N O T I C E

THIS DOCUMENT HAS BEEN REPRODUCED FROM MICROFICHE. ALTHOUGH IT IS RECOGNIZED THAT CERTAIN PORTIONS ARE ILLEGIBLE, IT IS BEING RELEASED IN THE INTEREST OF MAKING AVAILABLE AS MUCH INFORMATION AS POSSIBLE

(NASA-CR-163237) PATTERN CLASSIFICATION USING CHARGE TRANSFER DEVICES Final Report (North Carolina State Univ. at Raleigh.) 138 p HC A07/MF A01 CSCL 14E

N80-25760

Unclas
G3/43 22432

DEPARTMENT OF ELECTRICAL ENGINEERING

NORTH CAROLINA STATE UNIVERSITY

RALEIGH, NORTH CAROLINA

PATTERN CLASSIFICATION USING CHARGE TRANSFER DEVICES

Final Report

on Grant NSG 1535

to the

National Aeronautics and Space Administration
Langley Research Center
Hampton, Virginia

June 16, 1980

1 INTRODUCTION

1.1 Project Goals

For the past two and a half years, we have performed a study of the feasibility of the use of Charge Transfer Devices (CTDs) in the classification of multi-spectral image data. This work consisted of two primary stages: 1) An evaluation of particular devices to determine their suitability in a matrix multiplication subsystem of a pattern classifier and 2) The design of a prototype of such a system if a suitable device was found. The work centered around "analog-analog correlator" devices which consist of two tapped delay lines, on chip multipliers, and a summed output; these devices will be discussed in more detail in Section 2 of this report.

1.2 Summary of Results

In general, our results have been encouraging; one of the tested devices showed performance characteristics which warranted further development, the design of the system was accomplished, and construction was begun. Reference is made to two previous progress reports dated December 20, 1977 and August 28, 1979. These papers contain detailed reports on the results to the time of their issue, and the material contained in them will only be summarized in this report.

The previous reports indicated the following findings:

(1) The first device evaluated, the Reticon AAC-32 was found to be not suitable because of serious linearity problems.

(2) A second series of devices, the Reticon R5402/5403, was tested and was found to have acceptable accuracy and reasonable linearity indicating a need to explore its use in a pattern classifier.

(3) An architectural design and part of the detailed design for a multispectral classifier using the Reticon devices and controlled by an LSI-11 microcomputer was completed.

(4) Software was developed to support communication between the LSI-11 and a VAX medium scale computer. This software allowed the reading of LANDSAT tapes and the subsequent transfer of the multispectral images to the LSI-11 floppy disk for use by the classifier.

(5) Software was also developed to support the display of false colored images on a Chromatics graphics terminal.

Additional accomplishments since the August report are:

(1) The design for the classifier was completed.

(2) A printed circuit layout for the analog boards was completed, and the boards were fabricated. The other boards will be wire-wrapped.

(3) A test jig for the analog board was built and check-out begun.

(4) System software development was begun.

2 CTD THEORY

This section contains a brief overview of Charge Transfer Device characteristics. For a detailed explanation of CTD construction and operation see the 1977 report and its references.

A CTD is simply a monolithic integrated circuit which moves packets of charge linearly in synchronism with a clock. Depending upon the application, these charge packets may be used to represent either digital or analog information. CTDs have been used for a wide variety of purposes, particularly analog signal processing, digital memories, and imaging arrays.

Two very common types of CTDs are the "Bucket Brigade Device" (BBD) and the "Charge Coupled Device" (CCD). The major difference between these two devices is the manner in which the charge packets are stored and transferred from cell to cell. This difference results in slightly differing performance characteristics between the two types.

A common use of CTDs is signal processing has been as simple analog delay lines or shift registers. By modifying the clocking electrodes complex functions of the input rather than simple delays have been realized. These functions are of the form: $\Sigma\ a_i b_i$ where the b_is are samples of the input signal and the a_is are weighting coefficients. Such devices are known as "fixed tap weight devices" (from their filtering applications) because the weights are determined at manufacture and cannot be changed afterward. Fixed tap weight devices have been used to produce a variety of functions such as matched filtering, correlation, or the magnitude of the discrete Fourier transform.

The versatility of the CTD may be extended by using two delay lines, adding analog multipliers at each point, and providing a mechanism for summing the outputs of the multipliers. Such a device may be used to perform sum-of-product operations in which both of the operands are arbitrary. Experimental samples of these "variable tap weight devices" are now available, and these devices are the subject of this study.

3 PATTERN CLASSIFICATION USING CTDS

A major bottleneck in pattern classification operations has been the matrix multiplication required in the calculation of the discriminate function:

$$[\bar{x}-\bar{\mu}_i]^T \; C_i^{-1} \; [\bar{x} - \bar{\mu}_i]$$

where $\bar{x}$ is the data point vector, $\bar{u}$ is the mean vector of the ith class, and C_i (also commonly signified by Σ_i) is the covariance matrix of the ith class.

Figure 3.1 illustrates a proposed layout of sum of product CTDs (SOPs) which may be used to calculate the discriminant function. In this arrangement each of the first group of SOP devices (Σ_i --Σ_m) perform one of the row-column multiplications of the $\bar{x}-\bar{\mu}_i \; {}^T C_i^{-1}$ term of the discriminant function calculation. The results from these operations are then multiplexed into the final SOP device along with the $\bar{x}-\bar{\mu}_i$ term producing the desired final result. It is seen that this configuration requires at least two CTD loading times to perform the calculation.

A reduction of the calculation time to one SOP device loading time would be highly desirable. Figure 3.2 shows an arrangement which accomplishes this reduction by eliminating the multiplexer and the final CTD. The diagonal symmetry of the covariance matrix allows this simplification because the matrix may be transformed into upper triangular form. In this case the SOP devices perform the $C_i^{-1} \; \bar{x}-\bar{\mu}_i$ calculation as before. These results are then each squared and the results summed to produce the answer.

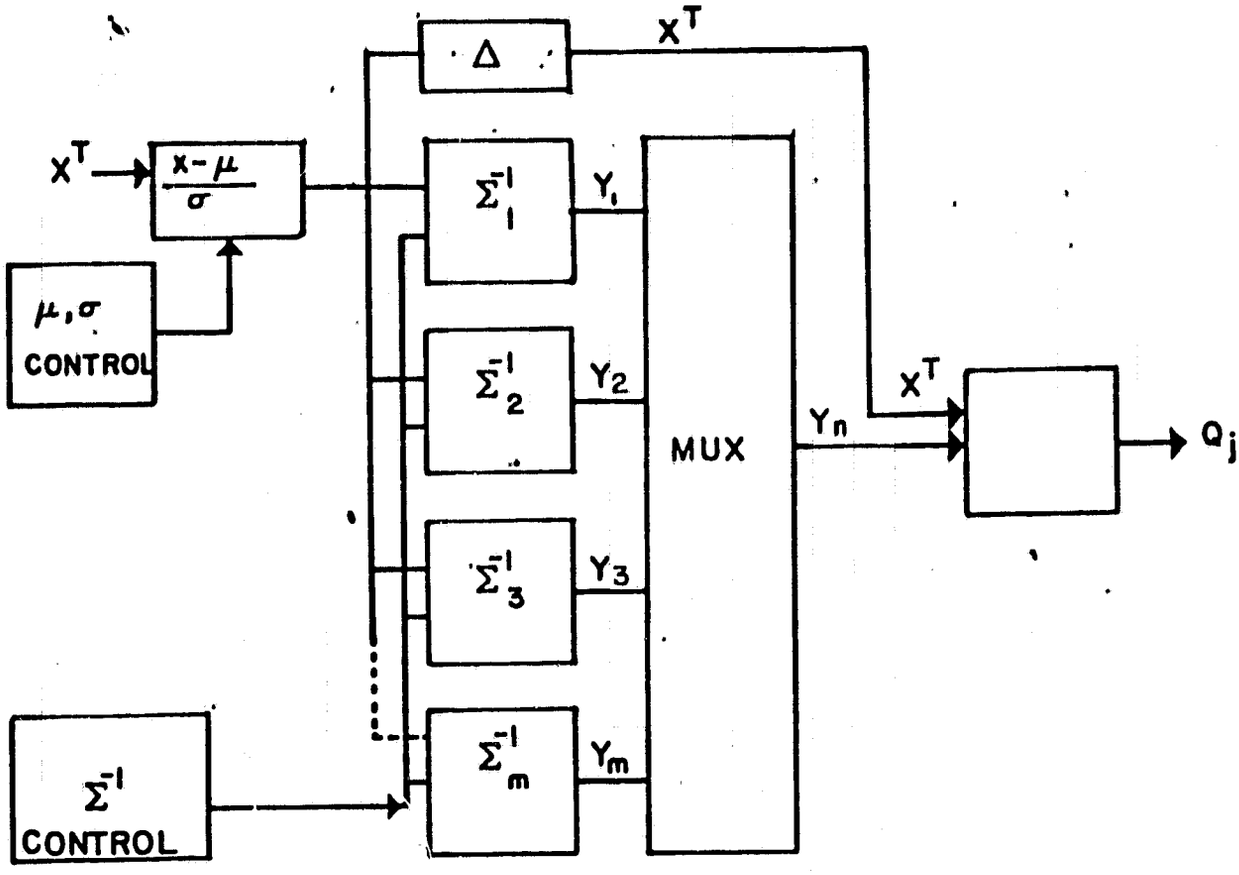


Figure 3.1 Functional Layout of Classifier

ORIGINAL PAGE IS
OF POOR QUALITY

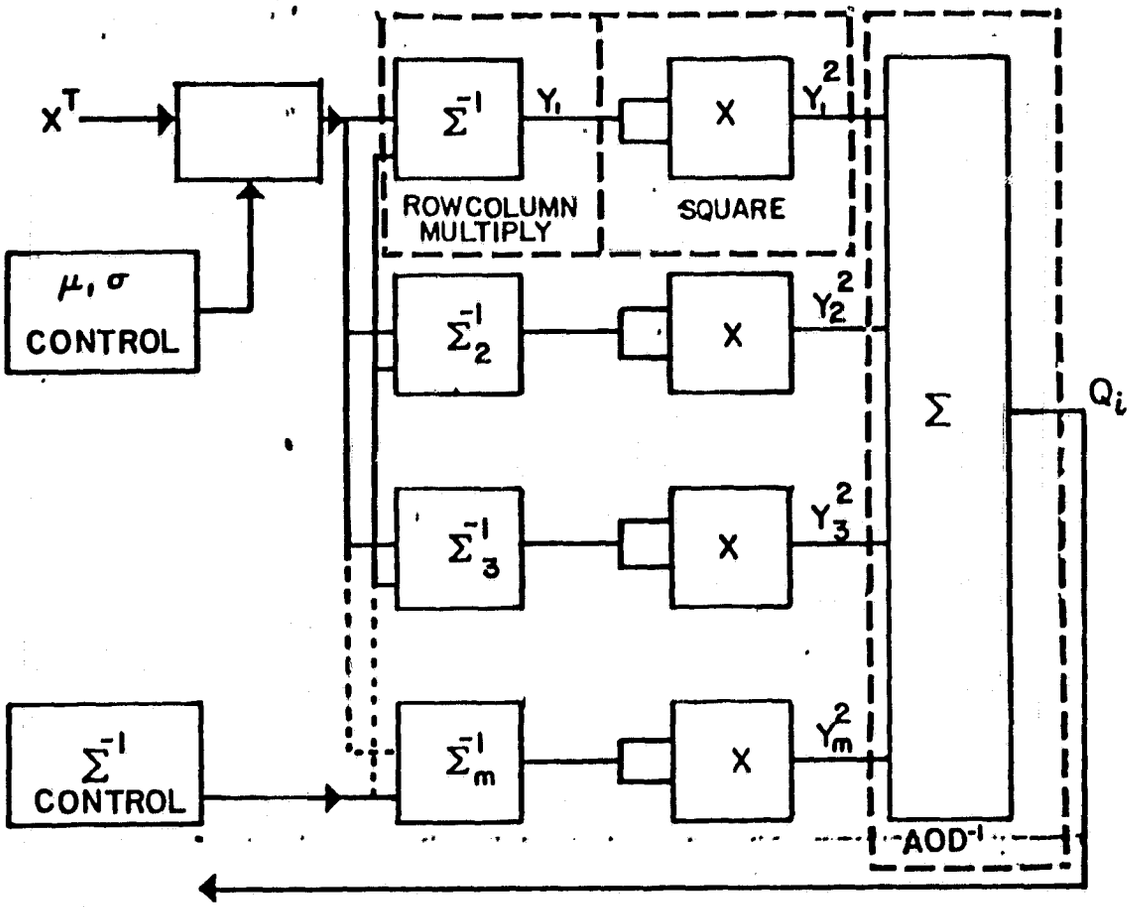


Figure 3.2 Layout of Classifier with Higher Speed Capability

4 DEVICE EVALUATIONS

4.1 AAC-32

Once the matrix multiplier configuration was derived an evaluation of the available devices was needed. A detailed account of the evaluation techniques and testing set-up hardware is given in the 1977 report.

The first device evaluated was the Reticon AAC-32, a bucket brigade device. This chip contains two tapped 32 cell delay lines with an untapped or "dead" cell at the start of one delay line and the end of the other. Thirty-two on-board multipliers and a summer compute the sum-of-products function.

After determining values for the various bias voltages and currents required by the chip, tests were made to determine proper signal levels and to evaluate the performance of the device. Zeroing and signal level adjustments were satisfactorily executed, then linearity, accuracy, and repeatability were examined.

The results showed an accuracy of about 5 bits and a repeatability of about 8-9 bits with the variance appearing to be the result of random noise. Also noted were short-term and long-term drifts when the device was left unclocked and a start-up inaccuracy. These results all appeared to indicate that the device was acceptable for a matrix multiplication application.

The AAC-32 however, showed a exhibited a serious linearity problem: in one quadrant, the output was severely clipped. Adjustments and reductions of the input could not remove the clipping. The failure appeared to be caused by the multipliers; observations of the delayed signal after it passed through the

bucket brigade showed no distortion problems. This non-linearity caused the AAC-32 to be ruled unsuitable.

