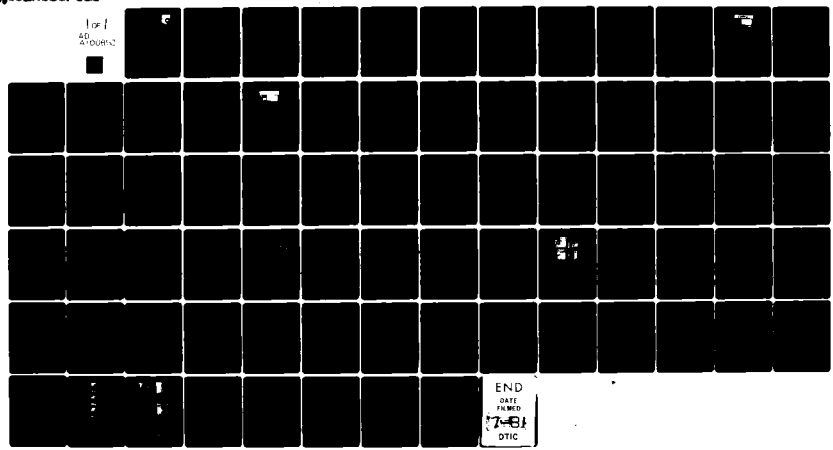
AD-A100 852 TEXAS INSTRUMENTS INC DALLAS CENTRAL RESEARCH LABS F/G 9/2
PROGRAMMABLE IMAGE PROCESSING ELEMENT.(U)
NOV 80 T F CHEEK, W L EVERSOLE, J F SALZMAN F33615-79-C-1763
AFWAL-TR-80-1209 NL

UNCLASSIFIED

1 OF 1
AD
A100852



AD A100852

LEVEL

Report AFWAL-TR-80-1209

PROGRAMMABLE IMAGE PROCESSING ELEMENT

T.F. Cheek, W.L. Eversole, J.F. Salzman
Texas Instruments Incorporated
Central Research Laboratories
13500 North Central Expressway
Dallas, Texas 75222

November 1980
Final Technical Report Period 1 August 1979–31 July 1980

Approved for public release; distribution unlimited.

DTIC FILE COPY

JUL 1 1981
A

Avionics Laboratory
Air Force Wright Aeronautical Laboratories (AFSC)
Wright-Patterson Air Force Base, Ohio 45433

81 6 30 014

NOTICE

When Government drawings, specifications, or other data are used for any purpose other than in connection with a definitely related Government procurement operation, the United States Government thereby incurs no responsibility nor any obligation whatsoever; and the fact that the Government may have formulated, furnished, or in any way supplied the said drawings, specifications, or other data, is not to be regarded by implication or otherwise as in any manner licensing the holder or any other person or corporation, or conveying any rights or permission to manufacture, use, or sell any patented invention that may in any way be related thereto.

This report has been reviewed by the Office of Public Affairs (ASD/PA) and is releasable to the National Technical Information Service (NTIS). At NTIS, it will be available to the general public, including foreign nations.

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

Guy D. Couturier

GUY D. COUTURIER
PROJECT ENGINEER

Robert E. Conklin

ROBERT E. CONKLIN, CHIEF
PROCESSOR TECHNOLOGY GROUP

FOR THE COMMANDER

Stanley E. Wagner

STANLEY E. WAGNER, CHIEF
MICROELECTRONICS BRANCH
ELECTRONIC TECHNOLOGY DIVISION

"If your address has changed, if you wish to be removed from our mailing list, or if the addressee is no longer employed by your organization please notify AFWAL/AADE-1 *W-PAFB, OH 45433 to help us maintain a current mailing list."*

Copies of this report should not be returned unless return is required by security considerations, contractual obligations, or notice on a specific document.

AIR FORCE/56780/17 June 1981 — 90

SECURITY CLASSIFICATION OF THIS PAGE (When Data Entered)

REPORT DOCUMENTATION PAGE

READ INSTRUCTIONS BEFORE COMPLETING FORM

1. REPORT NUMBER: AFWAL-TR-80-1209
2. GOVT ACCESSION NO.: AD-A100852
3. RECIPIENT'S CATALOG NUMBER:

4. TITLE (and Subtitle): Programmable Image Processing Element

5. TYPE OF REPORT & PERIOD COVERED: Final Technical Report, 1 Aug[redacted]79 – 31 Jul[redacted]80

6. PERFORMING ORG. REPORT NUMBER:

7. AUTHOR(s): T.F. Cheek W.L. Eversole J.F. Salzman

8. CONTRACT OR GRANT NUMBER(s): F33615-79-C-1763

9. PERFORMING ORGANIZATION NAME AND ADDRESS:
Texas Instruments Incorporated
Central Research Laboratories
13500 North Central Expressway
Dallas, Texas 75222

10. PROGRAM ELEMENT, PROJECT, TASK AREA & WORK UNIT NUMBERS:

11. CONTROLLING OFFICE NAME AND ADDRESS:
United States Air Force
Air Force Avionics Laboratory
Air Force Systems Command
Wright-Patterson AFB, Ohio 45433

12. REPORT DATE: Nov[redacted]80

13. NUMBER OF PAGES: 72

14. MONITORING AGENCY NAME & ADDRESS (if different from Controlling Office):

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

15a. DECLASSIFICATION/DOWNGRADING SCHEDULE:

16. DISTRIBUTION STATEMENT (of this Report):

Approved for public release; distribution unlimited.

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

18. SUPPLEMENTARY NOTES:

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

Image Processing
Large-Scale Integrated Circuits
User-Programmable
Nonvolatile Memory
KOM-Accumulate Algorithms
Sum-of-Products Operation

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

The objective of this 12-month exploratory development effort was to develop a general-purpose, digital, large-scale integrated circuit device, called the programmable image-processing element (PIPE), which can be programmed to perform a variety of signal-processing functions such as Cosine and Hadamard transforms, edge extraction, unsharp masking, pole-zero filtering, and signal smoothing. The PIPE devices can operate on 8 × 1 or 3 × 3 element data blocks. The PIPE LSIC is suited for those airborne applications where data rate, size, power, and weight restrictions prohibit the use of general-purpose microprocessors or high-speed digital multipliers. This contract addressed only the design and photomask fabrication of the PIPE LSIC.

DD FORM 1 JAN 73 1473 EDITION OF 1 NOV 65 IS OBSOLETE

SECURITY CLASSIFICATION OF THIS PAGE (When Data Entered)

BLOCK 20 (Continued)

During the design phase of the program, several architectures were investigated as candidates for implementing the PIPE LSIC. The read-only-memory (ROM)-accumulate architecture was selected, which maximizes flexibility in algorithm implementation and minimizes external control and timing logic.

Also, during the design phase, investigations of ROM technologies and designs to ensure a user-oriented image-processing LSIC were conducted. The N-channel metal-oxide-semiconductor (NMOS) technology was selected to implement the PIPE LSIC because it is an established, cost-effective technology capable of providing the high-circuit density and low speed-power product. The user-erasable, programmable memory (EPROM) was judged to be the best memory technology to meet the goals of the PIPE LSIC. The primary advantages of NMOS EPROM technology are:

- User-programmable (electrically)
- Erasable (ultraviolet light)
- Nonvolatile
- Single supply voltage operation
- Static on-chip NMOS logic
- Military grade
- Established technology.

These features make EPROM technology ideal for the PIPE design.

The PIPE LSIC design completed during this contract has five major sections: input latch/parallel-to-serial shift register, EPROM, shift-and-accumulate, tri-state output latch, and controller.

The total bar size is approximately 240 × 270 mils^2, with a total estimated power of 600 milliwatts.

PREFACE

This report was prepared by Texas Instruments Incorporated, Dallas, Texas under Air Force Contract No. F33615-79-C-1763. The work under this contract was administered and funded by the Air Force Systems Command. Mr. Guy Couturier (AFWAL/AADE-1) of the Avionics Laboratory, a part of Air Force Wright Aeronautical Laboratories, Wright-Patterson Air Force Base, Ohio, was the Project Engineer.

At Texas Instruments the work was performed in the Advanced Technology Laboratory under the direction of Dr. Tom F. Cheek, Manager of the signal processing branch in the Advanced Technology Laboratory, Equipment Group.

This is the Final Technical Report for the contract. It covers work done from 1 August 1979 to 30 July 1980. It was submitted by the authors in October 1980.

A

TABLE OF CONTENTS

Section	Title	Page
I	INTRODUCTION	1
	A. Overview	1
	B. Objective	1
	C. Summary	1
II	TECHNICAL DISCUSSION	5
	A. PIPE LSIC Design	5
	1. Introduction	5
	2. Input Latch/Parallel-to-Serial Shift Register	11
	3. EPROM	18
	4. Shift-and-Accumulate	21
	5. Output Latch/Tri-State Buffer	31
	6. Controller	31
	7. Programmable Sum-of-Products Operator	40
	B. PIPE LSIC Applications	44
	1. Introduction	44
	2. Input/Output Considerations	46
	3. Matrix Operation	47
	4. Neighborhood Operators	53
	5. Experimental Results	62
	6. Operational Characteristics	64
	REFERENCES	71

LIST OF ILLUSTRATIONS

Figure	Title	Page
1	PIPE LSIC Layout	3
2	Block Diagram of the Multiplier-Accumulator Algorithm for Implementing Sum of Products Operator	6
3	Block Diagram of the ROM-Accumulator Algorithm for Implementing Sum of Products Operator	7
4	Block Diagram of ROM-Accumulator Algorithm With Parallel-to-Serial Shift Register, Buffer Output, and Controller	7
5	Simplified Block Diagram of the PIPE LSIC Architecture	8
6	Diagram of the PIPE LSIC Bar Map	9
7	CALCOMP Plot of PIPE LSIC	10
8	Block Diagram of the Input Structure	11
9	Block Diagram of the Input Multiplexer Latch Parallel-to-Serial Register	12
10	Timing Diagram of the Input, Load, and Shift Sequences	13
11	Circuit Diagram of the Input Multiplexer	14
12	Typical NMOS Latch Circuit	14
13	Schematic Diagram of the Input Multiplexer	15

14 Partial Block Diagram of the Input Stage Architecture (3 Stages of 9) 16
15 Partial Block Diagram of the Parallel-to-Serial Conversion (4 Bits of One 8-Bit Word) 16
16 Circuit Schematic for Memory Address Driver and Latch 17
17 V_{pp} Voltage Divider Circuit 18
18 Programming Multiplexer 18
19 TTL-to-NMOS Level Input Buffer With Static Protection 19
20 Simplified Block Diagram of the EPROM Section 19
21 Block Diagram of the 512 × 12 EPROM Configuration 20
22 EPROM Cell Structure 20
23 Schematic Diagram of the X and Y Matrix Decoders 21
24 Schematic Diagram of the Read, Write, and Buffer Circuitry 22
25 Block Diagram of the Shift-and-Accumulate Section 23
26 Block Diagram of the Shift-and-Accumulate Section 25
27 Logic Diagram of the Full Adder 27
28 Schematic Diagram of the Full Adder With Input Data Latch 27
29 Schematic Diagram of the Carry/Sum Latch 28
30 Schematic Diagram of a 4-Bit NMOS With Dynamic Look-Ahead Carry 29
31 Schematic Diagram of the Dynamic Adder/Accumulator and Latch (DYNAD) 30
32 Logic Diagram of the Output Latch/Tri-State Buffer 31
33 Schematic Diagram of the Output Latch/Tri-State Buffer 32
34 Simplified Block Diagram of the Controller Section 33
35 Block Diagram of the Shift Register Controller 34
36 Controller Set/Reset Latch Diagram 35
37 Controller Timing Diagram 35
38 Schematic Diagram of the Control Multiplexer 37
39 Controller D-Latch Diagram 39
40 Schematic Diagram of the Clock Driver 39
41 NOR Gate Bus Driver Diagram 40
42 Programmable Sum of Products Unit 41
43 Programmable Sum of Products Breadboard 45
44 PIPE LSIC I/O Considerations 46
45 Programmable Image Processing Element Configured to Implement Y-WX 48
46 Weighting Coefficient Matrices for Various 8 × 1 Transforms 49
47 Implementation of Eight-Point Fourier Transform Using PIPE LSICs 51
48 Implementation of Two-Dimensional Transform Using PIPE LSIC 52
49 Using PIPE LSIC to Implement Second-Order Filter Function 53
50 Two-Dimensional Spatial Convolution for 3 × 3 Neighborhood 54
51 Low-Pass Weighting Arrays 55
52 High-Pass Weighting Arrays 55
53 Weighting Arrays for Various Differential Edge Detectors 56
54 Implementation of Differential Edge Detectors Using PIPE LSIC 57
55 Template Match Edge Detector Weighting Arrays 58
56 Implementation of Template Match Edge Detectors Using PIPE LSIC 59
57 Edge-Gradient Amplitude Response and Detected-Edge Orientation as Functions of Actual Edge Orientation 60
58 Probability of Detection Versus Probability of False Detection 61
59 Figure of Merit as a Function of Signal-to-Noise Ratio for Edge Detectors 63
60 Operation of Programmable Sum of Products Breadboard as 3 × 3 Neighborhood Operator 64

61 Sobel and Prewitt Edge Detection Using Programmable Sum-of-Products Breadboard 65
62 Template Match Edge Detection Using Programmable Sum of Products Breadboard 66
63 Real-Time Transform Processing Using Parallel PIPE LSIC 69
64 Block Diagram of Parallel PIPE LSIC to Implement Real-Time Neighborhood Operator Calculations 69
65 Block Diagram of the PIPE LSIC Demonstration Brassboard 70

LIST OF TABLES

Table *Title* *Page*

1 User-Defined Controls 9
2 Area and Estimated Power of the PIPE LSIC 10
3 PIPE LSIC Input/Output Data Rates 68

SECTION I
INTRODUCTION

A. OVERVIEW

Rapid advances have been made during the past several years in large-scale integrated circuit (LSIC) technology. These advances have had a significant impact on many military signal processing functions found in such applications as forward-looking infrared (FLIR) radar, guidance and control, and ECM systems. In particular, image processing system studies for video bandwidth reduction, FLIR automatic cueing, 3-D target classification, and image understanding have consistently recommended using LSIC technologies to perform critical image processing functions. A general-purpose algorithm[1] which uses the linear operation

$$y = \sum_{i}^{M} W_i X_i$$

on a single video line or on an n by n block of picture elements is an ideal candidate to be implemented with LSIC technologies.

While such algorithms can be executed easily at low data rates using general-purpose minicomputers or even commercial microprocessors, it is usually not possible to execute them in real time in an airborne environment because of excessive size, weight, power dissipation, and cost. The key to effective system design is to apply LSIC technology to minimize the overall component count and variety of components while absorbing as much as possible of the control and timing logic onto the information processing chip themselves. The solution is optimum when the same chips can be used for a multitude of other applications to provide a high volume market. These requirements have lead to the desire for a programmable chip architecture, and in turn, to the concept of a parallel/serial input bus. The discovery of the read-only memory (ROM) accumulate[2,3,4] algorithm to implement the above-mentioned linear operator could have significant impact on image processing systems.

B. OBJECTIVE

The objective of this 12-month exploratory development effort was to develop a general-purpose digital LSIC device called the programmable image processing element (PIPE), which can be programmed to perform a variety of signal processing functions such as Cosine and Hadamard transforms, edge extraction, unsharp masking, pole-zero filtering, and signal smoothing on 8 by 1 or 3 by 3 element data blocks at full TV data rates of 10 megasamples per second. Such a device should find widespread application in anti-jam video data links, FLIR automatic cuers, target classifiers, and digital filter processors. Hence, the PIPE LSIC is suited for those airborne applications where data rate, size, power and weight restrictions prohibit the use of general-purpose microprocessors or high-speed digital multipliers. This contract will address only the design and photomask fabrication of the PIPE LSIC.

C. SUMMARY

During the design phase of this contract, investigations of ROM technologies and designs to ensure a user-oriented image processing LSIC were conducted. The results of this investigation are

detailed in the PIPE LSIC design discussion given in Subsection II.A. A brief summary of the PIPE LSIC follows.

Texas Instruments was particularly well qualified to execute this contract successfully because of its low risk implementation approach. This low risk implementation approach is a result of:

- User-oriented PIPE LSIC architecture
- Established, cost-effective LSIC technology
- Proven erasable, programmable ROM design
- Breadboard emulator of PIPE LSIC.

Several architectures were investigated as candidates for implementing the PIPE LSIC. The architecture selected maximizes flexibility in algorithm implementation and minimizes external control and timing logic.

The N-channel metal-oxide-semiconductor (NMOS) technology was selected to implement the PIPE LSIC because it is an established, cost-effective technology capable of providing the high-circuit density, low speed-power product, and user erasable programmable read-only memory (EPROM) needed for the user-oriented PIPE LSIC architecture.

The NMOS erasable programmable read-only memory was judged to be the best technology to meet the goals of the PIPE LSIC. The advantages of NMOS EPROM technology are:

- User-programmable (electrically)
- Erasable (ultraviolet light)
- Nonvolatile
- Single supply voltage operation
- Static on-chip NMOS logic
- Military grade
- Established technology.

Texas Instruments has available in production quantities a family of military-grade EPROMs ranging in size from 8K (1K = 1024 bits) to 32K. This, along with the features of EPROMs, makes the EPROM technology ideal for the PIPE LSIC program.

The PIPE LSIC design completed during this contract is shown in Figure 1. As illustrated in this figure, there are five major sections of this integrated circuit:

- Input latch/parallel-to-serial shift register
- EPROM
- Shift-and-accumulate
- Tri-state output latch
- Controller.

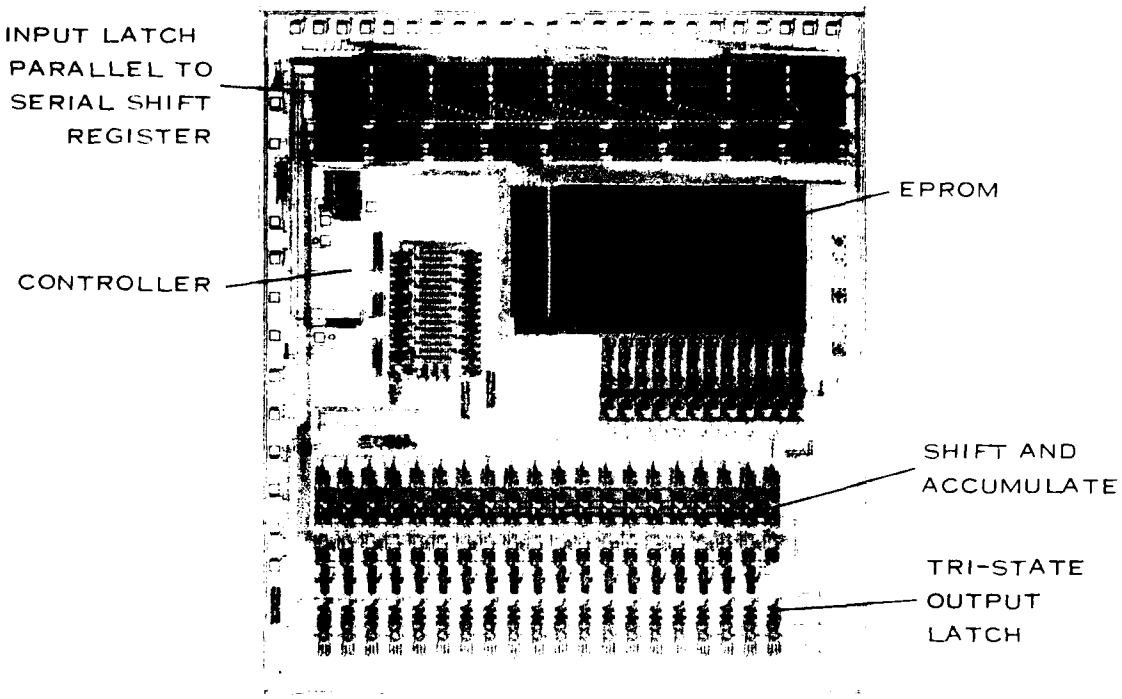


Figure 1. PIPE LSIC Layout

Each of these functions is discussed in detail in Subsections II.A.2 through II.A.6, respectively. The total bar size is approximately 240 by 270 mil^2, with a total estimated power of 600 milliwatts.

Texas Instruments believes the proposed Phase II PIPE program is very relevant and timely for the development of a programmable image processing element and has carefully considered the applications of the PIPE LSIC. The PIPE LSIC is ideal for 3 by 3 window operations and 8 by 1 or 9 by 1 matrix operations. A more detailed discussion of the applications of the PIPE LSIC is presented in Subsection II.B.

SECTION II
TECHNICAL DISCUSSION

A. PIPE LSIC DESIGN

Texas Instruments has designed a programmable image-processing-element (PIPE) large-scale integrated circuit (LSIC) during Phase I of the PIPE program, contract F33615-79-C-1763. Details of this design are discussed in this section.

1. Introduction

Many image-processing algorithms[1] require a sum of products operator of the form

$$Y = \sum_{i=1}^{M} W_i X_i \tag{1}$$

where the W_i represents a set of fixed programmable weighting coefficients, X_i represents a set or sequence of input values, and M represents the number of inputs. This mathematical function can be used to calculate the coefficients of various transforms used in many image-processing applications such as Fourier, Cosine, Hadamard, Haar, and others. The sum of products expression can also be used in determining many neighborhood operators which perform such operations as noise smoothing, edge enhancement, and edge crispening. For many image-processing applications such as video bandwidth reduction, forward-looking infrared (FLIR) autocueing, target classification, and image understanding, algorithms based on the sum of products operator are required.

The sum of products operation of Equation 1 can be implemented using digital integrated circuit (IC) multipliers and an accumulator, as shown in Figure 2. However, the size and power required to perform the multiplications with digital IC multipliers at video data rates are prohibitive for many airborne image-processing applications. Hence, investigations have been performed recently on techniques for the realization of the sum of products operation without using digital multipliers.[2,3,4] These distributed arithmetic techniques are a table look-up procedure to perform the multiplication in Equation 1. This table look-up operation replaces the digital multipliers with a read-only-memory (ROM) function. The principle of operation of the ROM-accumulate algorithms is discussed below.

As noted in Equation 1, the weighting coefficients, W_i, are fixed and known while the input words, X_i, are variable. In binary arithmetic, the input variable, X_i, can be expressed as

$$X_i = \sum_{j=0}^{N} e_{ij} 2^j \tag{2}$$

where the term e_{ij} is the binary value in each "digit" position; that is,

$$e_{ij} \in \{0, 1\}$$

By substituting Equation 2 into Equation 1, the following expression is obtained:

$$Y = \sum_{i=1}^{M} w_i \sum_{j=0}^{N} e_{ij} 2^j \tag{3}$$

Rearranging the terms yields

$$Y = \sum_{i=1}^{M} \left[\sum_{j=1}^{N} w_i e_{ij} \right] 2^j \tag{4}$$

Examination of the bracketed term in Equation 4 reveals only 2^M possible values for this term. Since the W_i are known *a priori* and the term e_{ij} can be only a value of 1 or 0, the bracketed term can be calculated for all possible combinations and stored in an ROM. Therefore, the particular e_{ij} from the incoming data can be used to address the ROM and fetch the associated value of the bracketed term. Once the particular value of the bracketed term is obtained, Equation 4 can be computed by shifting this bracketed term by one bit position before accumulating the previous sum. A block diagram of this ROM-accumulate algorithm is shown in Figure 3. Note that the X_i terms are bit-serial due to the rearrangement of terms in Equation 4. The size of the ROM required in Figure 3 is 2^M words, where M is the maximum number of input values, as given in Equation 1. The number of bits per word in the memory is the sum of the word length of W_i and the value of $\log_2 M$.

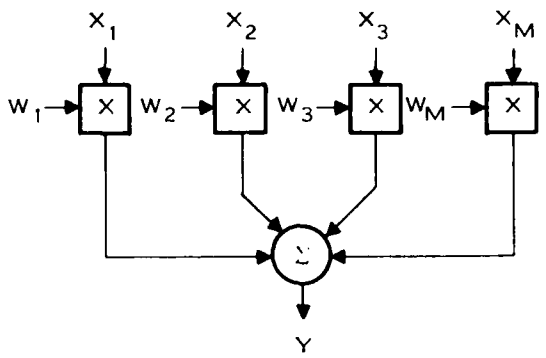


Figure 2. Block Diagram of the Multiplier-Accumulator Algorithm for Implementing Sum of Products Operator

As noted in the above discussion, the input data to the ROM is bit-serial. However, in most applications, the input data in a typical image-processing application is given in a parallel word format. Hence, the block diagram of Figure 3 is modified to change the input data from parallel format to bit-serial format, as shown in Figure 4. This input data can be reformatted with a parallel-to-serial shift register circuit. Also shown in Figure 4 is an output buffer, which buffers the outputs of the shift-and-accumulate function, and a controller function, which provides the necessary controls to the other functions. A brief discussion of the design of the PIPE LSIC follows.

The following design goals were used in the development of the PIPE LSIC. The PIPE LSIC will accept input data being ≤8 bits at a 20-MHz data rate. Data entry can be in 3 × 3 blocks or 9 × 1 blocks defined by the user. Data is entered via an input strobe at a 20-MHz rate. Output data is presented on a tri-state bus being ≤20 bits parallel. Output timing is compatible with the input, allowing device chaining. All inputs and outputs are TTL- and CMOS-compatible with a fanout of three. A master clock controls all data sequencing and timing. The PIPE LSIC will be user-programmable.

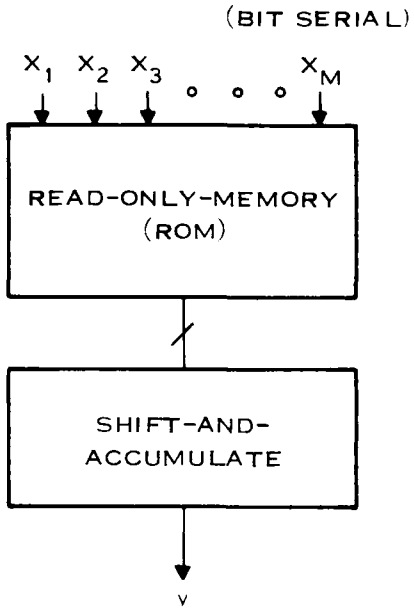


Figure 3. Block Diagram of the ROM-Accumulator Algorithm for Implementing Sum of Products Operator

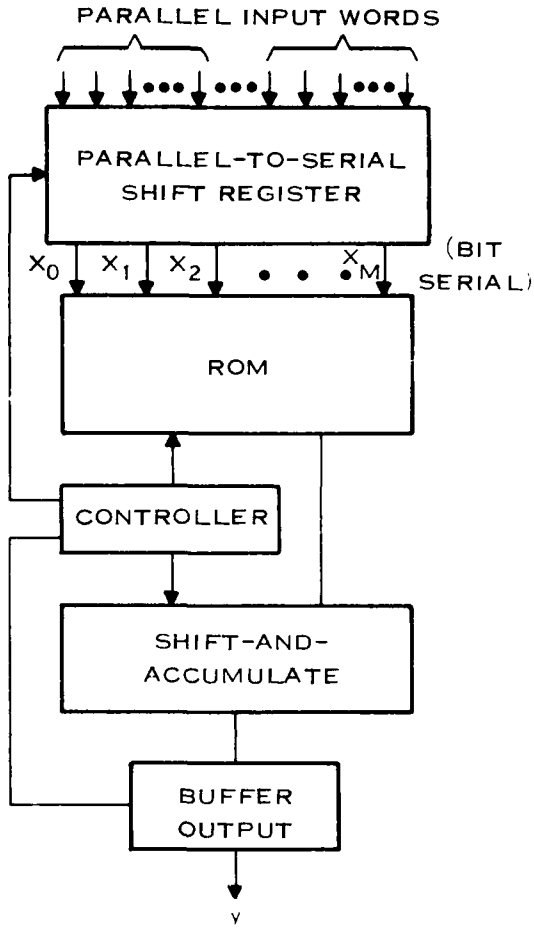


Figure 4. Block Diagram of ROM-Accumulator Algorithm With Parallel-to-Serial Shift Register, Buffer Output, and Controller

The simplified block diagram of the PIPE LSIC architecture is shown in Figure 5. As shown in Figure 5, the input data S, T, and P are parallel words that can be loaded into the input latches serially or in parallel. The particular mode of operation is controlled by the user through the use of one of the control input lines. This allows the PIPE LSIC to operate on 9 X 1 blocks or 3 X 3 blocks of data. In the serial mode, all data is loaded through the P input pins and latch, and then is sequentially clocked through the other latches. In the parallel mode, the data is loaded through the S, T, and P input pins into three separate input latches. The input data is then sequentially clocked into the other latches. In the serial mode, nine sample periods are required to load all of the input latches; in the parallel mode, three sample periods are required. This method of data entry eliminates the need for latch address pins and facilitates convolution and sliding-window operations with a single part.

