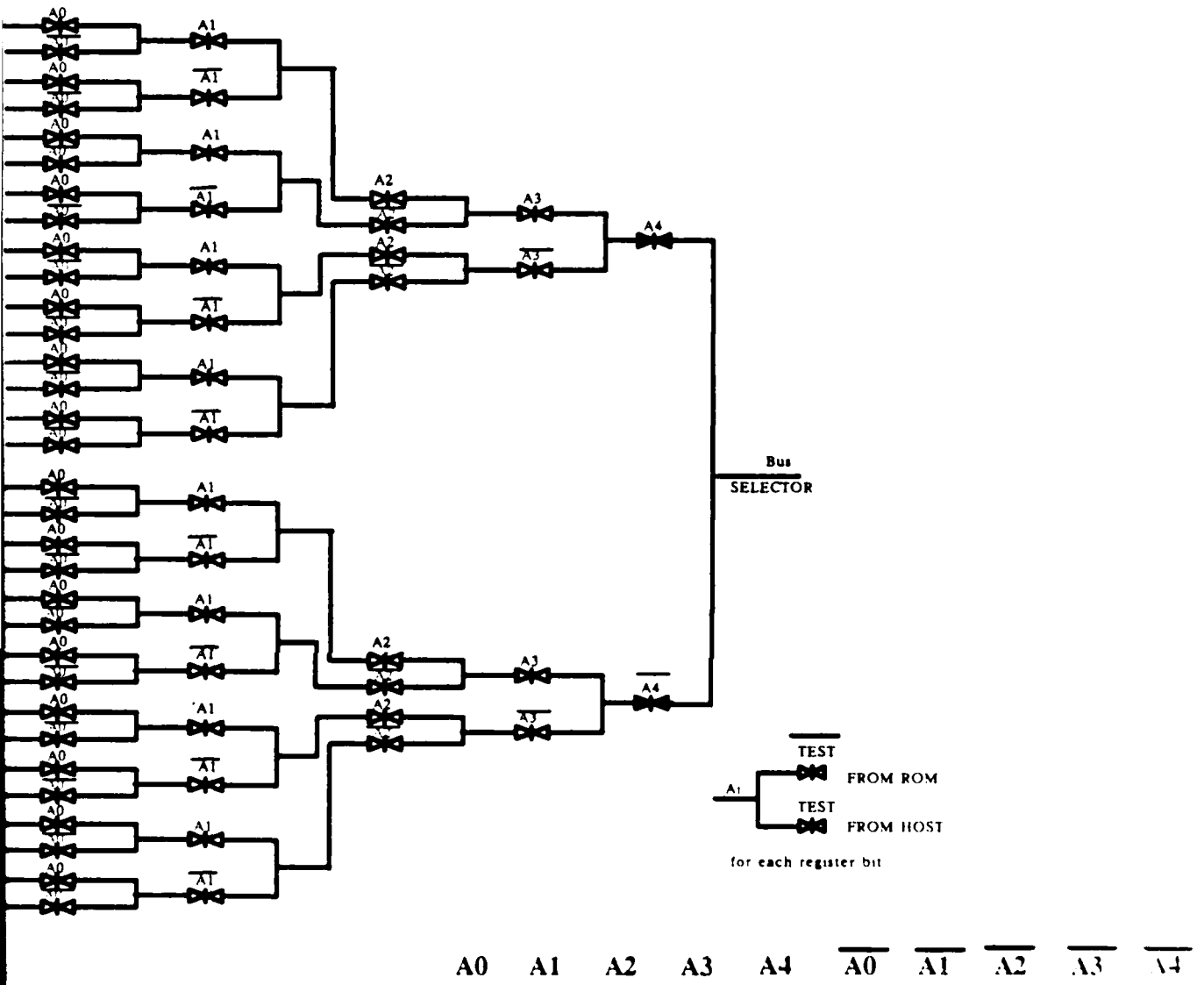


Figure 3-8 ADDRESS SELECTION CIRCUITRY

block code discussed will fulfill the error detection and correction requirements of single error correction and double error detection. The parity-check matrix P for a linear

block code is shown in Figure 3-9. Each parity bit can be generated with the circuit shown in Figure 3-10. All data bits are only used twice and the longest path is three levels of exclusive-ORs.

The parity check portion of the systematic is:

P MATRIX

	V0	V1	V2	V3	V4	V5	V6	V7
U0	1	1	0	0	0	0	0	0
U1	0	1	1	0	0	0	0	0
U2	0	0	1	1	0	0	0	0
U3	0	0	0	1	1	0	0	0
U4	0	0	0	0	1	1	0	0
U5	0	0	0	0	0	1	1	0
U6	0	0	0	0	0	0	1	1
U7	1	0	0	0	0	0	0	1
U8	1	0	1	0	0	0	0	0
U9	0	1	0	1	0	0	0	0
U10	0	0	1	0	1	0	0	0
U11	0	0	0	1	0	1	0	0
U12	0	0	0	0	1	0	1	0
U13	0	0	0	0	0	1	0	1
U14	1	0	0	0	0	0	1	0
U15	0	1	0	0	0	0	0	1
U16	1	0	0	1	0	0	0	0
U17	0	1	0	0	1	0	0	0
U18	0	0	1	0	0	1	0	0
U19	0	0	0	1	0	0	1	0
U20	0	0	0	0	1	0	0	1
U21	1	0	0	0	0	1	0	0
U22	0	1	0	0	0	0	1	0
U23	0	0	1	0	0	0	0	1

K rows

N-K columns

Encoder equations for the parity bits are:

V0 = U0 + U7 + U8 + U14 + U16 + U21

V1 = U0 + U1 + U9 + U15 + U17 + U22

V2 = U1 + U2 + U8 + U10 + U13 + U23

V3 = U2 + U3 + U9 + U11 + U16 + U19

V4 = U3 + U4 + U10 + U12 + U17 + U20

V5 = U4 + U5 + U11 + U13 + U18 + U21

V6 = U5 + U6 + U12 + U14 + U19 + U22

V7 = U6 + U7 + U13 + U15 + U20 + U23

1. 40 GATES REQUIRED
2. THREE LEVELS OF TWO INPUT EXCLUSIVE ORs
3. ALL DATA BITS ONLY USED TWICE

Figure 3-9 PARITY CHECK MATRIX P

The decoder consists of a syndrome calculation circuit and an error pattern detection circuit. The matrix for generating syndrome digits is shown in Figure 3-11. Each syndrome digits can be generated with the circuit shown in Figure 3-12. All the data bits are only used twice and the longest path is three levels of exclusive-ORs.

The matrix for generating the error bits is shown in Figure 3-13. The error bits are generated as outputs of a PLA. Each error bit is the product of the syndrome digits. For a "one" in the matrix AND the syndrome digit. For a "zero" in the matrix AND the inverse of the syndrome digit. The PLA has eight inputs and 24 outputs (we don't need the error bits for the parity bits). The outputs of the PLA are combined with the data bits output directly from memory. The outputs of the exclusive-ORs are the corrected output.

3.2.7. **Switching circuitry.**

The switching circuitry includes circuits for switching address selection signals as well as switching data signals. The address selection line switching is accomplished at

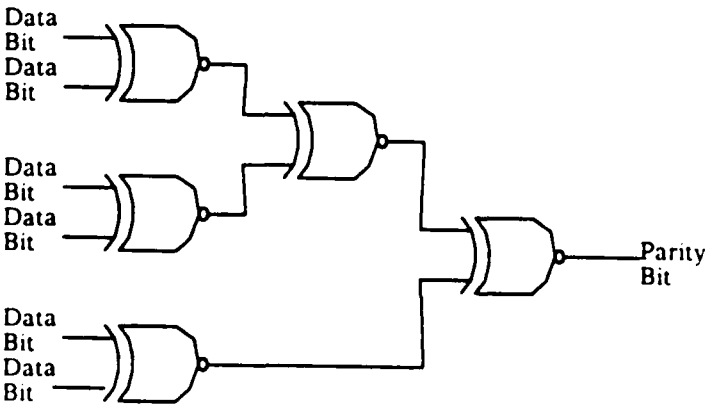


Figure 3-10 ENCODER CIRCUITRY

	S0	S1	S2	S3	S4	S5	S6	S7
R0	1	0	0	0	0	0	0	0
R1	0	1	0	0	0	0	0	0
R2	0	0	1	0	0	0	0	0
R3	0	0	0	1	0	0	0	0
R4	0	0	0	0	1	0	0	0
R5	0	0	0	0	0	1	0	0
R6	0	0	0	0	0	0	1	0
R7	0	0	0	0	0	0	0	1
R8	1	1	0	0	0	0	0	0
R9	0	1	1	0	0	0	0	0
R10	0	0	1	1	0	0	0	0
R11	0	0	0	1	1	0	0	0
R12	0	0	0	0	1	1	0	0
R13	0	0	0	0	0	1	1	0
R14	0	0	0	0	0	0	1	1
R15	1	0	0	0	0	0	0	1
R16	1	0	1	0	0	0	0	0
R17	0	1	0	1	0	0	0	0
R18	0	0	1	0	1	0	0	0
R19	0	0	0	1	0	1	0	0
R20	0	0	0	0	1	0	1	0
R21	0	0	0	0	0	1	0	1
R22	1	0	0	0	0	0	1	0
R23	0	1	0	0	0	0	0	1
R24	1	0	0	1	0	0	0	0
R25	0	1	0	0	1	0	0	0
R26	0	0	1	0	0	1	0	0
R27	0	0	0	1	0	0	1	0
R28	0	0	0	0	1	0	0	1
R29	1	0	0	0	0	1	0	0
R30	0	1	0	0	0	0	1	0
R31	0	0	1	0	0	0	0	1