4.2 R5402/5403 Series

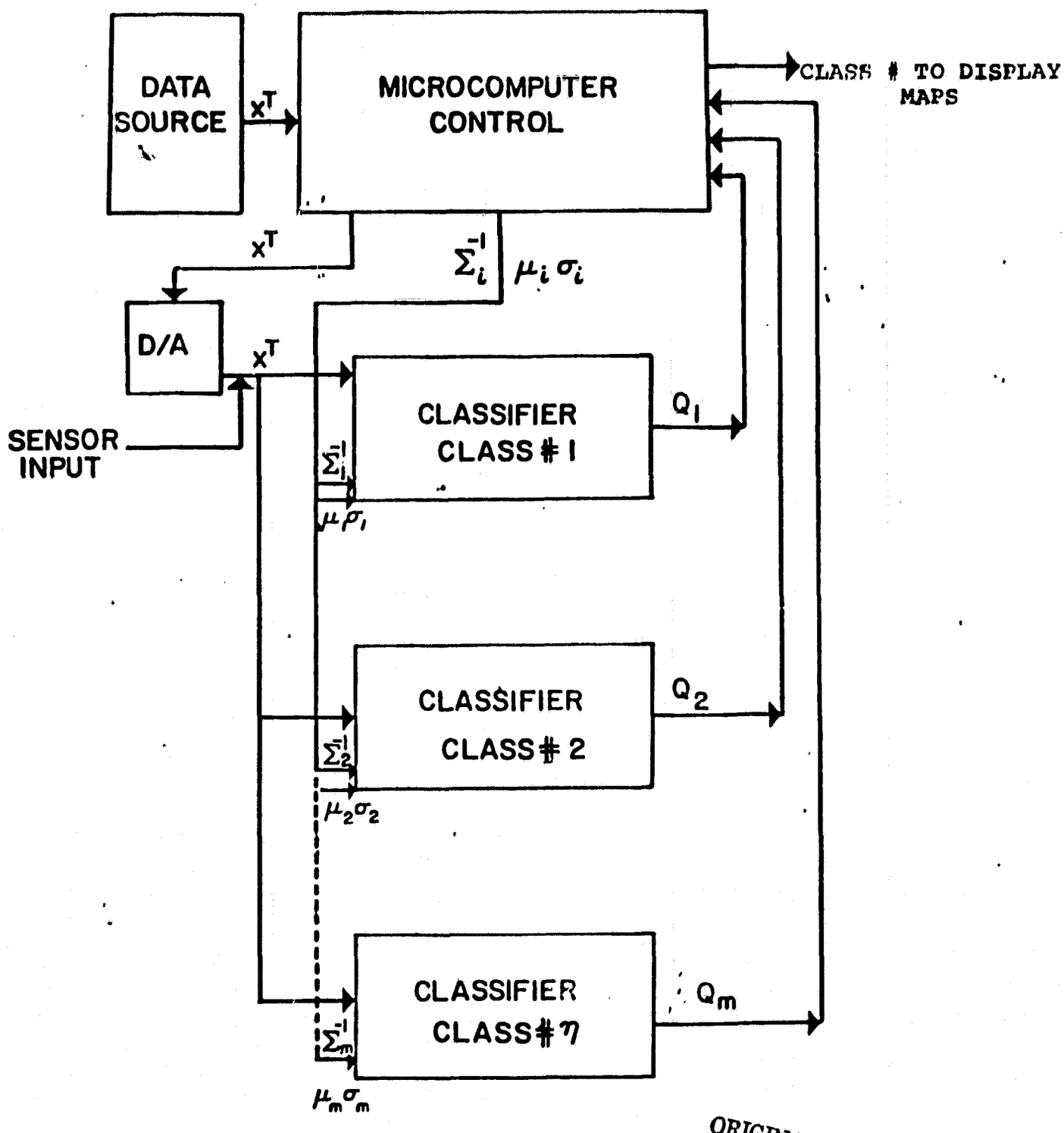
The Reticon R5402/5403 chips are revised versions of the AAC-32 which do not exhibit many of the earlier chip's problems. The difference between the 5402/5403 chips is in the number of taps: the R5402 has 16 taps; the R5403 has 32. A major change in both chips from the AAC-32 is the presence of a string of storage capacitors on one side of the device. The input signal on that side passes down the string of capacitors and is passed in parallel to the bucket brigade upon a sample and hold strobe signal.

The tests performed on the AAC-32 were repeated on the R5402/5403, and the results were encouraging. The chips exhibited acceptable accuracy (~7-8 bits) and repeatability and did not have the linearity problems of the AAC-32. The conclusion of the evaluation of the R5402/5403 was that the chips' performance was marginally satisfactory and that the design of a classifier around these devices should proceed.

5 CLASSIFIER SYSTEM DEVELOPMENT

5.1 Organization

Figures 5.1 and 5.2 illustrate the functional blocks and data flow of the classifier system. Figure 5.1 shows the set-up of a general classifier system while Figure 5.2 shows the set-up of the prototype system designed around the Reticon devices.



ORIGINAL PAGE IS
OF POOR QUALITY

Figure 5.1 Block Diagram of
General Classification System

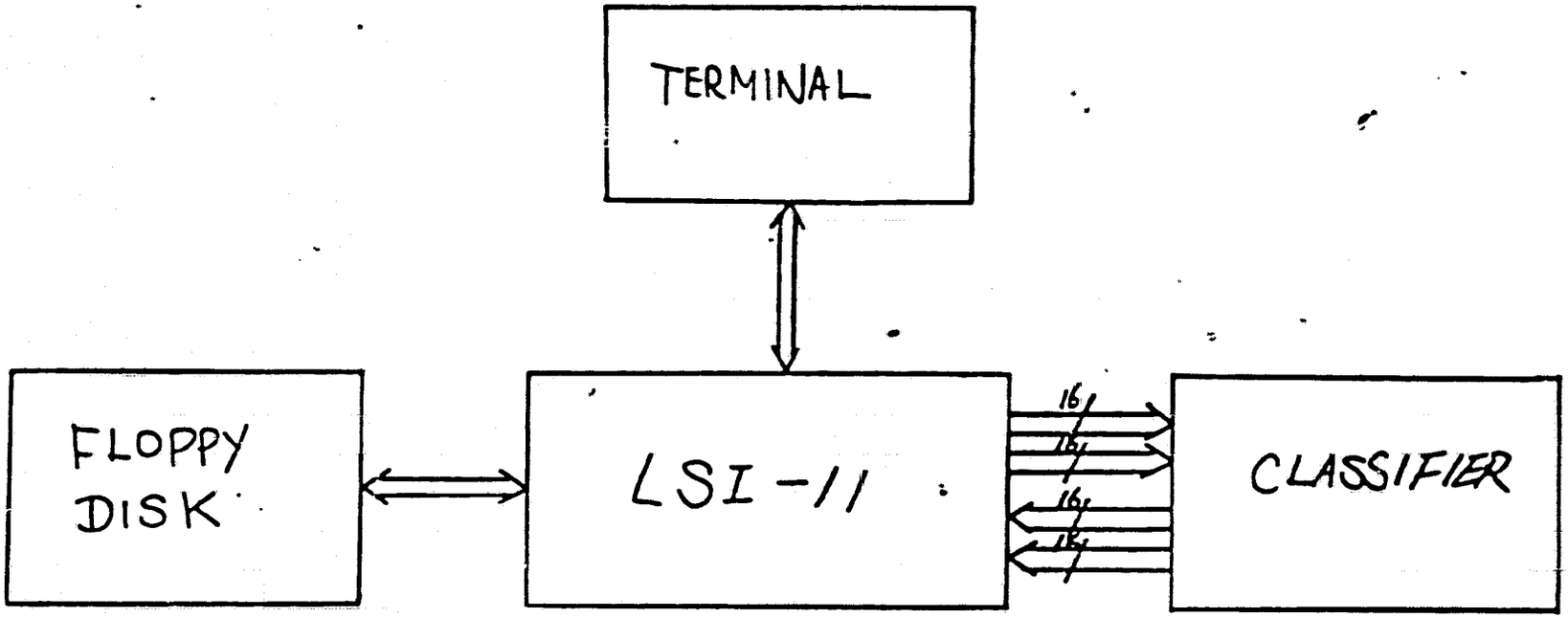


Figure 5.2 Block Diagram of Prototype System

In Figure 5.1 the sensor input could be directly from a LANDSAT sensor after optical and geometric corrections have been made, and the data source could be RAM built into the system for storing the necessary mean and covariance information. In the prototype system, however, the LSI-11 is the controlling microcomputer while the floppy disk serves as the sensor input by storing a LANDSAT image. All of the classifier hardware: CTD's, signal conditioning and conversion circuitry, RAM, and controlling circuitry is contained in one block, and communication with the LSI-11 is over two parallel data buses (16 bits in each direction per interface). The terminal is a 512x256 point color graphics display and serves as an output for the classified images. A more detailed description of the system is given in the 1979 report.

5.2 Hardware

5.2.1 Classifier Module Organization

This section contains a description of the classifier modules and their operation, These modules are: analog boards, timing and control/interface, and analog-to-digital conversion. Figure 5.3 indicates how these modules are integrated to form the classifier system. The modules all plug into a backplane over which the digital TTL level signals and power pass. The analog signals are not sent over the backplane over which the digital TTL level signals and power pass. The analog signals are not sent over the backplane to reduce potential noise problems; instead, they are passed from board to board by shielded cables. Except fo the A/D converter data output, a single board provides the required interface with the LSI-11.

5.2.2 Analog Board

The major functional component of the classifier is the analog or sum-of-products board. Eight of these boards form the core of the classifier; each contains a CTD and squaring circuit; the necessary D/A conversion, signal conditioning, and bias circuitry, and the RAM which is loaded with the appropriate column of the covariance matrix inverse. Upon receipt of the proper data and control signals each of these boards will cycle the CTD producing the row-column vector dot product and squaring the result. An analog multiplexer is included for diagnostic monitoring of critical on-board signal points.

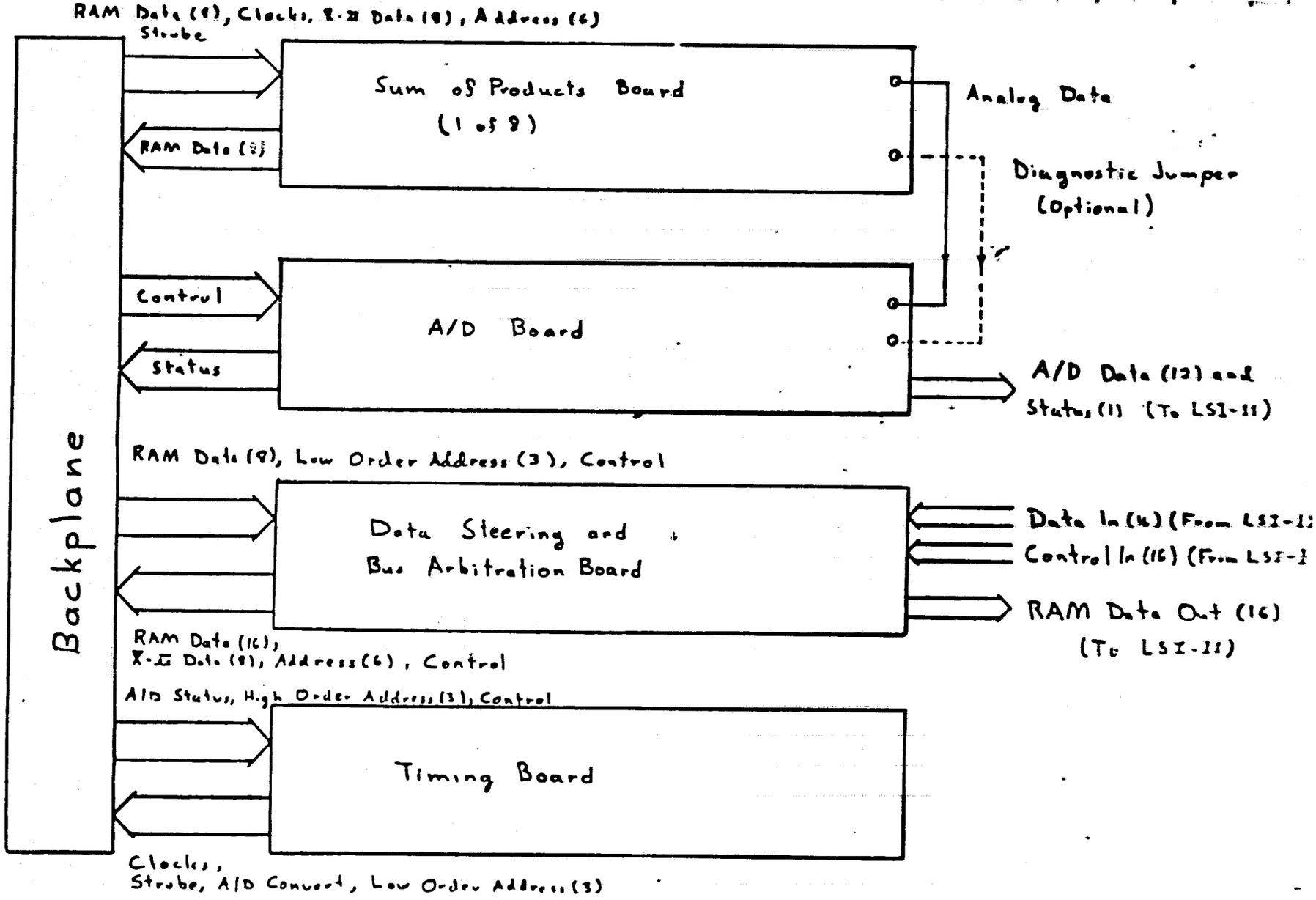


Figure 5.3 Classifier Module Organization and Data Flow

ORIGINAL PAGE IS
OF POOR QUALITY

Appendix A contains the schematic of the analog board along with a diagram of the printed circuit card layout for the board and a parts list.

5.2.3 Timing and Interface Modules

The functions of the timing and control module and the interface module are closely intertwined; for that reason, they will be discussed together in this section. Clocking, control, data steering, and interfacing with the LSI-11 are accomplished by these modules. Ten circuit groups make up the two modules:

(1) Transition Detector (TD)
(2) Counter Clock Reset/Enabler (CDREC)
(3) Counter and Decoder Circuit (CDC)
(4) Clock Generator Circuit (CGC)
(5) Address Bus Arbitrator (ABA)
(6) Mode Decode (MD)
(7) Subtractor (SUB)
(8) A/D Conversion Decoder (ADCD)
(9) Master Clock Generator (MCG)
(10) Chip Enable Decoder (CED)

The instructions arrive from the LSI-11 in the form of a 16-bit control word (see Figure 5.4). There are three basic operating modes for the classifier: LOAD, DIAGNOSE, and RUN. In LOAD mode the LSI-11 has control of the address bus and is loading new information for the classifier to process. In the DIAGNOSE mode the LSI-11 is driving different parts of the classifier to test their performance. In both of these modes the LSI-11 controls the backplane bus, and the CTD output data is invalid.

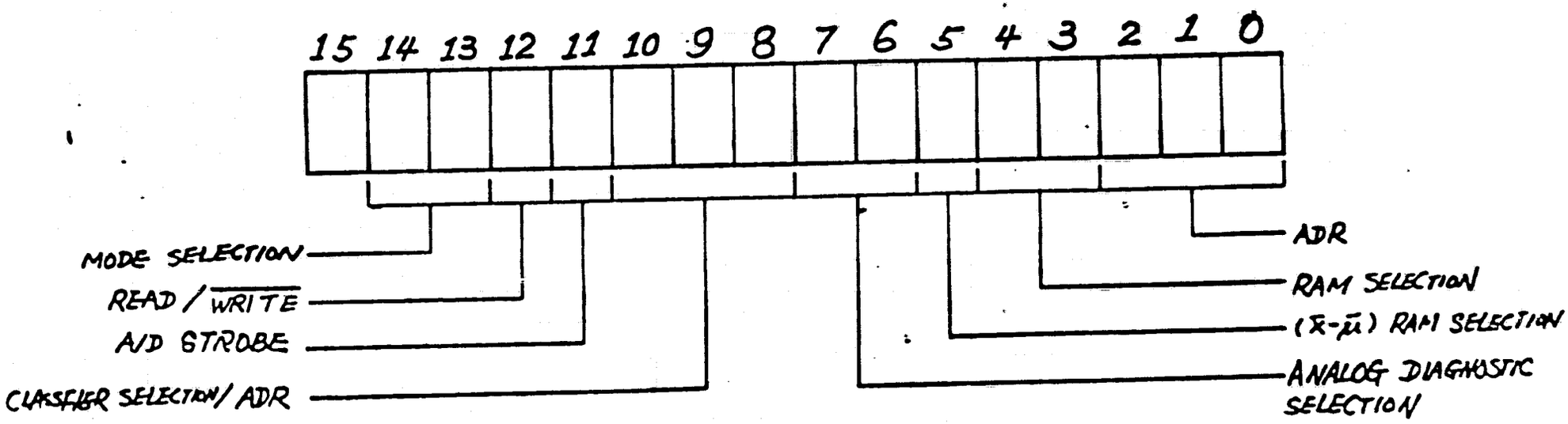


Figure 5.4 Control Word Format

In the RUN mode classifier operations proceed normally with the backplane bus controlled by the timing circuitry. These modes are selected by the mode selection bits (13-14) of the control word which are decoded by the Mode Decoder.

The Address Bus Arbitrator is a three-line multiplexer that has, as inputs, the three low order address bits of the control work and the three low order bits from the CDC, and has the three low order bits of the backplane address bus as outputs. When the classifier is in LOAD or DIAGNOSE mode the Arbitrator allows the LSI-11 to control these three bits, otherwise the timing circuitry exercises control.

The RAM selection bits (3,4,5 of the control word) go to the Chip Enable Decoder. This circuit has 6 output lines; the RAMs on the analog boards are enabled, two at a time, by four of these, the x-RAM is enabled by the fifth, and the μ-RAM is enabled by the last. During RUN mode this circuit is bypassed and all RAMs are enabled.