The bit-parallel words in the input latches are converted into bit-serial words by the parallel-to-serial registers. The outputs of the parallel-to-serial registers form the 9-bit memory address. The memory outputs are shifted and accumulated to complete the sum of products operation. Tri-state

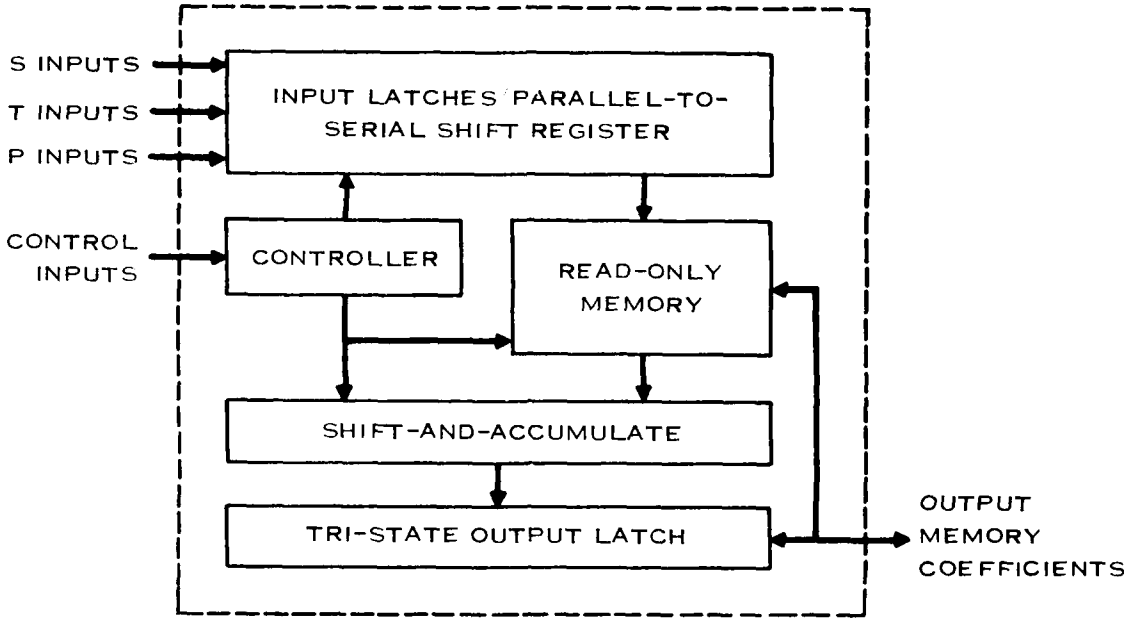


Figure 5. Simplified Block Diagram of the PIPE LSIC Architecture

output latches are provided for off-chip buffering. All timing and control pulses for the parallel-to-serial registers, the memory, the shift-and-accumulate, and the output buffers are generated on-chip using a simple shift register controller with a load pulse and master clock as inputs. The shift-and-accumulate circuitry will operate with either unsigned magnitude data or 2's complement data as selected by the user. The design is very user-oriented, both from the memory technology considerations as well as the number of control lines required to operate the chip. Table 1 defines the control inputs of Figure 5.

The PIPE LSIC design and layout have been completed. Figure 6 is the diagram of the IC barmap which is representative of the actual layout. Figure 7 is the CALCOMP plot of the PIPE LSIC. The major sections of this layout are:

- Input latches/parallel-to-serial shift register
- EPROM
- Shift-and-accumulate
- Tri-state output latch
- Controller.

The area and estimated power for each of these sections is given in Table 2. The area for each function in this table does not include the area used for lead routing. The total bar size is approximately 240 × 270 mils^2 with a total estimated power of 600 mW. Each of the functions listed above is

TABLE 1. USER-DEFINED CONTROLS

Control Line(s)	Function
Word length (3-bit BCD code)	Defines the word length in bits of the input data.
Parallel serial	Determines mode of chip operation; 3 × 3 or 9 × 1 operations.
Master clock	A square-wave clock provided for system timing (≤20 MHz).
Load	Initiates the parallel-to-serial data conversion (≤20 MHz).
Input strobe	Indicates valid input data and latches it in the input latches (≤20 MHz).
2's complement	Defines signed or unsigned magnitude data operation.
Enable	Used to tri-state or enable the output bus.
V_{DD}	Single +5-V operating supply
V_{PP}	Normally at 5 V but taken to 25 V for EPROM programming.
Data valid	An output signal indicating a complete computation.
Two's complement coefficients	Used to set the sign bits of the output word when <8-bit input data is used.

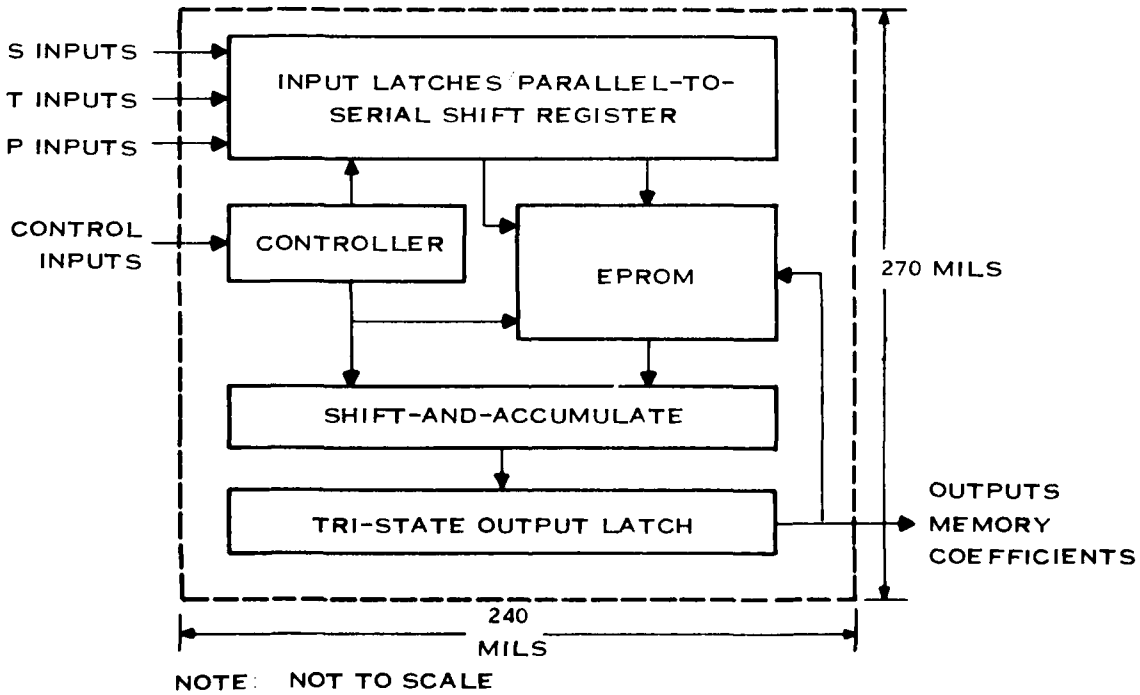


Figure 6. Diagram of the PIPE LSIC Barmap

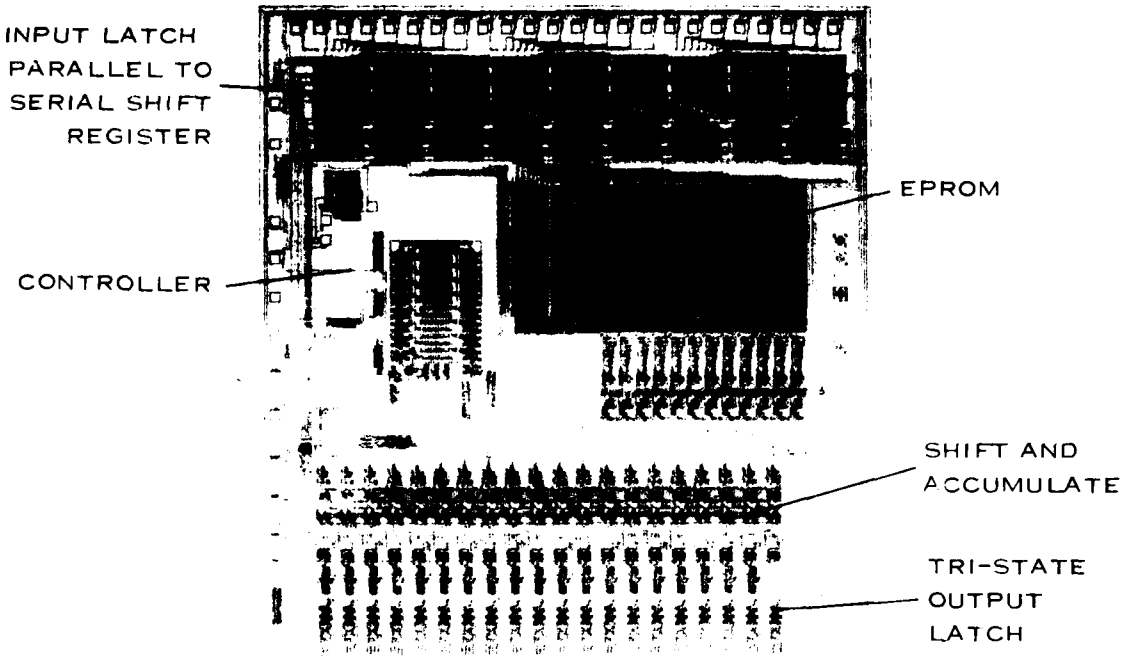


Figure 7. CALCOMP Plot of PIPE LSIC

TABLE 2. AREA AND ESTIMATED POWER OF THE PIPE LSIC

Function	Area (mil^2)	Power (mW)
Input latch, P-to-S shift register	8,400	175
EPROM memory	12,000	150
Shift-and-accumulate	10,800	95
Tri-state output latch	3,600	70
Control and timing	2,400	110

discussed in detail in Subsections II.A.2 through II.A.6, respectively. A parallel effort of a hardware demonstration of the sum of products operator using the ROM-accumulate algorithm is discussed in Subsection II.A.7.

2. *Input Latch Parallel-to-Serial Shift Register*

This subsection discusses the input latch parallel-to-serial shift register function of the PIPE LSIC. The block diagram of this input structure is shown in Figure 8. The main parts of this structure are:

- Input multiplexer and latch
- Parallel-to-serial shift register
- Memory address drivers
- Memory address select.

A functional block diagram of a single input stage (1 of 9) is shown in Figure 9. Transistors (M1 through M16) form the input multiplexer; latches L0 through L15) form the input latch; registers Reg 1 through Reg 14 form the parallel-to-serial shift register; Reg 0 is the memory driver register with increased drive capabilities; and M17 is the address select multiplex transistor used in programming the EPROM. The input multiplexer switch transistors (M1 through M16) select a data path for serial or parallel data input operation. This selection is made possible by the parallel or serial (PAR/SER) select line (TTL level) which is buffered to an MOS level by inverter I1 as shown in Figure 9. Inverter I2 provides the complement of the PAR/SER line for selecting the multiplexer

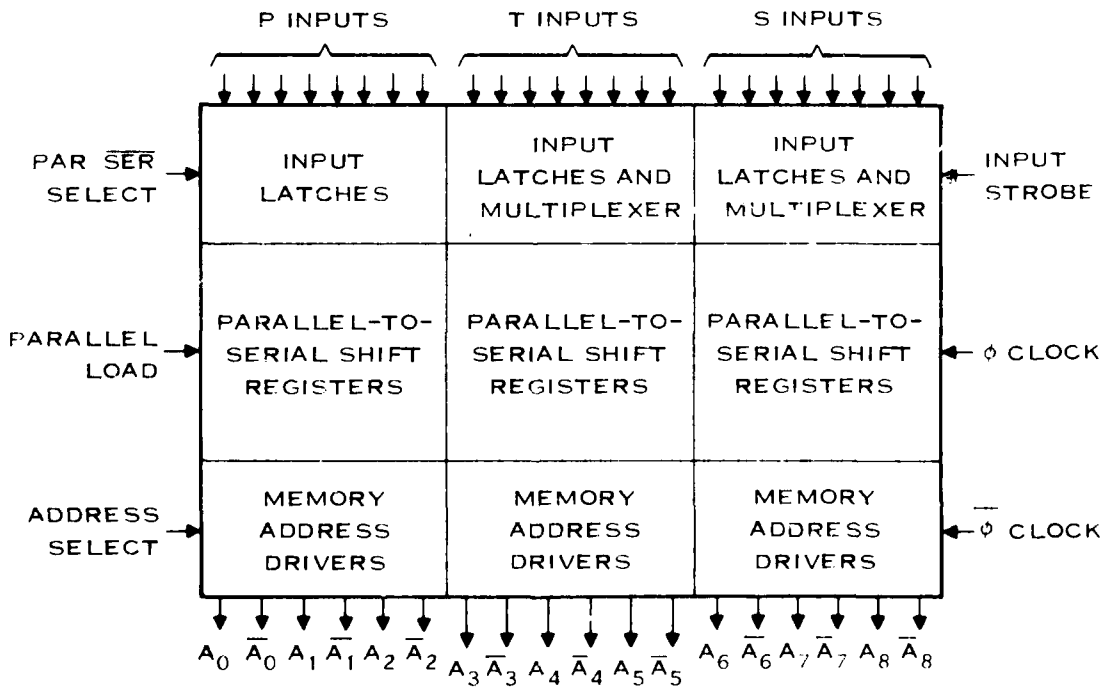


Figure 8. Block Diagram of the Input Structure

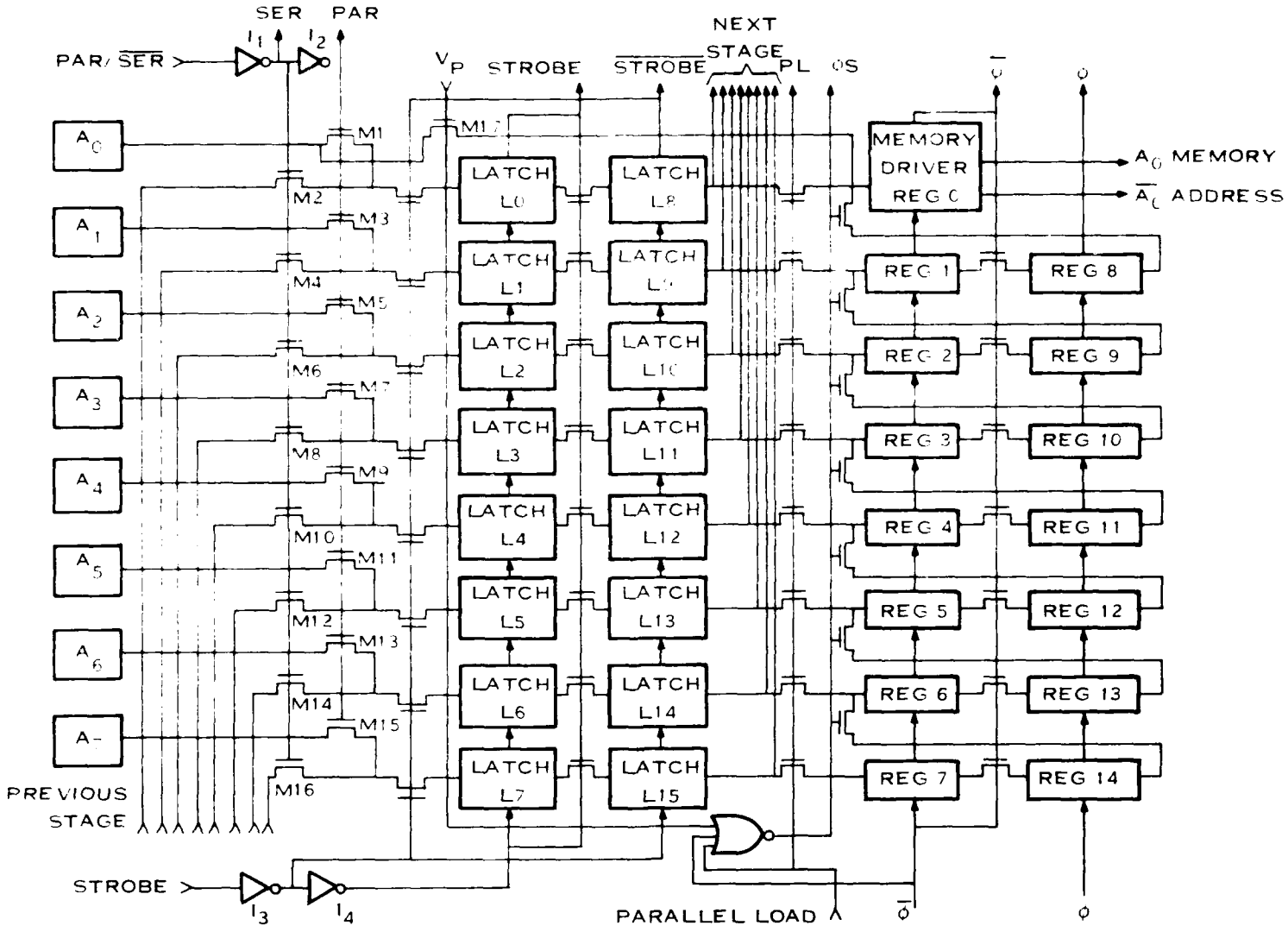


Figure 9. Block Diagram of the Input Multiplexer Latch Parallel-to-Serial Register

switch transistors (M1 through M16). Once a data path has been selected, the data proceeds through latches, L0 through L7 and awaits entry to latches L8 through L15. The data is latched into these registers when a strobe pulse is given. This latch takes place on the rising edge of the strobe (STROBE) pulse. The data is held in these latches regardless of changes appearing on the input data line since latches L0 through L7 are disabled on the rising edge of the strobe pulse. The strobe pulse (TTL level) is buffered by inverter I3 and complemented by inverter I4 to provide the proper levels. After the data is latched (I9 through L15), it is then ready for the parallel-to-serial The parallel-to-serial conversion takes place in registers R1 through R14. On each phase shift (ΦS) clock, data is transferred upward toward the memory address driver. Figure 10 shows a timing diagram of this function.

A detailed discussion of the input multiplexer and latch, parallel-to-serial shift register, memory address drivers, and memory address select circuits follow. The last topic discussed in this subsection is the level converter and input protection circuit.

a. Input Multiplexer and Latch

Figure 11 shows a single-input multiplexer. Transistors M1 and M2 provide two different data paths (A or B). The SELECT line is presented to the gate of M1 and its complement to M2. A high logic level on the SELECT line will turn transistor M1 on and allow data on the A line to propagate to point C. Similarly, taking the SELECT line low will produce a logic high on the gate of M2; then, data on line B is allowed to propagate to point C.

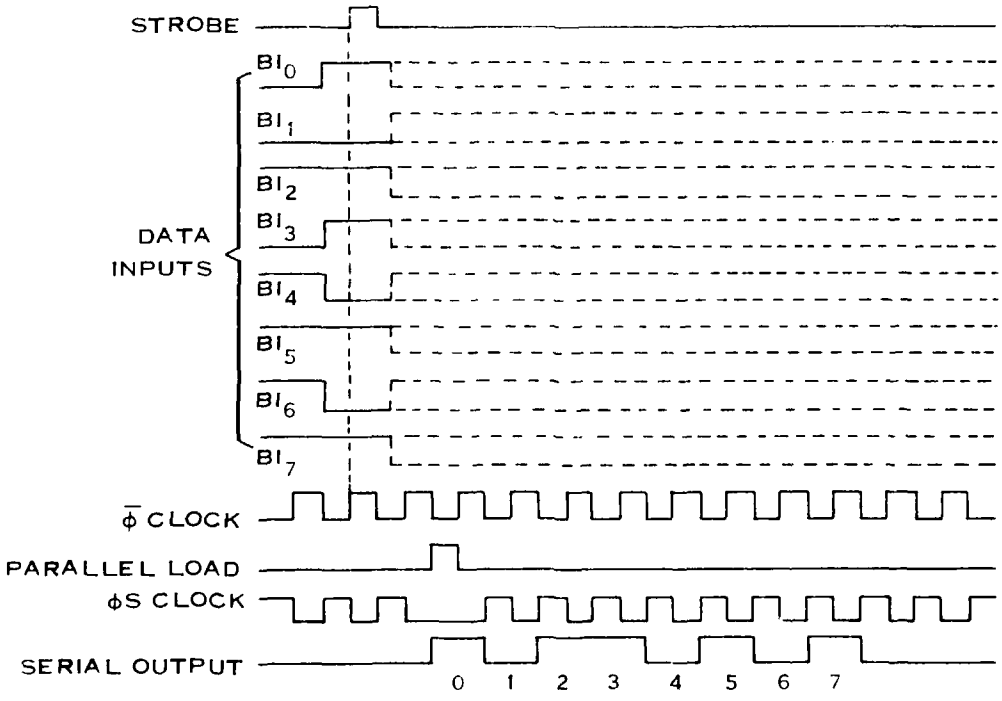


Figure 10. Timing Diagram of the Input, Load, and Shift Sequences

The input latch circuitry is implemented by using two N-channel MOS (NMOS) latch structures. Since the NMOS latch structure is the basic building block that is used in the input latch circuitry as well as in other portions of the PIPE LSIC, a brief discussion of this circuit is given.

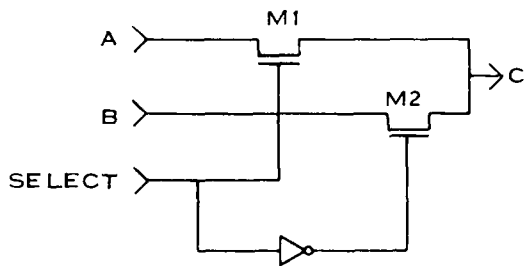


Figure 11. Circuit Diagram of the Input Multiplexer

The block diagram and schematic of an NMOS latch are shown in Figure 12. This latch is composed of an input data path, transistor M1, followed by two NMOS inverters (M2, M3) and (M4, M5), and a feedback path through transistor M6. These two inverters are constructed with depletion loads (M2, M4) and enhancement drivers (M3, M5).

The operation of the latch circuit is as follows. Data enters the latch through M1 and appears at node 1 when the LATCH line is high (logic 1). If the data entering is high, then M3 will conduct, pulling node 2 low (logic 0). Since the gate of M5 is connected to node 2, M5 will turn off transistor and let transistor M4 pull node 3 high, which is the same information at the input. The $\overline{\text{LATCH}}$ line can now be pulled low, causing the LATCH line to go high. This high level on the gate of M6 will cause M6 to conduct, providing a feedback path from node 3 to node 1. Since M1 is now in a

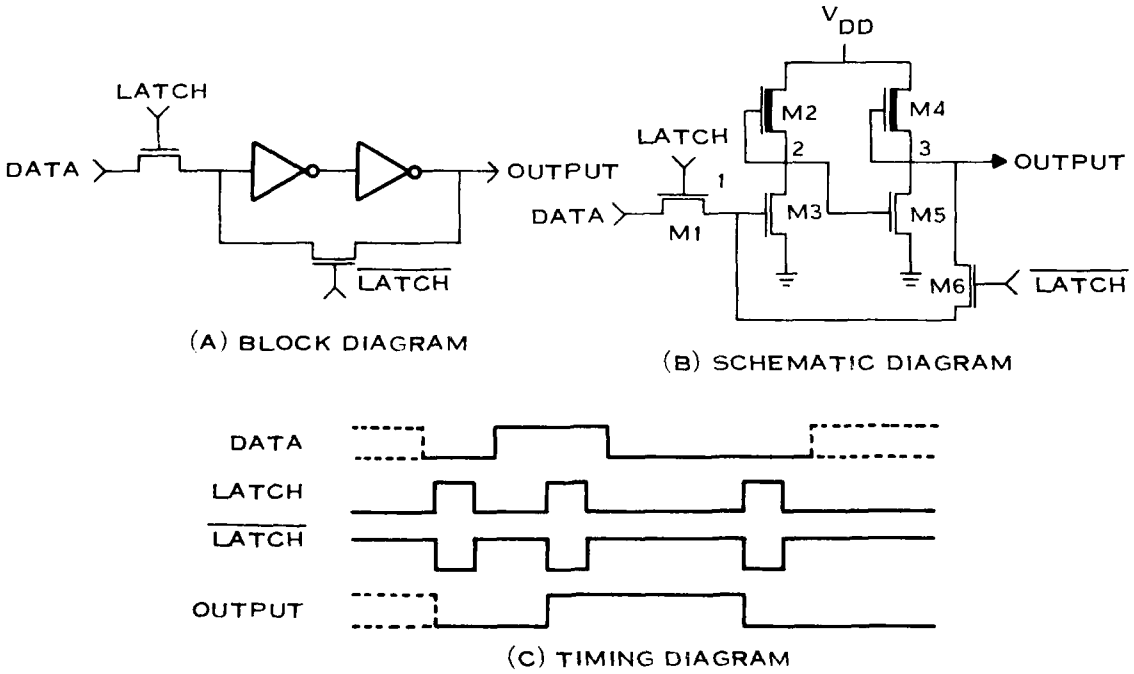


Figure 12. Typical NMOS Latch Circuit

nonconducting state, data changing on the input DATA line will have no effect on the latched data. If the data initially entering the latch is low, a low level will be latched similarly. A timing diagram of possible inputs and latching pulses is also shown in Figure 12.

The input latch circuitry incorporates a D-type latch action, as shown in Figure 13. This edge-triggered circuit provides greater noise immunity and shorter data-valid lengths than a level trigger latch. The D-type latch function is implemented by using two NMOS latches, as discussed above, clocked on opposite phases of the strobe (STROBE) pulse. The STROBE pulse is provided by the user while $\overline{\text{STROBE}}$ is generated on the LSIC.

Figure 14 is a partial block diagram of the PIPE input structure. The input data can enter this structure either through the S input lines or from a previous stage determined by the parallel or serial (PAR/SER) control line. Data enters latch DL2 when the strobe line is pulsed. On each successive strobe pulse, data shifts to the next latch. For example, consider that the strobe line is pulsed high and data enters from the S inputs to latch DL2. On the next strobe pulse, the data now in DL2 will shift to DL1 and the new data will enter DL2. On the next strobe pulse, data in DL1 enters DL0, data in DL2 enters DL1, and the new data enters DL2. Once all of the input latches have been loaded, the parallel-to-serial conversion can be initiated.

b. Parallel-to-Serial Shift Registers

A partial block diagram of the parallel-to-serial shift registers is shown in Figure 15. Parallel input data enters through a control multiplexer when the parallel load (PARALLEL LOAD) line is

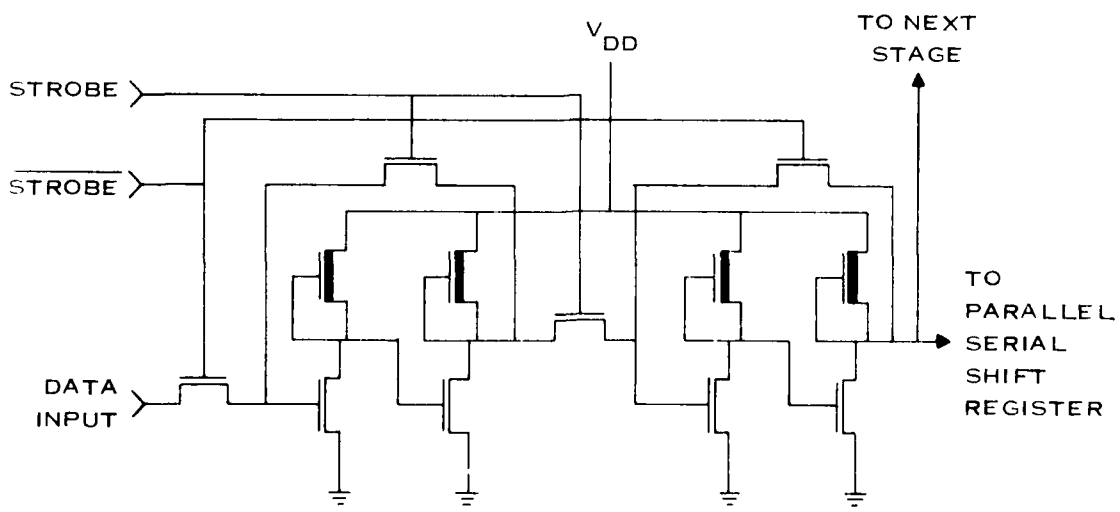


Figure 13. Schematic Diagram of the Input D-Latch

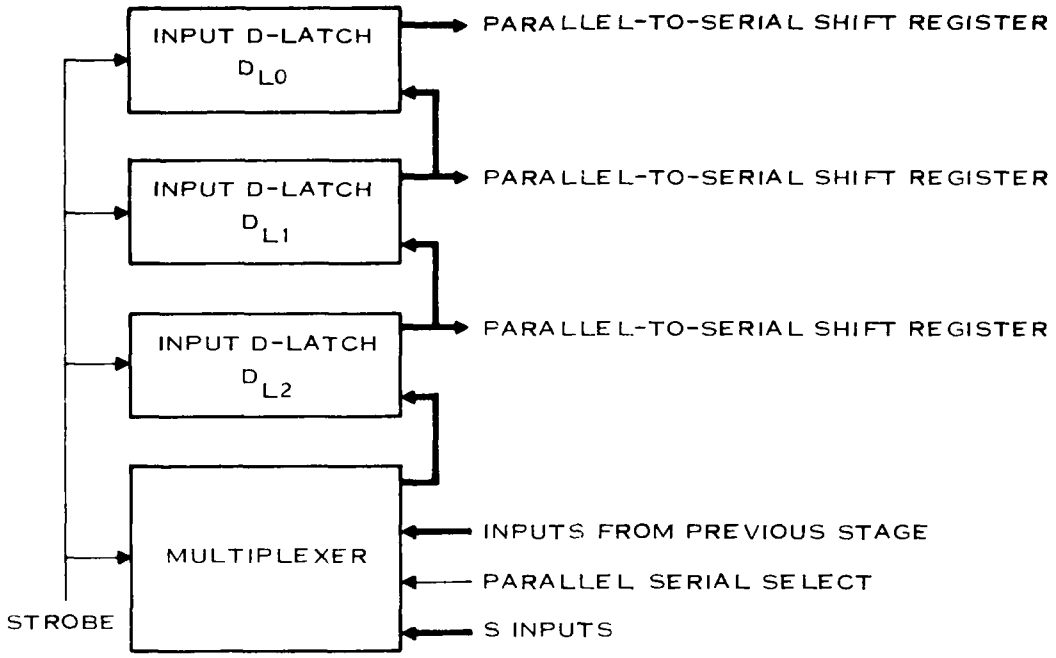


Figure 14. Partial Block Diagram of the Input Stage Architecture (3 Stages of 9)

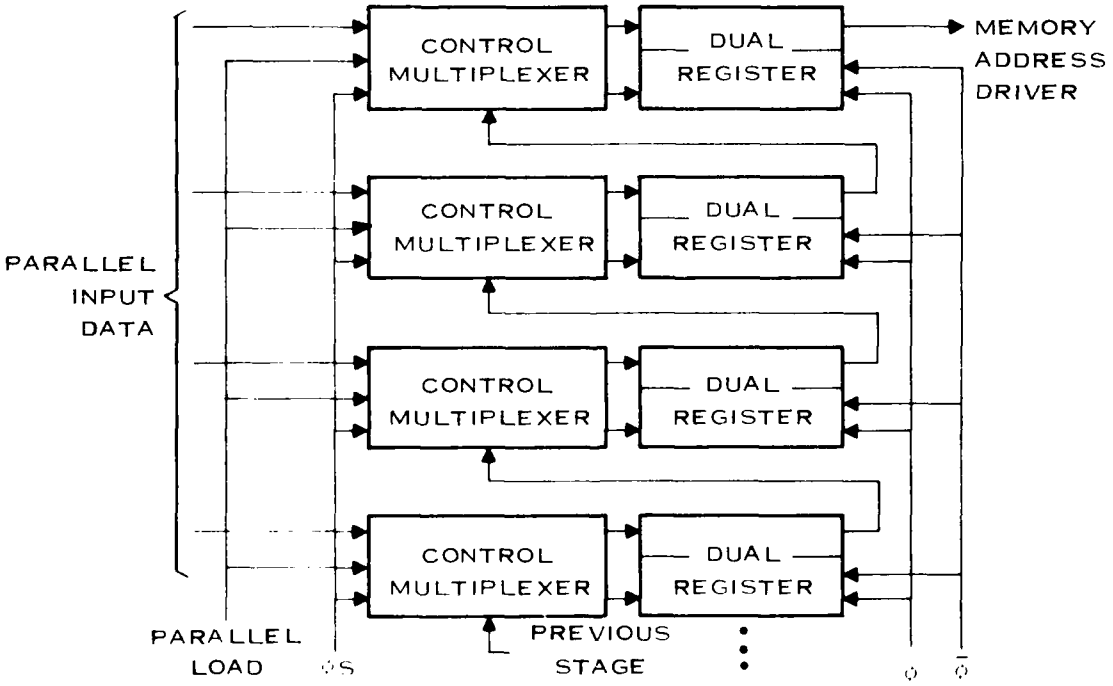


Figure 15. Partial Block Diagram of the Parallel-to-Serial Conversion (4 Bits of One 8-Bit Word)

pulsed, and enters the dual register (two NMOS latches). Once the PARALLEL LOAD line returns to a low logic state, the phase shift (ΦS) clock will shift data bits from the previous register to the next register. The phase shift clock is generated on-chip by OR-ing the parallel load, Φ clock, and the V_p voltage level. This operation is synchronous with the two-phase nonoverlapping clocks (Φ and $\bar{\Phi}$). These clock pulses are generated on-chip from the master clock supplied by the user.

c. *Memory Address Drivers*

The memory address driver is the last stage in the parallel-to-serial shift register. The circuit diagram of the memory address driver is shown in Figure 16. This circuit receives each shifted input data bit and latches this data for one clock cycle. The latched data is buffered to drive the memory address decoders.