N-K columns

SYNDROME DIGITS ARE :

S0 = R0 + R8 + R15 + R16 + R22 + R24 + R29

S1 = R1 + R8 + R9 + R17 + R23 + R25 + R30

S2 = R2 + R9 + R10 + R16 + R18 + R26 + R21

S3 = R3 + R10 + R11 + R17 + R19 + R24 + R27

S4 = R4 + R11 + R12 + R18 + R20 + R25 + R28

S5 = R5 + R12 + R13 + R19 + R21 + R26 + R29

S6 = R6 + R13 + R14 + R20 + R22 + R27 + R30

S7 = R7 + R14 + R15 + R21 + R23 + R28 + R31

N ROWS

Figure 3-11 MATRIX FOR SYNDROME BIT GENERATION

the address input pads. There are nine input pads for the read address and nine input pads for the write address. Each input pad has a 1:2 multiplexer. One multiplexer out-

put goes to the appropriate address line in the left decoder and the other output goes to the corresponding address line in the right decoder. The output selection is controlled by the RIGHT/LEFT signal. RIGHT/LEFT = 1, selects read right bank and write left bank.

The input data lines are not switched. The input data is sent to the input data bus. The input data bus is connected to the inputs of the encoder and to the inputs of the write circuitry of the right and the write circuitry of the left memory bank. However, only the write circuitry of the memory bank selected to write will utilize the input data.

The output data is placed on an output data bus. The outputs of the sense amplifiers are connected to the output data bus with tri-state drivers. The outputs of the sense amplifiers of the memory bank not selected to read are put into a high impedance state.

The signal RIGHT/LEFT is used to derive the signal READ/WRITE. For the right memory bank READ/WRITE is the same as RIGHT/LEFT but for the left

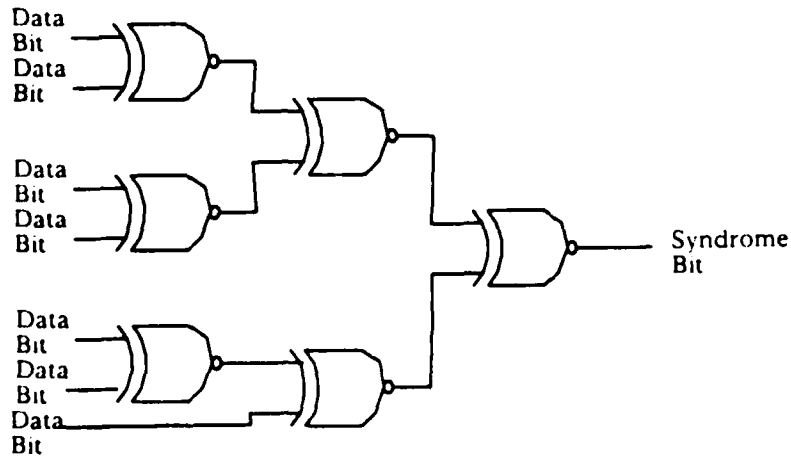


Figure 3-12 SYNDROME BIT GENERATION CIRCUITRY

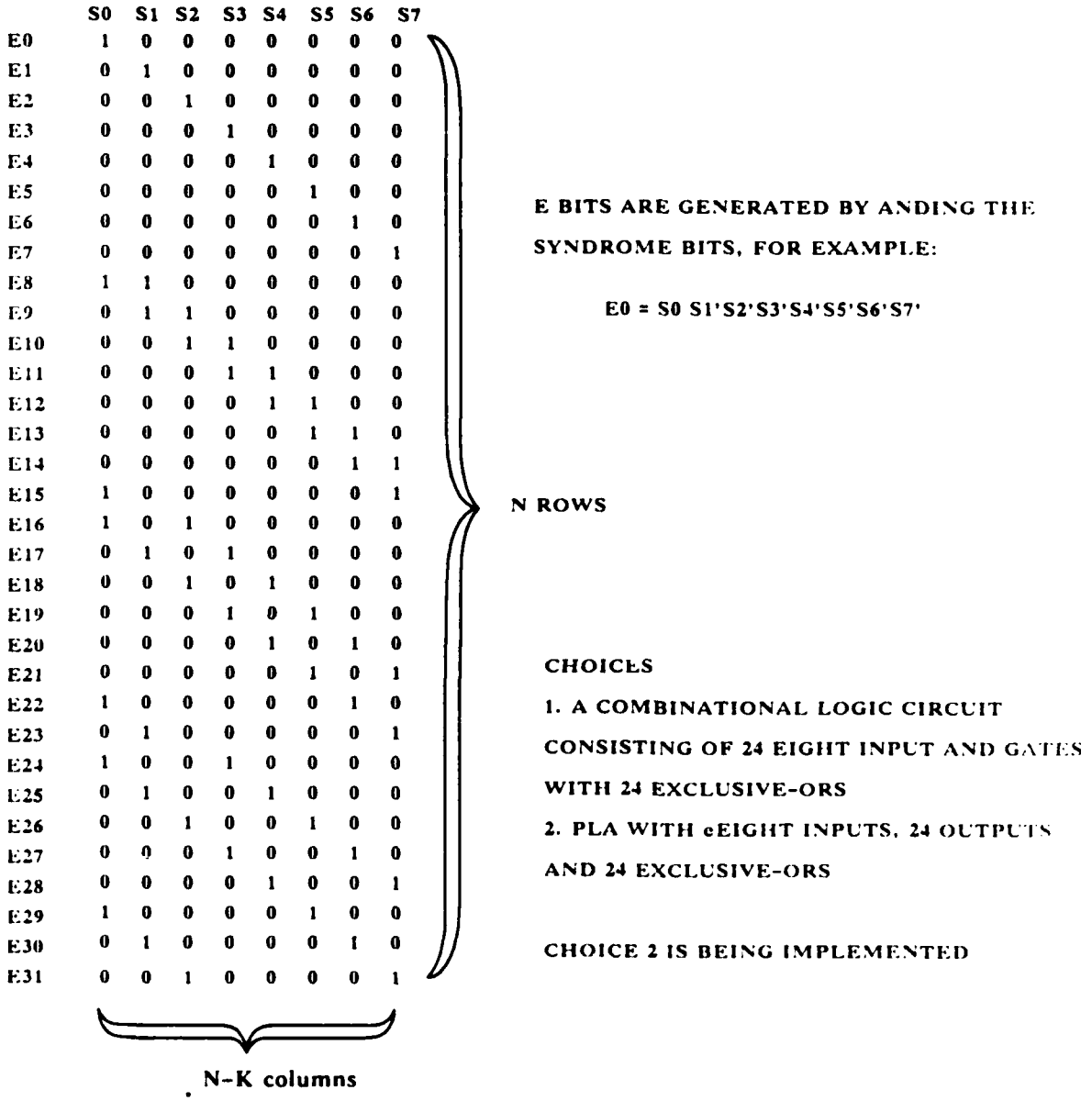


Figure 3-13 MATRIX FOR ERROR BIT GENERATION

memory bank READ/WRITE is the inverse of RIGHT LEFT

The precharge signals, PREL for the left memory bank and PRER for the right memory bank, are also derived from the write enable (WE) and read enable (RE).

3.3. Prototype System.

A prototype system will be built to evaluate the performance of the pipeline components. This prototype system will be configured as shown in Figure 3-14. The system shown will reduce the memory requirements and still permit the calculation of a 272-point DFT. A 272-point DFT only requires 544 64-bit words (272 64-bit words in each bank). Using a 3-micron fabrication process, the RAM array would be 11424 microns by 7104 microns. The PFAC I/O requirements would be reduced, enabling the use of a 84-pin configuration instead of the 144-pin package for a 4080-point DFT.