The Data Output Bus Enable selects the pair of RAM output bus drivers to be enabled. This selection is necessary to avoid having two RAMs attempting to control the backplane bus. This circuit is located on each of the analog boards.

The Subtractor is simply an ALU set to continually perform the necessary subtraction to obtain the $\bar{x}$-$\bar{\mu}$ term. The inputs come from the x-RAM which is loaded once per pixel and the μ-RAM which is loaded once per image.

The Transition Detector generates a single pulse to initiate a new cycle of the CTDs to produce the discriminate function results. This pulse is generated when the classifier is in the RUN mode and a change is detected on the three high order address bits.

The Counter Clock Reset/Enabler generates the signals to clock the CTD. When the CDC receives a Counter Enable signal from the CDREC it will begin the loading of the CTD. The Clock Enable signal is sent to the CGC to enable the CTD clocks.

In addition to the Counter Enable signal, the CDC receives the Master Clock from the MCG. The data is loaded into the CTD twice (assuming an eight-feature vector); this method should yield better accuracy than loading once and padding with zeros. During loading four outputs are produced. Three of these are low-order addresses and go to the address arbitrator. The other output is the Count Finished signal issued when the loading cycle is complete.

In addition to the Clock Enable and Count Finished signals the CGC receives the Master Clock signal and the three decoded mode signals. The clocks are always enabled during the LOAD and DIAGNOSE modes, and during RUN mode the clocks are started when the Count Enable signal is true. When the Count Finished signal is received one more clock cycle is applied to the A side of the CTD to correctly align the data because of the "dead" cell. The clocks are now halted, and the CTD is strobed to pass the data from the capacitors to the bucket brigades. The Load Ready strobe is now sent to the A/D Conversion Decoder.

The A/D Conversion Decoder generates the convert command that activates the A/D converter. In Run mode this command is controlled by the Load Ready Strobe, and in the other modes it is under the control of the LSI-11. The signal "A/D status" is generated by the A/D converter upon completion of its operation. This signal is used to restart the CTD clocks to avoid drifts and to signal the LSI-11 to read the A/D.

5.2.4 A/D Board

The A/D converter is the same 12 bit, 30μsec Analog Devices module used in the chip evaluations and operates in the same fashion.

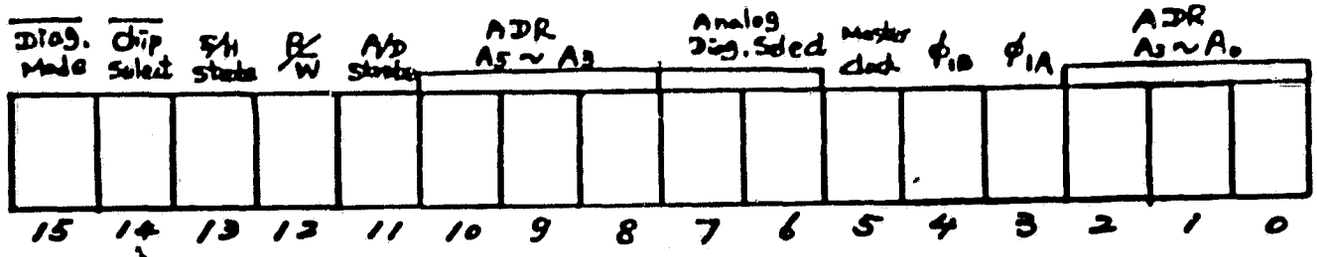
In addition to the A/D converter the board contains the summers for the eight signals from the analog boards. Switching is also provided between the summer output and a diagnostic signal jack used for monitoring various points on the analog boards during testing.

5.3 Testing Software

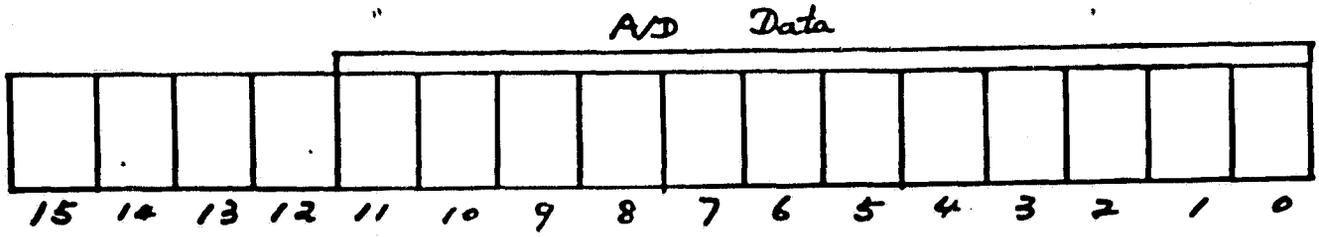
Special arrangements were necessary for testing of the analog boards. A testing stand was constructed to allow the board to be exercised without the full classifier. A subset of the control word signals which was necessary for the operation of the analog board was used. Clocks, control, and data which are normally provided internally were added. The revised communication word formats are shown in Figure 5.5. The purposes of the system were to check out the RAM on the board and to adjust the signal levels and offsets of the CTD inputs.

The adjustment algorithms are similar to those used for the device evaluation. These algorithms are discussed in the previous reports. We are including two new RAM testing programs here; Appendix D contains program listings.

FIRST DRV-11

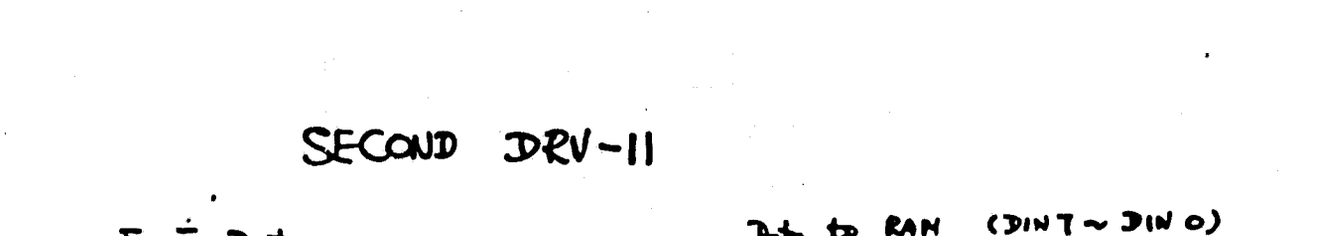


OUTPUT BUFFER
ADR = 167772_8

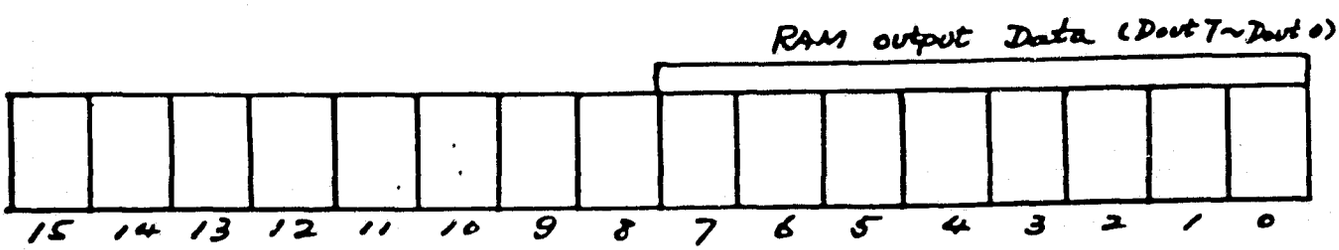


INPUT BUFFER
ADR = 167774_8

SECOND DRV-11



OUTPUT BUFFER
ADR = 167762_8



INPUT BUFFER
ADR = 167764_8

Figure 5.5 Revised Communication Word Formats for Testing

1. Program RAMTSB

RAMTSB checks the RAM in a bit-by-bit fashion. It loads the RAM with 1 at the first bit of the first location then checks the output. It then loads the RAM with 0 and checks the output. If the outputs are not consistant with the input data an error is logged. The testing sequence is from the first bit of the first location to the first bit of the last location (63). The program then checks the second and remaining bits in the same manner.

2. Program RAMTST

This program is an abbreviated version of RAMTSB which tests the RAM in a byte-by-byte fashion. It will not find all possible errors, but it may serve as a quick first test of the RAM operation. The program operates by loading each location with all 1's and all 0's and examining the result.

6 CONCLUDING REMARKS

The principal goal of this project has been the evaluation of Charge Transfer Devices and their potential use in pattern classifiers. As such, much of the work has been centered around testing the devices rather than construction of a final system. Since the devices were experimental and the application was new the work often involved a "ground-up" approach. In the case of the AAC-32, for example, very little information was available regarding optimum operating points, etc. In addition, some traditional types of tests were found to produce misleading results, for

example, the correlation of two sampled analog sine waves appeared to produce very good performance i.e., a high signal-to-noise ratio. It was not until the testing methodology was rethought and a different form of test was applied that the problems with the device became evident.

Three general conclusions may be drawn from the work:

(1) With the advent of "variable tap weight" devices there is a strong potential for the use of CTDs in pattern classifiers. They can be used to provide matrix multiplication subsystems. Suitable architectures were developed in this project, and their potential performance is good. The time required to produce an answer has been reduced to one CTD loading time. Further, such a classifier architecture allows the parallel computation of all of the discriminate functions (i.e. up to eight in this case) at once. Such a system of very fast, low power classifiers could be of tremendous benefit in processing data from sources such as LANDSAT satellites, particularly by making on-board classification feasible. This could make the use of pattern classification techniques and satellite data much more widespread, and could open the door to new uses of the techniques which are not possible now because of prohibitive computational requirements and their resulting delays or high cost.

(2) The evaluation of the Reticon AAC-32 clearly showed that it is not suitable for use in a classifier application. The failure did not, however, rule out the use of CTDs in classifiers because it appeared to result from a design problem in the analog multipliers rather than the CTD technology. It is interesting to note that if the failing quadrant was avoided, the AAC-32 did show usefulness in other applications such as programmable transversal filters. (See Appendix C).

The Reticon R5402/5403 indicated the validity of the conclusions since it did not have the problems of the AAC-32. The performance of these new devices indicated that they were candidates for matrix multiplication applications.

(3) A design for a micro-computer controlled classifier using the R5402/5403 chips was accomplished, and the paper system has a reasonable size, complexity, and power consumption. The further devleopment of the system for testing and evaluation purposes is recommended.

This project has by no means concluded that a CTD based classifier will operate well, only that such a system appears to be possible. The only was to accurately guage the performance of a CTD based classifier is to build and test it as is recommended. Several questions remain which cannot be answered simply by testing devices. These questions include: the stability of the system, particularly the analog interface and signal conditioning circuitry; the effects of long term drifts and aging on the accuracy of the total system when added together; the speed at which the total system may be operated, and the reliability of the devices when operated in such a system. The CTD technology is developing rapidly, and performance characteristics can certainly be expected to improve in the future. This project and other novel applications of the devices will certainly aid in indicating areas of technology development which need improvement and would increase the chances of the production of future devices which could operate even better.

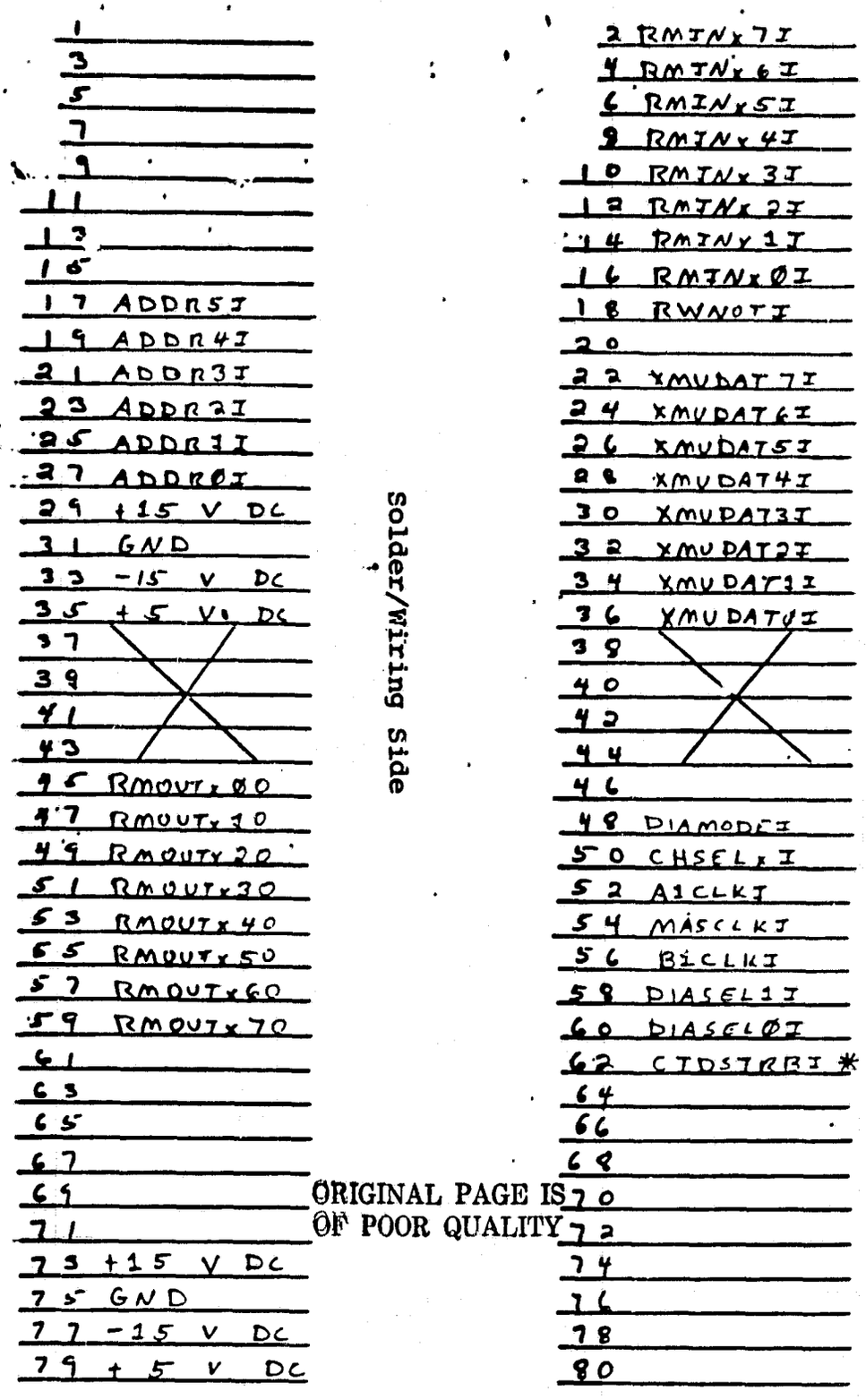
APPENDIX A

ANALOG BOARD SCHEMATIC AND PARTS LIST

The schematic is oversized and has been sent to NASA under separate cover.

Analog Board Pin Definitions

Edge View from Pins



ORIGINAL PAGE IS
OF POOR QUALITY

All signals TTL compatible unless otherwise specified.