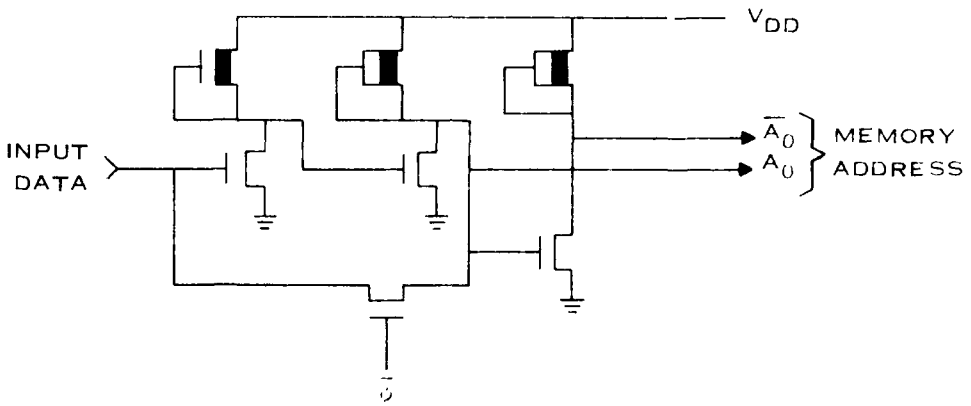


Figure 16. Circuit Schematic for Memory Address Driver and Latch

d. *Programming Address Select*

The V_{pp} line is taken to 25 volts during the programming of the EPROM. A special V_{pp} voltage divider (Figure 17) enables the programming address select multiplexer so that the input data lines can be used as memory address inputs (Figure 18). The V_{pp} voltage divider circuit also provides the program enable (PE) for the EPROM. This will make the memory easy to program and no special input bit shifting is required. In normal operation, the V_{pp} line is kept at a 5-volt level.

e. *Level Converter and Input Protection*

The PIPE IC operates on a single +5-volt supply (V_{DD}), and all inputs and outputs are CMOS/TTL-compatible. Most of the on-chip circuitry requires a 0- to 5-volt level for logic lows and highs, respectively. Therefore, a level inverter is necessary to convert TTL levels to the MOS levels needed. A TTL-to-MOS input buffer is shown in Figure 19. It consists of two inverters in series. Also, input protection from static damage is provided on the input M1 and R1. This level buffer is used on the appropriate inputs.

3. EPROM

Several semiconductor ROM technologies were considered as candidates for implementing the memory function of the ROM-accumulate algorithm. Prime considerations in the selection of the optimum ROM technology for the PIPE LSIC are:

- User-programmable
- Ease of reprogramming
- Military environment constraints
- Single supply voltage operation
- Static on-chip logic
- Established technology
- Nonvolatile.

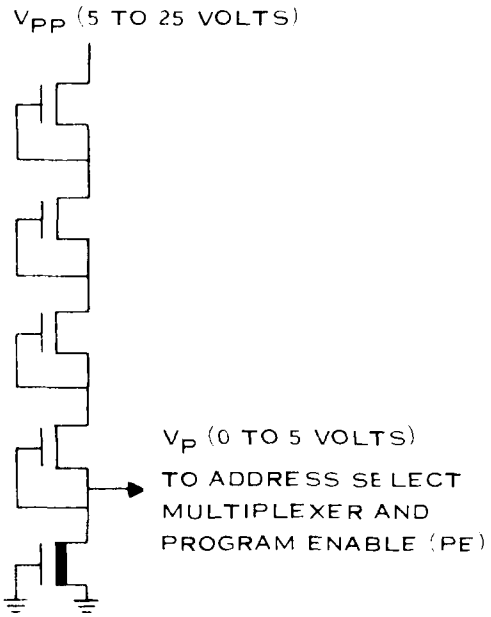


Figure 17. V_{pp} Voltage Divider Circuit

The electrically erasable, programmable ROM (EPROM) N-channel MOS (NMOS) technology was judged to be the best technology to meet the requirements of the program. The advantage of an EPROM technology is that it is user-programmed and not mask-programmed. Another important advantage of the EPROM is that the program can be electrically erased. A transparent quartz window covers the EPROM package. Erasing the EPROM is a simple matter of exposing the window to ultraviolet light. After erasing, the EPROM can be programmed again.

The EPROM technology allows for a common chip in the inventory to be personalized for a particular algorithm but it can also be easily reused later for an entirely different algorithm.

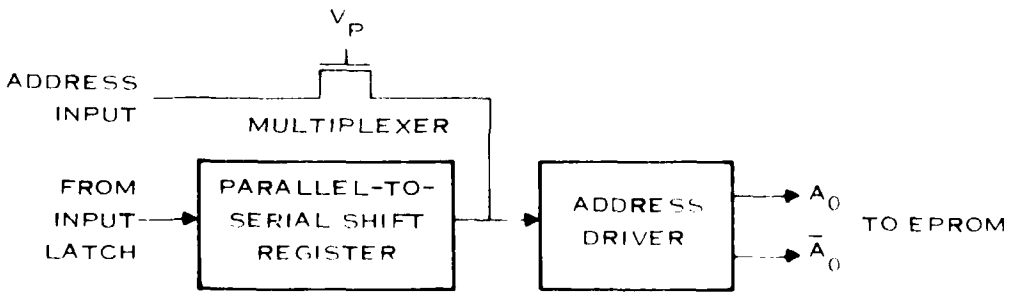


Figure 18. Programming Multiplexer

Another important consideration is that EPROM devices are available which operate over the full military range of −55°C to +125°C. Texas Instruments has in production a military EPROM family ranging from 8K (1K × 1024 bits) to 32K sizes. The SMJ2508 device is an 8K part organized as a 1K × 8 erasable and reprogrammable memory. This EPROM technology is also non-volatile, which has the advantage that the supply voltage can be removed from the chip but the memory coefficients still remain in the memory. Furthermore, this EPROM technology has a single 5-volt power supply requirement. The NMOS technology allows for static on-chip logic. The above mentioned EPROM technology is in high volume production.

For the above-mentioned reasons, the EPROM technology [illegible] used for the SMJ2508 device [illegible] is used for the PIPE-ESIC. This ROM technology [illegible] for [illegible] resource of the alterable reprogrammable memory, thus increasing the [illegible] of the PIPE-ESIC.

The simplified block diagram of the EPROM [illegible] is [illegible] in Figure 20. [illegible] address inputs are obtained from the bit serial input [illegible] outputs are presented to the shift-and-accumulate circuit in order to complete [illegible] operation. The control signals used in the memory are derived by the controller [illegible] (see Subsection II.A.6). The memory can be programmed by loading the addresses provided by the programming address select circuit (see Subsection II.A.2) while the memory coefficients are [illegible] through the tri-state output lines.

Figure 21 is the functional block diagram of the EPROM [illegible] PIPE-ESIC. The memory architecture has been arranged in a 512 × 64 × 8 × 12 [illegible] to optimize chip layout considerations. The central memory core is an array of 12 bit planes, each [illegible] composed of a 64 × 8 matrix. A single memory cell of one 64 × 8 bit plane, shown in Figure [illegible] consists of eight identifiable storage transistors. These transistors are physically built [illegible] floating gate structure (Figure 22B) where charge may be stored on this gate, depending on the logic state desired. This logic state may be defined when the device is in the programming mode. Figure 22C shows the schematic representation of the EPROM storage devices. The Texas Instruments military SMJ 2508 (1K × 8) process and design rules were used in the design and layout of the 512 × 12 memory structure. In fact, the memory cell is identical to the SMJ 2508 memory cell.

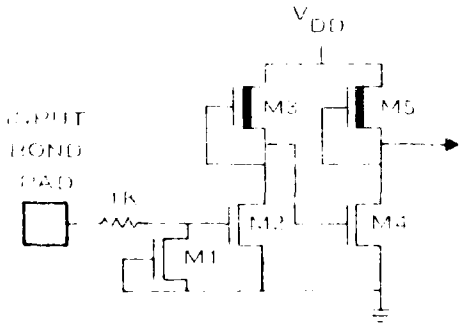


Figure 19. TTL-to-NMOS Level Input Buffer With Static Protection

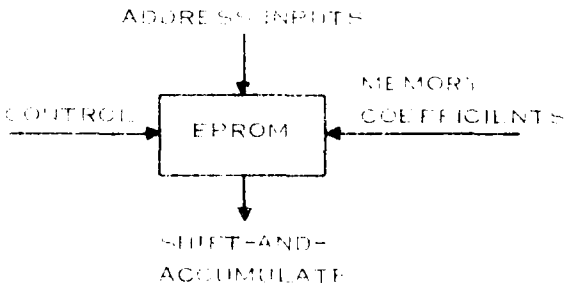


Figure 20. Simplified Block Diagram of the EPROM Section

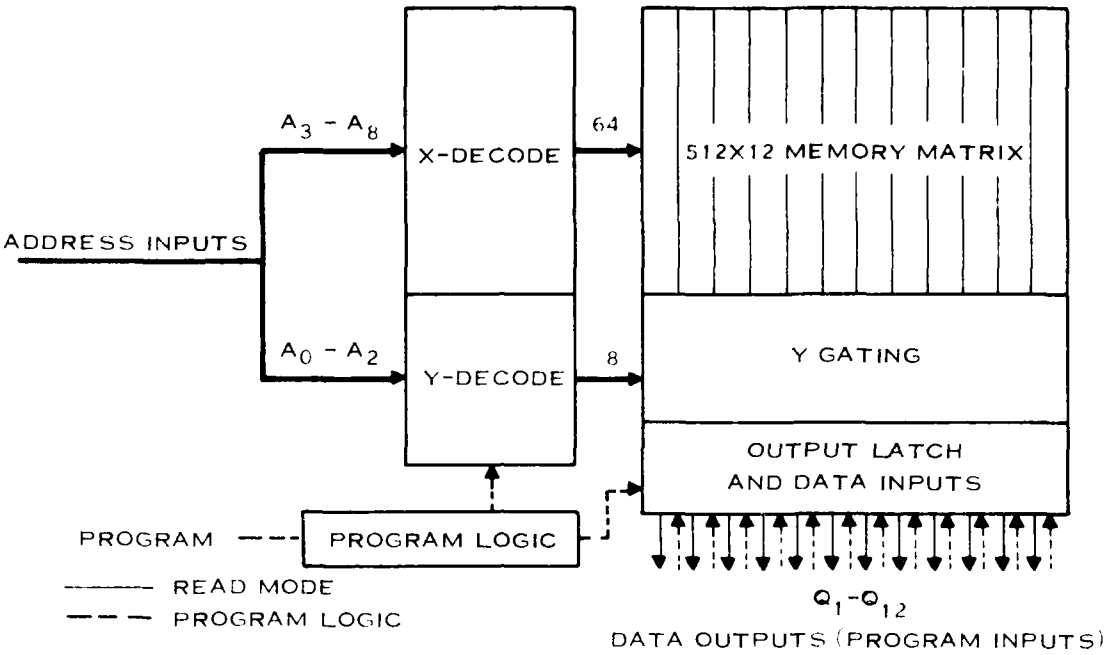


Figure 21. Block Diagram of the 512 × 12 EPROM Configuration

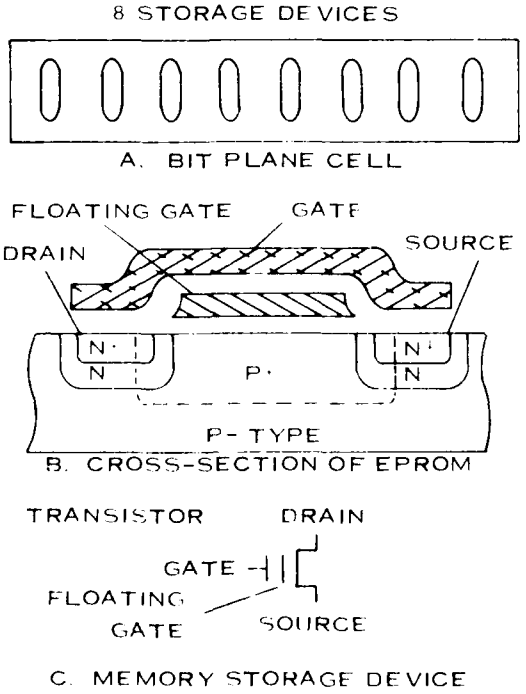


Figure 22. EPROM Cell Structure

By supplying the correct addresses (A_0 through A_8), the X- and Y-decoders select the proper 12 memory storage devices where logic highs or logic lows have been previously stored. These 12 logic levels (bits) are then latched into the memory data latch, which feeds the shift-and-accumulate circuitry.

The X and Y decode circuitry is shown in Figure 23. The input address and its data complement are hard-wired to the decode driver transistors (M0 through M8), providing the necessary BCD to decimal decode. The six drivers for the X decode will provide 2^6 different combinations, resulting in 64 possible selections. At the same time, the Y decode provides 2^3 combinations, resulting in eight possible selections; thus, by using a 9-bit address, it is possible to select 2^9 or 512 storage locations.

Once the address is supplied and the proper storage location has been selected, it is

then necessary to determine the logic level stored, being either a logic high or a logic low. This function is performed by the sense amplifier shown in Figure 24.

When the proper selection has been made by the decode circuitry, the sense amplifier determines the logic state stored there by measuring the charge stored on the floating gate of the storage transistor. Once this level is detected, it is buffered to drive the output memory latch that feeds the shift-and-accumulate circuitry.

Before programming, the memory is erased by exposing the chip through the transparent window to high-density ultraviolet light (wavelength 2,537 angstroms). The recommended minimum exposure dose is 15 watt-seconds per square centimeter. After erasure (all bits are in logic high state), logic lows are programmed into the desired locations. A low level can only be erased by ultraviolet light. The programming mode is achieved when V_{PP} is taken to 25 volts. Data is presented in parallel 12-bit words on Q_1 through Q_{12} of Figure 21. The corresponding low bits are programmed into the memory by taking the lines low for 50 ms.

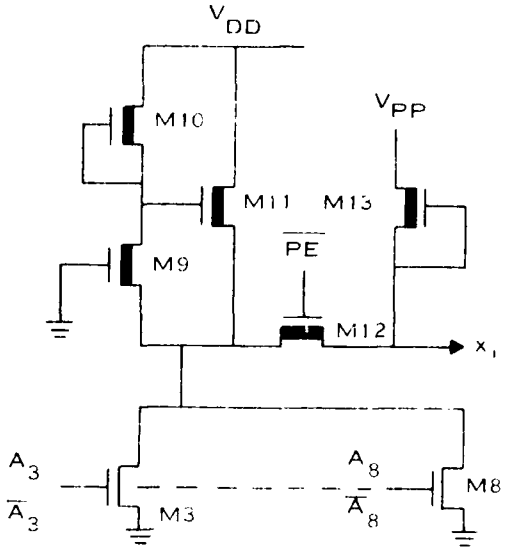
4. Shift-and-Accumulate

Figure 25 is a block diagram of the shift-and-accumulate circuitry which constructs a sum-of-products from a bit-by-bit parallel serial arithmetic operation.

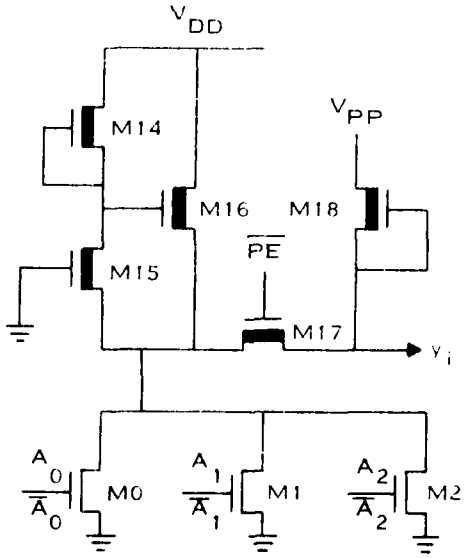
There are 12 data inputs that come from the EPROM section, 8 control lines, and 20 output lines. Since the ROM-accumulate technique produces no round-off error, all 20 bits are saved and presented as outputs.

Figure 26 shows the operational structure of the shift-and-accumulate circuitry.

Twelve-bit memory data is presented to the full adder "A" inputs on the rising edge of the phase clock, Φ, and latched on the falling



A. X-DECODE
(1 OF 64)



B. Y-DECODE
(1 OF 8)

Figure 23. Schematic Diagram of the X and Y Matrix Decoders

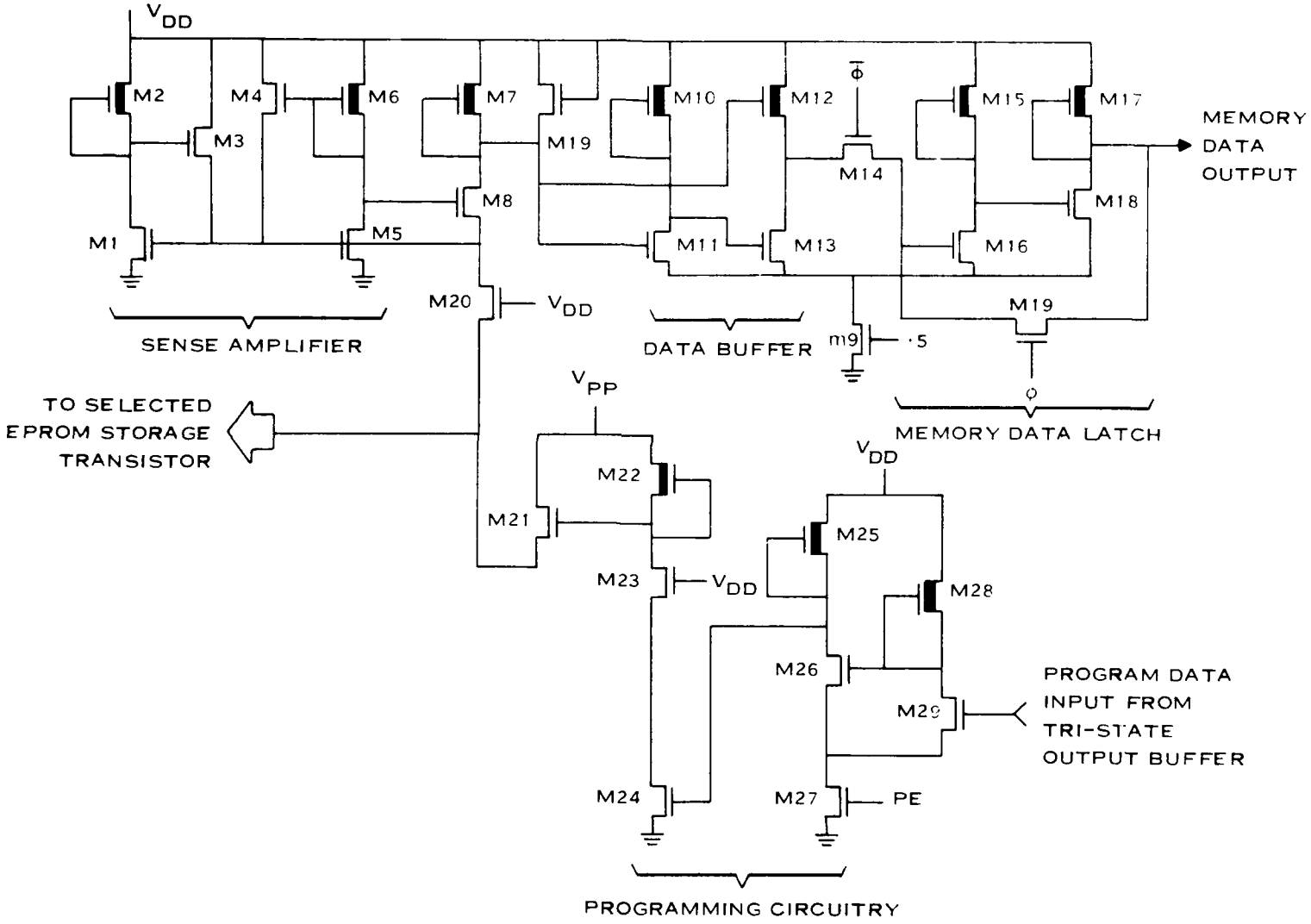


Figure 24. Schematic Diagram of the Read, Write, and Buffer Circuitry

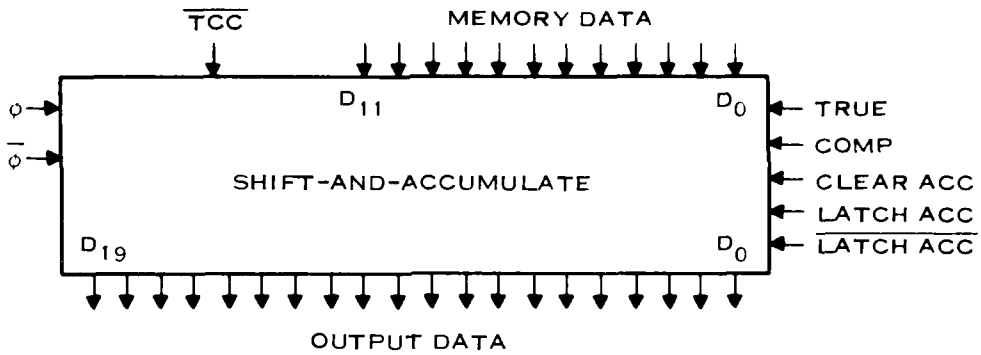


Figure 25. Block Diagram of the Shift-and-Accumulate Section

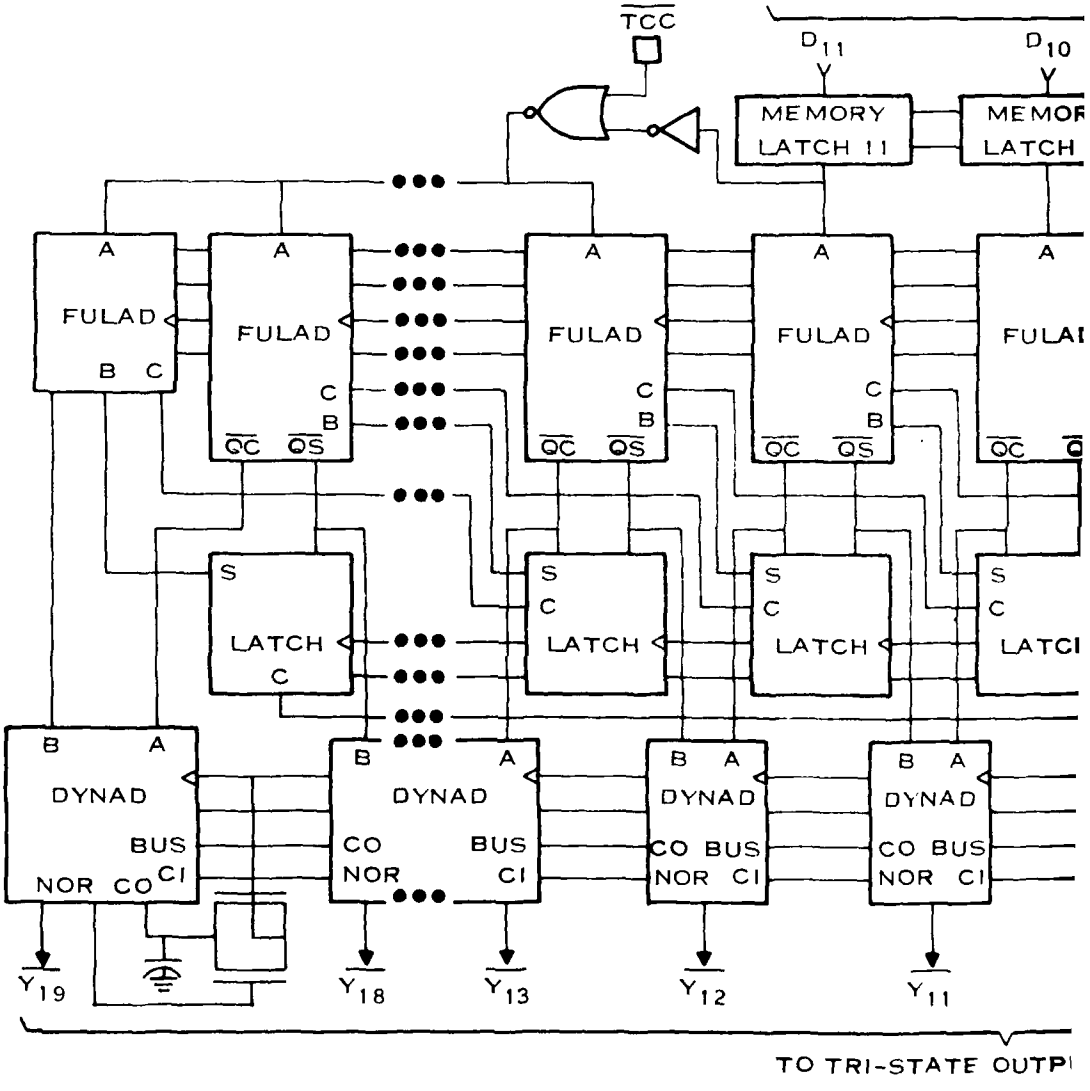
edge of Φ. The sum-and-carry results are latched into a master-slave register on the next rising edge of Φ. By feeding the sum-and-carry latch outputs forward (left) to remaining full adder inputs, a binary multiplication is performed per each clock cycle.

The sum-and-carry latches are cleared on the rising edge of the clear accumulator (CLEAR ACC) pulse which initiates the shift-and-accumulate operation for a given input data block. Sign magnitude or 2's complement memory data can be used by activating the true (TRUE) pulse or complement (COMP) pulse full adder control lines, respectively. Individual sums and carries are latched into the output adder on the falling edge of the latch accumulate (LATCH ACC) pulse, and the final result is valid on the next rising edge of the LATCH ACC pulse.

The total computation requires 2B clock cycles per input data block, where B is equal to the number of bits/input word.