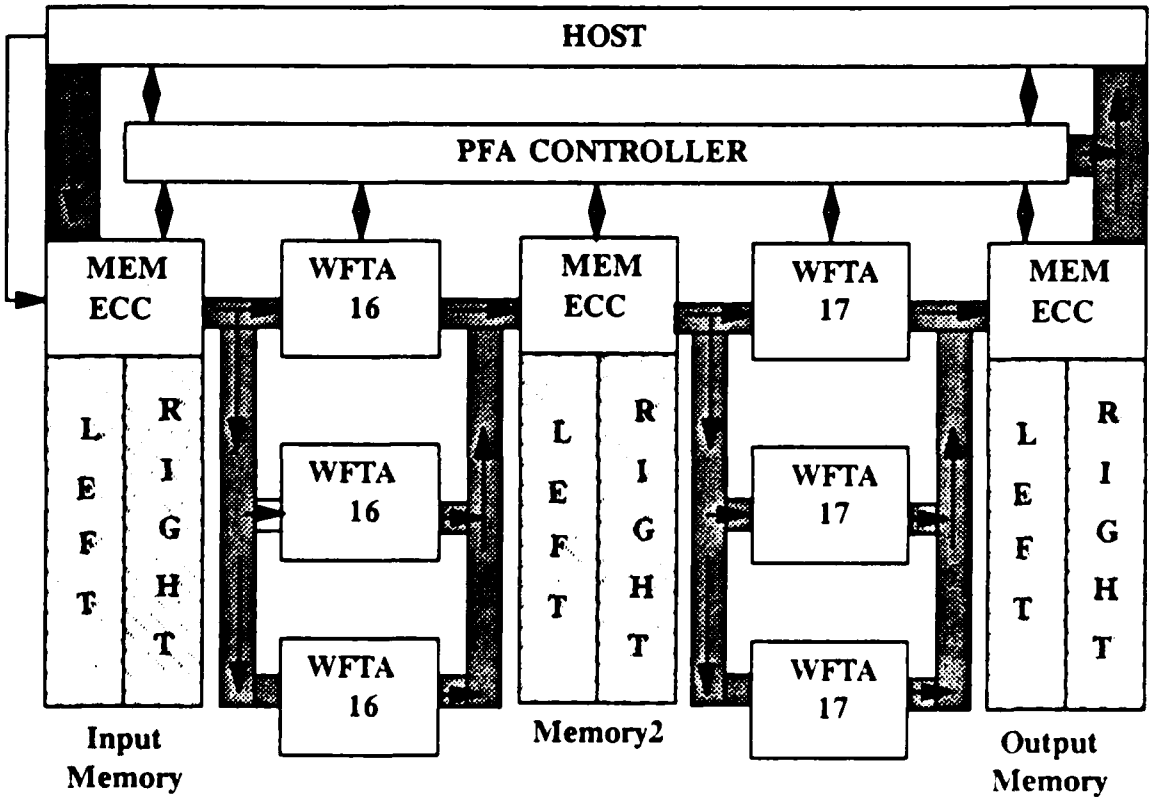


Figure 3-14 272-Point PFA Processor

CHAPTER 4

VLSI DESIGN

4.1. Method

The PFA controller was designed but not implemented in this thesis effort. The design utilized components that are available in the AFIT VLSI design library. The initial design of the PFA controller has flexibility and has the capability for fault detection, isolation, and correction.

The dual-ported double buffered memory with error control coding was specified in Caltech Intermediate Form (CIF). Although previous design projects at AFIT were implemented using CMOS technology, the design rules for fabrication of the basic cells have changed. Before any previously designed microcells could be used, much time and effort was taken to design the critical cells to minimize the area used, insure functionality, and maximize speed.

4.1.1. Tools.

Several Computer-Aided Design (CAD) tools were used in designing the cells required for the PFA pipeline memory.

The circuits were created and edited using Caesar (Ousterhout, 1983), an interactive VLSI design tool that utilizes a color graphics display terminal and a digitizing pad.

Lyra was used to perform design rule checking on the caesar circuits (Arnold, 1983).

The SPICE simulation program was used to determine if the data paths of the design operated at the required speed.

Mextra (Fitzpatrick, 1983) and Cstat, a CMOS version of stat (Baker, 1985) developed at AFIT provided information on unconnected or aliased nodes and

transistors which could not affect the outputs or be affected by the inputs.

4.1.2. **Design Rules.**

The circuits were designed using a scalable design rule set allowing for either a 1.2 or 3 micron implementation. The CMOS design rules used are listed in Appendix A.

4.2. **PFA Controller.**

In chapter 3, the high level design of the PFA controller was presented. In this chapter, the transistor level designs of the cells and modules are presented.

4.2.1. Processor Unit. The main modules of the processor unit are an ALU, twenty-eight processor registers, and bus selection circuitry.

The design of the ALU is a combinational logic circuit. The unit has a regular pattern that can be broken down into identical stages. Each stage consists of a full-adder with the input to each full-adder specified by the Boolean functions:

A typical ALU stage is implemented as shown in Figure 4-1.

The boolean input functions can be implemented, using CMOS Inverter Transmission Gate Logic (CITGL), with a total of sixteen transistors. The x input is implemented with a 2:1 multiplexer, an AND gate, and an OR gate. The Y input is implemented with only a 2:1 multiplexer. The Z input only requires a transmission gate for implementation.

The full-adder is implemented, using CITGL, with a total of nineteen transistors. The sum output is obtained with an exclusive-OR gate and an exclusive-NOR gate. The carry output is realized with two AND gates and an OR gate.

The twenty-eight processor registers are implemented with sixteen master-slave flip-flops which in turn consist of two D flip-flops. Master-slave flip-flops (MSFF) are required so that the new data can be arriving at the input to the master while the old