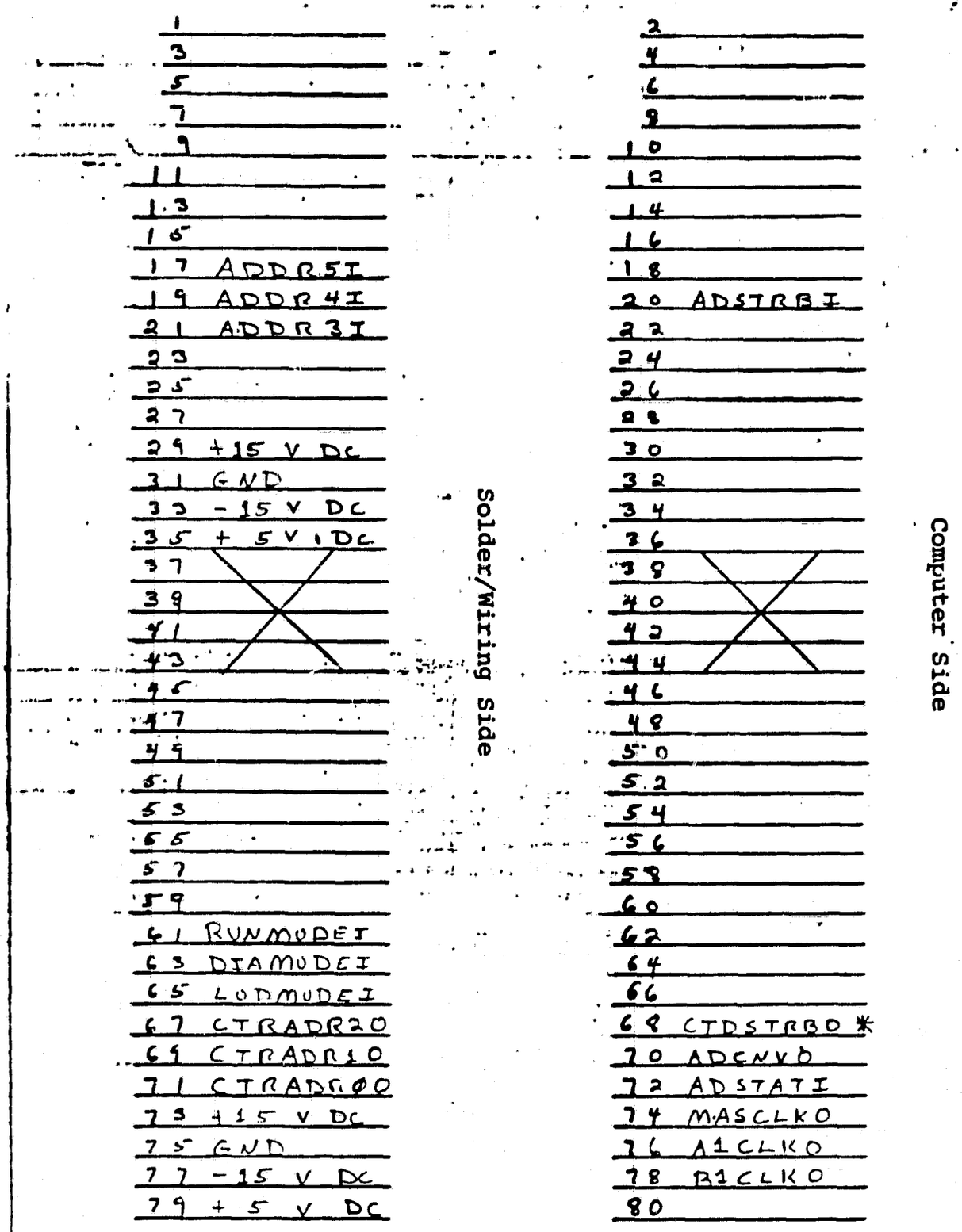
All unnamed pins reserved for future expansion.

x Depends upon slot

* This signal is 0-+15V

Timing Board Pin Definitions

Edge View from Parts



All signals TTL compatible unless otherwise specified.

All unnamed pins reserved for future expansion.

*This signal is 0-+15V

Interface and Data Steering Board Pin Definitions

Edge View From Pins

1	RMINL70	2	RMINR70
3	RMINL60	4	RMINR60
5	RMINL50	6	RMINR50
7	RMINL40	8	RMINR40
9	RMINL30	10	RMINR30
11	RMINL20	12	RMINR20
13	RMINL10	14	RMINR10
15	RMINL00	16	RMINR00
17	ADDR50	18	RWNRTO
19	ADDR40	20	ADSTRB0
21	ADDR30	22	XMUDAT70
23	ADDR20	24	XMUDAT60
25	ADDR10	26	XMUDAT50
27	ADDR00	28	XMUDAT40
29	+15 V DC	30	XMUDAT30
31	GND	32	XMUDAT20
33	-15 V DC	34	XMUDAT10
35	+5 V DC	36	XMUDAT00
37		38	
39		40	
41		42	
43		44	
45	RMOUTR0I	46	RMOUTL0I
47	RMOUTR1I	48	RMOUTL1I
49	RMOUTR2I	50	RMOUTL2I
51	RMOUTR3I	52	RMOUTL3I
53	RMOUTR4I	54	RMOUTL4I
55	RMOUTR5I	56	RMOUTL5I
57	RMOUTR6I	58	RMOUTL6I
59	RMOUTR7I	60	RMOUTL7I
61	RUNMODE0	62	CHSEL30
63	DIAMODE0	64	CHSEL20
65	LOPMODE0	66	CHSEL10
67	CTRADR2I	68	CHSEL00
69	CTRADR1I	70	DIASCL10
71	CTRADR0I	72	DIASEL0C
73	+15 V DC	74	MASCLKI
75	GND	76	
77	-15 V DC	78	
79	+5 V DC	80	

Left →

← Right

Solder/Wiring Side

Component Side

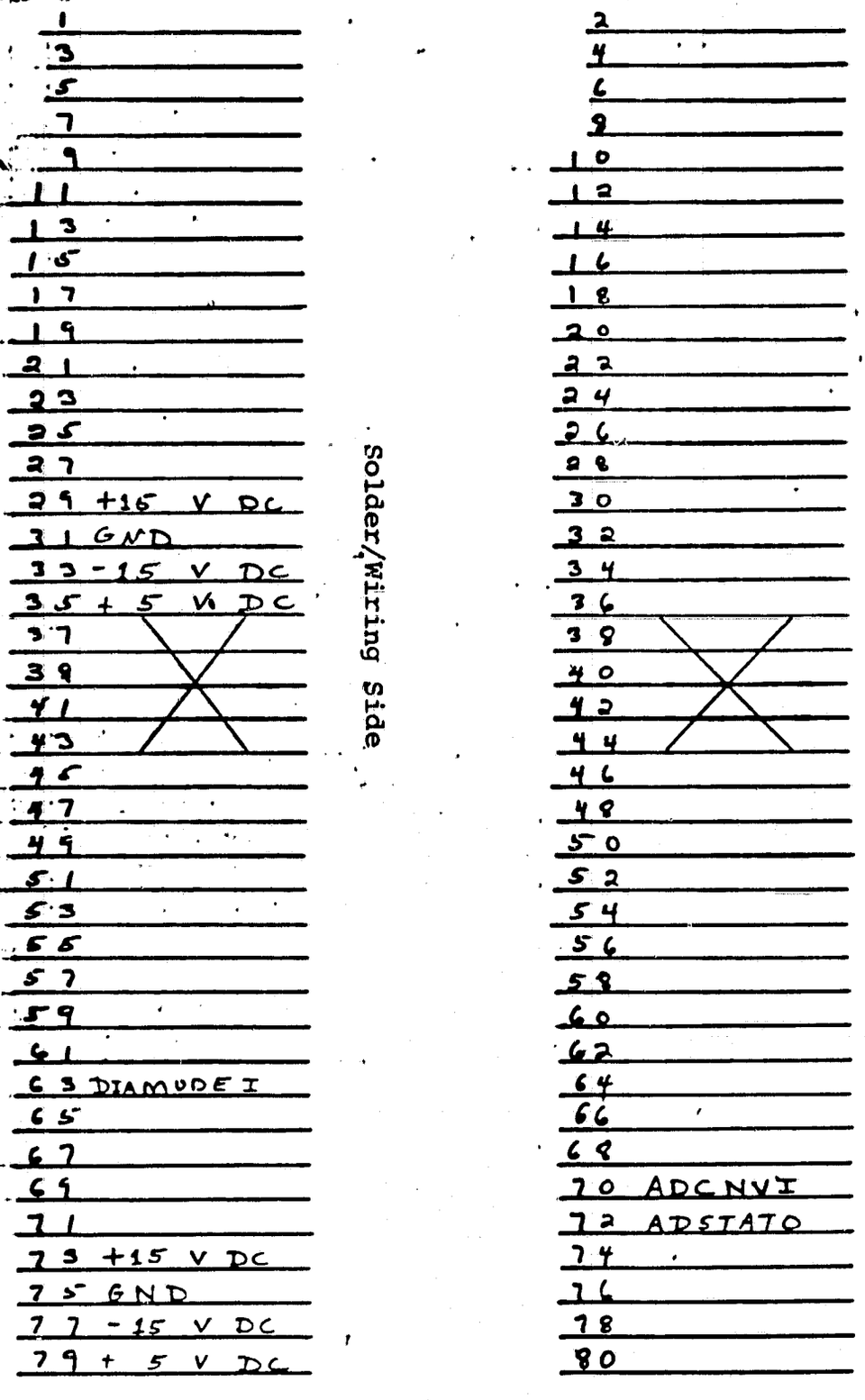
All signals TTL compatible unless otherwise specified.

All unnamed pins reserved for future expansion.

A/D Board Pin Definitions

ORIGINAL PAGE IS
OF POOR QUALITY

Edge View from Pins



All signals TTL compatible unless otherwise specified.

All unnamed pins reserved for future expansion.

LAYOUT OF THE ANALOG BOARD

The board used is a double sided PCB, with 72 possible gold finger contact evenly distributed in both sides, the space between contacts is .125 in., the base material used is FR4, the overall size of the board is 9.1 x 5.2 in.

PARTS LIST

RESISTANCES

The following values are given in Kohms and are assumed to be of 5% tolerance and 1/4 W maximum dissipation, unless otherwise especified.

location	value(Kohms)	location	value(Kohms)
R1	1.	R41	150
R2	10.	R42	62.
R3	20.	R43	62.
R4	1.	R44	100.
R5	10.	R45	15.
R6	20.	R46	2.
R7	10.	R47	68.
R8	1.	R48	300.
R9	1.0	R49	33.
R10	10.	R50	100.
R11	20.	R51	100.
R12	1.	R52	1.
R13	20.	R53	100.
R14	20.	R54	1.
R15	2.	R55	10.
R16	10.	R56	5.1
R17	20.	R57	1.
R18	2.	R58	10.
R19	10.	R59	5.1
R20	20.	R60	1.
R21	10.	R61	.62
R22	4.7	R62	.62
R23	20.	R63	.62
R24	10.	R64	.62
R25	4.7	R65	.62
R26	10.	R66	.62
		R67	.62
R28	20.	R68	.62
R29	1.	R69	.62
R30	56.	R70	2.
R31	15.	R71	2.
R32	100. ohms	72	2.
R33	20.	R73	2.
R34	270.	R74	5.1ohms
R35	100.	R75	5.1ohms
R36	62.	R76	.47
R37	160.	R77	.47
R38	120	R78	.47
R39	150.	R79	.47
R40	100.	R80	20.

CAPACITORS

The following capacitors are given in uF with 15 Vdc ratings and tolerances better to 10% unless otherwise specified.

location	value	location	value
C1	15pf	C28	0.1
C2	0.1	C29	0.1
C3	0.1	C30	0.1
C4	15pf	C31	0.1
C5	0.1	C32	0.1
C6	0.1	C33	0.1
C7	0.1	C34	0.1
C8	15pf	C35	0.1
C9	0.1	C36	0.1
C10	0.1	C37	0.1
C11	15pf	C38	0.1
C12	0.1	C39	100pf
C13	0.1	C40	0.1
C14	15pf	C41	0.1
C15	0.1	C42	0.1
C16	0.1	C43	100pf
C17	15pf	C44	0.01
C18	0.1	C45	0.01
C19	0.1	C46	0.01
C20	15pf	C47	0.01
C21	0.1	C48	0.1
C22	0.1	C49	0.1
C23	15pf	C50	0.1
C24	0.1	C51	0.1
C25	0.1	C52	0.1
C26	15pf	C53	0.1
C27	0.1		

DIODES

The selected diodes must supply 0.7 forward biased voltage.

location	model
D1	GE 914
D2	GE 914
D3	GE 914
D4	GE 914

INTEGRATED CIRCUITS

location	model
IC1	AD509JH
IC2	AD509JH
IC3	AD509JH
IC4	AD509JH
IC5	AD509JH
IC6	AD509JH
IC7	AD509JH
IC8	AD509JH
IC9	AD509JH
IC10	RC4200NB
IC11	DG509CJ
IC12	R5402 E 146
IC13	MC1408L8
IC14	MC1408L8
IC15	N82S09N

IC16	74LS240ND
IC17	SN7432J
IC18	SN7474N
IC19	DS0026CN
IC20	DS0026CN

JUMPERS

J1	takes the multiplexer output to the BNC connector
J2	takes the output of the squaring circuit to the BNC connecto
J3	joins both paths of ground

ORIGINAL PAGE IS
OF POOR QUALITY

APPENDIX B

PUBLICATIONS

BOOKS

Snyder, Benz, and Reece, "Pattern Classification Using Charge Transfer Devices", in Remote Sensing of Earth from Space: Role of "Smart Sensors", AIAA Progress in Aeronautics and Astronautics Series, vol. 67.

CONFERENCES

Snyder, Reece, and Benz, "Multispectral Classification Using Charge Transfer Devices", AIAA Smart Sensors Conference, Langley Research Center, 1978.

Snyder, Husson, and Benz, "Satellite Pattern Classification Using Charge Transfer Devices", IEEE Conference on Pattern Recognition and Image Processing, Chicago, 1979.

Snyder, Reece, and Benz, "Pattern Classification Using CTDs", Government Microcircuits Applications Conference, Monterrey, 1978.

Snyder, Rajala, and Hirzinger, "Image Modeling" The Continuity Assumption and Tracking", Submitted to the 5th International Conference on Pattern Recognition, Miami Beach, December 1980.

Snyder and Tang, "Optimal Computation of Image Gradients Using Eigenvector Techniques and 3x3 Neighborhoods, Submitted to the 5th International Conference on Pattern Recognition, Miami Beach, December 1980.

CONFERENCES (con't.)

Snyder and Cowart, "An Interative Approach to Region Growing," Submitted to the 5th International Conference on Pattern Recognition, Miami Beach, December 1980.

Snyder and Hirzinger, "Techniques for Processing Time-Varying Images", to be presented at the International Computer Technology Conference, San Francisco, August 1980.

PATTERN CLASSIFICATION USING CHARGE TRANSFER DEVICES

W. E. Snyder* and J. H. Reece†
North Carolina State University, Raleigh, N. C.

and

H. F. Benz‡
NASA Langley Research Center, Hampton Va.

Abstract

The potential uses of charge transfer devices (CTD's) in pattern classification operations are explored. The needs for a hardware-based pattern classifier are established, and a matrix multiplication subsystem based upon a sum of products CTD is presented. An evaluation process for sum-of-products devices (particularly analog-analog correlators) is developed, and the feasibility of employing a particular device in a pattern classifier is determined. Finally, the possible impact of future trends in technology is considered.

I. Introduction

Recent technological innovations are making general purpose computers cheaper and more accessible. Witness, for example, the dramatic increase in computational complexity available per dollar in just the last five years. These same technological innovations are making instrumentation packages simpler to use, more computationally dense, and much less expensive. It thus is becoming more and more reasonable to talk about special purpose, dedicated pattern recognition equipment.

We can discuss only a small subset of the "pattern recognition problem" in this context, since that larger problem is far from well defined, much less solved, and "special purpose equipment" implies that we are trading away flexibility in exchange for speed and/or simplicity of use. We have chosen to deal with the problem of multispectral satellite image classification. Under certain assumptions, this problem can be considered well defined, and a pressing need exists for special equipment which can deal rapidly with the vast amounts of data coming from satellites every minute.

NASA has been and continues to be concerned about the fact that present (general purpose computer-based) techniques are too slow and too expensive to begin to deal with more than a tiny fraction of the LANDSAT data which are currently available. This paper is one of the results of an ongoing

study conducted by NASA to investigate technologies which might contribute to the solution of this data processing bottleneck. This paper discusses the use of metal-oxide-semiconductor (MOS) technology in the construction of special purpose equipment for pattern classification.

II. Device Background

Charge transfer devices may be divided into two classes: the "bucket brigade device" (BBD) and the "charge coupled device" (CCD) with various subclasses within the major groupings. The two types of CTD's differ in the manner in which charge is stored and transferred from cell to cell and have slightly differing performance characteristics. Most CCD's have lower noise figures and higher transfer efficiencies (the percentage of charge in the original cell which is transferred to the new cell) than BBD's.