The logic implementation and truth table for the PIPE LSIC full adder function are shown in Figure 27. Note that the outputs of this circuit are complemented sum (QS) and carry (QC), allowing a minimum number of components. The full adder (FULAD) schematic diagram is shown in Figure 28. The addend input includes a latch from which the true or complemented data may be selected. Transistors M1, M2 through M5, M6, and M7 and M8 are a pass gate, a two-stage inverter, a feedback gate, and select gates, respectively. Transistor M14 is the load device for the carry NOR, with the series string of transistors, M9 and M10, ANDing the addend and augend inputs, the paralleled devices, M12 through M13, ORing the addend and augend, and the series device, M11, ANDing the OR result with the previous stage carry. Comprising the sum NOR are the following: the load device, M22, the paralleled transistors, M15 through M17, that OR the three inputs; the series transistor, M18, that ANDs the OR result with the $\overline{\text{CARRY}}$ output; and the series string of transistors, M19 through M21, that ANDs the three inputs. This circuit requires only true level inputs and a total of 16 devices, excluding the latch.

A $\overline{\text{TCC}}$ (2's complement coefficient) control line is provided by the user. When operating with signed magnitude input data less than 8 bits in word length, the $\overline{\text{TCC}}$ line can be taken to a digital low to ensure that the upper sign bits are set correctly.


TCC
D11
D10
MEMORY
LATCH 11
MEMOR
LATCH
A
FULAD
B C
C
B
QC QS
FULAD
FULAD
FULAD
FULAI
S
LATCH
C
LATCH
LATCH
LATCI
B A
DYNAD
BUS
NOR CO CI
CO
NOR
CO BUS
NOR CI
Y19
Y18
Y13
Y12
Y11
TO TRI-STATE OUTP

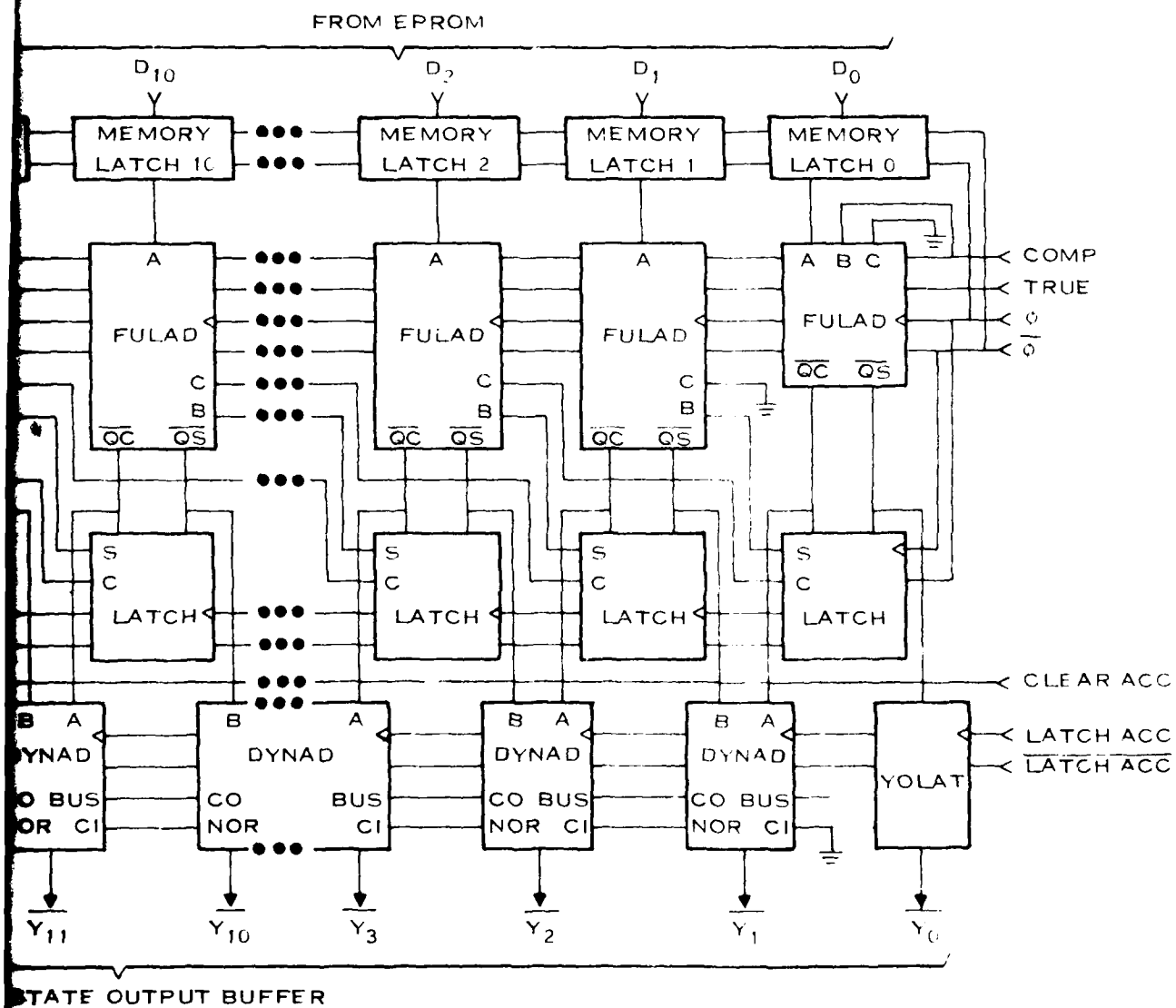
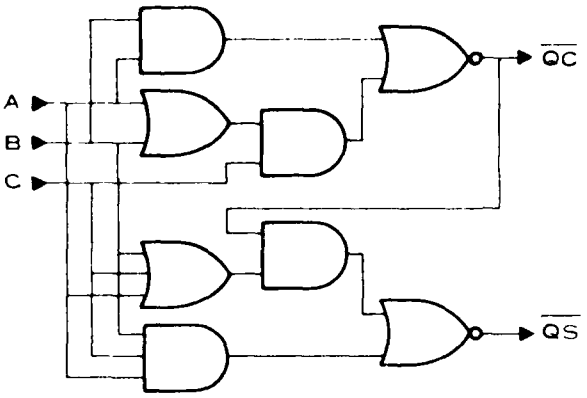
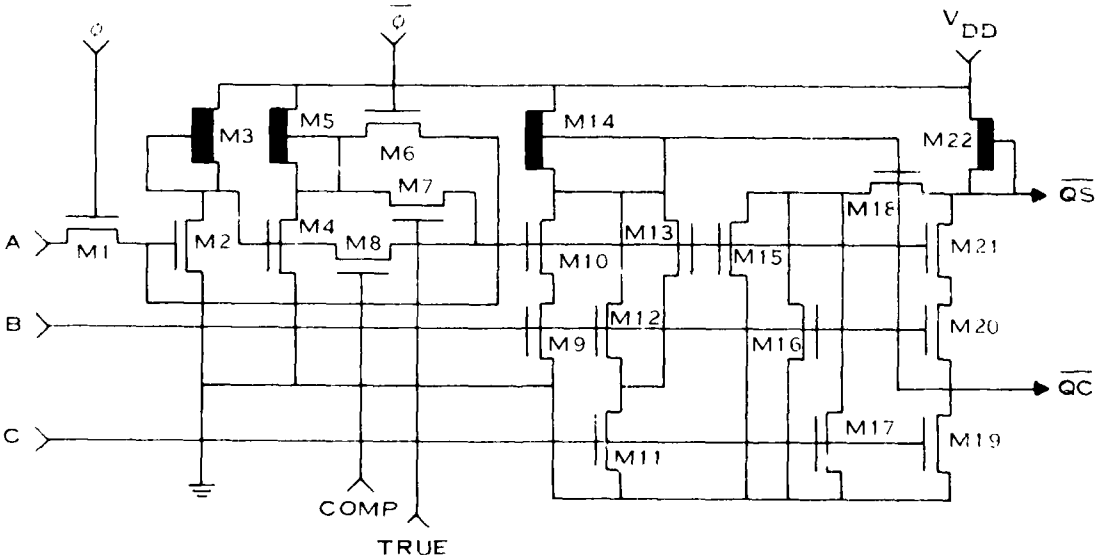


Figure 26. Block Diagram of the Shift-and Accumulate Section



INPUTS			OUTPUTS		
A	B	C	$\overline{QC}$	$\overline{QS}$	
0	0	0	1	1	
1	0	0	1	0	
0	1	0	1	0	
1	1	0	0	1	
0	0	1	1	0	
1	0	1	0	1	
0	1	1	0	1	
1	1	1	0	0	

Figure 27. Logic Diagram of the Full Adder



$\overline{QC} = \overline{AB + (A + B)C}$

$\overline{QS} = \overline{(A + B + C)\overline{QC} + ABC}$

Figure 28. Schematic Diagram of the Full Adder With Input Data Latch

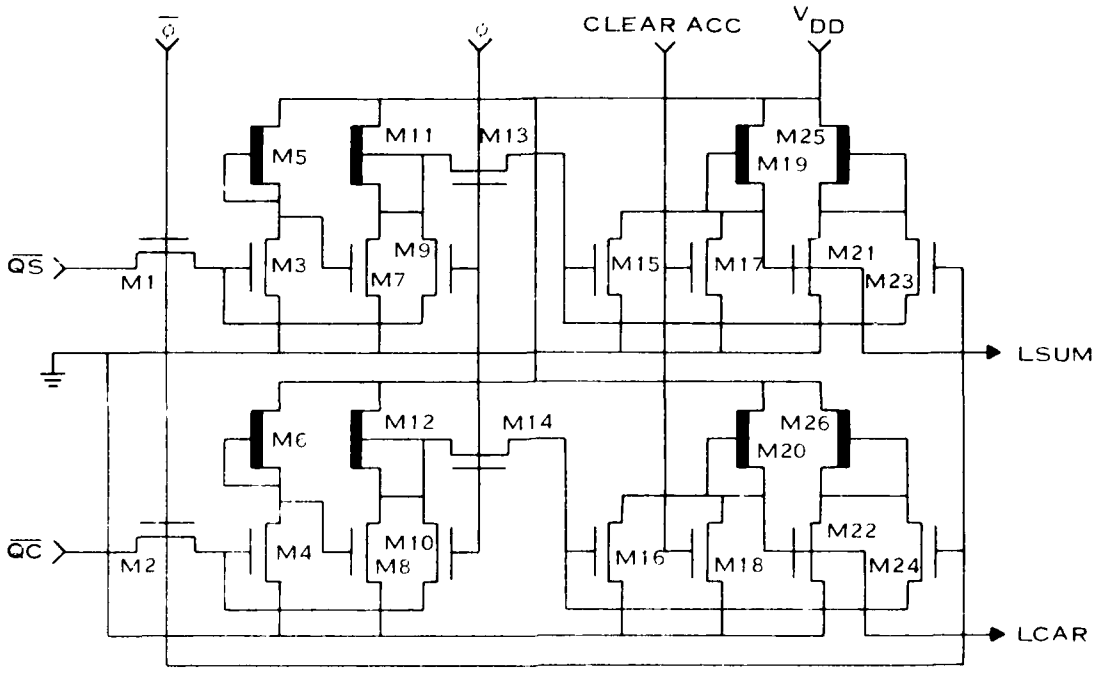


Figure 29. Schematic Diagram of the Carry/Sum Latch

The sum-and-carry latch shown in Figure 29 is an inverting dual master-slave configuration which accepts data on the falling edge of Φ, presents complemented data at the output on the next rising edge of Φ, and latches the output on the subsequent falling edge of Φ. Pass transistors M1 and M2 and feedback transistors M23 and M24 are enabled by Φ, while pass devices M13 and M14 and feedback devices M9 and M10 are enabled by Φ. Transistors M3 through M8 and M11 and M12 implement the dual master two-stage inverters, and transistors M15 and M16, M19 through M22, and M25 and M26 implement the dual slave two-stage inverters. Devices M17 and M18 are paralleled with M15 and M16, respectively, in the dual first-stage slave inverters to provide a clear capability.

The output adder circuitry uses the charge storage capabilities of NMOS devices to implement a compact, high-speed, look-ahead carry function. Figure 30 shows a 4-bit 2's complement adder. The exclusive OR (XOR) circuitry is static NMOS logic.

The NOR gates shown are a subset of the XOR gates and, therefore, require no extra devices to realize a carry decode. The fast carry is done with dynamic ratioless circuitry (M5 through M15), where capacitive nodes n_0 through n_{11} can temporarily store clocked signals. A LATCH ACC pulse precharges nodes n_0 through n_3 to a logic high voltage through transistors M5 through M8. When LATCH ACC pulse goes low, nodes n_0 through n_3 are selectively discharged, based on the output voltages of the input NOR and XOR gates existing on nodes n_4 through n_{11}. When nodes n_0 through n_3 have settled, the complemented sum, S, of A and B appears at the outputs of the final XOR gates. The dynamic ratioless carry allows the adder to operate at high speeds with a minimum number of transistors and low power dissipation.

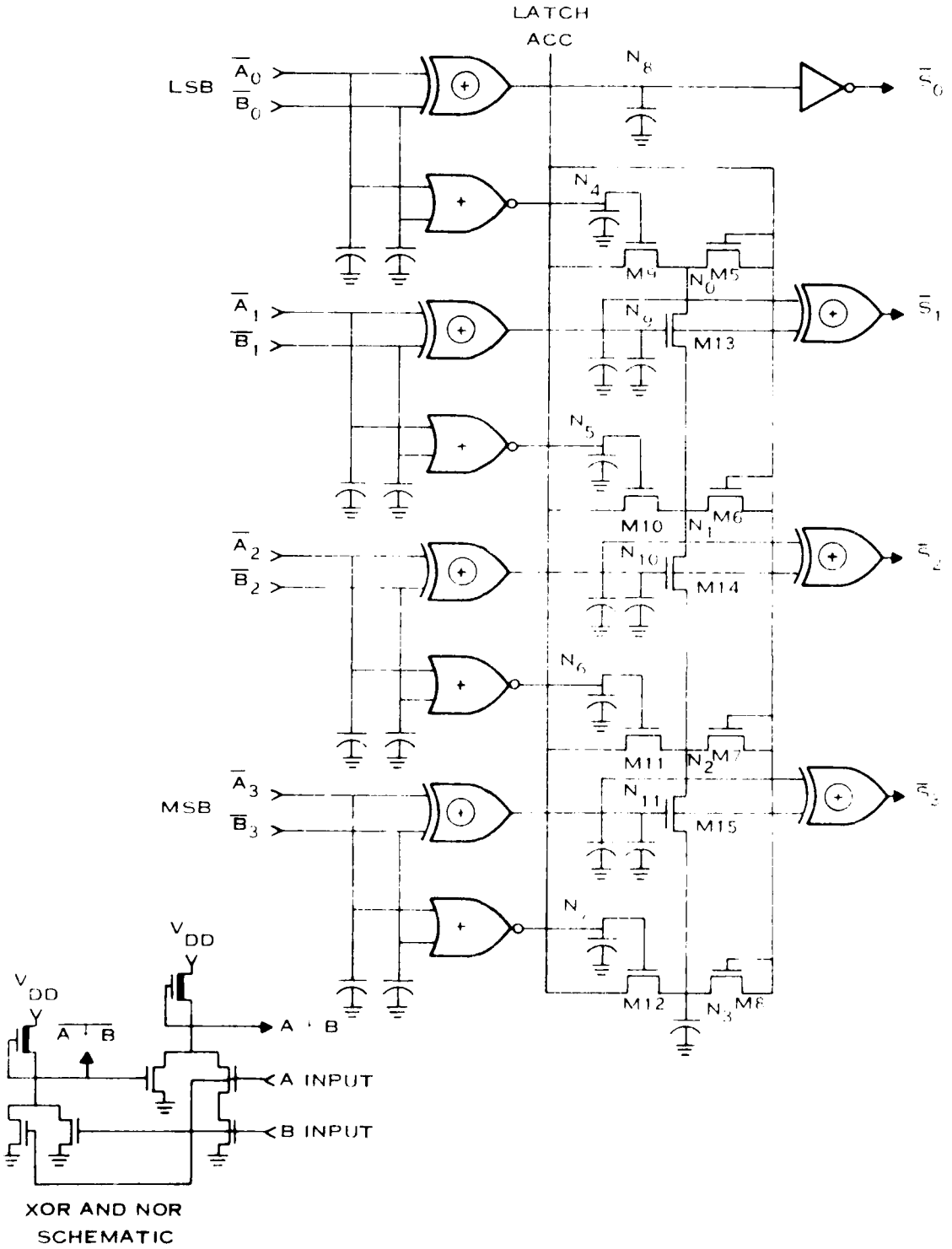


Figure 30. Schematic Diagram of a 4-Bit NMOS With Dynamic Look Ahead Carry

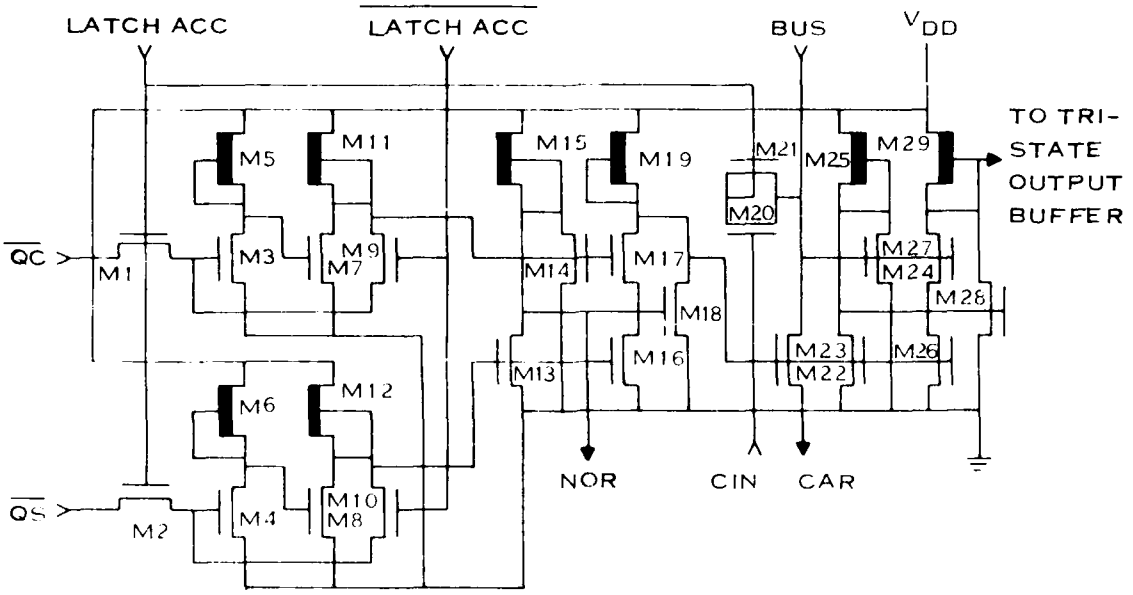


Figure 31. Schematic Diagram of the Dynamic Adder Accumulator and Latch (DYNAD)

The speed of the adder circuit is limited by the time required to precharge nodes n_0 through n_{19}, the propagation delay of the two NOR gates, and the time required to propagate the carry through the dynamic circuitry. The last of these limitations is dominant for a large number of sum bits. The worst case delay occurs when transistors M13 through M15 are all on and only one of the transistors M9 through M12 is on. Thus, nodes n_0 through n_{19} must all discharge through a series string of transistors. Computer simulations indicate that a 20 bit addition can be done in less than 350 ns.

Figure 31 is a schematic of a single-bit stage dynamic adder (DYNAD). The addend and augend are presented to the input NOR (M13 through M15) when LATCH ACC is pulsed high and then latched on the falling edge of the LATCH ACC pulse. The NOR output is the decoded bit carry to be fed forward to the next bit stage. LATCH ACC pulse precharges the BUS line through transistor M21, and the BUS line is discharged through transistor M20 when LATCH ACC pulse goes low and carry in (CIN) pulse from the preceding stage is high. The carry out (CO) propagates through transistor M22 if the input NOR output is high. The complemented bit sum appears at the second NOR output when BUS has settled. Except for the first stage, an adder requires 17 transistors per bit, no more than a static ripple through carry cell but with a significant overall speed advantage in cascaded configurations.

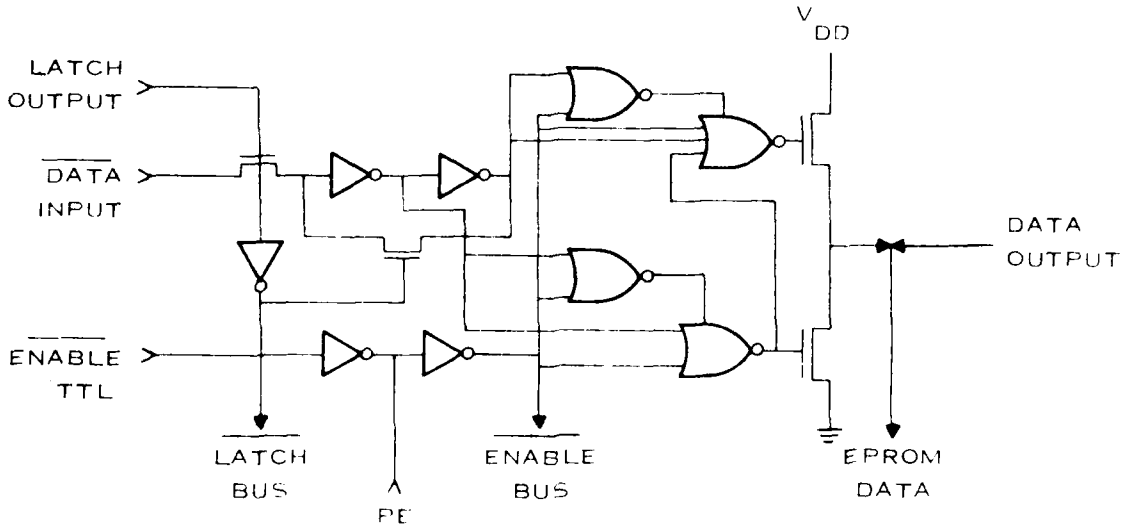


Figure 32. Logic Diagram of the Output Latch/Tri-State Buffer

5. Output Latch Tri-State Buffer

Figure 32 is a functional diagram of the static output latch tri-state buffer. Data from the shift-and-accumulate circuit enters the input latch when the LATCH OUTPUT line goes high. Data is latched into the input latch when the OUTPUT LATCH line is taken back low. An ENABLE line is applied to two NOR gates which enable two output-driver NOR circuits. These two output-driver NOR circuits control the push-pull output-driver transistors. A program enable (PE) line is also provided for programming the EPROM. This line will go high during the programming of the EPROM which, in turn, will tri-state the output bus. This enables the output bus to be used as data inputs to the memory, thus reducing the number of I/O lines required. Figure 33 is a schematic diagram of the output latch tri-state buffer. The output latch tri-state buffer has a fanout of three TTL/LS loads.

6. Controller

A simplified controller block diagram is shown in Figure 34. This controller has three inputs (word length, shift clock, and load) and seven outputs (parallel load, Φ_1, Φ_2, clear accumulator, latch accumulator, latch output, and data valid). The shift register controller has the function of providing the correct pipeline timing for the PIPELSIC

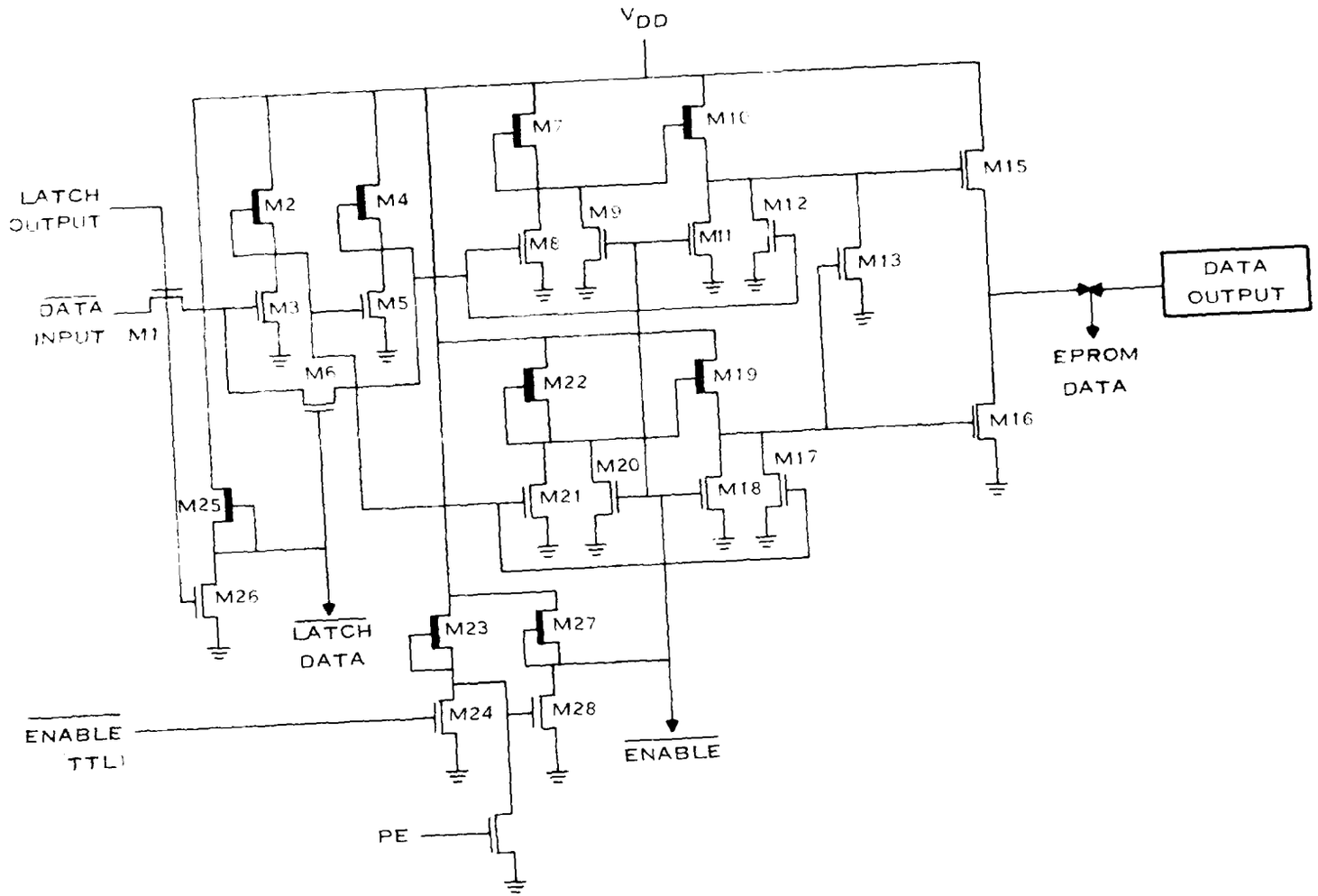


Figure 33. Schematic Diagram of the Output Latch/Tri-State Buffer

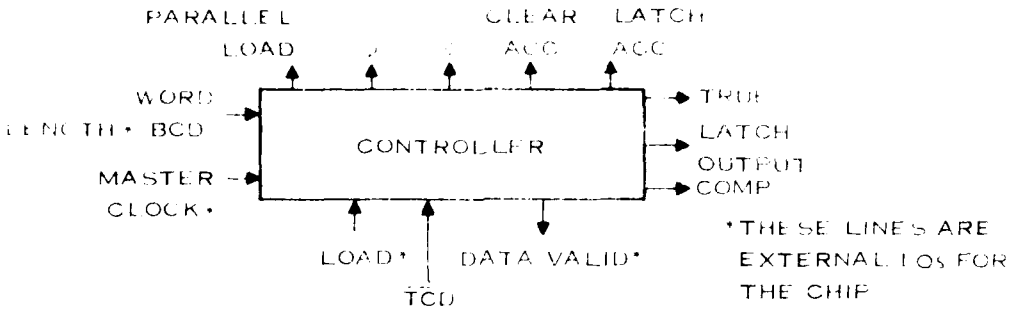


Figure 34. Simplified Block Diagram of the Controller Section

The Block diagram of the shift register controller is shown in Figure 35. The load pulse and [illegible] clock may be asynchronous and are provided externally to the chip along with the input data WORD LENGTH. The TTL level master clock is buffered by buffer (BF_1) to an MOS [illegible] from which two phase clocks are derived $(\Phi_1$ and $\Phi_2)$. These two clocks are used to [illegible] the necessary synchronization to each on chip section. The controller must also [illegible] information on the input data WORD LENGTH [illegible]. This information is provided to the multiplexers (MUX 1, MUX 2) via the WORD LENGTH inputs $(B_1$ through $B_4)$. These inputs are decoded to provide the proper timing pulses for each possible input word length.