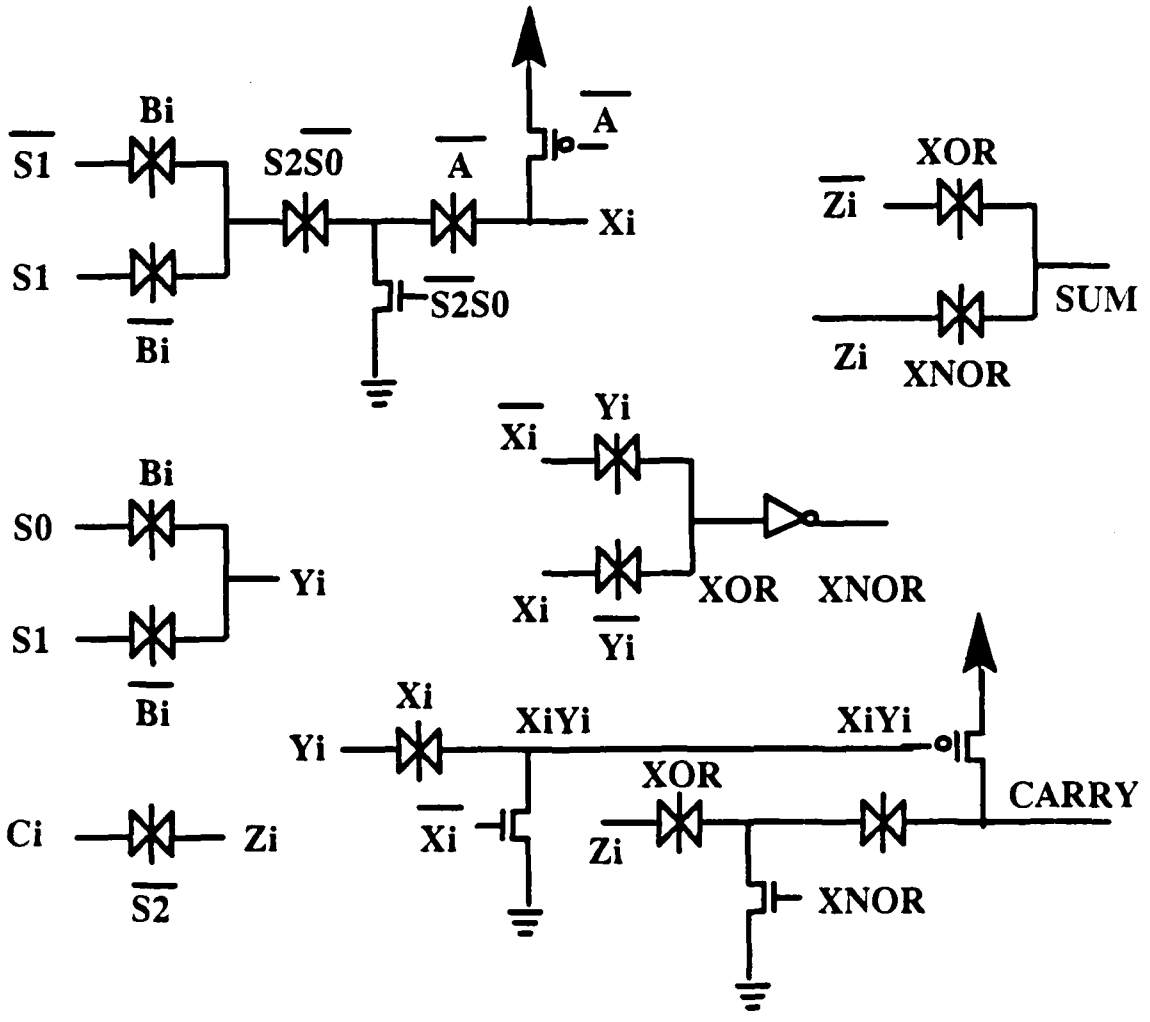


Figure 4-1 TYPICAL STAGE OF ALU

Xi=Ai+S2*S1'*So'*Bi+S2*S1*So'*Bi'
Yi=So*Bi+Si*Bi
Zi=S2'*Ci

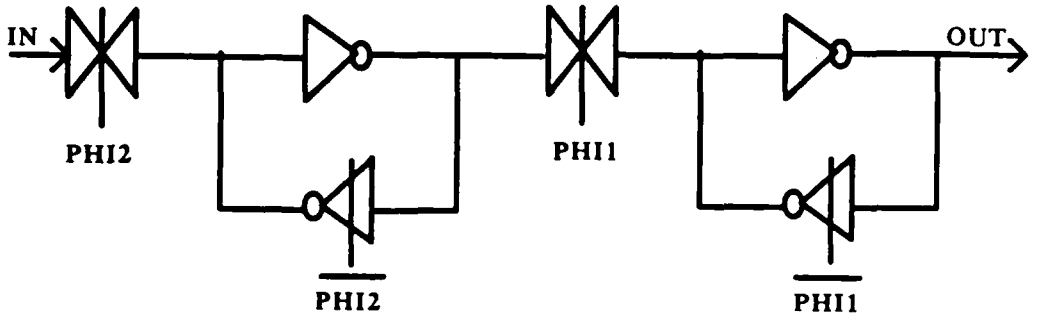


Figure 4-2 MASTER-SLAVE FLIP-FLOP

data is still stable at the output of the slave. The master-slave flip-flop functionality can be realized using CITGL. Only sixteen transistors are required to implement a master-slave flip-flop as shown in Figure 4-2. Two-phase clocking is used to guarantee no overlap between the two phases and thus prevent any possibility of racing through the MSFF. All the registers, with one exception, are implemented in this manner. The one exception is the permanent scale register. This register is a shift register, with parallel load, that does a five bit shift. Several of the registers have data input directly. The data is gated to the inputs through transmission gates. All the registers including special circuitry are shown in Appendix C.

The bus selection circuitry consists of a combination of a 32-bit decoder and a 2:1 multiplexer as shown in Figure 4-3. The 32-bit decoder enables the selection of any one of 32 addresses. The 2:1 multiplexor enables the host to specify addresses, during a test, instead of the control memory.

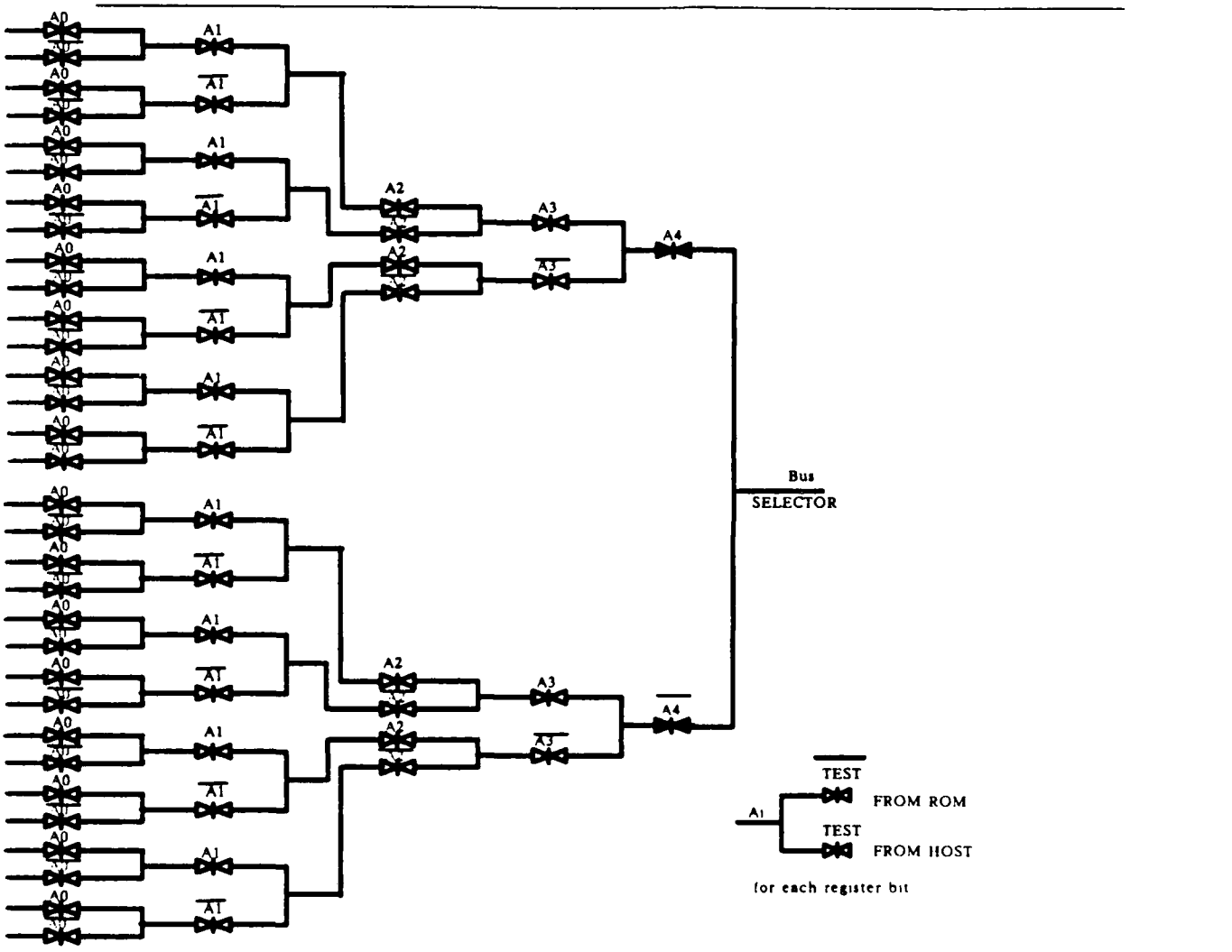


Figure 4-3 BUS SELECTION CIRCUITRY

4.2.2. Microprogrammed Control Unit.

The microprogrammed control unit consists of a control memory and an address sequencer as shown in Figure 3-4.

4.2.2.1. Control Memory.

As previously stated, the microinstructions for the RISC are stored in a control memory. The control memory is implemented with a very dense ROM design. This dense ROM design is one that uses a very compact XROM cell (Wilson and Schroeder, 1978; McKenny, 1980). The XROM cell receives its name from the X pattern of its transistors, as shown in Figure 4-4. The XROM cell is capable of storing four bits in a cell that is just under 8x8 microns square (using the design rules in Appendix A and fabricating with the 1.2 micron process). A data word can be retrieved from the XROM in 45 ns for a 3 micron process [Fr86]

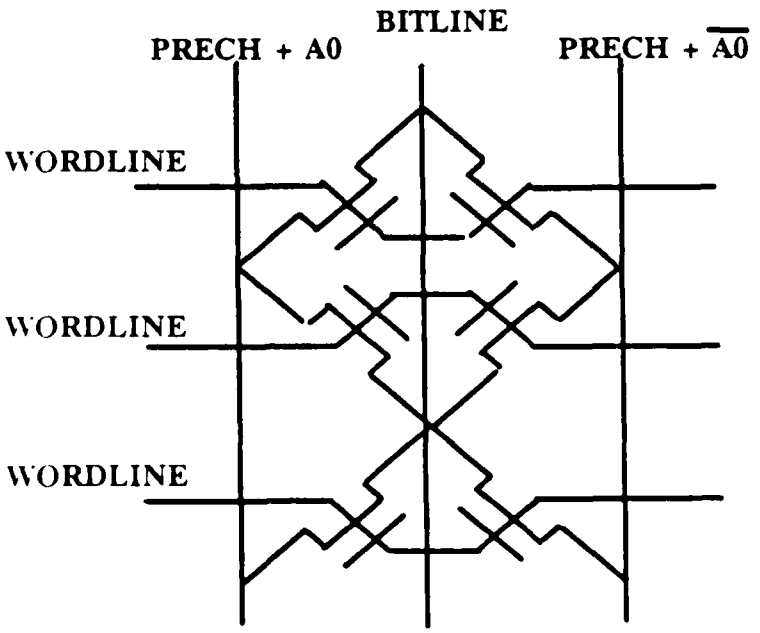


Figure 4-4 XROM CELL

4.2.2.2. **Address Sequencer.**

The address sequencer controls the generation of the address of the next stored microinstruction to be executed. The address sequencer consists of an address multiplexer, a control address register, an incrementer, a subroutine return register, a branch-test multiplexer, and branch logic, as shown in Figure 3-4.