When used as analog signal processors, charge transfer devices may be employed to obtain complex functions of the input waveform. One such function is $\sum_i a_i b_i$, where the b's are samples of the input waveform and the a's are weighting coefficients. This function is found in recursive filtering, correlation, convolution, and a number of other operations.

As shown in Fig. 1 and photographed in Fig. 2, the corresponding cells of two CTD delay lines can be connected to multipliers and the multiplier outputs then summed. In this case the result is again $\sum_i a_i b_i$, but the weighting coefficients are determined by the data stored in the second CTD and may be changed simply by clocking in new data.

Such variable tap weight devices may be used as sampled correlators of continuous analog signals. More important to this project, however, is the fact that, since they operate on discrete data samples, variable tap weight devices may be used to generate the vector dot product,

$$\bar{X} \cdot \bar{Y} = \sum_{i=1}^{n} x_i y_i$$

where n is the dimension of the vector and may be as large as the number of cells in the CTD. This product is obtained simply by loading one vector into one side of the CTD and the second vector into the other. The answer thus obtained is the result of one row by column operation of a matrix multiplication.

Unlike conventional filtering application, in which a useful new result is available each clock cycle, dot product operations with two arbitrary vectors require that the entire CTD be loaded before a useful answer is produced. Thus, for a typical 16 component vector, 16 clock cycles are required to load the device. However, 5-Mhz clock rates are quite reasonable, making it possible to perform 16 multiply and add operations in 3.2 μsec.

III. Pattern Classification Hardware

In the case of multispectral image classification, the input X is a vector resulting from measurements of light intensity in several different spectral ranges. Each vector X corresponds to a single point (pixel) in a scene. A typical LANDSAT scene consists of an array of over one million ordered pixels.

It has been shown[1] that, for a given class, all pixels belonging to that class may be reasonably described by a multivariate normal distribution. With this assumption, the probability that a vectbr X belongs to a class i is

$$P(i|X) = \frac{1}{(2\pi^{N/2})(|\Sigma_i|^{1/2})} \exp[-\tfrac{1}{2}(X-\mu_i)^T \Sigma_i^{-1}(X-\mu_i)]$$

where Σ_i and μ_i are the covariance matrix and mean vectors, respectively, which describe the statistics of class i.

Taking the logarithm of the probability gives a discriminant function

$$g_i(X) \triangleq \text{ln}[P(i|X)] = -1/2(X-\mu_i)^T \Sigma_i^{-1}(X-\mu_i) + \text{ln}|\Sigma_i|^{-1/2} + \text{ln}(2\pi^{-N/2})$$

Since the logarithm function is monotonic, the class having the largest discriminant function for a given measurement X will also be the class having the largest probability P(i|X) that X belongs to that class. $\text{ln}(2\pi^{-N/2})$ is a constant for all classes and therefore does not contribute to discriminating one class from another. Furthermore, the term $\text{ln}|\Sigma|^{-1/2}$ needs to be computed only once for each class. $(X-\mu_i)^T \Sigma_i^{-1}(X-\mu_i)$, however, must be computed for each of the

millions of measurements made in an image. Consequently, this matrix computation with a general purpose computer is very time consuming.

Figure 3 shows a block diagram for a hardware configuration of a system for classifying multispectral data. Data may come directly from a sensor array in analog form, or for purposes of testing, from a digital data source. The microcomputer is the control element for the system. In a training mode, it derives the statistics which describe the various classes. In classification mode, the microcomputer loads those statistics into an array of parallel CTD classifiers, controls steering of data input to those classifiers, and examines their outputs.

The individual classifiers are shown in Figs. 4 and 5. In Fig. 4, each of the row-column dot product operations is performed in a charge transfer device. The outputs are multiplexed together and fed to one more CTD for the post-multiplication dot product.

In contrast, Fig. 5 depicts a hardware simplification which also results in increased speed, since it eliminates the multiplexer and a delay. This simplification is made feasible by the diagonal symmetry of the covariance matrix, which may be transformed into upper triangular form.

The matrix operation shown in Fig. 5 can be decomposed into cellular substructures as shown in Figs. 6 and 7. At the conclusion of the training mode, the processor loads one row of the covariance matrix into the first-in-first-out memory associated with each cell. The hardware then takes over and under control of the clock generator, performs the entire discriminant function computation.

On a pixel-by-pixel basis, the output of the individual hardware classifiers is digitized and read by the processor, which then classifies the pixel as belonging to the class whose discriminant function was maximized. Using 128 CTD's, a pixel described by 16 multispectral measurements may be classified into one of nine categories in 3.2 μsec.

IV. In Situ Cell Qualification

The critical element in the pattern classifier system is the charge transfer device that performs the row-column multiply. Extensive testing has been performed on prototype units which have recently become available from semiconductor manufacturers.

The hardware test station shown in Fig. 8 has been implemented. It tests individual devices in a cellular

structure similar to the structures used in the classifier. Typical results of one of the tests on one of the devices are shown in Fig. 9. The figure shows the aggregate linearity and offset for a Reticon 5402, a 16-element analog-analog sum of products device. Plotted is the computed output in expanded scale and inverted decimal form) versus the B input values, where all of the B cells are filled with the fractional numbers indicated. Similarly, the A cells are filled with the numbers indicated, and these point values are connected to form families. It is apparent from this figure that there is a small offset, the point where the curves all intersect. There is also a small nonlinearity, where the points fall off the curve. The rotated appearance of the curve is caused by the small offset in the B side being multiplied and summed by the data in the A side. These test results demonstrate adequate linearity and offset at the 8 equivalent bit input and 8 equivalent bit output to continue further development. Figure 10 is a photograph of the test system showing the chip under test, the microcomputer and the associated data conversion circuitry.

V. Conclusion

This paper has shown one method of implementing dedicated hardware for pattern classification. Recent technological developments have made such classifiers feasible using sampled analog processing. Test results have indicated that prototype development should continue. It is expected that continued technological improvement will lead to more compact, lower power, and even faster system configurations.

Acknowledgements

This work was supported by NASA Research Grant NSG 1353.

References

[1]Kriegler, F. T., et al., "Midas, Prototype Multivariate Interactive Digital Analysis System - Phase I," Vol. 1, NASA CR-132463, ERIM 195800-25-F, Aug. 1974, p. 72.

Presented as Paper 78-1723 at the AIAA/NASA Conference on "Smart" Sensors, Hampton, VA., Nov. 14-16, 1978. This paper is declared a work of the U.S. Government and therefore is in the public domain.

*Assistant Professor of Electrical Engineering, North Carolina State University.

+Graduate Student, North Carolina State University.

≠Aerospace Technologist, NASA Langley Research Center.

ORIGINAL PAGE IS
OF POOR QUALITY

SATELLITE PATTERN CLASSIFICATION USING CHARGE TRANSFER DEVICES

W.E. Snyder

North Carolina State University
Raleigh, North Carolina

C. Husson and H.F. Benz

NASA Langley Research Center
Hampton, Virginia

Abstract

The potential uses of Charge Transfer Devices (CTDs) in pattern classification operations are explored. The needs for a hardware-based pattern classifier are established, and a matrix multiplication subsystem based upon a sum-of-products CTD is presented. Applications of the subsystem to the classification of multi-modal Gaussian distributions in general and to LANDSAT data processing in particular are discussed. Finally, the potential impact of this technology on satellite data processing methodologies is discussed.

Key words: Gaussian Classifier, Charge coupled device

1. Introduction

Recent technological innovations are making general purpose computers cheaper and more accessible. Witness, for example, the dramatic technological increase that has been made in just the last five years. These same technological innovations are making instrumentation packages simpler to use, more computationally dense, and much less expensive. It is thus becoming more and more reasonable to talk about special purpose, dedicated pattern recognition equipment.

We can discuss only a small subset of the "pattern recognition problem" in this context since that larger problem is far from well defined, much less solved, and "special purpose equipment" implies that we are trading away flexibility in exchange for speed and/or simplicity of use. We have chosen to deal with the problem of multi-spectral satellite image classification. Under limiting assumptions, this problem can be considered well defined and a pressing need exists for special equipment which can deal rapidly with the vast amounts of data coming from satellites every minute.

NASA has been and continues to be concerned about the fact that present (general purpose computer-based) techniques are too slow and too expensive to reduce to classified images more than a tiny fraction of the LANDSAT data which is currently available. This paper is one of the results of an on-going study conducted by NASA to investigate technologies which might contribute to the solution of this data processing bottleneck.

Of particular interest to NASA are technologies which may lead to flyable "on-board" processors, units on the satellite which can classify a multi-spectral image in real-time and transmit to the ground only the classified image. A unit must meet several requirements before it can be placed in such an application. First, it must be capable of dealing with the analog data directly as it comes from the sensors. Second, since the unit is to be placed on a satellite, it must consume little power, be light in weight, and be highly reliable. Finally, the unit must be programmable from the ground and capable of deriving its own classification parameters.

This paper discusses the use of Metal-Oxide-Semiconductor (MOS) technology in the construction of special purpose equipment for pattern classification. The computational function of individual sum-of-products chips is first described, then, a scheme for the organization of the chips into a pattern classifier is shown.

The potential of on-board classification has both possible gains and hazards associated with it. With the obvious benefits of timely data availability comes the potential of the unavailability of the raw data for further processing. This difficulty is discussed in Section 4.

2. Device Background

Charge Transfer Devices may be defined for the purposes of this paper as devices which move charge linearly in synchronism with a clock. If the charge is quantized in a binary manner, CTDs may be used as digital delay lines or as shift register memories. It is, however, the ability of CTDs to move analog data that has resulted in their widest application. They have been used to acquire analog video data and to process analog data from other sources. Special classifier hardware fits into this last category.

Charge transfer devices may be divided into two classes: the "bucket brigade device" (BBD) and the "charge coupled device" (CCD) with various subclasses within the major groupings. The two types of CTDs differ in the manner in which charge is stored and transferred from cell to cell and

CH1428-2/79/0000-0246$00.75 © 1979 IEEE

have slightly differing performance characteristics.

Most CCDs have lower noise figures and higher transfer efficiencies (the percentage of charge in the original cell which is transferred to the new cell) than BBDs.

When used as analog signal processors, charge transfer devices may be employed to obtain complex functions of the input waveform. One such function is $\Sigma a_i b_i$ where the 'b's are samples of the input waveform and the 'a's are weighting coefficients. This function is found in recursive filtering, correlation, convolution, and a number of other operations.

Buss[1] shows a simple means for implementing "fixed tap weight" devices where the tap weights are the weighting coefficients of the $\Sigma a_i b_i$ function and are fixed at the time of manufacture. The tap weights are realized by splitting the transfer electrodes of one clock phase of the CTD in the ratio $(1 + a_i)$: $(1 - a_i)$ where a_i is the ith desired coefficient.

In an alternate and potentially somewhat more useful approach, as shown in Figure 1 and photographed in Figure 2, the corresponding cells of two CTD delay lines can be connected to multipliers and the multiplier outputs then summed. In this case the result is again $\Sigma a_i b_i$, but the weighting coefficients are determined by the data stored in the second CTD and may be changed simply by clocking in new data.

Such variable tap weight devices may be used as sampled correlators of continuous analog signals. More important to this application, however, is the fact that since they operate on discrete data samples, variable tap weight devices may be used to generate the vector dot product, $X \cdot Y = \sum_{i=1}^{n} X_i Y_i$ where n is the dimension of the vector and may be as large as the number of cells in the CTD. This product is obtained simply by loading one vector into one side of the CTD and the second vector into the other. The answer thus obtained is the result of one row by column operation of a matrix multiplication.

Unlike conventional filtering applications, in which a useful new result is available each clock cycle, dot product operations with two arbitrary vectors require that the entire CTD be loaded before a useful answer is produced. Thus for a typical 16 component vector, 16 clock cycles are required to load the device. However, 5 Mhz clock rates are quite reasonable, making it possible to perform 16 multiply and add operations in 3.2 μs.

3. Pattern Classification Hardware

It has been shown[2] that for the purpose of LANDSAT data classification, all pixels belonging to a given class may be described (typically), by a multimodel multivariate distribution. The multimodel distribution may be adequately decomposed into an aggregate of Normal distributions. Classification then consists of determining which of several Normal distributions a particular pixel is most likely to belong to and assign the pixel to the class having that distribution. With this assumption, the probability that a vector X belongs to a class i is

$$P(w_i|X) = \frac{1}{(2\pi)^{N/2}(|C_i|^{1/2})} \mathrm{EXP}(-1/2(X-\mu_i)^T C_i^{-1}(X-\mu_i))$$

Where C_i and u_i are the covariance matrix and mean vectors respectively which describe the statistics of class i.

The usual definition of the Normal distribution includes a term representing the a-priori probability $P(w_i)$ that a sample X belongs to a particular class w_i. Experience has shown that very satisfactory results can be had by treating all a-priori probabilities as equal. If this is the case, then for the purposes of classification, the a-priori probabilities may be neglected.

Taking the logarithm of the probability gives a discriminant function

$$g_i(X) = \ln P(w_i|X) = -1/2(X-\mu_i)^T C_i^{-1}(X-\mu_i) + \ln|C_i|^{-1/2} + \ln(2\pi^{-N/2})$$

Since the logarithm function is monotonic, the class having the largest discriminant function for a given measurement X will also be the class having the largest probability $P(w_i|X)$ that X belongs to that class.

$\ln(2\pi^{-N/2})$ is a constant for all classes and therefore does not contribute to discriminating one class from another. Furthermore the term $\ln|C|^{-1/2}$ needs to be computed only once for each class.

$(X-\mu_i)T\ C_i^{-1}(X-\mu_i)$ however must be computed for each of the millions of measurements made in an image. Consequently, this matrix computation with a general purpose computer is very time consuming.

Figure 3 shows a block diagram for a hardware configuration of a system for classifying multispectral data.

Data may come directly from a sensor array in analog form or, for purposes of testing, from a

digital data source. The microcomputer is the control element for the system. In a training mode, it derives the statistics which describe the various classes. In classification mode, the microcomputer loads those statistics into an array of parallel CTD classifiers, controls steering of the data input to those classifiers, and examines their outputs.

The individual classifiers are shown in figures 4 and 5. In figure 4, each of the row-column dot product operations is performed in a charge transfer device. The outputs are multiplexed together and fed to one more CTD for the postmultiplication dot product.

In contrast, figure 5 depicts a hardware simplification which also results in increased speed since it eliminates the multiplexer and a delay. This simplification is made possible by the following argument:

By the constructed diagonal symmetry of the C matrix, the product $z = [X^T]\ [C^{-1}]\ [X]$ may be rewritten as

$z = [X^T]\,[A]^T\,[A]\,[X]$ where A is upper triangular then

$$z = [X^T A^T]\,[AX]$$
$$= [Y^T]\,[Y]$$
$$= Y^2 .$$

The matrix operation shown in figure 5 can be decomposed into cellular substructures as shown in figure 6. If 8 features are assumed, then 8 of the sum-of-products structures in figure 6 are needed -- one for each row column operation.

At the conclusion of the training mode, the processor loads of the covariance matrix associated with each class into the covariance memory associated with each cell. If 8 possible classes are assumed, this memory is 64 x 8 bits as shown. During this time, the X-μ vector for each class is loaded into the X-μ RAM also.

At this point, the hardware controller takes over and performs the discriminant function computation. The outputs of all sum-of-products cells are summed and passed through the analog to digital converter whose output is read by the processor as shown in figure 7. This operation is repeated for each class simply by stepping to the next group of covariance matrix rows and X-μ vectors in RAM. The processor then classifies the pixel as belonging to the class where discriminant function was maximized.

Operating the classifier in this fashion with longer memories holding all of the statistical information allows the use of only a single classifier without the need to reload the covariance information for each discriminant function calculation. In this manner a significant reduction in hardware over the use of a separate classifier for each class is realized with only a slight reduction in operating speed.

6. The Potential of On-Board Classification

We have shown in this paper an architecture which makes feasible the possibility of on-board classification. An on board classifier offers significant potential gains in performance of the satellite system; data would be available to the user in minutes rather than months.

There are significant logistical and technical problems which must be overcome before these benefits could become reality. In this section, we demonstrate only a few and their potential solutions.