The actual timing pulse sequence begins when a load pulse is applied to the TTL level LOAD input. This load pulse triggers the data latch (DL_1) and sets a logic 1 on its output line (Q_1). This logic 1 is shifted into the next latch (DL_2) on the next Φ_1 clock, producing a parallel load pulse via OR gate (N_1) and providing a set pulse to latch (SR_1). This pulse on the set input of gate (SR_1) sets a logic 1 on the clear input of latch (DL_1) producing a logic 0 on its output line (Q_1). A timing diagram of LATCH (SR_1) is shown in Figure 36. The LOAD input line is now inoperative because of the logic 1 on the clear input of latch (DL_1). The logic 1 latched into latch (DL_2) is shifted [illegible] of latch (DL_3) on the next Φ_1 clock, producing a clear accumulator pulse via NOR gate (N_2). The TRUE and COMPLEMENT lines are also pulsed during this time via NOR gate (N_3) and [illegible] N_4. The complement line (CD) is low. During this shift operation, the [illegible] through the multiplexer (MUX_1) of the data WORD LENGTH [illegible] BCD [illegible] the logic 1 will continue to shift through the next latches $(DL_4$ through $DL_{[illegible]})$ [illegible] proper data pattern is provided by (MUX_1) to the SR latches $(SR_2$ [illegible] $DL_{[illegible]})$. When [illegible] is provided to the inputs $(D_1$ through $D_4)$ of (MUX_2), the logic 1 will appear at the output of (MUX_2). This logic 1 will now present an input to latch (DL_1) and will [illegible] (SR_1) [illegible] (Q_1) [illegible] output line (Q_1) [illegible] the clear input of latch (DL_1). The LOAD [illegible] operation is complete and is ready for another load pulse. The data [illegible] (MUX_2) produced its output also sets a logic 0 on the output line (Q_5) of latch (DL_5). [illegible] 0 on the input of NOR gate (N_2) will cause its output to provide a LATCH ACCUMULATOR CLEAR on the next Φ_1 clock.

The data stored in latch (DL_1) will continue to shift through (DL_2) [illegible] $(DL_{[illegible]})$ [illegible] as it did through latches $(DL_1$ through $DL_4)$. Multiplexer (MUX_1) [illegible] WORD LENGTH [illegible] provided to it as (MUX_2) and will provide [illegible] the proper decode address. The data will then appear at [illegible]

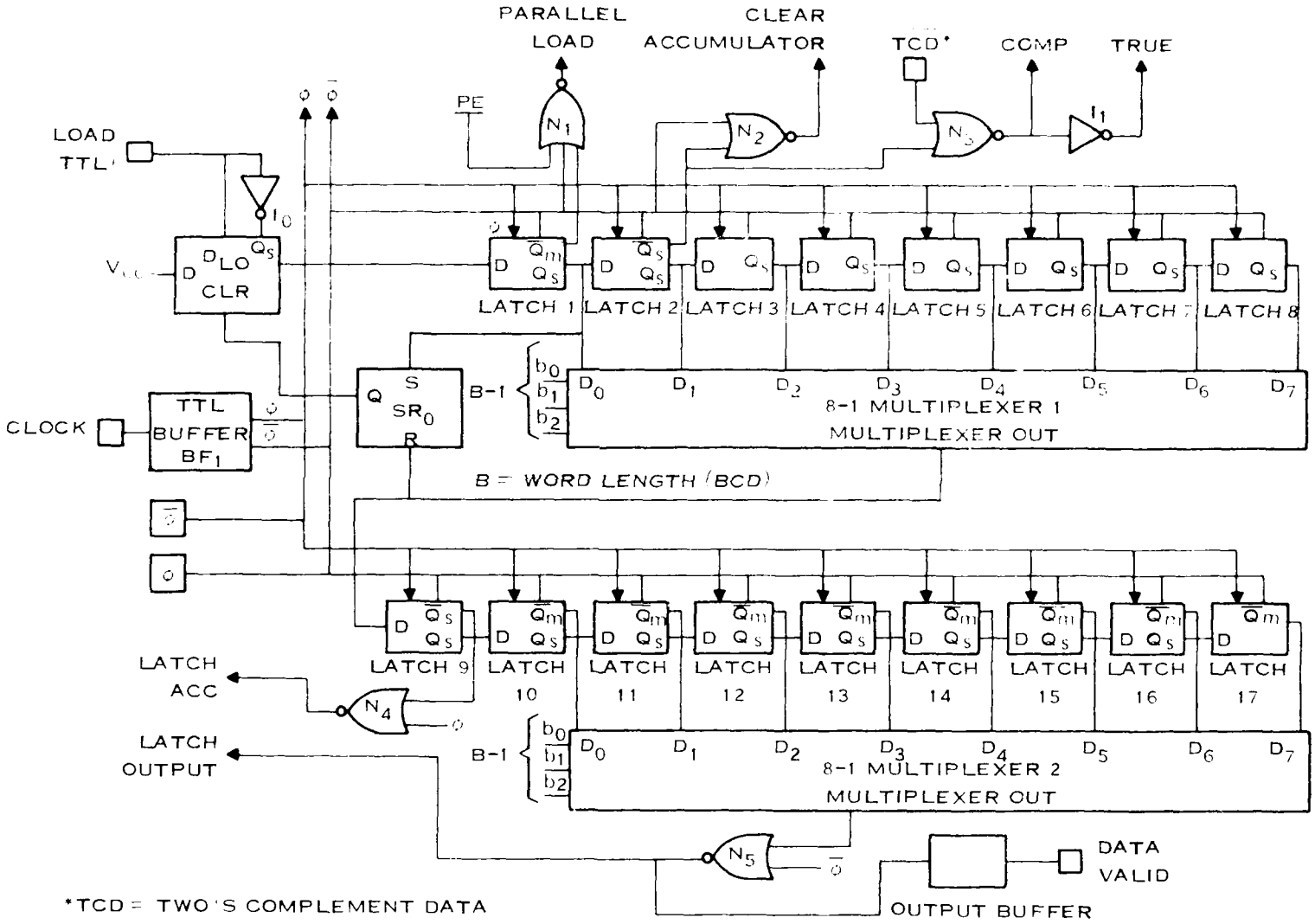


Figure 35. Block Diagram of the Shift Register Controller

MUX_2 and will produce a LATCH OUTPUT via NOR gate N_5 on the same Φ clock. The output of NOR gate N_5 is also buffered through buffer BF_2 to provide an external DATA VALID pulse. A timing diagram of this function is shown in Figure 37. The latch accumulator and latch output waveforms are shown for 1- through 8-bit input words.

The control multiplexer is shown schematically in Figure 38. This multiplexer consists of three input stages (TTL level) which provide the necessary control logic to drive the decode circuitry. The decode circuitry selects the proper data path by turning on the associated transistor (M20, M25, M30, M35, M40, M45, M50, and M55).

A block diagram of a D-controller latch is shown in Figure 39. This latch is implemented by using two NMOS latches in cascade, clocked on opposite phases (Figure 39B) which produces a D-type latch function.

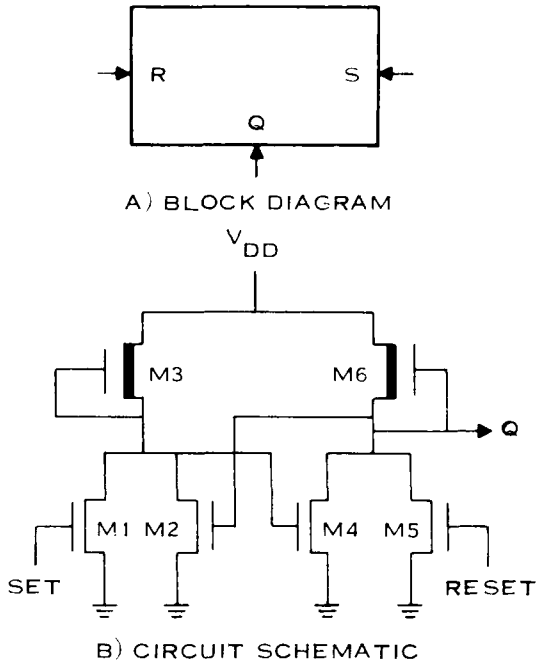
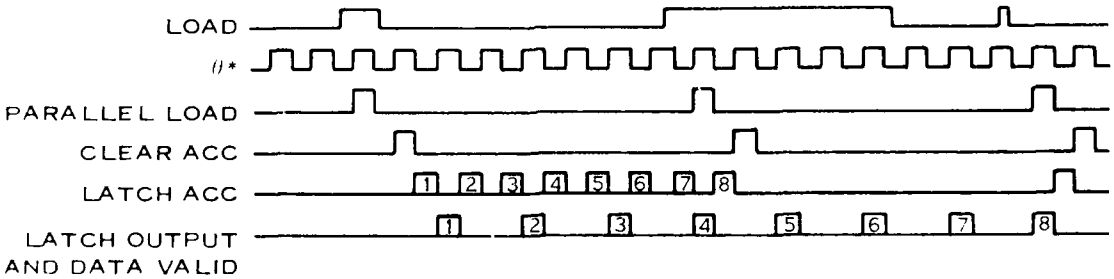


Figure 36. Controller Set/Reset Latch Diagram

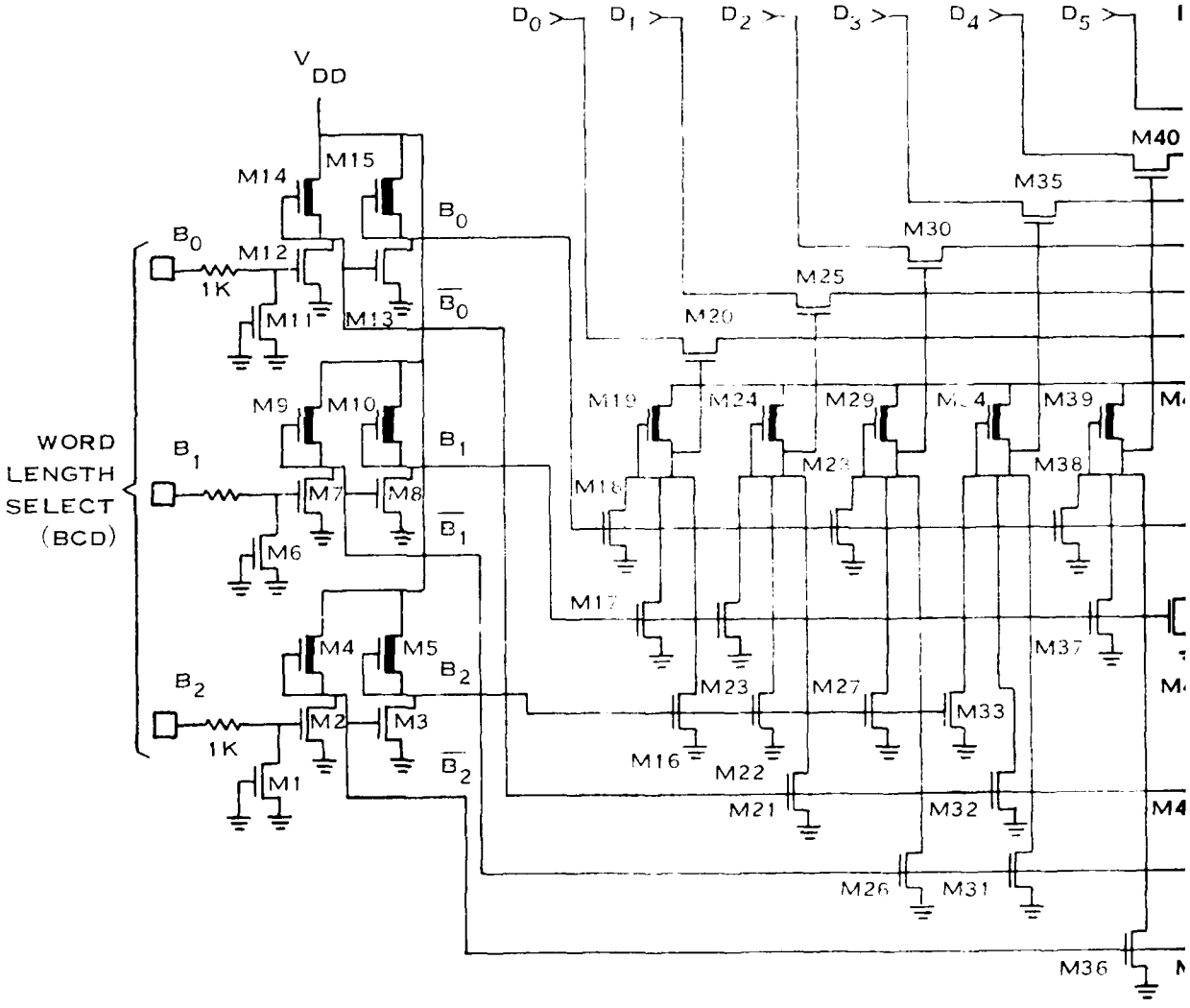
A circuit schematic of the clock buffer BF1 is shown in Figure 40. This buffer receives the master clock (TTL level) off-chip and derives the two phases 0 to 5 volts necessary to drive the on-chip circuitry. Input protection is provided by M14 and R1.

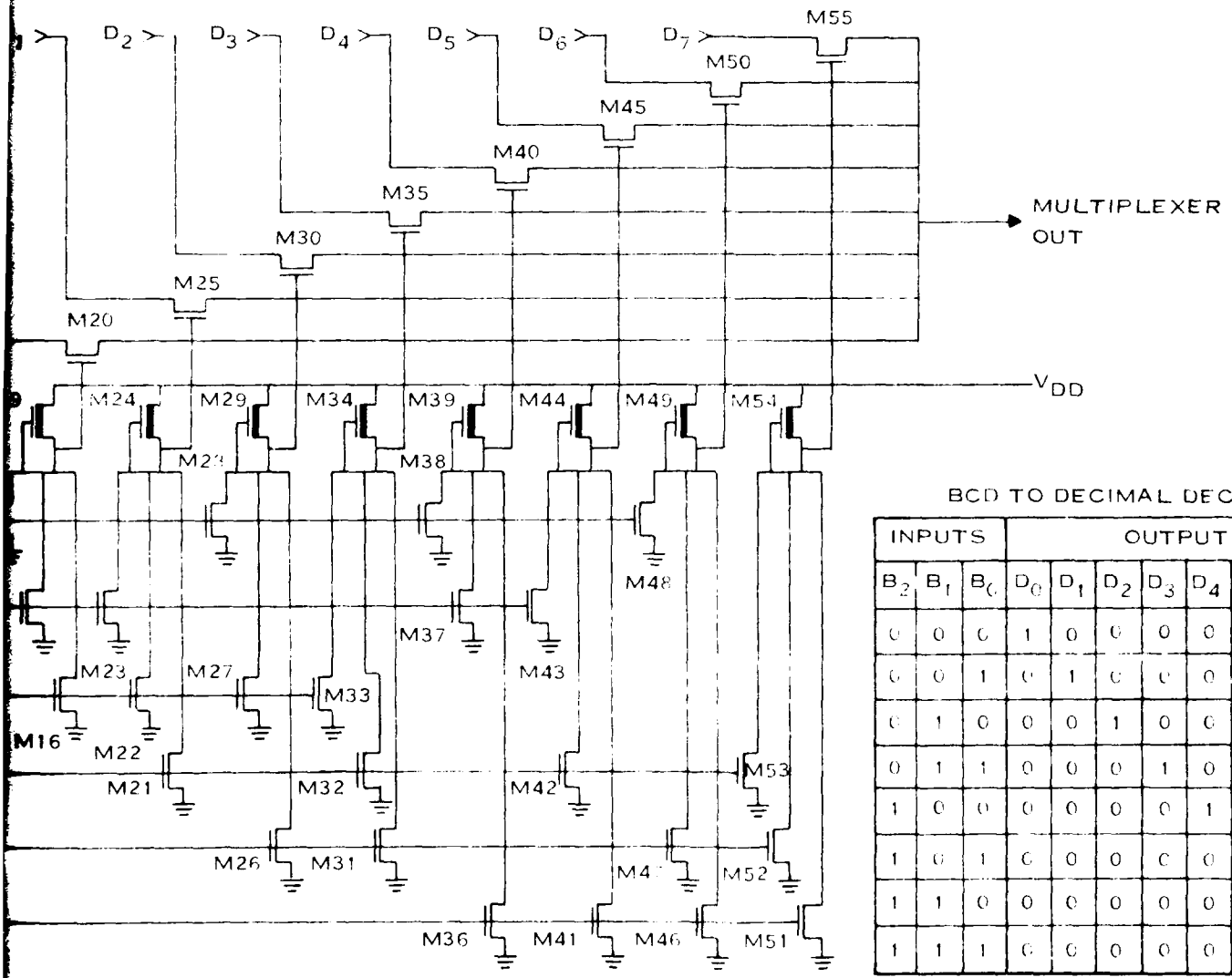


*20 MHZ MAX

LATCH ACC AND LATCH OUTPUT WAVEFORMS ARE SHOWN FOR SEVERAL POSSIBLE 8 BIT WORDS, ALONG WITH SEVERAL POSSIBLE LOAD WAVEFORMS.

Figure 37. Controller Timing Diagram


WORD
LENGTH
SELECT
(BCD)
V_{DD}
B_0
B_1
B_2
1K
1K
$\overline{B_0}$
$\overline{B_1}$
$\overline{B_2}$
D_0
D_1
D_2
D_3
D_4
D_5
M1
M2
M3
M4
M5
M6
M7
M8
M9
M10
M11
M12
M13
M14
M15
M16
M17
M18
M19
M20
M21
M22
M23
M24
M25
M26
M27
M29
M30
M31
M32
M33
M35
M36
M37
M38
M39
M40



BCD TO DECIMAL DECODE

INPUTS			OUTPUTS							
B_2	B_1	B_0	D_0	D_1	D_2	D_3	D_4	D_5	D_6	D_7
0	0	0	1	0	0	0	0	0	0	0
0	0	1	0	1	0	0	0	0	0	0
0	1	0	0	0	1	0	0	0	0	0
0	1	1	0	0	0	1	0	0	0	0
1	0	0	0	0	0	0	1	0	0	0
1	0	1	0	0	0	0	0	1	0	0
1	1	0	0	0	0	0	0	0	1	0
1	1	1	0	0	0	0	0	0	0	1

Figure 38. Schematic Diagram of the Control Multiplexer

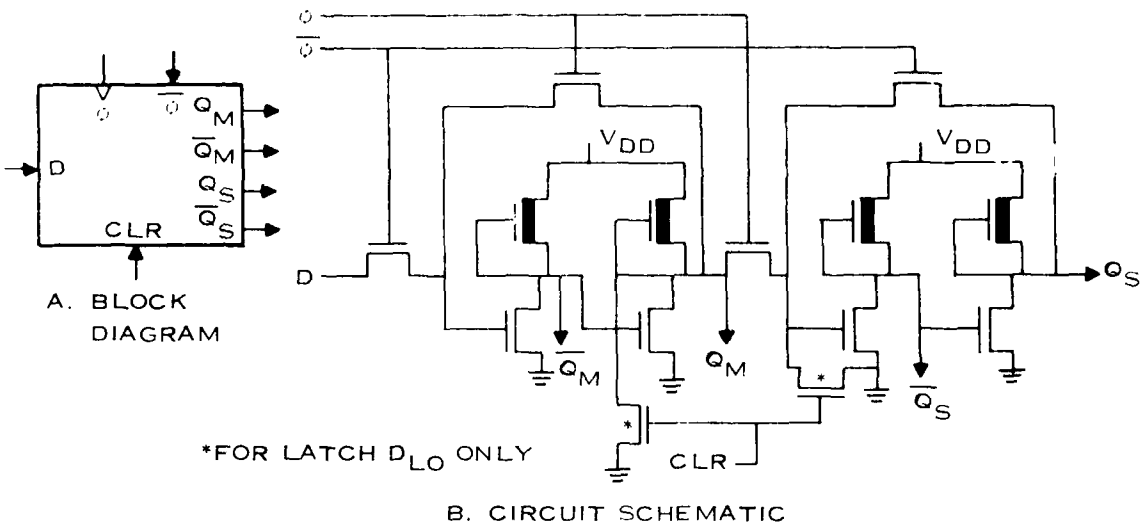


Figure 39. Controller D-Latch Diagram

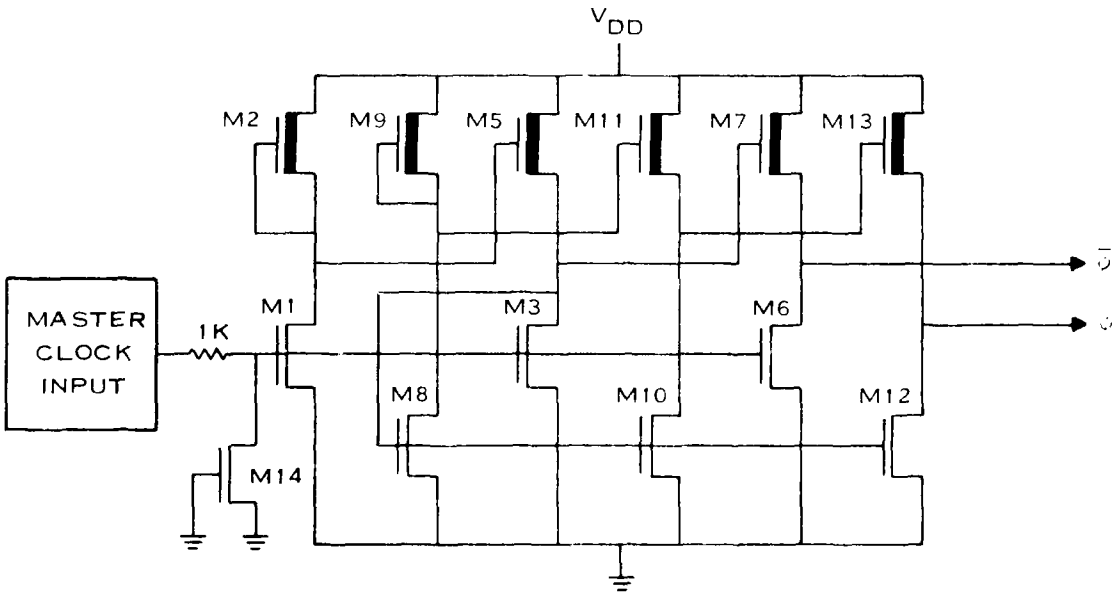


Figure 40. Schematic Diagram of the Clock Driver

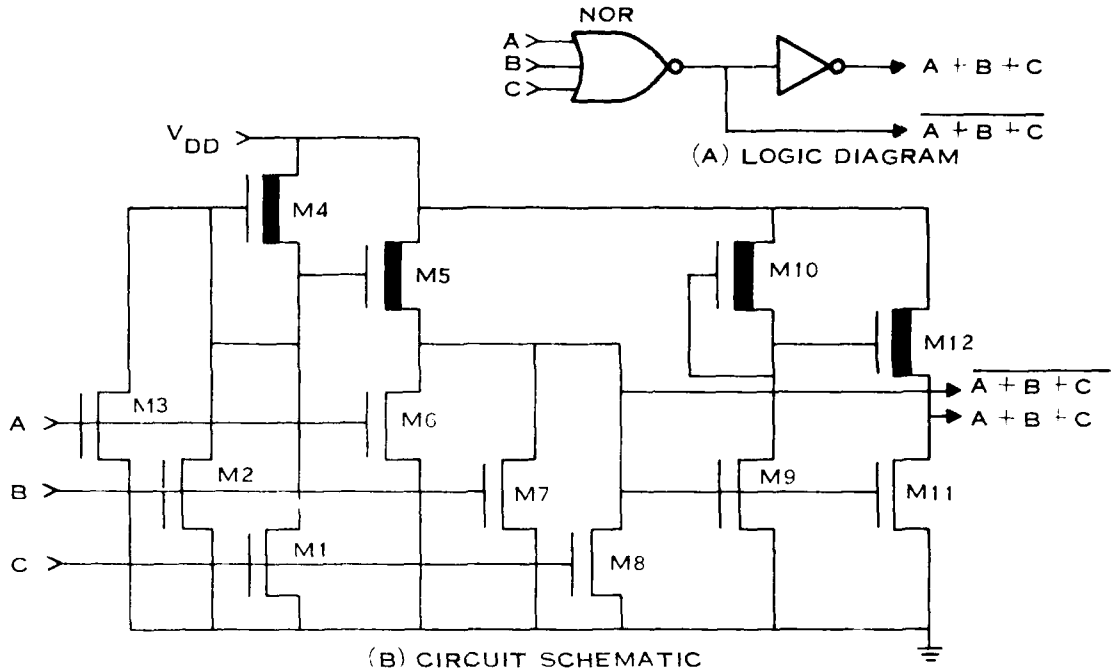


Figure 41. NOR Gate Bus Driver Diagram

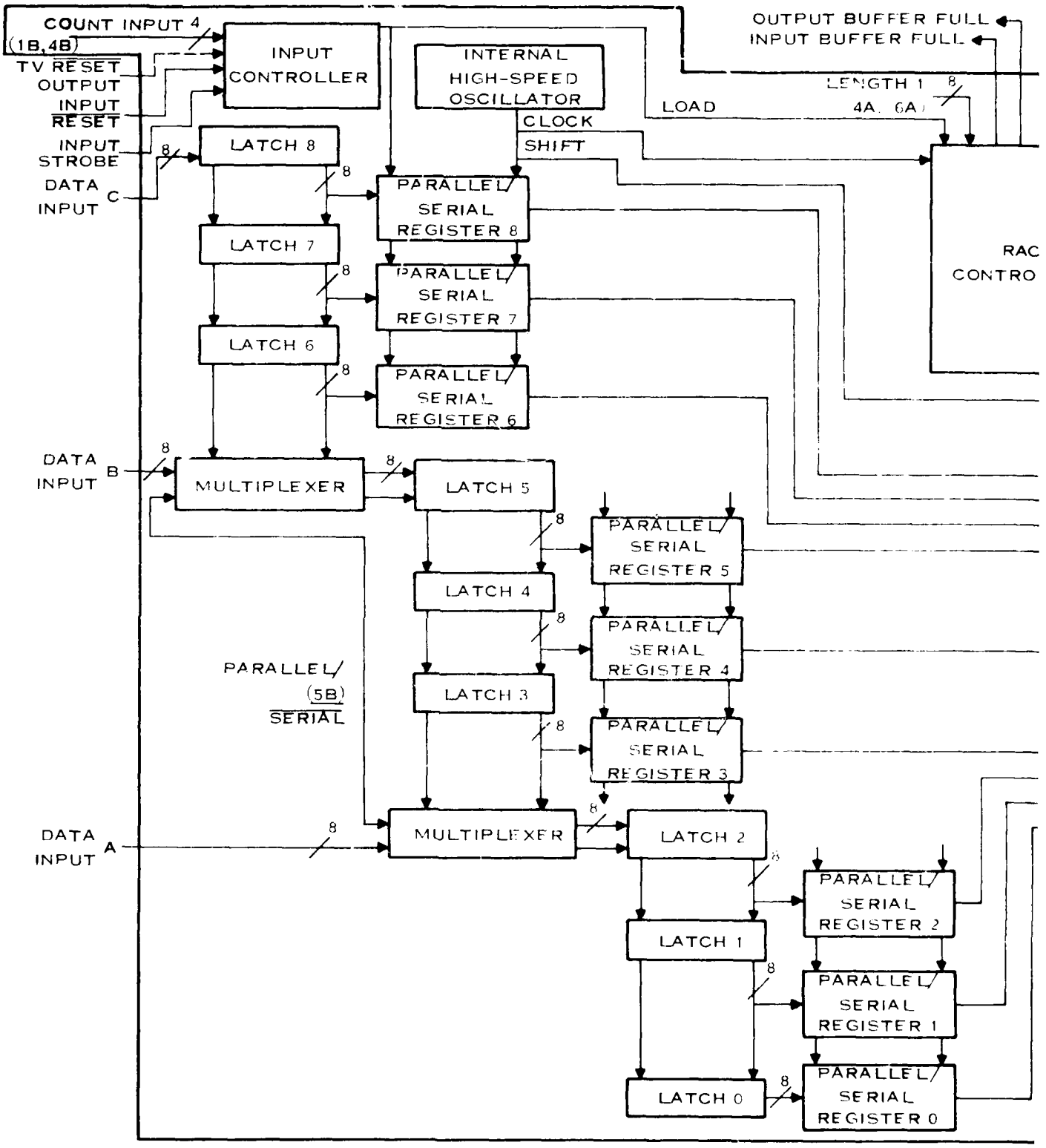
The NOR gates, N_1, N_2, N_3, and N_5, are required to drive internal buses and, therefore, not only provide an NOR function but also provide required buffering. A logic operation and circuit schematic is shown in Figure 41.

7. Programmable Sum-of-Products Operator

To fully explore the architectural implications of a programmable sum-of-products operator, a hardware implementation of Equation 1 has been designed and fabricated under a parallel contract with Carnegie-Mellon University. The breadboard can operate on either sliding or fixed blocks of data. A block diagram of the hardware implementation is shown in Figure 42. The breadboard consists of nine input latches, nine parallel-in/serial-out shift registers, a fast 512 × 12-bit memory for temporary storage of the partial products, an EPROM for permanent storage of the partial products, shift-and-accumulate circuitry, tri-state output latches, and control circuitry.

The input latch structure is hardware- or software-selectable for either serial data entry at DATA INPUT P or parallel data entry at DATA INPUT S, DATA INPUT T, and DATA INPUT P. This facilitates the implementation of a nine-point transversal filter or a 3 by 3 sliding window operator, respectively. The input data word length is hardware-selectable from 1 bit to 8 bits and is hardware- or software-selectable as 2's complement or sign magnitude format.