The address multiplexer is a four to one multiplexer that selects a nine-bit address and sends the selected address to the control address register, as shown in Figure 4-5. The design shown is obtained with fifty four transmission gates. Segregating the input busses as shown minimizes the amount of bus crossover.

The control address register is implemented with nine master-slave flip-flops which in turn consist of two D flip-flops.

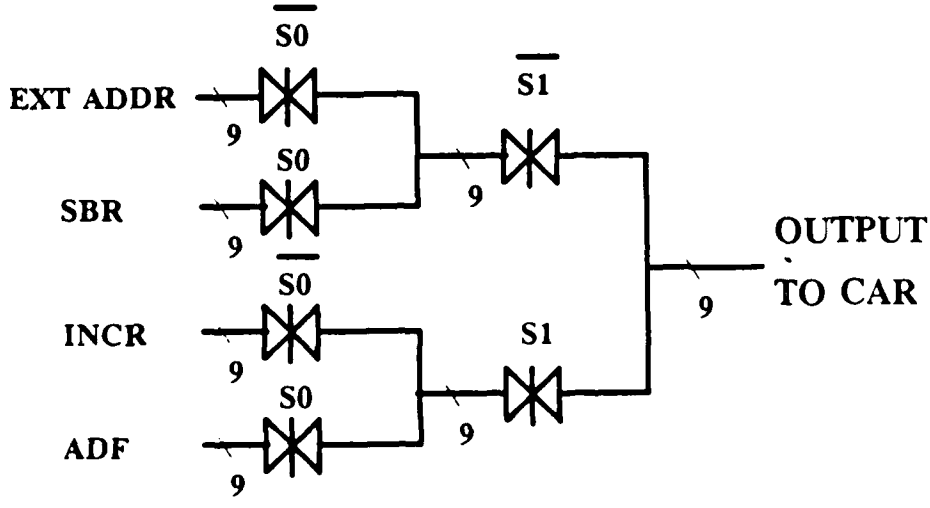


Figure 4-5 ADDRESS MULTIPLEXER

The incrementer is a combinational logic circuit constructed by cascading a series of half-adder circuits. One input of the least significant stage must always be a "one". The carry from each stage is connected to one of the inputs of the next stage. A half-adder can be constructed, using CITGL, with seven transistors, as shown in Figure 4-6.

The subroutine return register consists of a last-in, first-out (LIFO) register stack. The address lines are input to a series of MSFFs. The outputs of each MSFF has a feedback loop to the input of the previous MSFF. The feedforward path is enabled by the PUSH signal and the feedback path is enabled by the POP signal. An empty full stack detector is by constructing an additional column in the stack. A FULL signal is generated by ANDing the output of the deepest stage with the PUSH signal. An

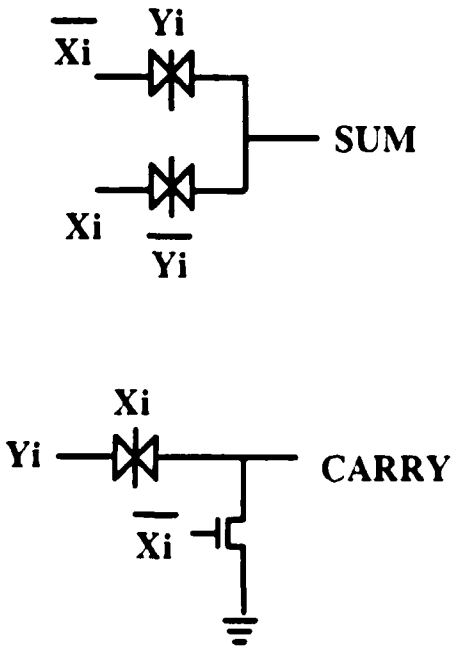


Figure 4-6 HALF-ADDER

EMPTY signal is generated by ANDing the output of the first stage with the POP signal. The SBR is shown in Figure 4-7.

The branch-test multiplexer is the same circuit as the 32-bit address selection decoder except the roles of input and output are reversed.

The branch logic is implemented with five two input AND gates, an OR gate, and an inverter, as shown in Figure 4-8.

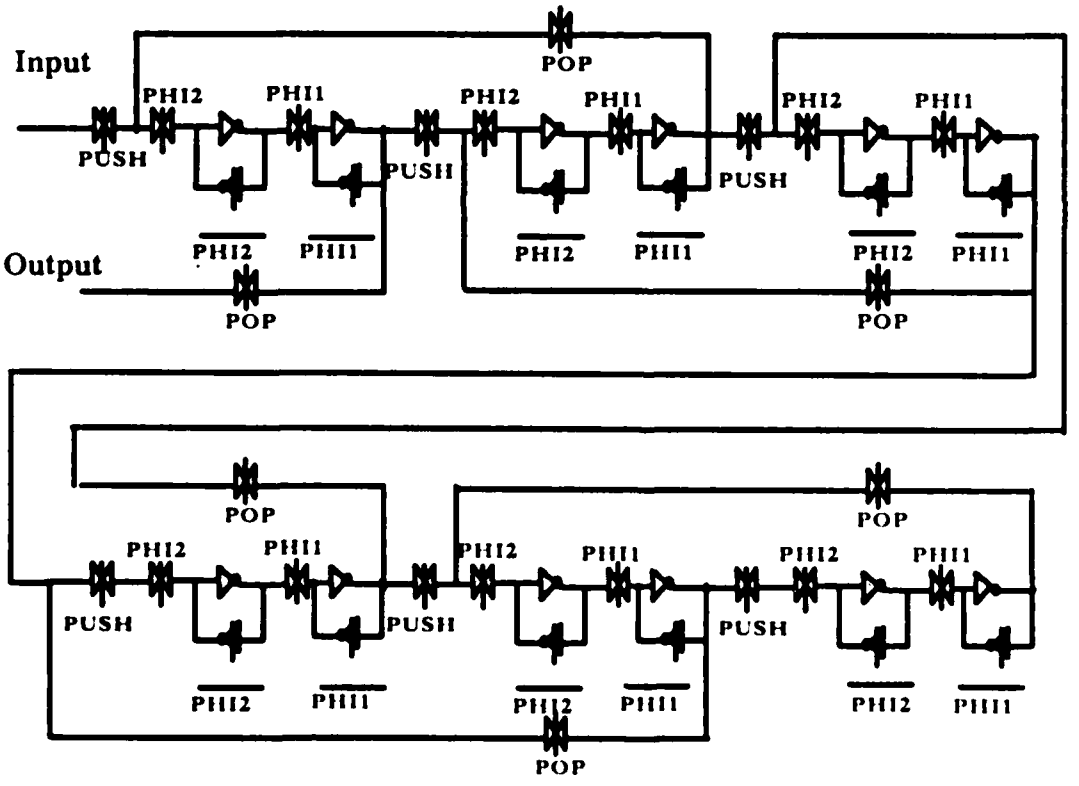


Figure 4-7 SBR LIFO STACK

4.3. Memory

The memory chip circuits were presented in Chapter 3. The main components are: the RAM cell, the read circuitry, the write circuitry, the error detection and correction circuitry, and the switching circuitry.

4.3.1. **RAM Cell** A cifplot of the six transistor RAM cell is shown Figure 4-9. The following sizes assume the use of the 1.5 micron process. The RAM cell is 42 micrometers high by 67.5 micrometers wide with stepping distances of 42 and 55.5 micrometers. The size of the RAM array is 7104 micrometers by 11424 micrometers for a dual bank, 272 x 64-bit word memory.

4.3.2. Read Circuitry

The read circuitry consists of the pull up transistors and the sense amplifier.

A CIFplot of the pull up transistors are shown in Figure 4-10. Because the purpose of the pull up transistors is to not only raise BITLINE and BITLINE' but also, to ensure that they are the same by shorting the two lines together, a very compact cell was implemented. The active region of P3 connects the source of one transistor to the source of the other transistor.

The sense amplifier was designed to be arrayed at the same pitch as the RAM array. In a SPICE simulation of this two stage sense amplifier, the output inverter detected a differential voltage of 0.01 volts. The amount of time necessary to read a RAM cell is directly proportional to the magnitude of the differential voltage needed for detection.