A Scenerio

We will make this demonstration through a scenerio of how a typical classification might be performed:

(1) A county agricultural agent reserves the satellite for its next pass over. In so doing, he specifies the coordinates of some areas known to be corn, soybeans, and cotton.

(2) The coordinates of these training sets are transmitted to the satellite. As the satellite passes over, image data is acquired and stored. The on-board classifier performs a cluster analysis on the training sets and derives a Gaussian fit for each cluster.

In an alternative proposed system[3] the satellite clusters the entire scene, transmits the cluster statistics, and for each pixel, transmits the number of the cluster to which that pixel is assigned.

(3) Once appropriate statistics have been derived to describe training sets, the classifier described in section 3 is initialized by loading the statistics into the RAMS, and the data is then classified as belonging to one of the classes identified as corn, soybeans, or cotton. The results of the classification are then encoded and transmitted to the ground.

(4) The agriculture agent then can receive a false colored map of the area or a digital tape with the classification results.

It should be noted that this scenerio has passed over a significant amount of pre-processing which must be done to the sensor output prior to classification, including correcting for geometric distortion.

On board classification provides a tremendous potential benefit since it makes reasonable direct user interaction with the satellite; and provides data for the user in expeditious time. The one factor which some users may consider detremental in such a system is the fact that no longer does the ground user have the raw data to mull over at his leisure.

This factor does open up a new area of study, for in those instances when the user has both a computer and the time to study the image, he may

also to attempt to improve the classifier performance by doing post-classification processing to reduce errors. Since the raw data is missing, this might seem impossible, however, two pieces of data are available, the training sets, and the output of the on-board classifier. It is possible in this circumstance to improve classifier performance by using these data. We are currently studying this problem and will be publishing our results in the near future.

5. Conclusion

This paper has shown one method of implementing dedicated hardware for pattern classification. Recent technological developments have made such classifiers feasible using sampled analog processing.

Test results have indicated that prototype development should continue. It is expected that continued technological improvement will lead to more compact, lower power, and even faster system configurations.

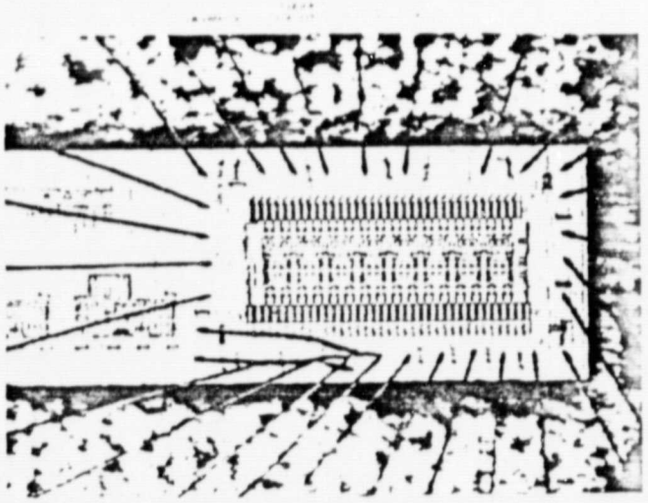
ORIGINAL PAGE IS
OF POOR QUALITY

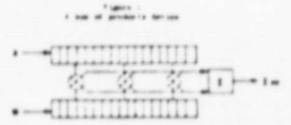
Bibliography

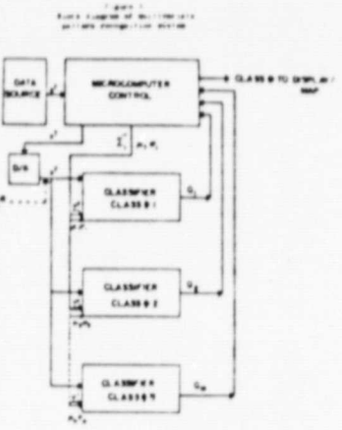
[1] Buss, D. IEEE Journal of Solid State Circuits, SC-8, p. 138 (1973).

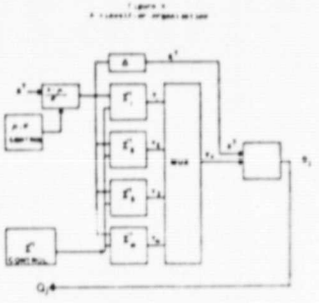
[2] Environmental Research Institute of Michigan, ERIM 10880-49-F NASA CR-2730, MIDAS, Prototype Multivariate Interactive Digital Analysis System for Large Area Earth Resources Surveys Vol. 1. Systems Description, Sept. 1976.

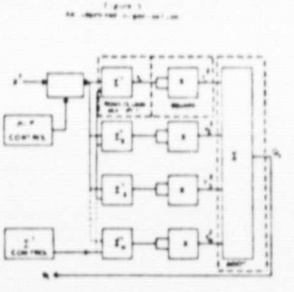
[3] Hilbert, Ed, "Cluster Compression Algorithm," JPL PR 77-43 Jet Propulsion Lab.

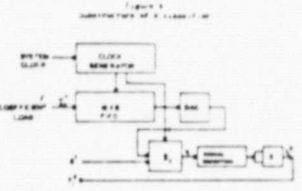












Pattern Classification Using Charge Transfer Devices*

W.E. Snyder and J.H. Reece
North Carolina State University,
Raleigh, NC 27650

Harry F. Benz
NASA-Langley Research Center,
Hampton, VA 23665

Abstract

The potential uses of Charge Transfer Devices (CTDs) in pattern classification operations are explored. The needs for a hardware-based pattern classifier are established, and a matrix multiplication subsystem based upon a sum of products CTD is presented. An evaluation process for sum of products devices (particularly analog-analog correlators) is developed, and the feasibility of employing a particular device in a pattern classifier is determined. Finally, the possible impact of future trends in technology is considered.

1. Introduction

Recent technological innovations are making general purpose computers cheaper and more accessible. Witness, for example, the dramatic increase in computational complexity available per dollar in just the last five years. These same technological innovations are making instrumentation packages simpler to use, more computationally dense, and much less expensive. It is thus becoming more and more reasonable to talk about special purpose, dedicated pattern recognition equipment.

We can discuss only a small subset of the "pattern recognition problem" in this context since that larger problem is far from well defined, much less solved, and "special purpose equipment" implies that we are trading away flexibility in exchange for speed and/or simplicity of use. We have chosen to deal with the problem of multispectral satellite image classification. Under certain assumptions, this problem can be considered well defined and a pressing need exists for special equipment which can deal rapidly with the vast amounts of data coming from satellites every minute.

NASA has been and continues to be concerned about the fact that present (general purpose computer-based) techniques are too slow and too expensive to begin to deal with more than a tiny fraction of the LANDSAT data which is currently available. This paper is one of the results of an ongoing study conducted by NASA to investigate technologies which might contribute to the solution of this data processing bottleneck.

This paper discusses the use of metal-oxide-semiconductor (MOS) technology in the construction of special purpose equipment for pattern classification.

*This work was supported by NASA Research Grant NSG 1353.

2. Device Background

Charge transfer devices may be divided into two classes: the "bucket brigade device" (BBD) and the "charge coupled device" (CCD) with various subclasses within the major groupings. The two types of CTDs differ in the manner in which charge is stored and transferred from cell to cell and have slightly differing performance characteristics.

Most CCDs have lower noise figures and higher transfer efficiencies (the percentage of charge in the original cell which is transferred to the new cell) than BBDs.

When used as analog signal processors, charge transfer devices may be employed to obtain complex functions of the input waveform. One such function $\Sigma a_i b_i$ where the 'b's are samples of the input waveform and the 'a's are weighting coefficients. This function is found in recursive filtering, correlation, convolution, and a number of other operations.

As shown in Figure 1 and photographed in Figure 2, the corresponding cells of two CTD delay lines can be connected to multipliers and the multiplier outputs then summed. In this case the result is again $\Sigma a_i b_i$, but the weighting coefficients are determined by the data stored in the second CTD and may be changed simply by clocking in new data.

Such variable tap weight devices may be used as sampled correlators of continuous analog signals. More important to this project, however, is the fact that since they operate on discrete data samples, variable tap weight devices may be used to generate the vector dot product,

$\bar{X} \cdot \bar{Y} = \sum_{i=1}^{n} x_i y_i$ where n is the dimension of the vector and may be as large as the number of cells in the CTD. This product is obtained simply by loading one vector into one side of the CTD and the second vector into the other. The answer thus obtained is the result of one row by column operation of a matrix multiplication.

Unlike conventional filtering application, in which a useful new result is available each clock cycle, dot product operations with two arbitrary vectors require that the entire CTD be loaded before a useful answer is produced. Thus for a typical 16 component vector, 16 clock cycles are required to load the device. However, 5 Mhz clock rates are quite reasonable, making it possible to perform 16 multiply and add operations in 3.2 μs.

3. Pattern Classification Hardware

In the case of multispectral image classification, the input, X, is a vector resulting from measurements of light intensity in several different spectral ranges. Each vector X corresponds to a single point (pixel) in a scene. A typical LANDSAT scene consists of an array of over one million ordered pixels.

It has been shown [1] that for a given class, all pixels belonging to that class may be reasonably described by a multivariate normal distribution. With this assumption, the probability that a vector X belongs to a class i is

$$P(i/X) = \frac{1}{(2\pi^{N/2})(|\Sigma_i|^{1/2})} \text{EXP}(-1/2(X-\mu_i)^T \Sigma_i^{-1}(X-\mu_i))$$

Where Σ_i and μ_i are the covariance matrix and mean vectors respectively which describe the statistics of class i.

Taking the logarithm of the probability gives a discriminant function

$$g_i(X) \triangleq \text{ln}(P(i/X)) = -1/2(X-\mu_i)^T \Sigma_i^{-1}(X-\mu_i) + \text{ln}|\Sigma_i|^{1/2} + \text{ln}(2\pi^{-N/2})$$

Since the logarithm function is monotonic, the class having the largest discriminant function for a given measurement X will also be the class having the largest probability P(i/X) that X belongs to that class.

$\text{ln}(2\pi^{-N/2})$ is a constant for all classes and therefore does not contribute to discriminating one class from another. Furthermore, the term $\text{ln}|\Sigma|^{1/2}$ needs to be computed only once for each class.

$(X-\mu_i)^T \Sigma_i^{-1}(X-\mu_i)$ however must be computed for each of the millions of measurements made in an image. Consequently, this matrix computation with a general purpose computer is very time consuming.

Figure 3 shows a block diagram for a hardware configuration of a system for classifying multispectral data.

Data may come directly from a sensor array in analog form or, for purposes of testing, from a digital data source. The microcomputer is the control element for the system. In a training mode, it derives the statistics which describe the various classes. In classification mode, the microcomputer loads those statistics into an array of parallel CTD classifiers, controls steering of data input to those classifiers, and examines their outputs.

The individual classifiers are shown in figures 4 and 5. In figure 4, each of the row - column dot product operations is performed in a charge transfer device. The outputs are multiplexed together and fed to one more CTD for the postmultiplication dot product.

In contrast, figure 5 depicts a hardware simplification which also results in increased speed since it eliminates the multiplexer and a delay. This simplification is made feasible by the diagonal symmetry of the covariance matrix which may be transformed into upper triangular form.

The matrix operation shown in figure 5 can be decomposed into cellular, substructures as shown in figures 6 and 7. At the conclusion of the training mode, the processor loads one row of the covariance matrix into the first-in-first-out memory associated with each cell. The hardware then takes over and under control of the clock generator, performs the entire discriminant function computation.

On a pixel by pixel basis, the output of the individual hardware classifiers are digitized and read by the processor, which then classifies the pixel as belonging to the class whose discriminant function was maximized. Using 128 CTDs, a pixel described by 16 multispectral measurements may be classified into one of nine catagories in 3.2 μs.

4. In Situ Cell Qualification

The critical element in the pattern classifier system is the charge transfer device which performs the row - column multiply. Extensive testing has been performed on prototype units which have recently become available from semiconductor manufacturers.

The hardware test station shown in figure 8 has been implemented. It tests individual devices in a cellular structure similar to the structures used in the classifier.

Typical results of some of these tests are shown in figure 9. Shown are results of testing three different devices, the Reticon AAC-32, a 32 element unit, the 5403, a modified AAC-32, and the 5402, a 16 element unit. The figure shows output numerical computations in families for varying input numbers.

In going from the AAC-32 to the 5403, Reticon significantly improved the numerical range of calculations.

5. Conclusion

This paper has shown one method of implementing dedicated hardware for pattern classification. Recent technological developments have made such classifiers feasible using sampled analog processing.

Test results have indicated that prototype development should continue. It is expected that continued technological improvement will lead to more compact, lower power, and even faster system configurations.

References

1. Kriegler, F. T., et al. Midas, Prototype Multivariate Interactive Digital Analysis System — Phase I, Volume I, NASA CR-132463 ERIM 195800-25-F, August 1974. p. 72.

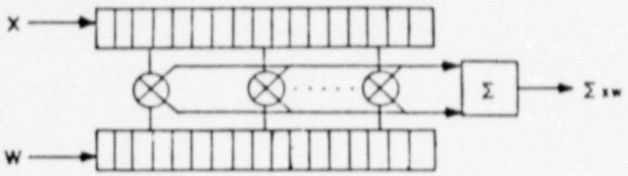


Figure 1. Sum of Products Device.

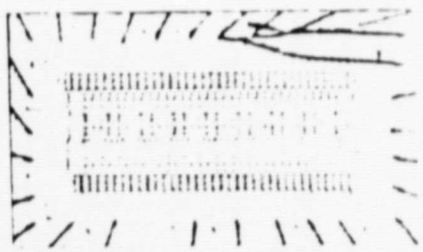


Figure 2. Photograph of Ketron K 5402.

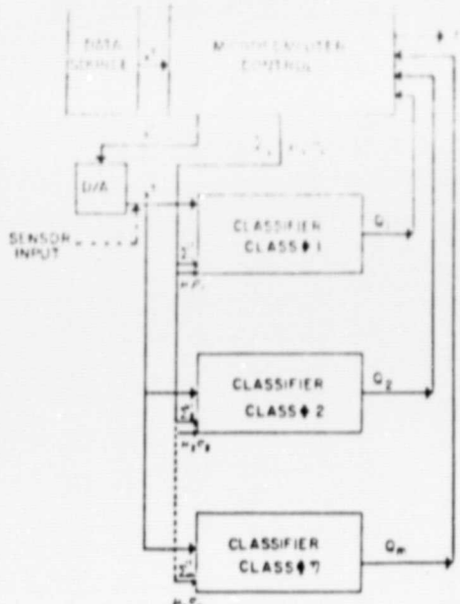


Figure 3. Block Diagram of Gaussian Multivariate Pattern Recognition System.

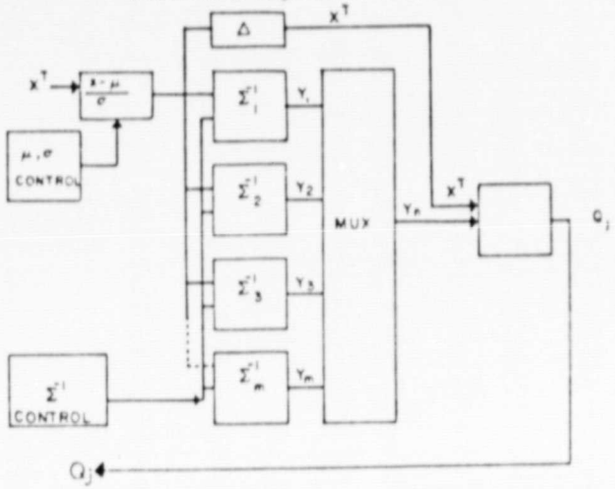


Figure 4. Classifier Functional Block Diagram.