The weighting coefficients, W_i, determine a set of partial products which are stored in a 512 × 12-bit high-speed radom access memory (RAM). Partial products may be down-loaded on the data bus from an external source by a controlling processor. The partial products are hardware-selectable as 2's complement or sign magnitude format. The partial products obtained from the


COUNT INPUT 4
(1B,4B)
TV RESET
OUTPUT
INPUT
RESET
INPUT
STROBE
DATA
INPUT C
INPUT CONTROLLER
INTERNAL HIGH-SPEED OSCILLATOR
CLOCK
SHIFT
LOAD
OUTPUT BUFFER FULL
INPUT BUFFER FULL
LENGTH 1
4A, 6A
RAC
CONTRO
LATCH 8
LATCH 7
LATCH 6
PARALLEL/ SERIAL REGISTER 8
PARALLEL/ SERIAL REGISTER 7
PARALLEL/ SERIAL REGISTER 6
DATA
INPUT B
MULTIPLEXER
LATCH 5
LATCH 4
LATCH 3
PARALLEL/ SERIAL REGISTER 5
PARALLEL/ SERIAL REGISTER 4
PARALLEL/ SERIAL REGISTER 3
PARALLEL/
(5B)
SERIAL
DATA
INPUT A
MULTIPLEXER
LATCH 2
LATCH 1
LATCH 0
PARALLEL/ SERIAL REGISTER 2
PARALLEL/ SERIAL REGISTER 1
PARALLEL/ SERIAL REGISTER 0
8

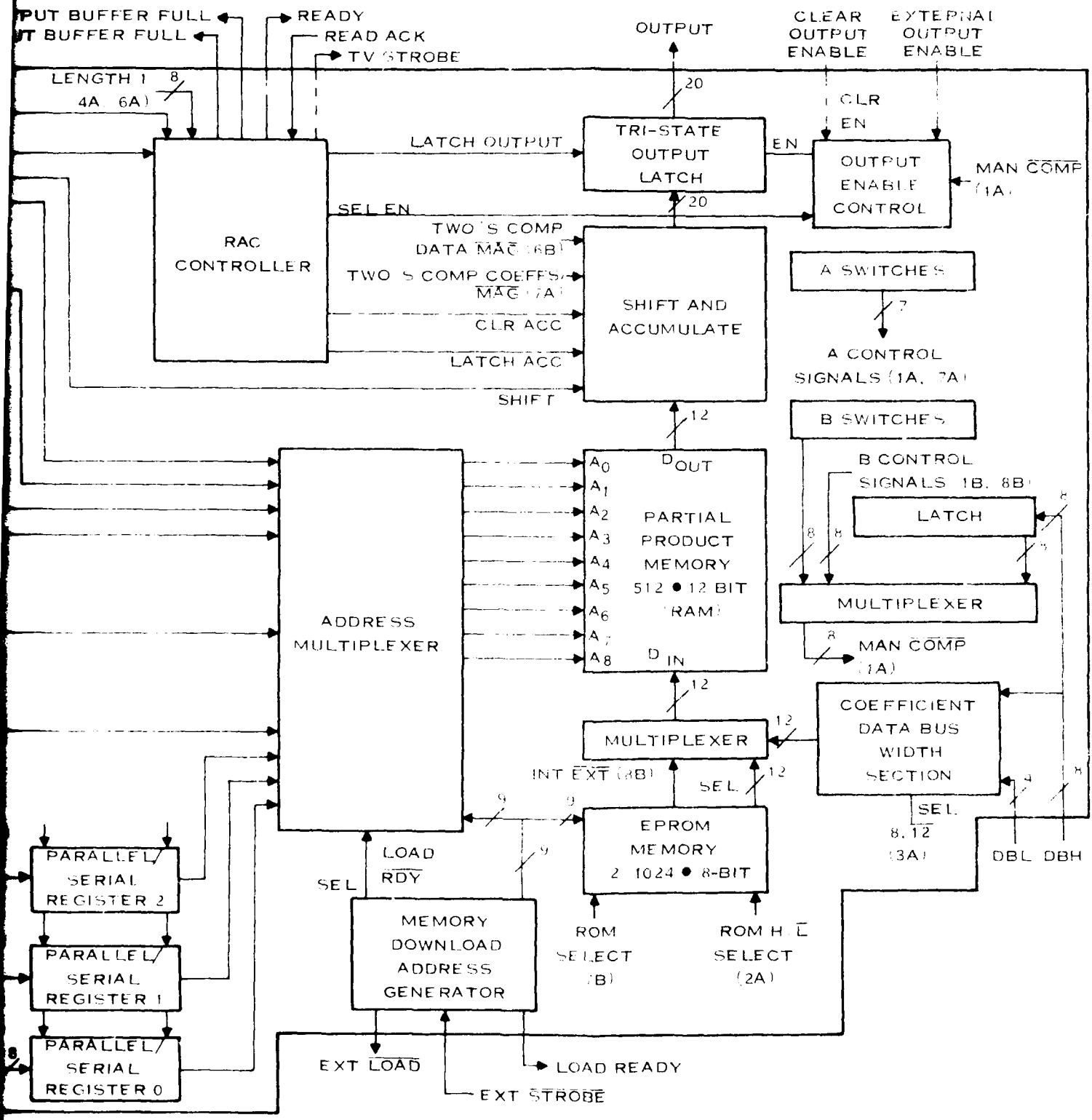


Figure 42. Programmable Sum of Products Unit

RAM during operation are summed in a carry-save accumulator with data shifting to weight the significance of each partial product. Output data (20 bits) is buffered by a tri-state output latch.

The data input section consists of nine latches connected to form a 9-stage-long-by-8-bit-wide shift register. The outputs of each stage are also connected to nine parallel-to-serial conversion registers which form the data for ROM-accumulate operation. The data input operation is controlled by the INPUT CONTROLLER. The controller has a hardware- or software-selected BCD value count as one of its inputs. The INPUT RESET line is pulsed low to initialize the controller. The INPUT STROBE line shifts data through the input latches and clocks the controller on its leading edge. After enough strobe pulses, the INPUT CONTROLLER generates a LOAD pulse which latches the data from the input latches into the parallel-to-serial registers and starts the high-speed asynchronous CONTROLLER. The LOAD pulse also resets the INPUT CONTROLLER so that new data can be shifted into the input latches while the ROM-accumulate operation is taking place. A second INPUT RESET pulse is not required.

The ROM-accumulate CONTROLLER operates from an asynchronous internal 16.7-MHz oscillator (60-ns period). This is the maximum clock rate for the components selected for the shift registers and accumulator. After a LOAD pulse is received from the INPUT CONTROLLER, the CONTROLLER sequences the operations of the PARALLEL-to-SERIAL REGISTERS, the PARTIAL PRODUCT MEMORY, the SHIFT-AND-ACCUMULATE function, and the OUTPUT LATCH. It also provides signals which can be monitored by an external processor if desired. After the final INPUT STROBE pulse, the result of the ROM accumulate calculation is available at the TRI-STATE OUTPUT LATCH in $[B_x + 6\frac{1}{2}]$ internal clock cycles. The additional $6\frac{1}{2}$ clock cycles is due to the pipelined architecture. Thus,

$$t_{\text{ROM-accumulate}} = [60 \times B_x + 390] \text{ ns}$$

where $t_{\text{ROM accumulate}}$ is the time required to process the data and B_x is the number of significant bits in each input data word. For 8 bit data, this is 870 ns. The ROM-accumulate CONTROLLER will bring the READY line high when the output is available and will not allow another result to overwrite the output latch data until the READ ACK input is pulled high by the controller processor. The OUTPUT BUFFER FULL line goes high, the INPUT STROBE line is inhibited, and the internal oscillator is stopped if an overwrite condition exists. The tri-state outputs are enabled by pulling the external OUTPUT ENABLE line low.

Owing to the pipeline organization of the ROM-accumulator hardware, the PARALLEL-SERIAL REGISTERS can be loaded as soon as B_x bits of data have been shifted out of the latches, B_x internal clock cycles or $60 \times B_x$ nanoseconds after the last INPUT STROBE pulse. For 8-bit data, this is 480 ns, which gives a throughput rate of about 2 MHz. If the INPUT CONTROLLER generates a LOAD pulse before the PARALLEL-SERIAL REGISTERS are emptied, the INPUT BUFFER FULL line will go high and the INPUT STROBE line will be inhibited until the registers are emptied to prevent overwriting any data.

From the above discussion, it can be seen that the maximum input data rate depends on the number of bits in the input data words (B_x) and the number of INPUT STROBE pulses. The number of INPUT STROBE pulses (N) between parallel-to-serial conversions depends on the type of operations performed. For sliding 3 × 3 or 9 × 1 filter applications only one strobe pulse is needed between parallel-to-serial conversions. For nonsliding 3 × 3 window operations, three strobe pulses

are needed and, for a 9 × 1 transform, nine strobe pulses are needed. The maximum input data rate is given by

$$f_{MAX_{IN}} = (N)(B_x \times 60\ ns)$$

For sliding window or transversal filter applications with 8-bit data, $f_{MAX_{IN}} = 2$ MHz. For application such as an 8 × 1 transform with 8-bit data, $f_{MAX_{IN}} = 16$ MHz.

The maximum output data rate is always given by

$$f_{MAX_{OUT}} = 1/(B_x \times 60\ ns)$$

The breadboard also can operate in parallel with two other breadboards to provide throughput at real time (TV) data rates. Three control lines, shown dashed in Figure 42, are provided to synchronize this operation. These lines, TV RESET OUTPUT, TV STROBE, and CLEAR OUTPUT ENABLE, are normally not connected when operating a single breadboard.

The programmable-sum-of-products breadboard is 12 × 12 × 5 inches, weighs 7 pounds, and dissipates 15 watts. Figure 43 is a photograph of this breadboard. For size comparison, a 40-pin dual-in-line package is shown adjacent to the electronics boards. With the exception of the input controller, output controller, and the EPROMs for program downloading, the entire electronic boards have been integrated onto the PIPE LSIC. This represents monolithically integrating approximately 75 discrete integrated circuits.

B. PIPE LSIC APPLICATIONS

This subsection is a discussion of applications of the PIPE LSIC described in the preceding section. The types of operations as well as operational constraints are discussed.

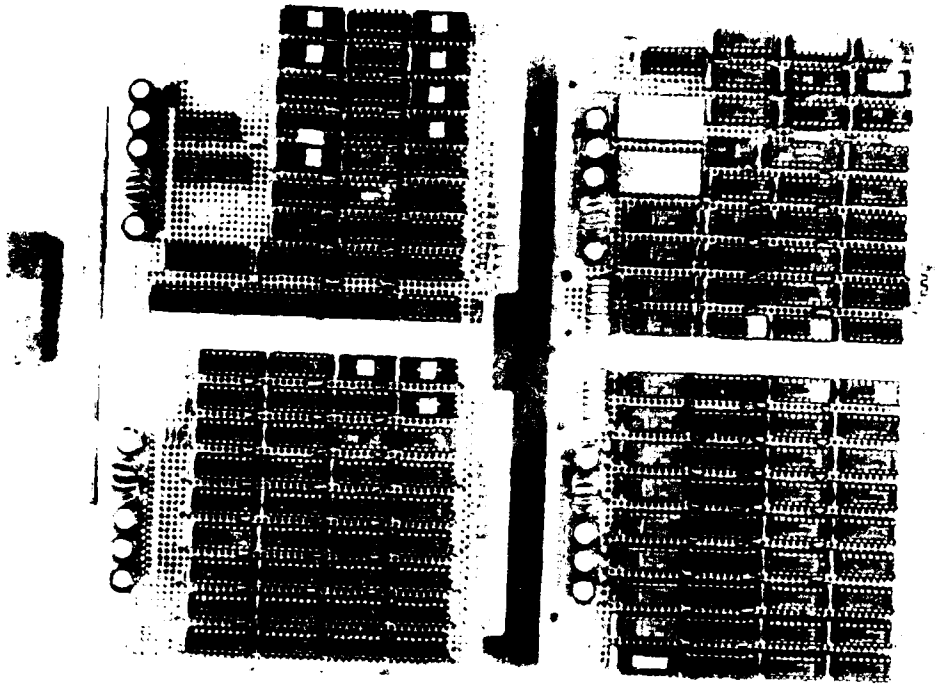
1. Introduction

Many digital signal-processing and image-processing algorithms require operations of the form

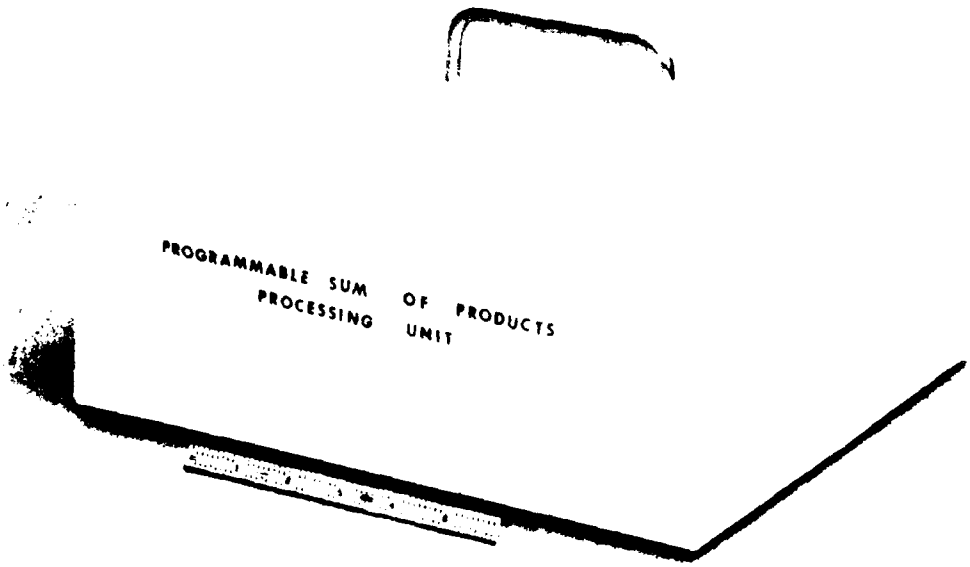
$$Y = \sum_{i=0}^{M-1} W_i X_i \tag{5}$$

where the W_i represents a set of fixed weighting coefficients and the X_i represents a set or sequence of input values. Equation 5 can be used to calculate the coefficients of various transforms such as Fourier, Cosine, Hadamard, Haar, etc. Where two-dimensional transforms are needed, successive one-dimensional transforms can be used if the transforms are separable. For image-processing applications, Equation 5 defines the discrete convolution of a two-dimensional input image with a convolution array. These mathematical operations are based on the adjacent pixel values and are termed neighborhood operators. Examples of neighborhood operators include noise smoothing, edge crispening, linear edge enhancement, etc.

The following subsections discuss the applicability of the PIPE LSIC to matrix operations for calculating transform coefficients and neighborhood operator calculations.



(A) ELECTRONICS BOARDS



(B) BREADBOARD

Figure 43. Programmable Sum of Products Breadboard

2 Input Output Considerations

Key to the full use of the PIPE LSIC is an understanding of the input output relationships.

Figure 44 shows the input output pins of the PIPE LSIC. The PAR/SER select pin determines whether the LSIC operates on 9 × 1 or 3 × 3 blocks of data, either of which may be sliding or nonsliding. The input word length is designated by the 3-bit WORD LENGTH pins. Eight-bit parallel input words S, T, or P are loaded into the input latches by the INPUT STROBE and converted into bit-serial words by the MASTER CLOCK and LOAD. The PIPE LSIC is capable of operating on 2's complement or sign magnitude data, as determined by the state of the 2's COMP pin. A DATA VALID pulse informs the user when the PIPE LSIC has completed a calculation. The 20-bit parallel output is obtained by bringing the enable (EN) pin low to activate the tri-state outputs.

The PIPE LSIC requires a single 5-volt power supply and, during programming of the EPROM, a 25-volt programming voltage.

A total of 58 pins is required for full 8-bit input and 20-bit output operations. If the input word length is reduced to 6 bits and only 8 bits of the output are used, a total of 40 pins are needed. The pins required for word-length selection, input type (parallel or serial), and data format (2's complement or magnitude) can be eliminated by on-chip bonding to the V_{DD} or ground pin for further pin reduction on fixed application.

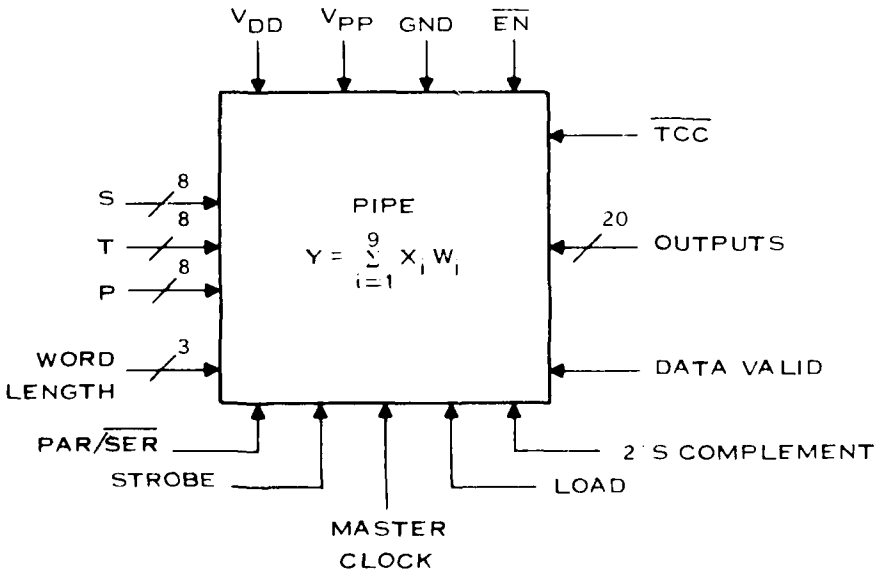


Figure 44. PIPE LSIC I/O Considerations

The operational characteristics of the PIPE LSIC are discussed in a later subsection. These characteristics result in part from the architecture selected to implement the PIPE LSIC and in part from the circuit design of the architecture. After a discussion of the types of calculations the PIPE LSIC is capable of, these characteristics are easily understood and appreciated.

3 Matrix Operation

The PIPE LSIC is ideally suited for matrix operations of the form:

$$Y = [W_0\ W_1\ W_2\ W_3\ W_4\ W_5\ W_6\ W_7\ W_8] \begin{bmatrix} X_0 \\ X_1 \\ X_2 \\ X_3 \\ X_4 \\ X_5 \\ X_6 \\ X_7 \\ X_8 \end{bmatrix} \qquad (6)$$

where Y represents the product of a row vector, W, and a column vector, X. Most signal-processing applications require the product of a weighting matrix and an input vector. This can be represented as:

$$Y = W\ X \qquad (7)$$

or, for the PIPE LSIC, as:

$$\begin{bmatrix} Y_0 \\ Y_1 \\ Y_2 \\ Y_3 \\ Y_4 \\ Y_5 \\ Y_6 \\ Y_7 \\ Y_8 \end{bmatrix} \quad \begin{matrix} W_{00} & W_{01} & W_{02} & \cdots & W_{08} \\ W_{10} & W_{11} & W_{12} & & W_{18} \\ \bullet & & & & \bullet \\ & & & & \\ \bullet & & & & \bullet \\ & & & & \\ \bullet & & & & \bullet \\ & & & & \\ W_{80} & W_{81} & W_{82} & & W_{88} \end{matrix} \quad \begin{bmatrix} X_0 \\ X_1 \\ X_2 \\ X_3 \\ X_4 \\ X_5 \\ X_6 \\ X_7 \\ X_8 \end{bmatrix}$$

A diagram of the PIPE LSIC configured to implement Equation (7) is shown in Figure 45. Nine PIPE LSICs are used, with each PIPE programmed with the weighting coefficients of one row of W. This eliminates reprogramming the PIPE LSIC, thus increasing the speed of the calculation. All the P inputs of the PIPE LSICs are connected and data is entered sequentially. Nine sample periods are required to load the PIPE LSICs. The y vector is calculated in parallel for the particular values of X. A valid output is available every sample period if operating on sliding 9-by-1 data while nine sample periods are required to produce an answer if the operation of Equation (7) is performed on nonsliding 9-by-1 data blocks.

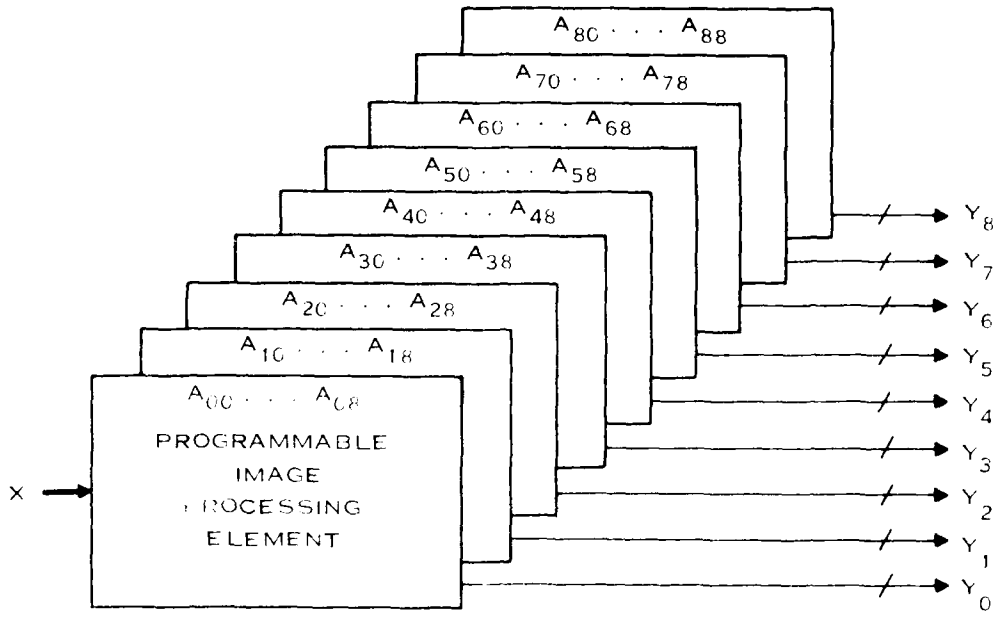


Figure 45. Programmable Image Processing Element Configured to Implement Y=WX

The configuration shown in Figure 46 can be used to calculate coefficients of various transforms.

a. One-Dimension Transforms

The discrete cosine transform of a data sequence is defined as

$$Y_0 = \frac{\sqrt{2}}{M} \sum_{i=0}^{M-1} X_i \tag{8}$$

$$Y_k = \frac{\sqrt{2}}{M} \sum_{i=0}^{M-1} X_i \cos\frac{(2i+1)k\pi}{2M} \quad k = 1, 2, \ldots, M-1$$

Assuming an 8 point transform is desired, Equation 8 simplifies to

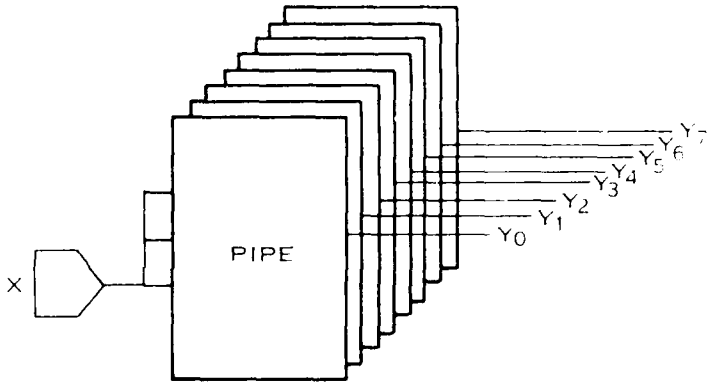
$$Y_0 = 0.177 \sum_{i=1}^{8} X_i \tag{9}$$

$$W = \begin{bmatrix} 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 \\ 1 & -1 & 1 & -1 & 1 & -1 & 1 & 1 \\ 1 & 1 & -1 & -1 & 1 & 1 & -1 & -1 \\ 1 & -1 & -1 & 1 & 1 & -1 & -1 & 1 \\ 1 & 1 & 1 & 1 & -1 & -1 & -1 & -1 \\ 1 & -1 & 1 & -1 & -1 & 1 & -1 & 1 \\ 1 & 1 & -1 & -1 & -1 & -1 & 1 & 1 \\ 1 & -1 & -1 & 1 & -1 & 1 & 1 & -1 \end{bmatrix}$$

HADAMARD

$$W = \begin{bmatrix} 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 \\ 1 & 1 & 1 & 1 & -1 & -1 & -1 & -1 \\ 1 & 1 & -1 & -1 & -1 & -1 & 1 & 1 \\ 1 & 1 & -1 & -1 & 1 & 1 & -1 & -1 \\ 1 & -1 & -1 & 1 & 1 & -1 & -1 & 1 \\ 1 & -1 & -1 & 1 & -1 & 1 & 1 & -1 \\ 1 & -1 & 1 & -1 & -1 & 1 & -1 & 1 \\ 1 & -1 & 1 & -1 & 1 & -1 & 1 & -1 \end{bmatrix}$$

WALSH



$$W = \begin{bmatrix} 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 \\ 1 & 1 & 1 & 1 & -1 & -1 & -1 & -1 \\ \sqrt{2} & \sqrt{2} & \sqrt{-2} & \sqrt{-2} & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & \sqrt{2} & \sqrt{2} & \sqrt{-2} & \sqrt{-2} \\ 2 & -2 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 2 & -2 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 2 & -2 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 2 & -2 \end{bmatrix}$$

HAAR

Figure 46. Weighting Coefficient Matrices for Various 8 × 1 Transforms

$$Y_k = 0.25 \sum_{i=0}^{7} X_i \cos \frac{(2i+1)k\pi}{16} \qquad k = 1, 2, \ldots 7$$

or, in matrix form

$$\begin{bmatrix} Y_0 \\ Y_1 \\ Y_2 \\ Y_3 \\ Y_4 \\ Y_5 \\ Y_6 \\ Y_7 \end{bmatrix} = \begin{bmatrix} 0.177 & 0.177 & \ldots & 0.177 \\ 0.245 & 0.208 & \ldots & 0.245 \\ 0.231 & \ldots & & 0.231 \\ 0.208 & \ldots & & 0.208 \\ 0.177 & \ldots & & 0.177 \\ 0.139 & & & 0.139 \\ 0.096 & & & 0.096 \\ 0.049 & \ldots & \ldots & 0.049 \end{bmatrix} \begin{bmatrix} X_0 \\ X_1 \\ X_2 \\ X_3 \\ X_4 \\ X_5 \\ X_6 \\ X_7 \end{bmatrix}$$

The discrete cosine transform can be implemented using Figure 45.

The weighting coefficient matrices for 8-by-1 Hadamard, Walsh, and Haar transforms are shown in Figure 41. The transforms described above all have real coefficients.

The Fourier transforms produces complex coefficients because of a complex weighting matrix. The discrete Fourier transform of a sequence is defined by

$$Y_k = \sum_{i=0}^{M-1} X_i e^{-j\frac{2\pi ik}{M}} \qquad k = 0, 1, 2, \ldots M-1 \tag{11}$$

Assuming an 8-point transform and defining

$$W = e^{-j\frac{2\pi}{8}} = \cos\frac{2\pi}{8} - j\sin\frac{2\pi}{8} \tag{12}$$

Equation 11 can be rewritten as

$$Y_k = \sum_{i=0}^{7} X_i W^{ik} \qquad k = 0, 1, 2, \ldots 7 \tag{13}$$

which is the same form as Equation 5. The weighting coefficients are now complex, requiring additional PIPE ESICs to calculate the real and imaginary Fourier coefficients. This is shown in Figure 47.

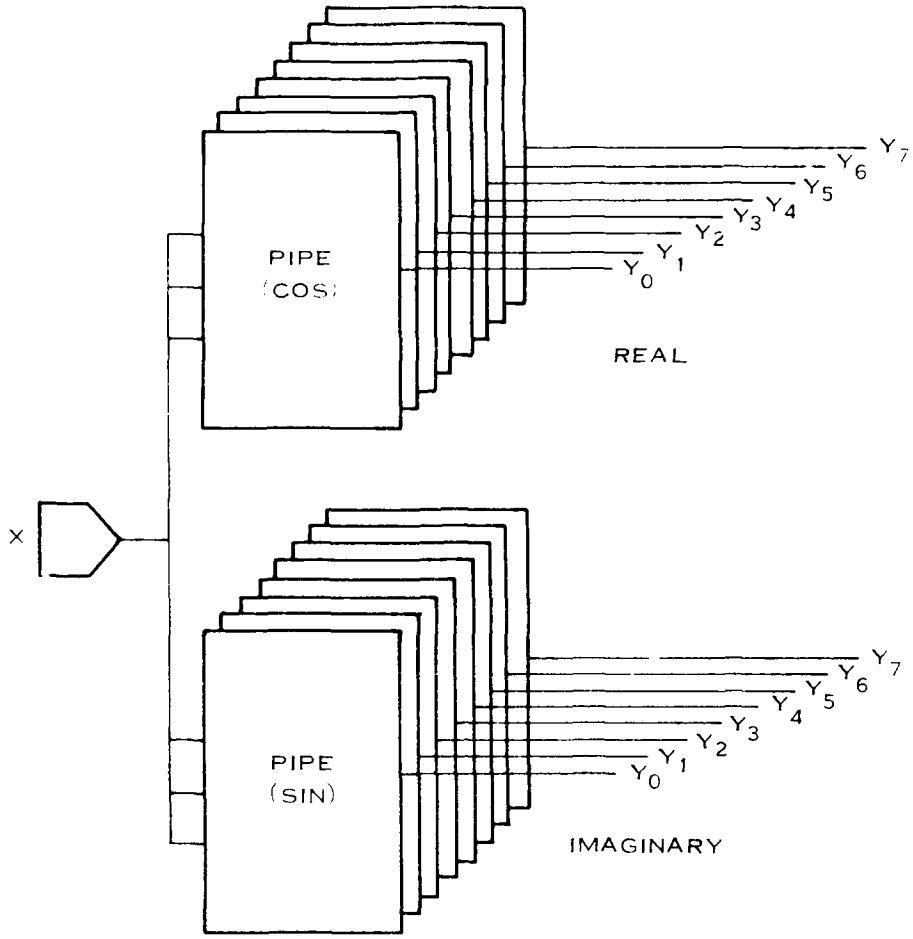


Figure 47. Implementation of Eight-Point Fourier Transform Using PIPE LSICs

b Two-Dimensional Transforms

For applications where two-dimensional data is used, two-dimensional transforms are required. If the transform is separable, a two-dimensional transform can be implemented by using a one-dimensional transform, first on the rows of the data and then followed by a transform of the columns of the coefficients of the first transform. Figure 48 shows an implementation of an 8 × 8 two-dimensional transform using PIPE LSICs. Data is loaded sequentially, one row at a time, into the row transform and, at the end of each row, the transform coefficient for that row appears as parallel output. These coefficients are parallel-loaded into a reformatting memory. The reformatting memory accumulates the 64 coefficients of the eight rows of the image and loads them sequentially into the column transform. Each eight coefficients of the row transform produces eight output coefficients.