4.3.3. Write Circuitry.

The write circuitry is implemented with two NAND gates and two NOR gates The write circuitry is arrayed at the same pitch as the RAM array

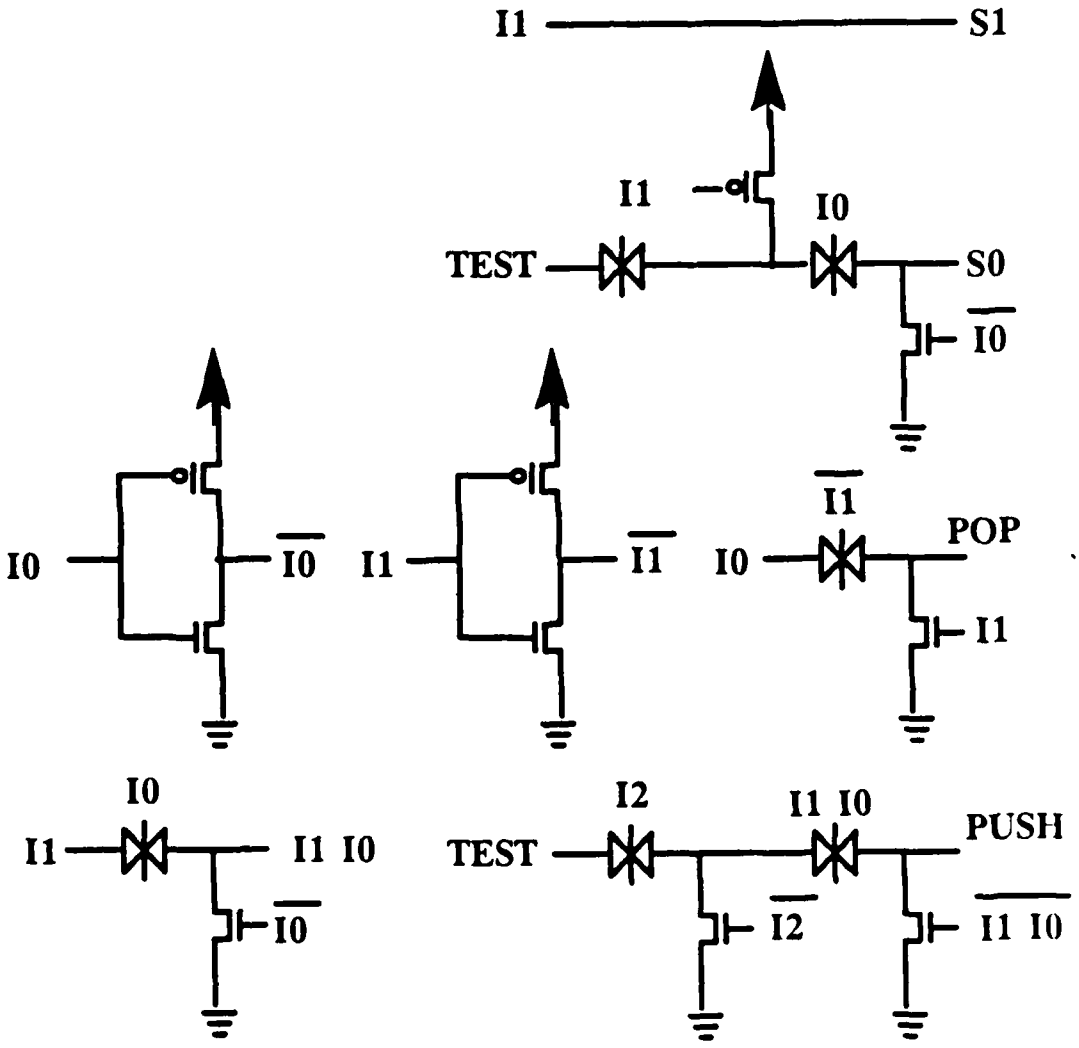


Figure 4-8 BRANCH LOGIC

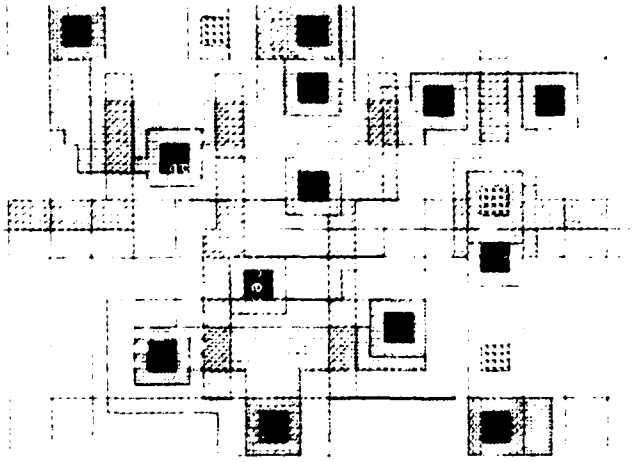


Figure 4-9 RAM CELL

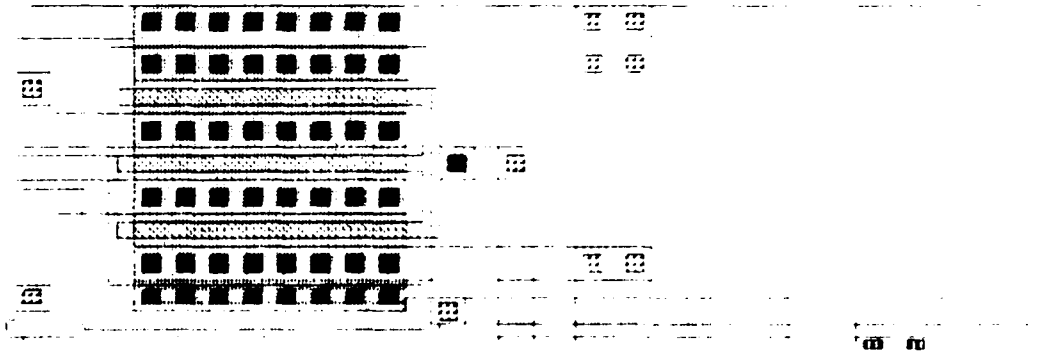


Figure 4-10 READ PULL UP TRANSISTORS

4.3.4. **Error Detection and Correction Circuitry.**

The error detection detection circuitry consists of a parity bit generation circuit, a syndrome bit generation circuit, and an error pattern detecting circuit. The CIFplot of a typical parity bit generator circuit is shown in Figure 4-11. The design of the syndrome bit generator circuit is the same as the parity bit generator circuit except that seven inputs are combined for each output. The error bit generator is an eight input 24 output PLA, as shown in Figure 4-12.

4.3.5. **Address Selection Circuitry**

The address selection circuitry consists of the decoder and the wordline driver The decoder is a NAND PLA with snaked gates to minimize the pitch of the cells The pitch of the decoder and the wordline driver is half the pitch of the RAM array

Figure 4-11 PARITY BIT GENERATOR

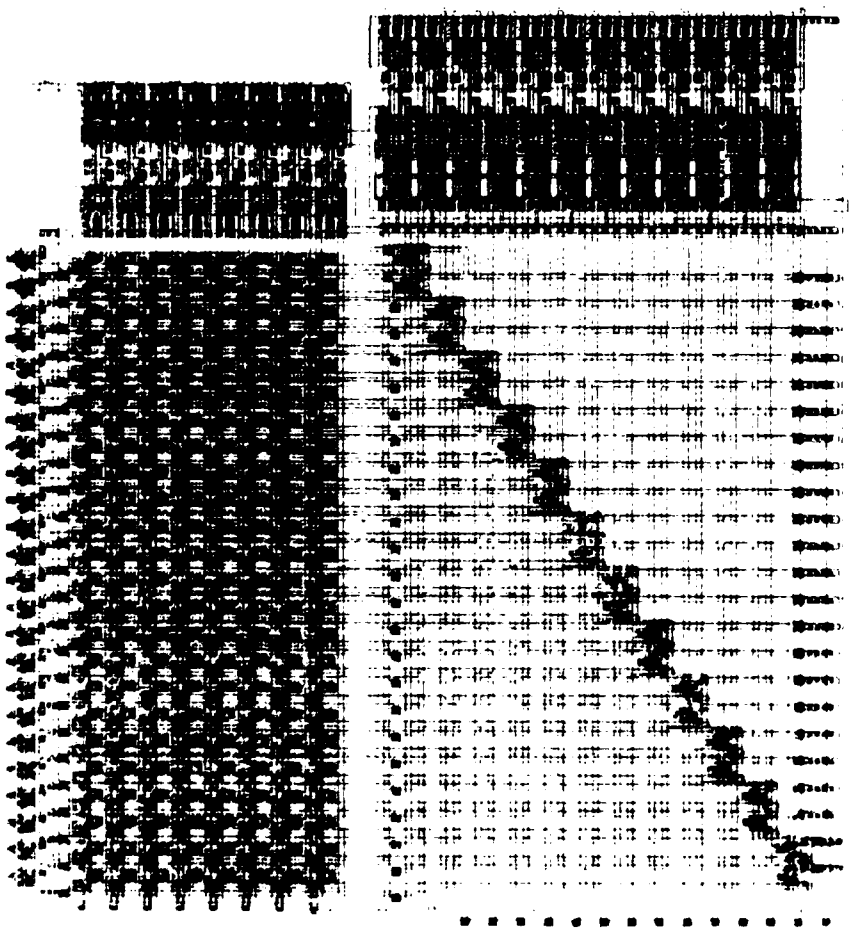


Figure 4-12 ERROR BIT GENERATOR PLA

CHAPTER 5

CONCLUSIONS and RECOMMENDATIONS

5.1. Conclusions

This thesis effort produced the design of the PFA Controller which will operate at a clock rate of 20 MHz. The RISC implementation possesses an advantage over other methods in flexibility and extensibility of the design. The RISC architecture of the controller is simple to implement. The instruction set is limited to the necessary micro-operations. If the control requirements change, the microprogram can be changed by changing the personalization of the control memory ROM. In addition, the design of the RISC can be used for control in many diverse situations. Since the RISC architecture minimizes hardware and software, the complexity of the design effort is also minimized. Also the PFA pipeline memory was designed with performance comparable to that of commercially available state-of-the-art CMOS RAMs. The worst case access times should be less than 28 ns.