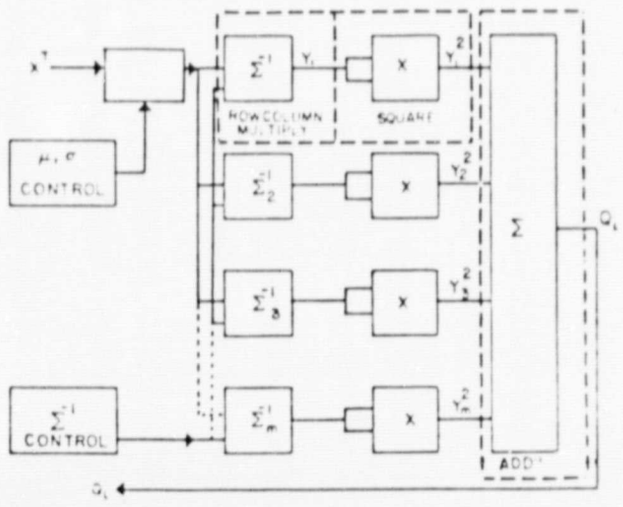


Figure 5. Alternative Form of Classifier Functional Block Diagram.

ORIGINAL PAGE IS
OF POOR QUALITY

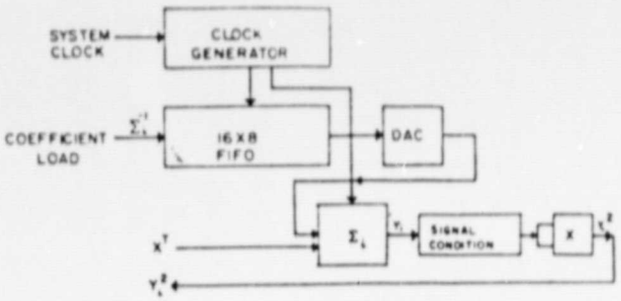


Figure 6. Block Diagram of Hardware Cell Configuration for Row-Column Multiply and Square Functions (One per Color and Feature).

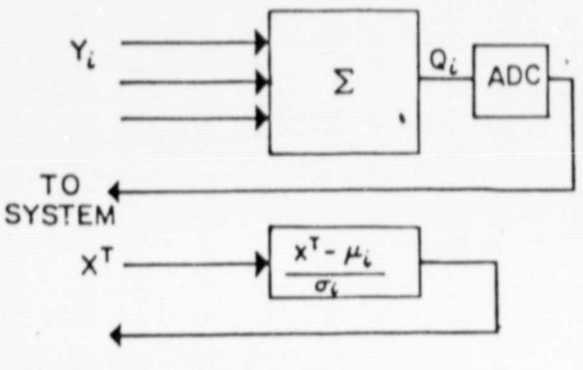


Figure 7. Block Diagram for Data Scaling, Summing and A/D Conversion for Subsequent Digital Comparison and Storage. (One per Feature)

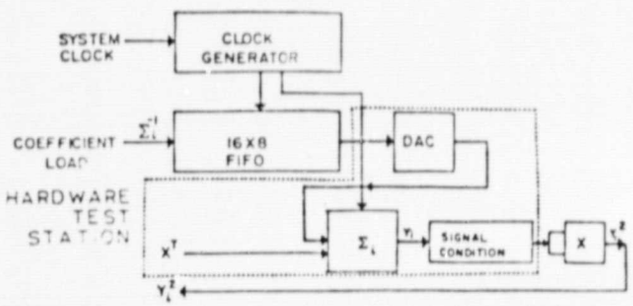
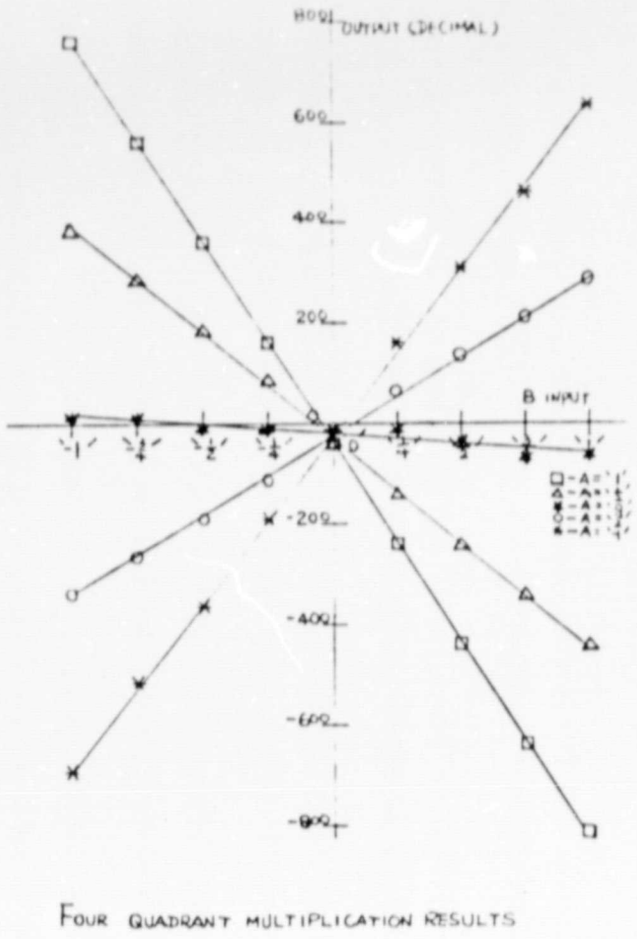


Figure 8. In Situ Testing of Devices.



FOUR QUADRANT MULTIPLICATION RESULTS

Figure 9. Representative Test Data.

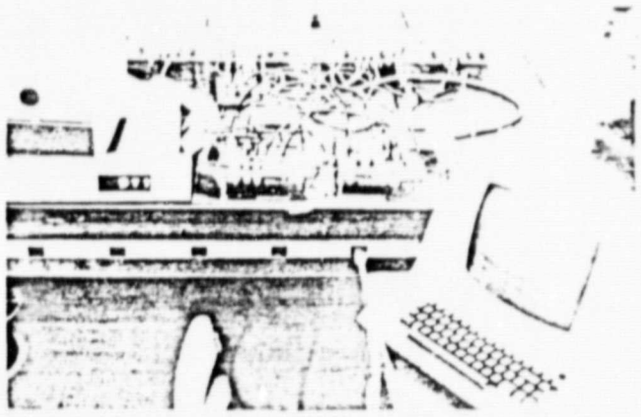


Figure 10. Photograph of Test Station.

Submitted to the 5th International Conference on Pattern Recognition, Miami Beach, December, 1980

IMAGE MODELING

The

Continuity Assumption and Tracking

by

Wesley E. Snyder

and

Sarah A. Rajala

Image Analysis Laboratory
Department of Electrical Engineering
North Carolina State University

and

Gerhard Hirzinger
Deutsche Forschungs-und Versuchsanstalt für
Luft-und Raumfahrt e.V.

This work was sponsored by the National Aeornautics and Space Agency under Research Grant NSG1535, and by the Deutsche Forschungs- und Versuchsanstalt für Luft-und Raumfahrt e.V.

ABSTRACT

This paper addresses a particular application, that of tracking moving objects with a television camera, (1, 2, 5, ...) and deals in this context with the interactions between motion, image gradients, and image representations. A representation for moving images in terms of Taylor's series expansions is developed, and application of this expression is shown to suffer from the assumption that images are continuously differentiable. A simple numerical example is given, and the velocity-segmentation work of Fennema and Thompson is explained in this context.

It is concluded that motion information results only from the motion of edges and that not understanding this principal can lead to problems.

Correlation trackers, and in particular, a tracker developed by Fitts', are shown to, in fact, correlate the motion only of edges.

Finally, in the appendix, it is shown how Fitts' tracker can be derived from first order Taylor's series assumptions.

Key words: Tracking, gradients, motion.

1. INTRODUCTION

Traditionally, the intensity function representing the brightness at a point in an image is considered to be a two dimensional continuous function, $I(x,y)$. Image operators derived using this model are then discretized by setting Δx and Δy to one in the finite increment model.

We will extend this model to consider the possibility of motion within a scene by allowing the intensity function to vary in x,y, and t. The images will be modeled in the context of digitized television images, therefore the t will be discretized to the frame-to-frame time interval.

We will examine a particular model for the image, the Taylor series expansion of the intensity function, and it will be shown how this expansion leads to a mechanism for segmentation and velocity detection. The capability of segmentation results from the concept that all the pixels having the same velocity will belong to the same region. Some work in this area has been done by Fennema and Thompson [3], and is extended here. A second method of determining pixel velocity will then be considered, a correlation tracker. This method was introduced by Fitts [4]. The correlation tracker assumes that at least some segmentation has already been done and, in fact, this method estimates the velocity of a group of pixels. We will then compare the velocity estimation accuracies of these two methods and come to some rather predictable conclusions that better tracking can be done if some segmentation has already been done. (See [6, 9, 11, 13] for related work).

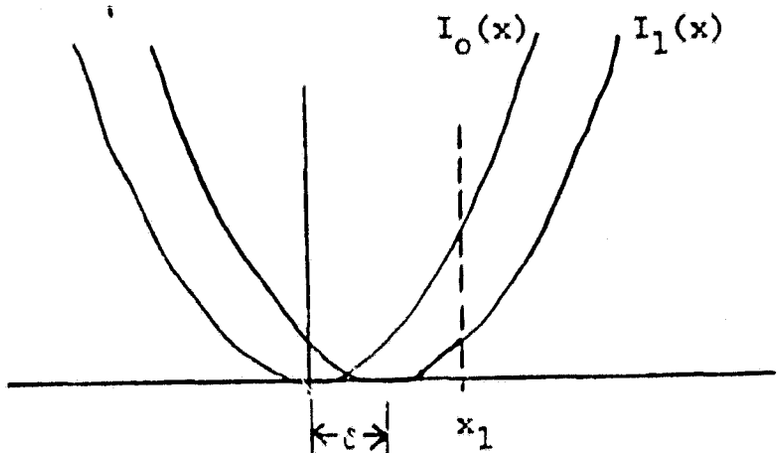
The fundamental weakness of both of these approaches will then be noted. We will generalize this to a philosophy for dealing with digitized images.

2. THE TAYLOR SERIES MODEL FOR TIME-VARYING IMAGES

For purposes of clarity, we will at first describe the intensity function I as a function of a single variable, x, and consider the case of shifting I(x).

I(x) thus, represents in one-dimension the grey scale values of the object to be tracked. It will be the same function in two sequential frames, but will be displaced in x.

Consider the following example, $I(x) = x^2$. This is shown in Figure 1.



At some point on the x axis, x_1, we measure $I_o(x_1)$ from the first frame, and later, at this same point x_1 we measure $I_1(x_1)$ from the second frame. But the two functions are the same except for a shift in the axis, so

$$I_o(x-\delta) = I_1(x). \qquad (1)$$

ORIGINAL PAGE IS
OF POOR QUALITY

Expanding $I_o(x - \delta)$ about x_1, we get

$$I_o(x_1-\delta) = I_o(x_1) - \delta I_o'(x_1) + \frac{\delta^2}{2} I_o''(x_1) - \ldots \qquad (2)$$

and from the equality (1) we find that

$$I_1(x_1) = I_o(x_1-\delta) = I_o(x_1) - I_o'(x_1)\delta + I_o''(x_1)\frac{\delta^2}{2} - \ldots .$$

and finally

$$I_1(x_1) = I_o(x_1) - \frac{\partial I_o}{\partial x}(x_1)\,\delta + \frac{\partial^2 I_o}{\partial x^2}(x_1)\frac{\delta^2}{2} - \ldots \qquad (3)$$

$I_o(x_1)$ is the value of the intensity function measured at point x_1 in the first frame, and likewise, $I_1(x_1)$ is measured in the second frame. Thus, if we can compute or accurately estimate the gradients in the first frame, we can solve the above equation for the displacement δ.

Now, let us consider the extension of this concept to two dimensions. Assume we have a function $I_o(x,y)$ and a second function $I_1(x,y)$ which results from shifting $I_o(x,y)$ through δ_x and δ_y. That is,

$$I_1(x,y) = I_o(x-\delta_x, y-\delta_y) . \qquad (4)$$

Expanding $I_o(x-\delta_x, y-\delta_y)$ at a point (x_1,y_1), we get

$$I_o(x_1-\delta_x, y_1-\delta_y) = I_o(x_1,y_1) - \left\{\frac{\partial I_o}{\partial x}(x_1,y_1)\,\delta_x + \frac{\partial I_o}{\partial y}(x_1,y_1)\,\delta_y\right\}$$

$$+ \tfrac{1}{2}\left\{\frac{\partial^2 I_o(x_1,y_1)}{\partial x^2}\delta^2_x + \frac{2\partial^2 I_o(x_1,y_1)}{\partial x \partial y}\delta_x\delta_y + \frac{\partial^2 I_o(x_1,y_1)}{\partial y^2}\delta^2_y\right\} - \ldots \qquad (5)$$

However, we know from (4) that this is equal to $I_1(x_1,y_1)$ which is measured in the second frame.

Now, let us consider the following notational simplifications and define:

$$G_x = \frac{\partial I_o(x_1,y_1)}{\partial x}, \qquad G_y = \frac{\partial I_o(x_1,y_1)}{\partial y},$$

$$G_{xx} = \frac{\partial^2 I_o(x_1,y_1)}{\partial x^2}, \quad G_{yy} = \frac{\partial^2 I_o(x_1,y_1)}{\partial y^2}, \quad G_{xy} = \frac{\partial^2 I_o(x_1,y_1)}{\partial x \partial y},$$

and $\Delta I = I_1(x_1,y_1) - I_o(x_1,y_1)$, and this reduces to

$$\Delta I = \frac{G_{xx}}{2} \delta_x^2 + \frac{G_{yy}}{2} \delta_y^2 + G_{xy}\delta_x \delta_y - G_x \delta_x - G_y \delta_y \qquad (6)$$

Thus, a single pixel (x,y) substituted into (6) will yield a quadratic equation in two unknowns, δ_x and δ_y. However, all pixels having the same velocity will have characteristic equations whose solutions pass through the same point.

In practice, due to noise, one finds that the solutions pass through nearly the same point, so that a clustering technique may be used to find the best approximation to that point of common intersection, thus yielding an algorithm for finding frame-to-frame displacemen

2.2 Several Numerical Examples

Let us consider the ability of this representation to determine the displacement of a simple signal. Assume displacement only occurs in the x direction so we may use the simpler one dimensional form

$$\Delta I = \tfrac{1}{2} G_{xx}\delta_x^2 - G_x\delta_x$$

or

$$\delta_x = \frac{G_x \pm \sqrt{G_x^2 + 2G_{xx}\Delta I}}{G_{xx}}$$

or, in the case $G_{xx} = 0$, we use

$$\delta_x = \frac{-\Delta I}{G_x}$$

First, we apply the technique to $I_o = x^2$ between $-6 \leq x \leq 6$

x	-6	-5	-4	-3	-2	-1	0	1	2	3	4	5	6
I_o	36	25	16	9	4	1	0	1	4	9	16	25	36
I_1	64	49	36	25	16	9	4	1	0	1	4	9	16
G_x	-12	-10	-8	-6	-4	-2	0	2	4	6	8	10	12
G_{xx}	2	2	2	2	2	2	2	2	2	2	2	2	2
ΔI	28	24	20	16	12	8	4	0	-4	-8	-12	-16	-20
δ_x	2	2	2	2	2	2	2	2	2	2	2	2	2

On this signal, the algorithm works exactly, Now consider it on a pulse.

x	-6	-5	-4	-3	-2	-1	0	1	2	3	4	5	6
I_o	0	0	0	2	4	4	4	2	0	0	0	0	0
I_1	0	0	0	0	0	2	4	4	4	2	0	0	0
G_x	0	0	1.18	2	1.18	0	-1.18	-2	-1.18	0	0	0	0
G_{xx}	0	.643	1	0	-1	-1.18	-1	0	1	.643	0	0	0
ΔI	0	0	0	-2	-4	-2	0	2	4	2	0	0	0
δ_x	0	0	2.35	1.00	-4.24	-1.84	2.35	1	1.18	.8	0	0	0
or					+1.88	+1.84			-4.24	-.8			

After experimenting with this algorithm with several different signals, we observed the following:

1) For signals continuous over the domain, $\hat{\delta}$ is computed exactly. For example,

$$S = x^2$$

$$y = (x-\delta_x)^2$$

Computing this function analytically for $s = x^2$ yields an exact result. That is $\hat{\delta} = \delta_x$ for any δ_x. We need to point out that S(x) is continuous in all derivatives and defined over the tire interval from $-\alpha$ to α for any α.