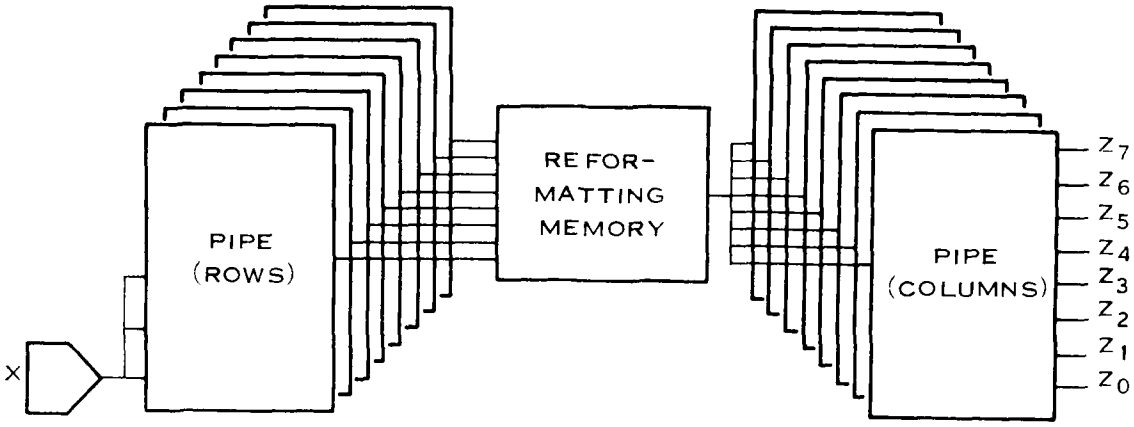


Figure 48 Implementation of Two-Dimensional Transform Using PIPE LSIC

c. *Pole-Zero Filtering*

Another important application of the PIPE LSIC is pole-zero filtering. Consider a system whose transfer function is given in the Z-domain by

$$H(Z) = \frac{\sum_{n=0}^{M} a_n Z^{-n}}{1 - \sum_{n=1}^{N} b_n Z^{-n}} = \frac{Y(Z)}{X(Z)} \tag{14}$$

The difference equation relating the input samples X_k to the output samples Y_k is given by

$$Y_k = \sum_{n=1}^{N} b_n Y_{k-n} + \sum_{n=1}^{M} a_n X_{k-n} \tag{15}$$

Using the PIPE LSIC to implement a pole-zero filter requires M to be less than or equal to 8 and N to be less than or equal to 9. As an example, consider a second-order filter with transfer function

$$H(Z) = \frac{a_0 + a_1 Z^{-1} + a_2 Z^{-2}}{1 - b_1 Z^{-1} + b_2 Z^{-2}} \tag{16}$$

$$H(Z) = \frac{Y(Z)}{X(Z)} = \frac{A_0 + A_1 Z^{-1} + A_2 Z^{-2}}{1 - B_1 Z^{-1} - B_2 Z^{-2}}$$

$$Y_K = A_0 X_K + A_1 X_{K-1} + A_2 X_{K-2} + B_1 Y_{K-1} + B_2 Y_{K-2}$$

$$Y_K = \begin{bmatrix} A_0 & A_1 & A_2 \end{bmatrix} \begin{bmatrix} X_K \\ X_{K-1} \\ X_{K-2} \end{bmatrix} + \begin{bmatrix} B_1 & B_2 \end{bmatrix} \begin{bmatrix} Y_{K-1} \\ Y_{K-2} \end{bmatrix}$$

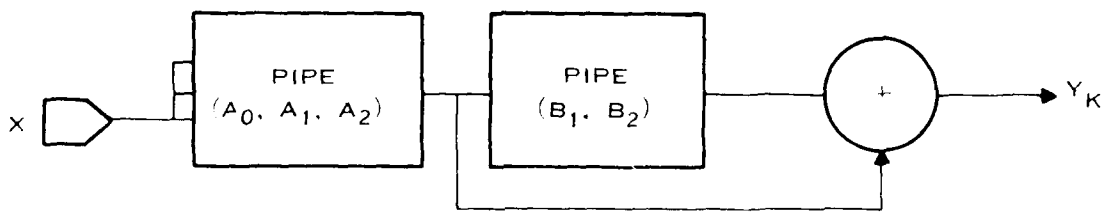


Figure 49. Using PIPE LSIC to Implement Second-Order Filter Function

The difference equation is

$$Y_k = a_0 X_k + a_1 X_{k-1} + a_2 X_{k-2} + b_1 Y_{k-1} + b_2 Y_{k-2} \tag{17}$$

or

$$Y_k = \begin{bmatrix} a_0 & a_1 & a_2 & b_1 & b_2 \end{bmatrix} \begin{bmatrix} X_k \\ X_{k-1} \\ X_{k-2} \\ Y_{k-1} \\ Y_{k-2} \end{bmatrix}$$

Equation 17 is the same form as Equation 6 and can be implemented using PIPE LSICs as shown in Figure 49. The X data sequence is loaded sequentially into the first PIPE LSIC containing weighting coefficients a_0, a_1, and a_2. The output of the first PIPE LSIC is loaded sequentially into the second PIPE LSIC to complete the calculation of Equation 17.

4. Neighborhood Operators

Many image-processing algorithms such as noise cleaning, edge detection, and edge enhancement can be implemented with the PIPE LSIC operating on sliding 3 × 3 pixel blocks of the input

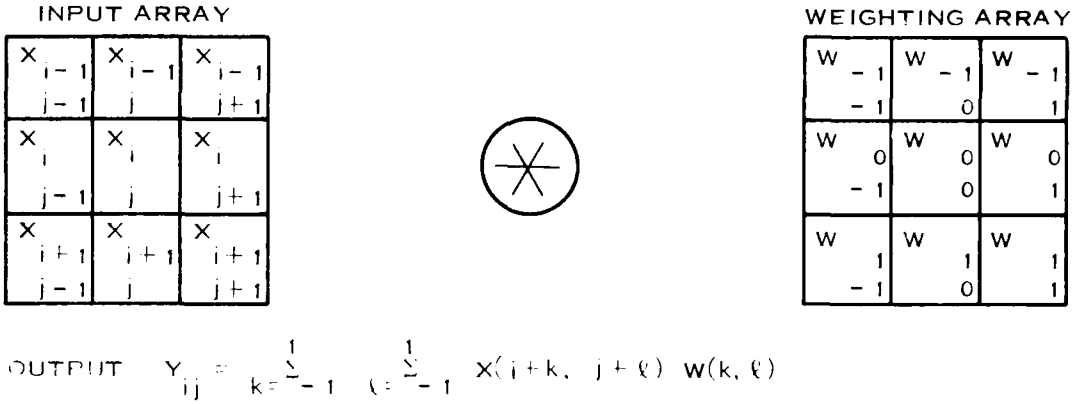


Figure 50 Two-Dimensional Spatial Convolution for 3 × 3 Neighborhood

image. A 3 × 3-input array is spatially convolved with a two-dimensional weighting array, as shown in Figure 50. This 3 × 3 two-dimensional spatial convolution is a very powerful image-processing calculation and one for which the PIPE LSIC is ideally suited.

The following subsections discuss some commonly used neighborhood operators and present some experimental results of the programmable sum of products breadboard described in Subsection II.A.7.

a. Noise Cleaning

Many images contain discrete pixel variations that are a result of noisy sensors and very objectional from a user viewpoint. Simple low-pass spatial filtering can eliminate or smooth most such noise since the noise is decorrelated spatially from its surrounding pixels. Figure 51 shows these low-pass weighting arrays that can be used as weighting coefficients for the PIPE LSIC. As can be seen, the weighting arrays are normalized to unit weighting to prevent an intensity bias into the processed image.

b. Edge Enhancement

Edge enhancement is used to accent edges of an image to provide a more subjectively pleasing image. Since areas of high frequency (edges) are to be highlighted, the logical operation is a high-pass filter. This can be done spatially, using the PIPE LSIC as a 3 × 3 neighborhood operator. Some weighting arrays that are of the high-pass form are shown in Figure 52. There is no need to normalize these arrays since their elements sum to unity.

c. Edge Detectors

One of the most distinguishing features of an image are edges because they provide information on the physical extent of objects within the image. Edges are defined as local discontinuities

W =

1/9	1/9	1/9
1/9	1/9	1/9
1/9	1/9	1/9

W =

1/10	1/10	1/10
1/10	2/10	1/10
1/10	1/10	1/10

W =

1/16	2/16	1/16
2/16	4/16	2/16
1/16	2/16	1/16

Figure 51. Low-Pass Weighting Arrays

W =

0	−1	0
−1	5	−1
0	−1	0

W =

−1	−1	−1
−1	9	−1
−1	−1	−1

W =

1	−2	1
−2	5	−2
1	−2	1

Figure 52. High-Pass Weighting Arrays

in the image luminance or amplitude level. There are basically two methods of edge detection: edge enhancement followed by thresholding and edge filtering.[1,5,6]

In the edge enhancement thresholding method, the input image is spatially convolved with a set of linear weighting arrays to produce a set of gradient functions which are, in turn, combined by a linear or nonlinear function to create an edge-enhanced array. To improve edge visibility, the gray level map is compared to a threshold, T: if the gray level is greater than T, an edge is assumed present; if the gray level is less than T, the decision is no edge. The selection of the threshold is very important; if it is too high, some edges will not be detected; if it is too low, noise will be detected as edges.

There are two types of edge-enhancement operators: differential edge detectors, such as Roberts, Prewitt, and Sobel; and template-matching edge-detection such as compass gradient, Kirsch, three-level, and five-level.

In edge fitting edge-detectors, subregions of the input image are fitted to a two-dimensional model of an edge. If the fit is close, an edge is assumed to exist with the same parameters as the edge model. Edge fitting cannot be implemented with the PIPE LSIC and is not discussed further.

	ROBERTS	PREWITT	SOBEL
HORIZONTAL	$\begin{bmatrix} 0 & -1 \\ 1 & 0 \end{bmatrix}$	$\begin{bmatrix} 1 & 0 & -1 \\ 1 & 0 & -1 \\ 1 & 0 & -1 \end{bmatrix}$	$\begin{bmatrix} 1 & 0 & -1 \\ 2 & 0 & -2 \\ 1 & 0 & -1 \end{bmatrix}$
VERTICAL	$\begin{bmatrix} -1 & 0 \\ 0 & 1 \end{bmatrix}$	$\begin{bmatrix} -1 & -1 & -1 \\ 0 & 0 & 0 \\ 1 & 1 & 1 \end{bmatrix}$	$\begin{bmatrix} -1 & -2 & -1 \\ 0 & 0 & 0 \\ 1 & 2 & 1 \end{bmatrix}$

Figure 53. Weighting Arrays for Various Differential Edge Detectors

d. Differential Edge Detectors

The Roberts edge detector is applied to a 2×2 neighborhood of pixels and can be implemented as two spatial convolutions of the input array with a weighting array. The outputs of the two convolutions are combined to produce an edge magnitude to be compared with the selected edge threshold, T. The orientation of the edge can also be calculated from the outputs of the two convolutions.

The Sobel and Prewitt edge operators are applied to 3×3 windows of pixels and are implemented by two spatial convolutions of the input image with a 3×3 weighting array. Again, the magnitude and orientation of the edge can be calculated from the two convolutions.

Another differential edge detection is the Laplacian operator; however, because of its sensitivity to points and lines, it is not a very efficient edge detector.[7,8]

The weighting arrays used for the Roberts, Sobel, and Prewitt edge detectors are shown in Figure 53. For each operator, the amplitude of the edge is given by

$$A(i,j) = \left\{ [Y^H(i,j)]^2 + [Y^V(i,j)]^2 \right\}^{1/2} \tag{18}$$

or

$$A(i,j) = |Y^H(i,j)| + |Y^V(i,j)|$$

where $Y^H(i,j)$ is the convolution of the input array and the horizontal weighting array and $Y^V(i,j)$ is the convolution of input array and the vertical weighting array.

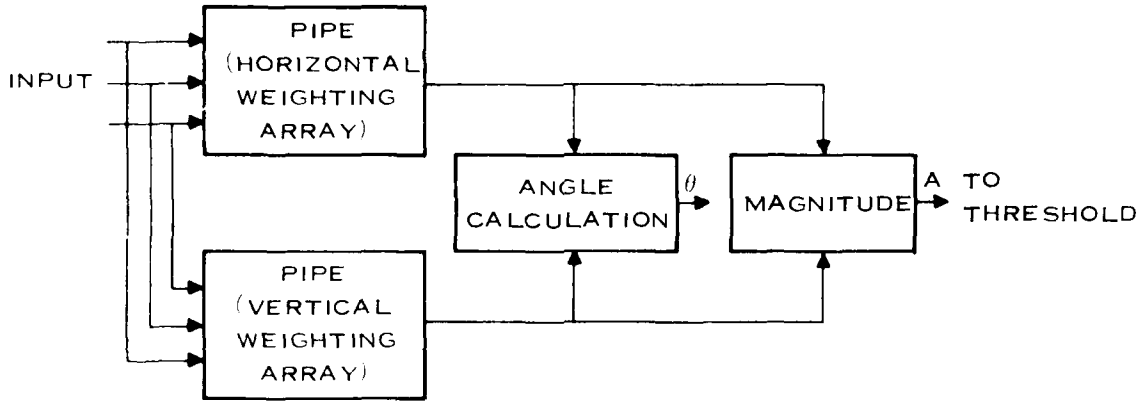


Figure 54. Implementation of Differential Edge Detectors Using PIPE LSIC

For the Roberts operator, the edge orientation is given by

$$\theta(i,j) = \frac{\pi}{4} + \tan^{-1}\left(\frac{Y^V(i,j)}{Y^H(i,j)}\right) \tag{19}$$

For the Sobel and Prewitt operator, the edge orientation is given by

$$\theta(i,j) = \tan^{-1}\left(\frac{Y^V(i,j)}{Y^H(i,j)}\right) \tag{20}$$

The PIPE LSIC can be used to implement the Roberts, Prewitt, or the Sobel edge detector. A block diagram of the implementation of these operators is shown in Figure 54. The input data is loaded as three sets of three parallel words. Three sample periods are required to initially load the PIPE LSIC and provide the first output. Succeeding outputs occur each sample period. For the Roberts operator, the weighting coefficients would be rearranged to take into account the 2 × 2 weighting array.

e. *Template Matching Edge Detectors*

In template-matching edge detection, a set of weighting arrays corresponding to the eight major compass directions (north, northeast, east, etc.) is convolved with the input image. The edge orientation is determined by the direction producing the maximum gradient response greater than the selected threshold. Examples of template-matching weighting arrays are given in Figure 55 for the compass-gradient, Kirsch, three-level, and five-level template-matching operators.

The PIPE LSIC can implement the template-matching edge detector easily, as shown in Figure 56. The input array is loaded as three sets of three parallel words in all eight PIPE LSICs.

1	1	1
1	-2	1
-1	-1	-1

NORTH

1	1	1
-1	-2	1
-1	-1	1

NORTHEAST

-1	1	1
-1	-2	1
-1	1	1

EAST

-1	-1	1
-1	-2	1
1	1	1

SOUTHEAST

-1	-1	-1
1	-2	1
1	1	1

SOUTH

1	-1	-1
1	-2	-1
1	1	1

SOUTHWEST

1	1	-1
1	-2	-1
1	1	-1

WEST

1	1	1
1	-2	-1
1	-1	-1

NORTHWEST

(A) COMPASS GRADIENT

3	3	3
3	0	3
-5	-5	-5

NORTH

3	3	3
-5	0	3
-5	-5	3

NORTHEAST

-5	3	3
-5	0	3
-5	3	3

EAST

-5	-5	3
-5	0	3
3	3	3

SOUTHEAST

-5	-5	-5
3	0	3
3	3	3

SOUTH

3	-5	-5
3	0	-5
3	3	3

SOUTHWEST

3	3	-5
3	0	-5
3	3	-5

WEST

3	3	3
3	0	-5
3	-5	-5

NORTHWEST

(B) KIRSCH

1	1	1
0	0	0
-1	-1	-1

NORTH

0	1	1
-1	0	1
-1	-1	0

NORTHEAST

-1	0	1
-1	0	1
-1	0	1

EAST

-1	-1	0
-1	0	1
0	1	1

SOUTHEAST

-1	-1	-1
0	0	0
1	1	1

SOUTH

0	-1	-1
0	0	-1
1	1	0

SOUTHWEST

1	0	-1
1	0	-1
1	0	-1

WEST

1	1	0
1	0	-1
0	-1	-1

NORTHWEST

(C) THIRD-LEVEL

1	2	1
0	0	0
-1	-2	-1

NORTH

0	1	2
-1	0	1
-2	-1	0

NORTHEAST

-1	0	1
-2	0	2
-1	0	1

EAST

-2	-1	0
1	0	1
0	1	2

SOUTHEAST

-1	-2	-1
0	0	0
1	2	1

SOUTH

0	-1	-2
1	0	-1
2	1	0

SOUTHWEST

1	0	-1
2	0	-2
1	0	-1

WEST

2	1	0
1	0	-1
0	-1	-2

NORTHWEST

(D) FIFTH-LEVEL

Figure 55. Template Match Edge Detector Weighting Arrays

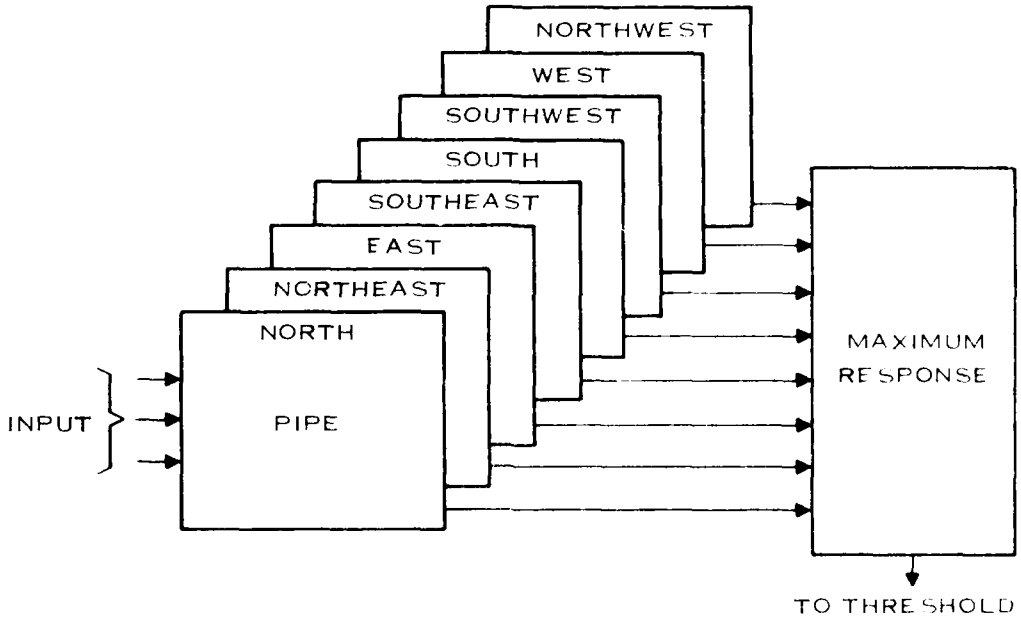


Figure 56. Implementation of Template Match Edge Detectors Using PIPE LSIC

The outputs are compared to determine the maximum response and, hence, the orientation of the edge.

f. *Edge Detector Performance Analysis*

In an attempt to determine the relative performance of the edge detectors discussed above, an evaluation has been performed comparing edge response as a function of actual edge orientation, the probability of correct detection as a function of the probability of false detection, and a figure of merit as a function of signal-to-noise ratio.[5,6]

Figure 57 shows edge detector response (amplitude and orientation) as a function of actual edge orientation.[6] For the Roberts, Prewitt, and Sobel amplitude response, both the square root sum of the squares and the sum of the absolute values (Equation 18) responses are shown. As can be seen, the Prewitt, Sobel, and template matching amplitude response is relatively invariant to edge orientation while the Sobel operator has the most linear response between actual edge orientation and detected edge orientation. For the template-matching operators, the difference between actual and detected edge orientation is large because the template-matching operators measure edge orientation in a quantized step.

The performance of various edge detectors in the presence of additive, white Gaussian noise can be compared using parametric curves of corrected detection probability versus false detection probability in terms of the detector threshold.[7] Figure 58 shows such curves for vertical and diagonal edges for both differential edge detectors and template-matching detectors with signal-to-noise ratios of 1.0 and 10.0. For the differential detectors, the Sobel and Prewitt perform better

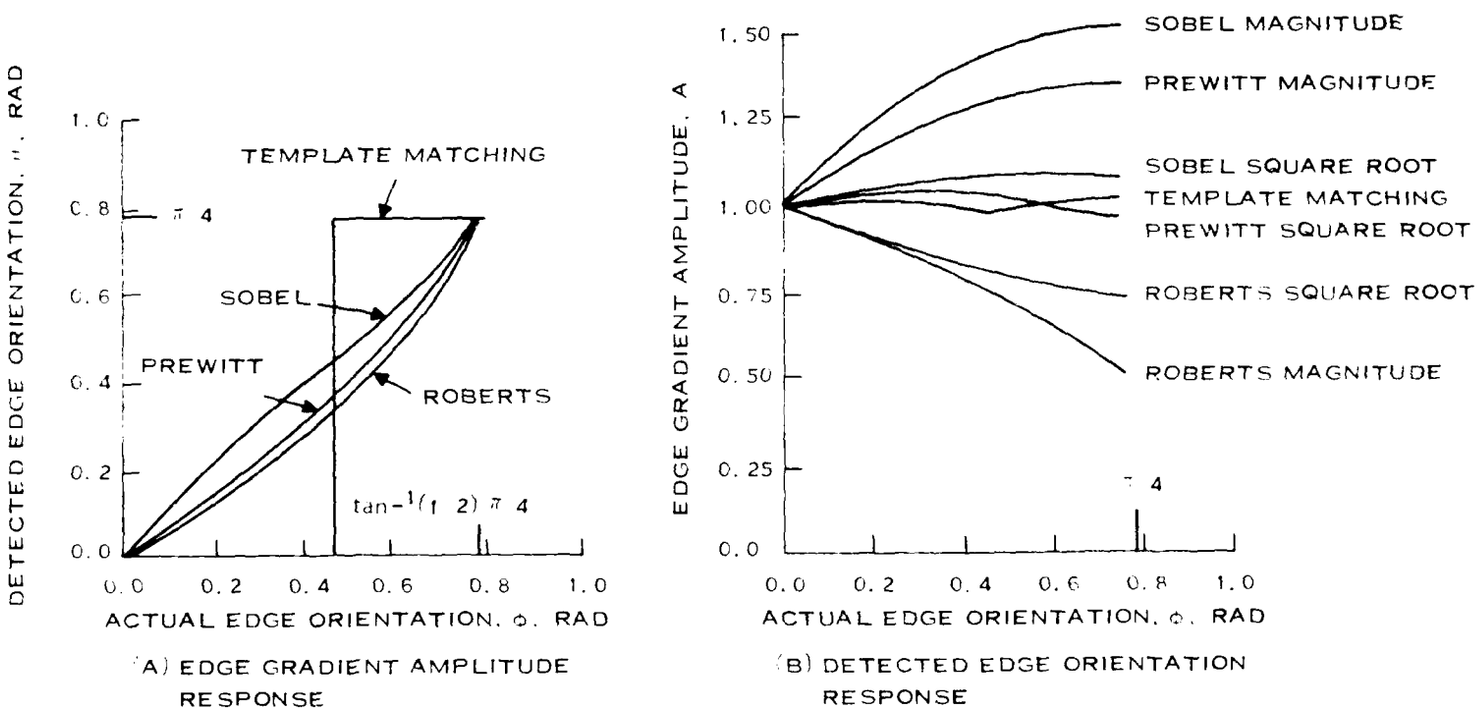


Figure 57. Edge-Gradient Amplitude Response and Detected-Edge Orientation as Functions of Actual Edge Orientation (From Reference 6)

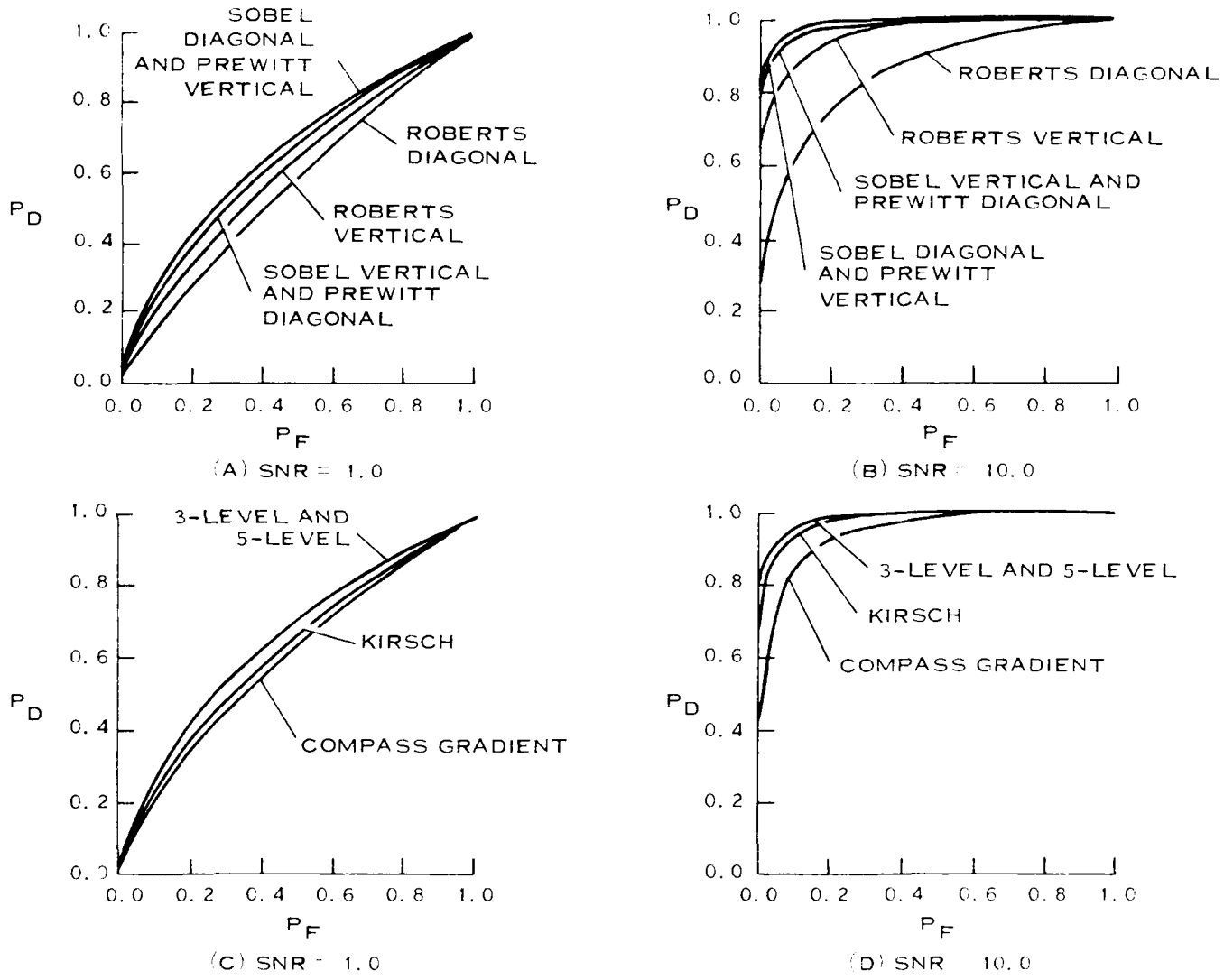


Figure 58. Probability of Detection Versus Probability of False Detection (From Reference 6)

than the Roberts, with the Prewitt operator performing better for vertical edges and the Sobel operator superior for diagonal edges. For the template matching edge detectors, the three-level and five-level operators perform best. In comparing the differential operators to the [illegible] operators, the Sobel and Prewitt perform slightly better than the three-level [illegible].