The realization of the PFA pipeline controller has its primary importance in the contribution it makes toward achieving the overall goal of implementing a PFA processor capable of computing 4080-point DFTs at a rate of over 8300 Hz (Taylor, 1985) while incorporating a high degree of fault tolerance into the system.

5.2. Recommendations

The following recommendations are proposed regarding future action:

1. The PFA controller should be implemented using the 3 micron process to enable performance evaluations. Extensive test results are required before the PFA controller is submitted for fabrication in the 1.2 micron process.

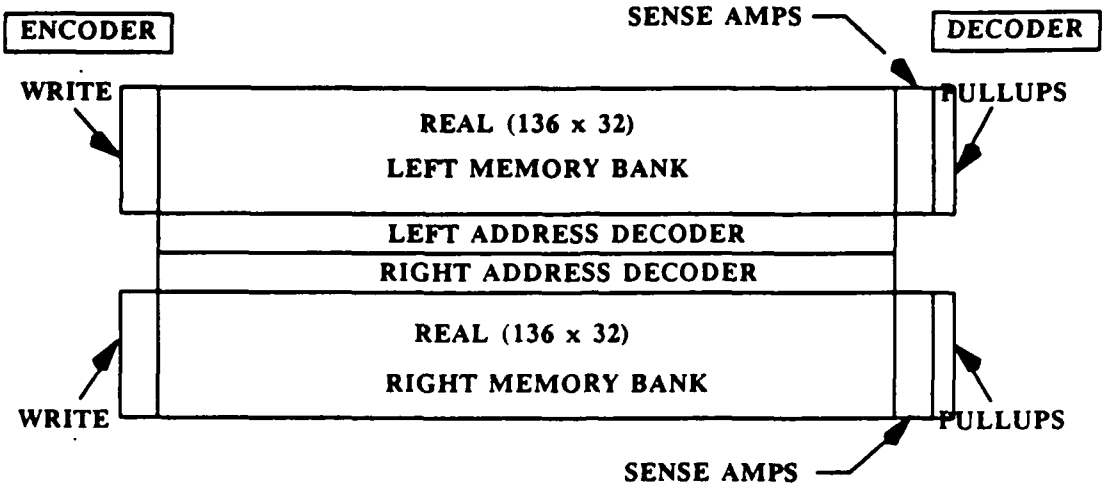


Figure 5-1 MEMORY EVALUATION CHIP

2. The memory chip should be implemented using the 3 micron process to evaluate the performance of the circuits. However, a 3 micron fabrication necessitates decreasing the size of the memory array. The memory could be fabricated as a set of four chips, two for the real numbers and two for the imaginary numbers. The address decoder would have the same number of address lines but the address lines would be broadcast to four chips, two chips for the real numbers and two chips for the imaginary numbers. A typical chip is shown in Figure 5-1. This arrangement would degrade the speed of operation but would permit testing of the functionality of the pipeline structure.

3. After the actual performance of the data switching circuits and error detection-correction circuits are evaluated, a dynamic RAM memory approach should be pursued to decrease the size of the memory chips.

4. To increase the ECC decoder's speed, use two input OR gates in lieu of the orplane shown in Figure 4-12.

REFERENCES

[Bl85] Blahut, Richard E., *Fast Algorithms for Digital Signal Processing*, Addison-Wesley Publishing Co., Reading, MA (1985).

[Br73] Breuer, M. A., "Testing for Intermittent Faults in Digital Circuits," *IEEE Transactions on Computing*, pp. 241-245 (March 1973).

[Co85 ...] Collins, James M., "Simulation and Modeling of a VLSI Winograd Fourier Transform Processor," *MS Thesis, GE/ENG/85D-9*, School of Engineering, Air Force Institute of Technology (AU) Wright-Patterson AFB, OH, (December 1985).

[Co86] Cooper, C. H., "Modeling and Simulation of the WFTA 16 Processor Using the VHSIC Hardware Description Language," *M.S. Thesis, AFIT/GE/ENG/86D-44* . School of Engineering, Air Force Institute of Technology (AU) Wright-Patterson AFB, OH, (Dec 1986).

[Co85] Coutee, Paul W., "Arithmetic Circuitry for High Speed VLSI Winograd Fourier Transform Processors," *MS Thesis, GE/ENG/85D-11*, School of Engineering, Air Force Institute of Technology (AU) Wright-Patterson AFB, OH, (December 1985).

[Fl68] Flehinger, B.J., "Reliability Improvement Through Redundancy at Various System Levels," *IBM Journal of R and D*, pp. 148-158 (April 1968).

[Fr86] French, L. E., "A RISC Controller for the CAM-PUTER System," *M.S. Thesis, GE-86D*, School of Engineering, Air Force Institute of Technology (AU) Wright-Patterson AFB, OH, (Dec 1986).

[La85] Lala, P. K., *Fault Tolerant and Fault Testable Hardware Design*, Prentice-Hall, Englewood Cliffs, N J (1985).

[Li83] Lin, S. and D. Costello, *Error Control Codes: Fundamentals and Applications*, Prentice Hall, Englewood Cliffs (1983).

[Li86] Linderman, R. W., H. W. Carter, J. DeGroat, and R. R. Gross, "Architecture and Design of High Speed VLSI Processors and Integrated Circuit Design Automation," *DARPA Semi-Annual VLSI Research Report*, p. 16 (March 1986).

[Ma79] Mano, M., *Digital Logic and Computer Design*, Prentice-Hall, Englewood Cliffs, N.J. (1979).

[Ma82] Mano, M., *Computer System Architecture*, Prentice-Hall, Englewood Cliffs, N.J. (1982).

[Mc79] McClellan, J. H. and C. M. Rader, *Number Theory in Digital Signal Processing*, Prentice-Hall, Englewood Cliffs, NJ (1979).

[Mo56] Moore, E.F., *Geganken-Experiments on Sequential Machines, Automata Studies.*, (34) pp. 129-153 Princeton Univ. Press, (1956).

[Ne56] Neumann, J. von, "Probablistic Logics and Synthesis of Reliable Organisms From Unreliable Components," *Annals of Mathematical Studies*, (34) pp. 43-98 Princeton Press, (1956).

[On84] Ong, D. G., *Modern MOS Technology: Processes, Devices, and Design*, McGraw-Hill, New York, New York (1984).

[Ra68] Rader, Charles M., "Discrete Fourier Transforms When the Number of Data Samples is Prime," *Proceedings of the IEEE* **6**(56) pp. 1107-1108 (June 1968).

[Ro85] Rossbach, Paul C., "Control Circuitry for High Speed VLSI Winograd Fourier Transform Processors," *MS Thesis, GE/ENG/85D-85*, School of Engineering, Air Force Institute of Technology (AU) Wright-Patterson AFB, OH, (December 1985).

[Se78] Sedmak, R.M. and H.L. Libergot, "Fault-Tolerance of a General Purpose Computer Implemented by Very Large Scale Integration," *Proc. Int. Symp. Fault-Tolerant Computing*, pp. 137-143 (1978).

[Sh86] Shephard, C. G., "Integration and Design for Testability of a High Speed Winograd Fourier Transform Processor," *M.S. Thesis, AFIT/GE/ENG/86D-46*, School of Engineering, Air Force Institute of Technology (AU) Wright-Patterson AFB, OH. (1983).

[Si84] Sieworek, Daniel, "Architecture of Fault Tolerant Computers," *Computer* **8**(17) pp. 9-18 (August 1984).

[So85] Soclof, S., *Analog Integrated Circuits*, Prentice-Hall, Englewood Cliffs, N J (1985).

[Ta85] Taylor, Kent, "Architecture and Numerical Accuracy of High-Speed DFT Processors," *MS Thesis, GE/ENG/85D-47*, School of Engineering, Air Force Institute of Technology (AU), Wright-Patterson AFB, OH, (December 1985).

[Tr82] Triebel, W. A. and Alfred E. Chu, *Handbook of Semiconductor and Bubble Memories*, Prentice-Hall, Englewood Cliffs, N J (1982).