Another method of comparing the various edge detectors is based on a figure of merit that penalizes for fragmented, smeared, and offset edges.[illegible] This figure of merit [illegible] weighting the different errors according to importance and adjusting each edge [illegible] to obtain its optimum performance. This figure of merit is defined by:[illegible]

$$R = \frac{1}{\mathrm{MAX}(I_I, I_A)} \sum_{i=1}^{I_A} \frac{1}{1 + \alpha d^2}$$

where I_I and I_A represent the number of ideal and actual edge map points [illegible] that can be adjusted to penalize edges which are localized but offset from the [illegible] and d is the separation distance [illegible] an actual edge point normal to a line of ideal edge points. The figure of merit is normalized so that $R = 1$ indicates a perfectly detected edge. Figure 59 shows [illegible] figure of merit versus signal-to-noise ratio for various differential and template matching edge detectors. For each edge detector, the edge threshold has been adjusted to obtain a maximum figure of merit. As shown in Figure 59A and 59B, the Prewitt and the Sobel operators [illegible] the Roberts. The Prewitt performs best on a vertical edge, with the Sobel performing slightly better on diagonal edges. For high signal-to-noise ratios, there is only a small difference between the Sobel and the Prewitt. For the template-matching edge detectors, the three-level, the five-level and the Kirsch are better than the compass gradient. For vertical edges, the best edge detector varies with signal-to-noise ratio while, for diagonal edges, the three-level edge detector gives a higher figure of merit. In comparing the differential to the template matching edge detectors, the Prewitt square root is slightly better than the three-level template match for vertical edges while, for diagonal edges, the three-level template matching operator gives a slightly higher figure of merit.

In summary, there is little difference between the Prewitt and the Sobel differential edge detectors and the three-level template matching edge detector. The three-level edge detector has the advantage of having the same performance for all edge orientations while the Prewitt and the Sobel require less computation.

5. Experimental Results

The programmable sum of product breadboard discussed in Subsection II.A.[illegible] has been interfaced to Texas Instruments Image Processing Laboratory to provide an experimental breadboard for the PIPE ESIC. The hardware interface to the laboratory's host computer was implemented with a standard 16 I/O data module and, consequently, is relatively slow. The input to the breadboard is down-loaded from the Image Processing Laboratory's extensive data base.

Figures 60 through 62 show the operation of the breadboard as a 3×3 neighborhood operator with various weighting coefficients. Figure 60 shows the result of low pass and high-pass filtering. In the low pass case, the image is somewhat blurred while, in the high pass case, only areas with high frequency content are preserved. Figure 61 shows the result of the Prewitt and the Sobel differential edge detectors, using the sum of the absolute values method to obtain the edge magnitude. No thresholding was performed. The results of convolving the North weighting array of

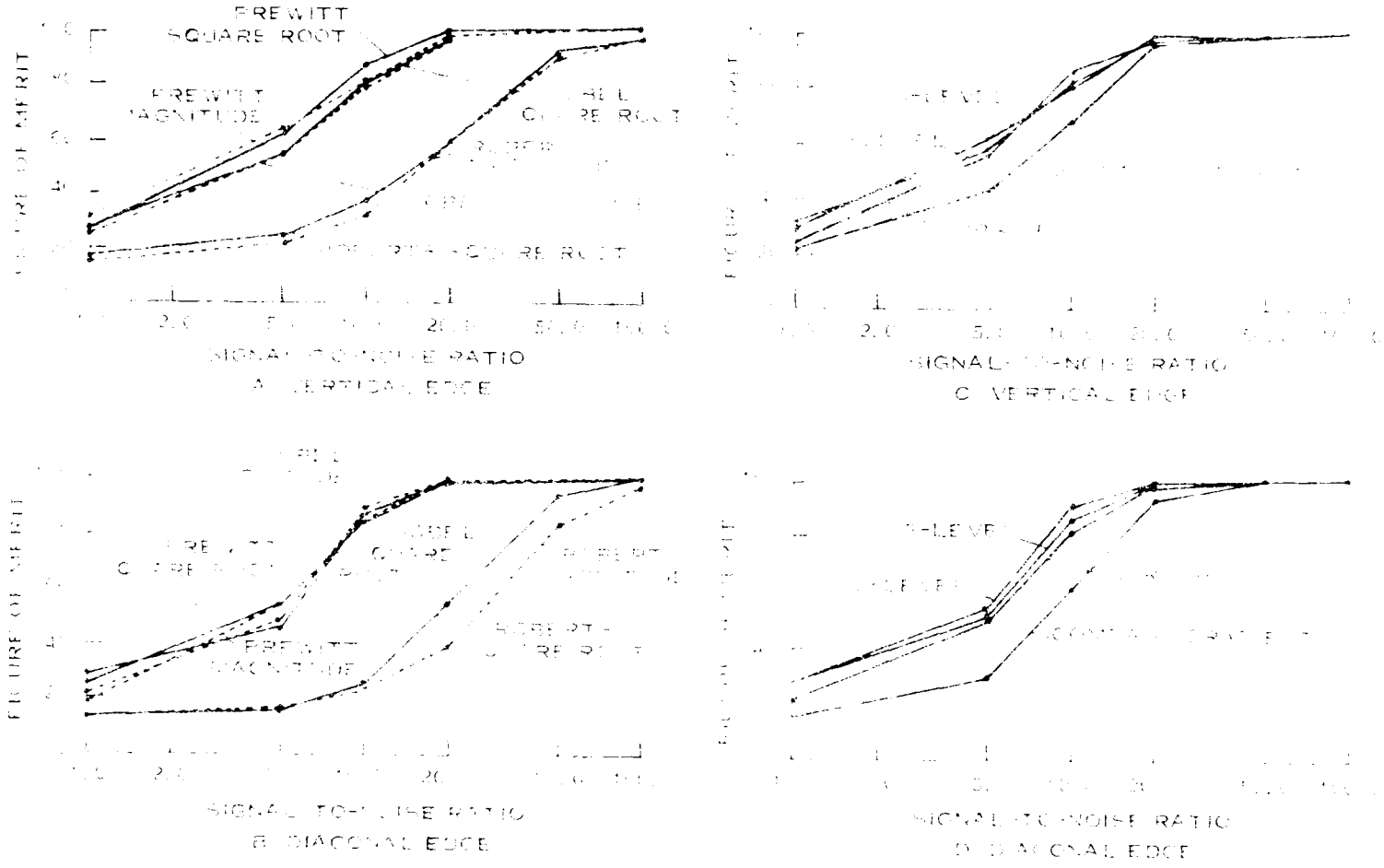


Figure 5b. Figures of Merit [illegible] of Signal-to-Noise Ratio for Edge Detectors (Text Reference 6)

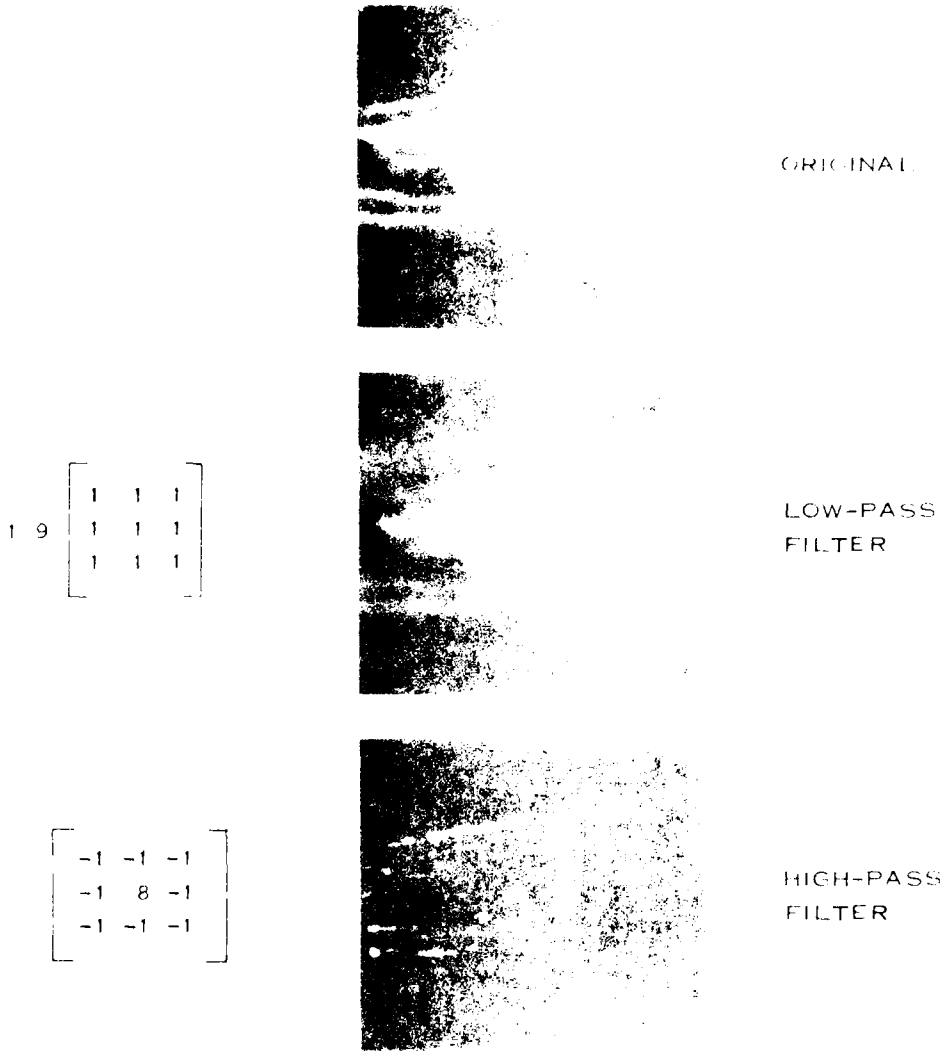


Figure 60. Operation of Programmable Sum of Products Breadboard as 3 × 3 Neighborhood Operator

the compass gradient, the Kirsch, the three-level and the [illegible] with the original image are shown in Figure 62.

6. Operational Characteristics

Several key parameters define the operation [illegible] the PIPE LSIC: the types of operations (serial or parallel, sliding or nonsliding), [illegible] of input word (16 bits, 8 bits, etc.), EPROM access time; accumulator time; and the strobe [illegible] input latches.

Since the PIPE LSIC must access the on-board [illegible] B_L times (where B_L is the length of the input word) to calculate one term of the answer [illegible] accumulator must sum these terms, the time required to output an answer from the PIPE LSIC

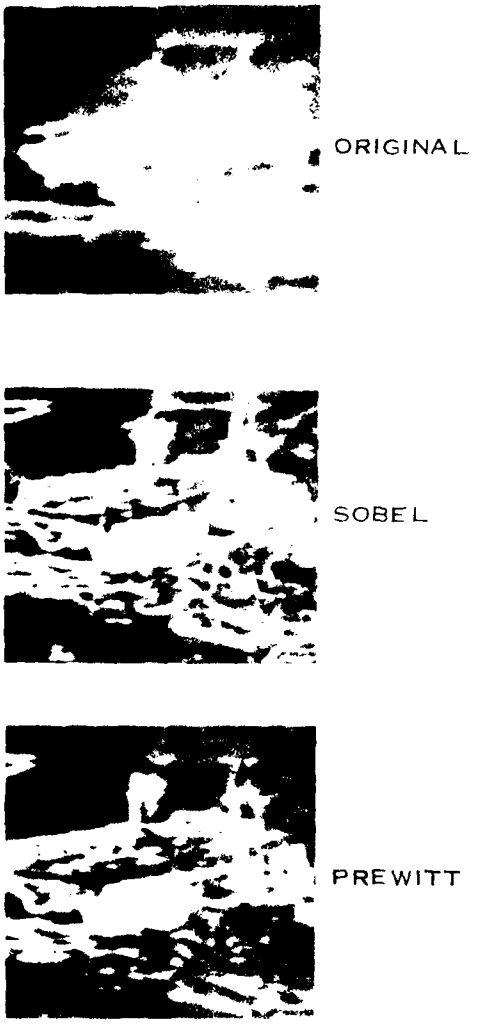


Figure 61. Sobel and Prewitt Edge Detection Using Programmable Sum-of-Products Breadboard

$$T_{out} = MAX(T_A \times B_X, T_{sum}) \tag{22}$$

where T_A is the on-chip EPROM access time, and T_{sum} is the time required by the accumulator. To prevent addressing the memory incorrectly, the time between load pulses must also satisfy Equation 22.

Recalling that the number of input strobe pulses between load pulses determines whether a sliding or nonsliding type of operation is performed, the time between strobe pulses (T_{STROBE}) can be given as

$$T_{STROBE} = MAX(T_{LOAD} \div N, T_{LATCH}) \tag{23}$$

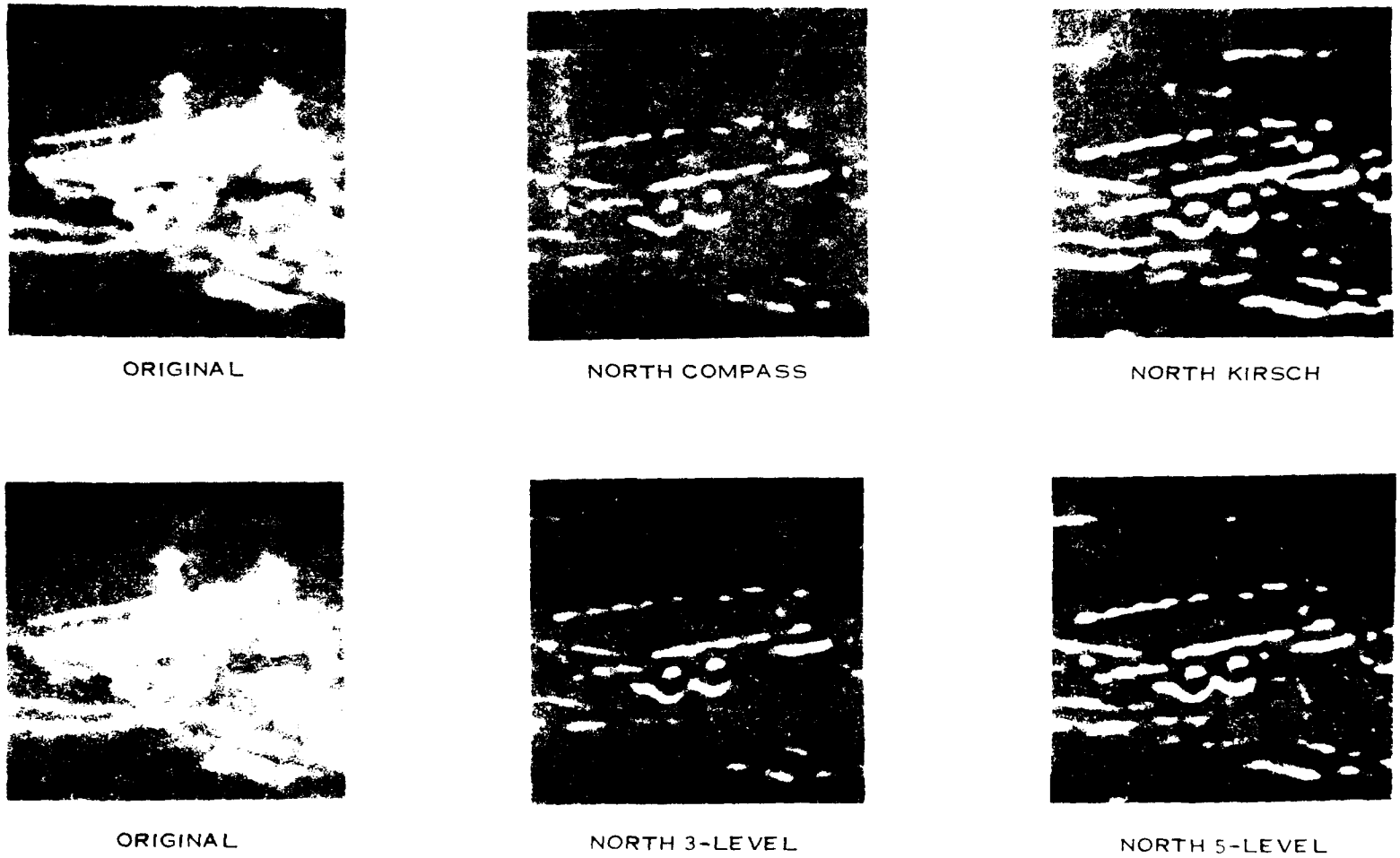


Figure 62. Template Match Edge Detection Using Programmable Sum of Products Breadboard

where N is the number of input strobes between load pulses, T_{LOAD} is the time between load pulses as determined by Equation 22, and T_{LATCH} is the operating time for the input latches.

To maintain proper on-chip timing, the master clock (T_{CLOCK}) must be greater than B_x times the load frequency, or

$$T_{CLOCK} > \frac{T_{LOAD}}{B} \tag{24}$$

Table 3 summarizes Equations 22 through 24 in terms of input and output data rates for different types of operations, word length, and EPROM access time (assuming 300-ns accumulator time), 20-MHz input latch, and 20-MHz master clock (design goals).

In the serial 8 × 1 sliding type of operation, one strobe pulse is needed between each load pulse, and the load frequency is calculated from Equation 22 to be 1.25 MHz for 8-bit data and 1.67 MHz for 6-bit data, assuming 100-ns EPROM access time. For serial 8 × 1 nonsliding type of operations, eight strobe pulses are required between each load pulse; therefore, the maximum load frequency is 1.25 MHz; however, since data can be loaded in the PIPE input latches independent of the parallel-to-serial registers, the maximum input data rate is determined by the input strobe frequency. This can be calculated from Equation 23 to be 10 MHz for both 8- and 6-bit data for 8 × 1 nonsliding type of operations.

For the parallel 3 × 3 sliding type of operation, only one strobe pulse is needed between load, thus, from Equations 22 and 23, the load frequency and input data rate (input strobe frequency) can be calculated to be 1.25 MHz for 8-bit data and 1.67 MHz for 6-bit data, with an EPROM access time of 100 ns. For parallel 3 × 3 nonsliding type of operations, three strobe pulses are required between load pulses; therefore, the input data rate is 10 MHz for both 8- and 6-bit data. The effect of EPROM access time on input data rate is also shown in Table 3.

The output data rate is given by Equation 22, regardless of the type of operation, and is 1.25 MHz for 8-bit data and 1.67 MHz for 6-bit data with 100-ns EPROM access time.

To achieve real-time (10-MHz) operation on 8-bit data with 50-ns EPROM access time, four PIPE LSICs can be operated in parallel as shown in Figure 63 for transform calculations and Figure 64 for neighborhood operators. In both implementations, the input data is demultiplexed by the input strobe pulse of the PIPE LSIC into the four parallel PIPEs and the tri-state outputs are multiplexed, using the enable pulse of each PIPE LSIC. A block diagram of the PIPE LSIC Demonstration Brassboard is shown in Figure 65.

TABLE 3. PIPE LSIC INPUT/OUTPUT DATA RATES

Type of Operation	Data Word Length							
	8-Bit				6-Bit			
	EPROM Access Time				EPROM Access Time			
	100 ns		50 ns		100 ns		50 ns	
Serial	**Input**	**Output**	**Input**	**Output**	**Input**	**Output**	**Input**	**Output**
8 × 1 Sliding	1.25 MHz	1.25 MHz	2.5 MHz	2.5 MHz	1.67 MHz	1.67 MHz	3.3 MHz	3.3 MHz
8 × 1 Nonsliding	10 MHz	1.25 MHz	10 MHz	2.5 MHz	10 MHz	1.67 MHz	10 MHz	3.3 MHz
Parallel								
3 × 3 Sliding	1.25 MHz	1.25 MHz	2.5 MHz	2.5 MHz	1.67 MHz	1.67 MHz	3.3 MHz	3.3 MHz
3 × 3 Nonsliding	10 MHz	1.25 MHz	10 MHz	2.5 MHz	10 MHz	1.67 MHz	10 MHz	3.3 MHz

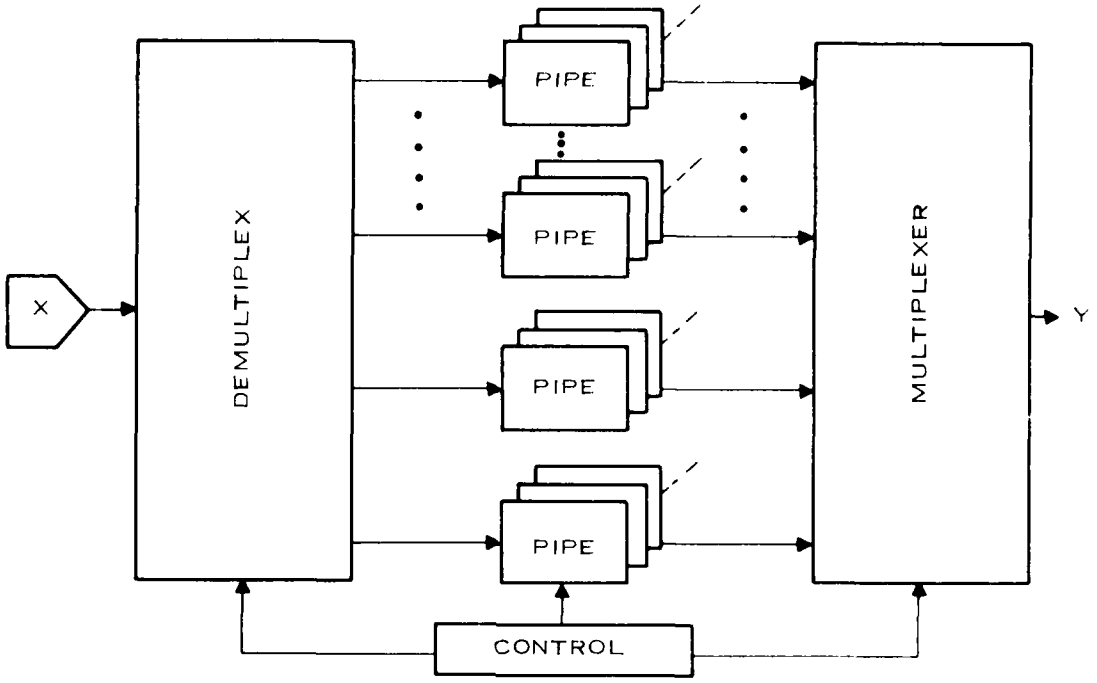


Figure 63. Real-Time Transform Processing Using Parallel PIPE LSIC

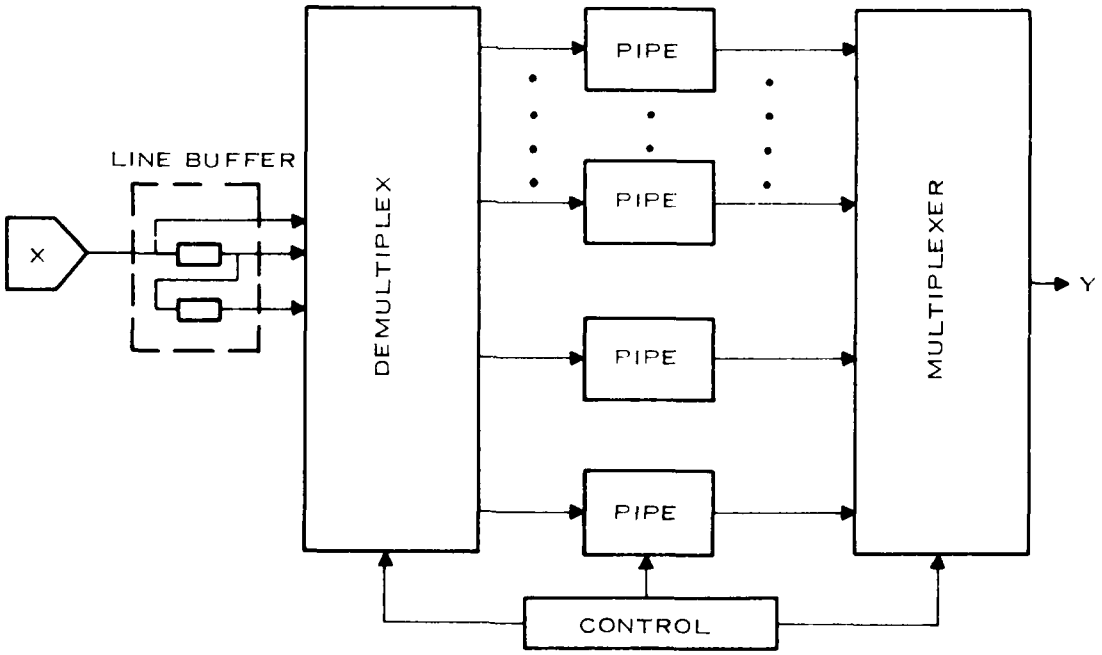


Figure 64. Block Diagram of Parallel PIPE LSIC to Implement Real-Time Neighborhood Operator Calculations

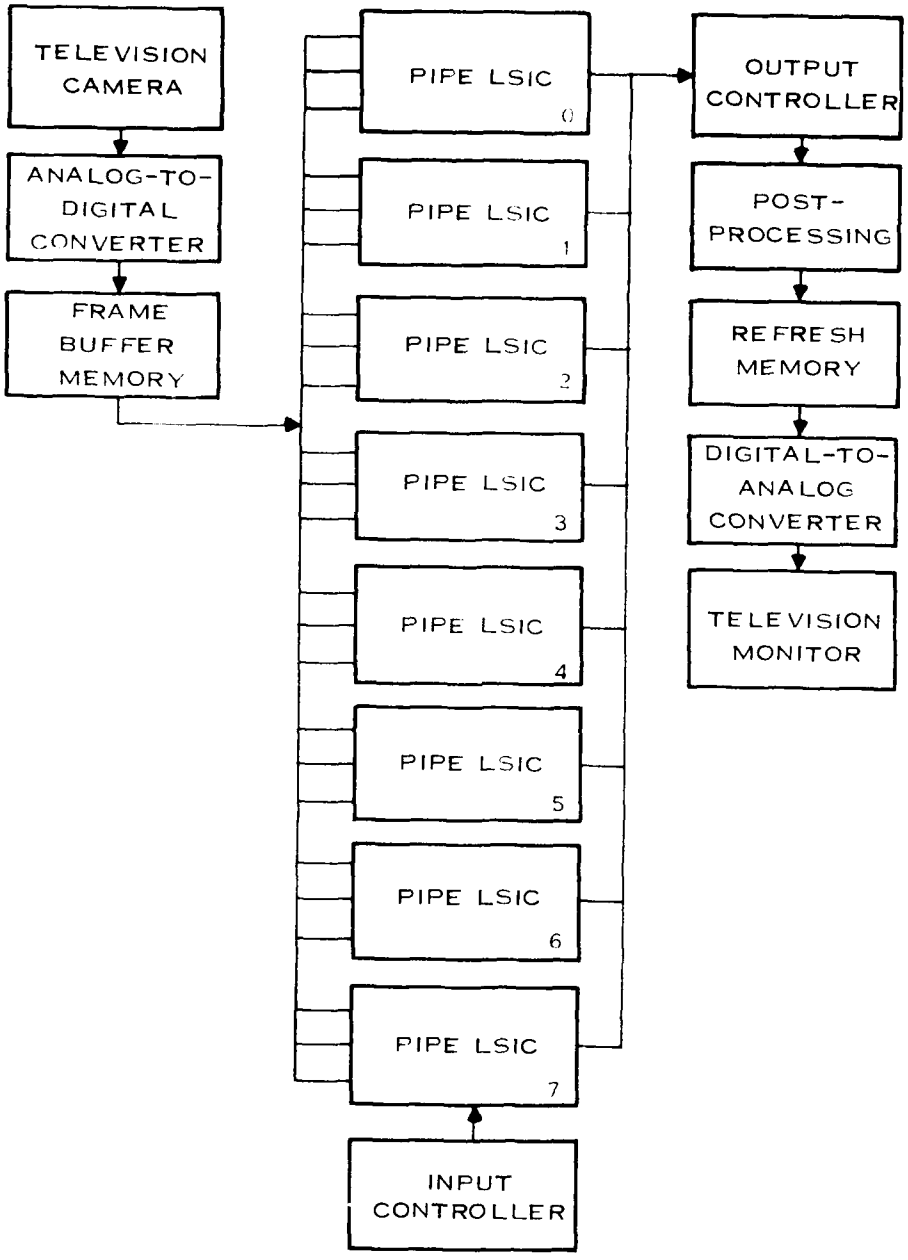


Figure 65. Block Diagram of the PIPE LSIC Demonstration Brassboard

REFERENCES

1. W.K. Pratt, *Digital Image Processing*, Wiley-Interscience, New York, 1978.

2. C.S. Burrus, "Digital Filter Structures Described by Distributed Arithmetic," *IEEE Transactions on Circuits and Systems*, Vol. CAS-24 (December 1977), pp. 674–680.

3. H.J. DeMan, C.J. Vandenbulcke, and M.M. Van Cappellen, "High Speed NMOS Circuits for ROM-Accumulator and Multiplier Type Digital Filters," *IEEE Journal Solid-State Circuits*, Vol. SC-13 (October 1978), pp. 565–572.

4. T.A.C. Classen, W.F.G. Mecklenbrauker, and J.B.H. Peek, "Some Considerations on the Implementation of Digital Systems for Signal Processing," *Phillips Research Reports*, Vol. 30 (1975), pp. 73–84.

5. W.K. Pratt, "Quantitative Design and Evaluation Methods for Edge and Texture Feature Extraction," *Image Understanding Workshop Proceedings*, Pittsburgh (November 1978), pp. 103–109.

6. I. Abdou, "Quantitative Methods of Edge Detection," University of Southern California, Image Processing Institute, USCIPI Report 800, Los Angeles, California, 1978.

☆U.S.Government Printing Office: 1981 — 757-002/573