[We85] Weste, N. H. E. and K. Eshraghian, *Principles of CMOS VLSI Design*, Addison-Wesley Publishing Company, (1985).

[Wi78] Winograd, S., "On Computing the Discrete Fourier Transform," *Math Comput.* **32** pp. 175-199 (Jan. 1978).

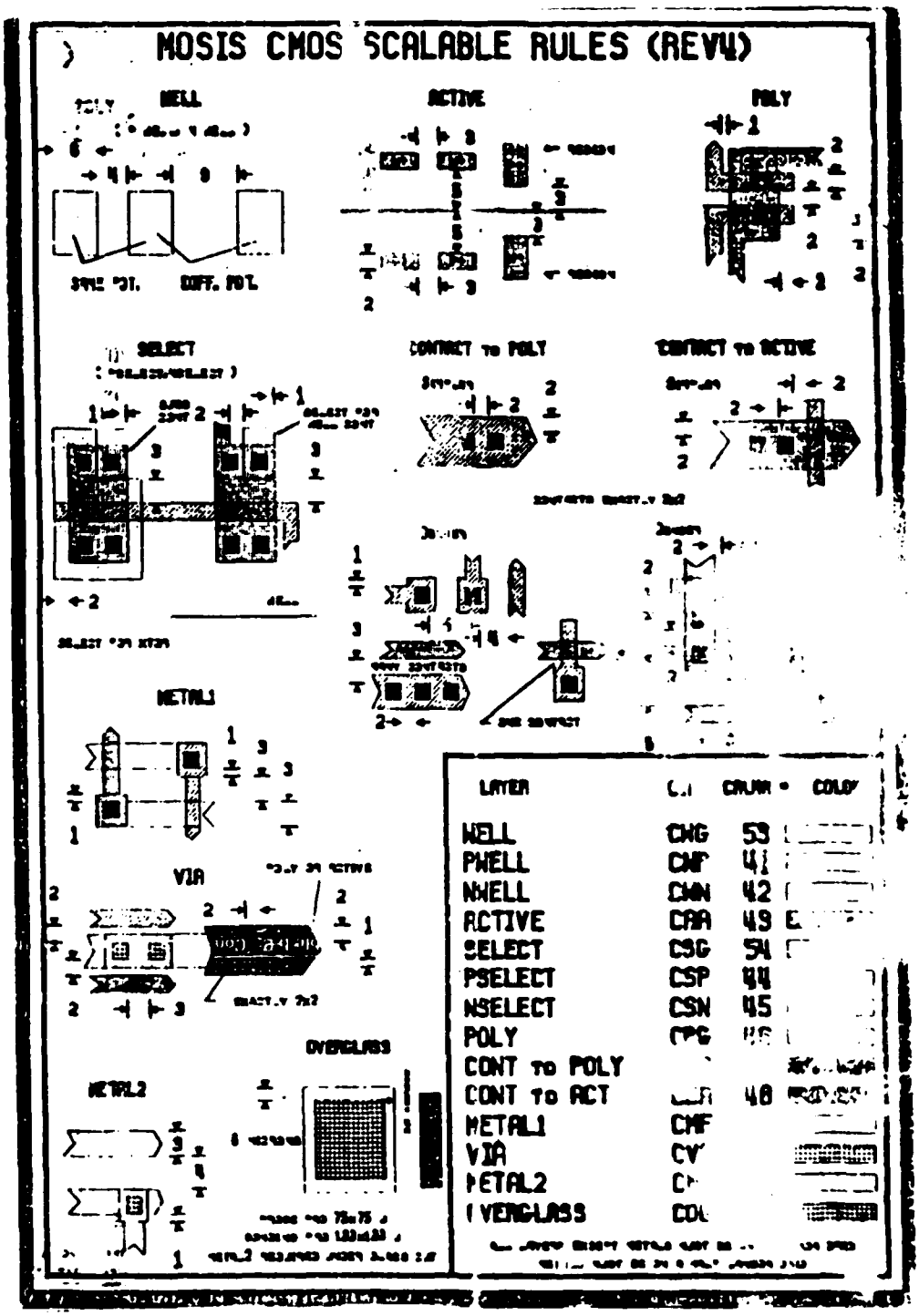
[Ya84] Yamada, Iunzo and et al., "A Submicron VLSI Memory with a 4b-at-a-Time Built-in ECC Circuit," *International Solid-State Circuits Conference Digest of Technical Papers*, pp. 104-105 Lewis Winner, (February 1984).

Vita

Captain Gary D. Hedrick was born on 14 January 1946 in Beckly, West Virginia. He graduated from West Geauga High School in Chesterland, Ohio in 1946. In 1965, he enlisted for four years in the U.S. Air Force. Between 1969 and 1980 he worked for various companies. During this time he received a Bachelor of Science in Electrical Engineering from Ohio State University (June, 1976). Upon graduation from the USAF Offficer Training School in December, 1980, he was commisssioned a Second Lieutenant in the United States Air Force. He served two years at Vandenberg AFB, California, working on the space shuttle launch computers. In 1982, he was assigned to the Foreign Technology Division at Wright-Patterson AFB, Ohio. In May, 1985, he entered the Air Force Institute of Technology to pursue a Master of Science in Electrical Engineering. In January, 1987, he will begin a joint assignment at the Armed Forces Radiobiology Research Institute in Bethesda, Maryland as the Chief of the Hardware Division.

APPENDIX A

MOSIS DESIGN RULE SET



SECURITY CLASSIFICATION OF THIS PAGE AD-A180 111

REPORT DOCUMENTATION PAGE

Form Approved OMB No. 0704-0188

1a. REPORT SECURITY CLASSIFICATION: UNCLASSIFIED	1b. RESTRICTIVE MARKINGS
2a. SECURITY CLASSIFICATION AUTHORITY	3. DISTRIBUTION/AVAILABILITY OF REPORT: Approved for public release; distribution unlimited
2b. DECLASSIFICATION/DOWNGRADING SCHEDULE	
4. PERFORMING ORGANIZATION REPORT NUMBER(S): AFIT/GE/ENG/86D-45	5. MONITORING ORGANIZATION REPORT NUMBER(S)
6a. NAME OF PERFORMING ORGANIZATION: School of Engineering; 6b. OFFICE SYMBOL (If applicable): AFIT/ENG	7a. NAME OF MONITORING ORGANIZATION
6c. ADDRESS (City, State, and ZIP Code): Air Force Institute of Technology, Wright Patterson AFB, OH 45433	7b. ADDRESS (City, State, and ZIP Code)
8a. NAME OF FUNDING/SPONSORING ORGANIZATION: Air Force Office of Scientific Res.; 8b. OFFICE SYMBOL (If applicable): AFOSR/XP	9. PROCUREMENT INSTRUMENT IDENTIFICATION NUMBER
8c. ADDRESS (City, State, and ZIP Code): Bolling AFB, Washington D.C. 20332	10. SOURCE OF FUNDING NUMBERS: PROGRAM ELEMENT NO. / PROJECT NO. / TASK NO. / WORK UNIT ACCESSION NO.

11. TITLE (Include Security Classification): See Box 19

12. PERSONAL AUTHOR(S): Gary D. Hedrick, B.S.E.E., CPT, USAF

13a. TYPE OF REPORT	13b. TIME COVERED	14. DATE OF REPORT (Year, Month, Day)	15. PAGE COUNT
MS Thesis	FROM _____ TO _____	1986 December	

16. SUPPLEMENTARY NOTATION

17. COSATI CODES			18. SUBJECT TERMS (Continue on reverse if necessary and identify by block number)
FIELD	GROUP	SUB-GROUP	
09	02		Digital Signal Processing, Prime Factor Algorithm, Computer Architecture, Fault Tolerance

19. ABSTRACT (Continue on reverse if necessary and identify by block number)

Title: Design of Fault Tolerant Prime Factor Algorithm Array Elements

Thesis Chairman: Richard W. Linderman
Assistant Professor of Electrical and
Computer Engineering

Approved for public release: IAW AFR 190-1.
LYNN E. WELAVER 7 April 87
Dean for Research and Professional Development
Air Force Institute of Technology (AFIT)
Wright-Patterson AFB OH 45433

20. DISTRIBUTION/AVAILABILITY OF ABSTRACT	21. ABSTRACT SECURITY CLASSIFICATION
☒ UNCLASSIFIED/UNLIMITED ☐ SAME AS RPT. ☐ DTIC USERS	UNCLASSIFIED
22a. NAME OF RESPONSIBLE INDIVIDUAL: Richard W. Linderman, Captain, USAF	22b. TELEPHONE (Include Area Code): 513-255-3576 / 22c. OFFICE SYMBOL: AFIT/ENG

DD Form 1473, JUN 86 Previous editions are obsolete. SECURITY CLASSIFICATION OF THIS PAGE: UNCLASSIFIED

END

5-87

DTIC