2) A test of the predictor was made numerically, again using $S = x^2$, over the interval -6 to 6. With $\Delta x = 1$, the following numerical values.

x	-6	-5	-4	-3	-2	-1	0	1	2	3	4	5	6
s(x)	36	25	16	9	4	1	0	1	4	9	16	25	36
y(x)	64	49	36	25	16	9	4	1	0	1	4	9	16
$\frac{\partial s}{\partial x}$	-12	-10	-8	-6	-4	-2	0	2	4	6	8	10	12
W	12	10	8	6	4	2	0	-2	-4	-6	-8	-10	-12
y-s	28	24	20	16	12	8	4	0	-4	-8	-12	-16	-20
W(y-s)	336	240	160	96	48	16	0	0	16	48	96	160	240
$\Sigma W(y-s) =$	1456												
$\left[\frac{\partial s}{\partial x}\right]^2$	144	100	64	36	16	4	0	4	16	36	64	100	144
$\Sigma \left[\frac{\partial s}{\partial x}\right]^2 = 728$													

$$\hat{\delta} = \frac{\Sigma w(y-s)}{\Sigma \left[\frac{\partial s}{\partial x}\right]^2} = \frac{1456}{728} = 2$$

This rather remarkable result, an exact answer, can be attributed to the fact that x^2 is a nicely behaved function. Let us try it with some other functions which are not quite so nice.

x	-6	-5	-4	-3	-2	-1	0	1	2	3	4	5	6
s	0	0	0	2	4	4	4	2	0	0	0	0	0
y	0	0	0	0	0	2	4	4	4	2	0	0	0
$\frac{\partial s}{\partial x}$	0	0	1.18	2	1.18	0	-1.18	-2	-1.18	0	0	0	0
W	0	0	-1.18	-2	-1.18	0	-1.18	2	1.18	0	0	0	0
y-s	0	0	0	-2	-4	-2	0	2	4	2	0	0	0
W(y-s)	0	0	0	4	4.72	0	0	4.72	4	0	0	0	0
$\Sigma(w)(y-s)$	= 17.44												
$\left[\frac{\partial s}{\partial x}\right]^2$	0	0	1.93	4	1.93	0	1.93	4	1.93	0	0	0	0
$\Sigma\left[\frac{\partial s}{\partial x}\right]^2$	15.72												
$\hat{\delta}$	$\frac{16}{13}$ = 1.1 a significant error from the correct value of 2												

Suppose the function s(x) is a longer duration pulse, but with the same edge values. In this case, we observe that if the object is homogeneous in the interior, the W term will be zero at those points and contribute nothing toward the sum.

Consider one more experiment, assume the signal is a step edge

x	-6	-5	-4	-3	-2	-1	0	1	2	3	4	5
s	0	0	0	0	4	4	4	4	4	0	0	0
y	0	0	0	0	0	0	4	4	4	4	4	0
$\frac{\partial s}{\partial x}$	0	0	0	2.56	2.56	0	0	0	-2.56	-2.56	0	0
W	0	0	0	-2.56	-2.56	0	0	0	2.56	2.56	0	0
y-s	0	0	0	0	-4	-4	0	0	0	4	4	0
W(y-s)	0	0	0	0	10.24	0	0	0	0	10.24	0	0
$\Sigma w(y-s) =$	20.48											
$\left[\frac{\partial s}{\partial x}\right]^2$	0	0	0	6.55	6.55	0	0	0	6.55	6.55	0	0
$\Sigma\left[\frac{\partial s}{\partial x}\right]^2 =$	26.2											
$\hat{\delta}$	.78 An even larger error											

We can now understand intuitively the operation of Fitt's predictor. When an object moves from one frame to the next, a signal is produced in the difference image y-s. For objects which are homogeneous in their interiors, no information is contained in those homogeneous regions, only at the edges. Thus, it is not the motion of an object which must be tracked, but the motion of the edges of that object.

With this, we see that

$$\int w(y-s)\, de = \int \left(\frac{-\partial s}{\partial e}\right) (y(e) - s(e))\, de$$

is in fact nothing more than the correlation of the edges of an object due to intensity changes with the edges due to motion.

Thus we can conclude:

1) Tracking is a gradient-based operation. Only the motion of edges is significant.
2) The motion of the object from one frame to the next must be

small, not small with respect to the size of the object, but small with respect to the distance over which meaningful information can be gleaned from the gradient. That is, motion must be less than the edge width.

These conditions result from the antithesis of the continuity assumption for images. Images are in fact NOT continuous functions of space and intensity, but do contain step discontinuities, and the fewer number of grey levels in the digitization process, the worse the continuity assumption becomes.

We now see more clearly why it was necessary for Fennema and Thompson [3] to blur their images, for this makes the continuity assumption closer to valid, but induces increased uncertainty in object location.

These two conditions above must hold if tracking is to be done utilizing only pixel grey value information. However, if some image preprocessing has been done, it may be possible to permit larger frame to frame displacements. One possibility would be to track some feature of the object which is more likely to be continuous than its intensity. A typical such measurement might be the vertical cross section.

The alternative is to admit that digital images are not continuous and that classical signal processing techniques simply are not appropriate, particularly if binary images are being tracked. We can then deal with the data with that thought in mind.

4. CONCLUSIONS

In general, we feel that segmentation is necessary for good tracking and conversely, motion information can be a good cue to direct a segmentation operator to the appropriate area. Thus, a feedback structure is necessary, containing both the elements of

segmentation and tracking, with the ability to average out the background over time and to extract trackable features from the region of interest.

5. APPENDIX

Derivation of the Fitts' tracker from a first order Taylor's series expansion.

The measured signal $y(x)$ is assumed to have the form

$$y(x) = s(x-\Delta) + n(x)$$

where the expected signal is $s(x)$ and $n(x)$ is noise. For purposes of this derivation, we will ignore the noise term. Then expanding $y(x) = s(x-\Delta)$ in a Taylor's series we get

$$y(x) = s(x) - \frac{\delta s}{\delta x}\Delta + \left(\frac{\delta s}{\delta x}\right)^2\Delta^2 - \ldots$$

$$\cong s(x) - \frac{\delta s}{\delta x}\Delta, \text{ or}$$

$$(1) \qquad \Delta \frac{\delta s}{\delta x} = s(x)-y(x)$$

We can make this equation independent of x by integrating over the non-zero domain of x. However, since

$$\int_a^b \frac{\delta s}{\delta x}\, dx = s(b)-s(a)$$

may be expected to be zero, we must first multiply both sides of eq (1) by $\frac{\delta s}{\delta x}$ (assuring a non zero integral), and then integrate

$$\Delta \int_a^b \left(\frac{\delta s}{\delta x}\right)^2 dx = \int_a^b \frac{\delta s}{\delta x}\,(s(x)-y(x))\, dx$$

$$\rightarrow \quad \Delta = \frac{1}{c}\int_a^b \frac{s}{x}\,(s(y)-y(x))\, dx; \text{ where } c = \int_a^b \left(\frac{\delta s}{\delta x}\right)^2 dx.$$

6. REFERENCES

1. J.K.Aggrawal and R.O. Duda, "Computer analysis of moving polygonal images," IEEE Trans. Computers, C-24, pp. 966-976, Oct., 1975.

2. N.I. Badler and J.K. Aggrawal ed., Abstracts of the Workshop on Computer Analysis of Time-Varying Imagery, April, 1979.

3. C.L. Fennema and W.B. Thompson, "Velocity determination in scenes containing several moving objects", Computer Graphics and Image Processing, Vol. 9, pp. 301-315, 1979.

4. Fitts, "Video Correlation Tracker", US Patent #4133004, 1979.

5. R. Jain, H.H. Nagel, "Analysis of a real world scene sequence using fuzziness," IEEE Conference on Decision and Control, New Orleans, Dec., 1977.

6. J.O. Limb, J.A. Murphy, "Estimating the velocity of moving images in television signals," Computer Graphics and Image Processing, Vol. 4, pp. 311-327, 1975.

7. W.N. Martin, J.K. Aggrawal, "Dynamic scene analysis: The study of moving images," Technical Report No. 184, Information Systems Research Lab., Electronics Research Center, The University of Texas at Austin, Jan., 1977.

8. H.H. Nagel, "Analysis techniques for image sequences," 41JCPR, pp. 186-211, Kyoto, Japan, Nov., 1978.

9. J.L. Potter, "Motion as a cue to segmentation," Milwaukee Symposium on Automatic Control, pp. 100-104, 1974.

10. J.W. Roach, J.K. Aggrawal, "Computer tracking of objects moving in space," IEEE Trans. on Pattern Analysis and Machine Intelligence, PAMI 1-1, pp. 127-134, 1979.

11. A.A. Sawchuk, "Space-Variant image motion degradation and restoration," Proc. of the IEEE, Vol. 60, pp. 854-861, 1972.

12. J.A. Stuller, A.N. Nehavali, "Transform Domain Motion Estimation," Abstracts of the Workshop on Computer Analysis of Time-Varying Imagery, Philadelphia, PA, pp. 82, April 1979.

13. J.B. Thompson, "Combing motion and contrast for segmentations," Technical Report 79-7, Computer Science Dept., University of Minnesota, Minneapolis, March 1979.

Submitted to the 5th International Conference on Pattern Recognition, Miami Beach, December, 1980
A short paper

Optimal Computation of Image Gradients using Eigenvector Techniques and 3x3 Neighborhoods[1]

by

Wesley E. Snyder[2]
North Carolina State University

and

I-Sheng Tang
North Carolina State University

[1] This research was supported in part by the National Aeronautics and Space Administration, under grant NSG 1535

[2] Currently affiliated with the Deutsche Forschungs- und Versuchsanstalt für Luft- und Raumfahrt

Introduction

The last thing it would seem that the image processing community needs is yet another way to compute image gradients. The subject would seem to have been exhaused, since it is discussed in numerous textbooks, (2, 3, 5) (just to list a few) and literally hundreds of papers (we will not list them) (1) provides one survey) yet here we are, about to report on another gradient operator and worse yet, an operator which is not computationally efficient! There is only one thing which motivated us to develop this operator, an intense need for accuracy. This is because we are modeling frame to frame motion, and using in some cases, the value of the gradient to predict the intensity function in the next frame. Consequently, we could not be satisfied with an edge detector, or some vector in the correct direction and proportional to the magnitude. For example, to represent an image by its Taylor's series,

$$I(x,y)=I(x_o,y_o) + \frac{\partial I}{\partial x}(x_o,y_o)(x-x_o) + \frac{\partial I}{\partial y}(x_o,y_o)(y-y_o) + \ldots$$

requires that the partial derivative terms be evaluated as closely as possible to the correct value, not only in direction, but also in magnitude.

The image processing literature is filled with techniques for computing gradients of pictures. There are linear gradients, mask gradients, statistical gradients, and many others. Virtually all of these return a number which is an indication of the direction and magnitude of the gradient vector.

Th simple mask

$$f(i,j)-f(i-1,j)$$

yields $\frac{\partial f}{\partial x}$ at point i,j, however, such operators are sensitive to noise, and operators which cover a larger area are desirable since they tend to filter noise.

Sobel (2,P271) proposes finding $\frac{\partial f}{\partial x}$ at (i,j) by computing

$$\tilde{f}(i+1,j) - \tilde{f}(i-1,j).$$

Where $\tilde{f}(i,j)$ is an averaged version of f(i,j) computed by taking a weighted average of pixels about (i,j) in the vertical (y) direction. This gradient has good smoothing properties, and therefore is not so sensitive to noise, however some observations need to be made:

1) The Sobel operator actually returns twice the gradient, since it is taken over a space of two pixels.

2) The operator achieves smoothing by throwing away information, the center pixel is ignored.

Some authors [Prewitt, Rosenfeld] point out that one way to find the gradient is to fit a curve to the set of points, and then to differentiate that curve. In this paper, we propose to fit a plane to a 3x3 set of points, and the slope of that plane will be the gradient.

We will compute $\frac{\partial f}{\partial x}$ and $\frac{\partial f}{\partial y}$ separately, and in fact reduce the problem of fitting a plane to two line fitting problems.

For purposes of clarity, the following explanation considers only $\frac{\partial f}{\partial x}$. The argument for the y axis is symmetric.

<u>Definition:</u> f(i,j) is the value of intensity at point i,j, where i corresponds to the x coordinant and j to the y coordinant.

<u>Definition:</u> $\tilde{f}(i,j) \triangleq (f(i,j-1)+2f(i,j)+f(i,j+1)/4$ is the weighted average (as in the Sobel operator) of the intensity at a point, averaging being done only in the y direction.

Thus, in a 3x3 neighborhood, we can compute three horizontal intensity values (for notational convenience, we will rename these variables) $\tilde{f}(i-1,j) \triangleq I_a$, $\tilde{f}(i,j) \triangleq I_b$, and $\tilde{f}(i+1,j) \triangleq I_c$.

We now wish to fit a straight line to these three points. The slope of that line will be the gradient at i,j. To avoid the coordinant-dependent degeneracy problems pointed out by Duda and Hart [2], we will use eigenvector line fitting.

We will consider each point as a vector spanning the space $\langle x, f(x) \rangle$ (having eliminated the y coordinant for the time being.

I_{11}	I_{12}	I_{13}
I_{21}	I_{22}	I_{23}
I_{31}	I_{32}	I_{33}

$\Longrightarrow I_a \mid I_b \mid I_c$

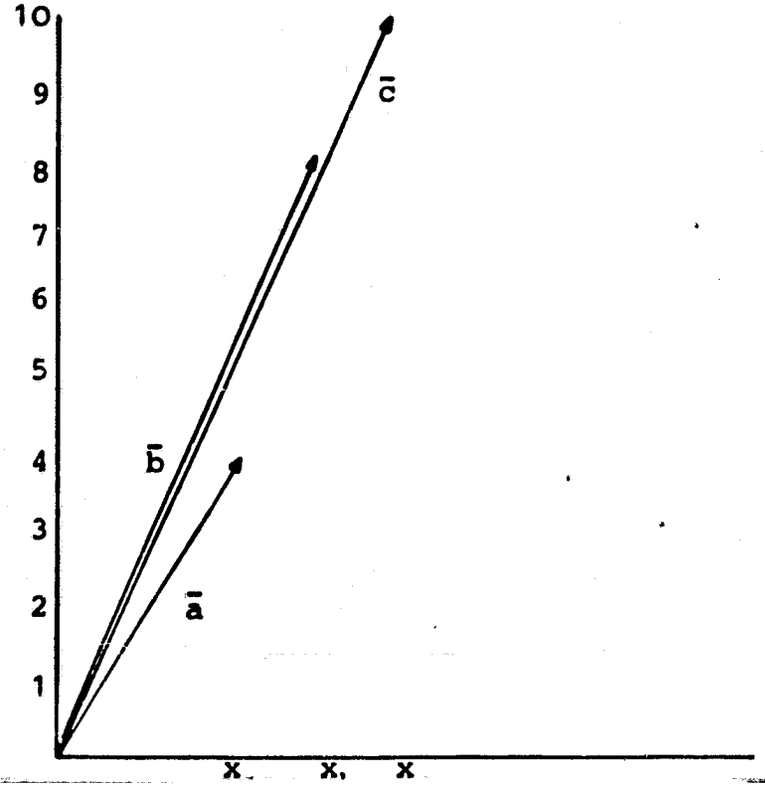
example

3	8	11
4	8	9
5	8	11

$\Longrightarrow 4 \mid 8 \mid 10$

Now I_c=10, I_b=8, and I_a=4

Then the vectors a, b, and c can be defined:



Let the mean vector $\bar{m}^T \triangleq \langle m_x, m_I \rangle$, where $m_x = x_b$ and $m_I = (I_a + I_b + I_c)/3$. The best fitting line will pass through this point.

We now transform the coordinant system to put the origin at $\bar{m}$ by subtracting $\bar{m}$ from each point.

$$v_a^T = \langle (x_a - x_b), (I_a - m_I) \rangle$$

$$v_b^T = \langle \quad o \quad , (I_b - m_I) \rangle$$

$$v_c^T = \langle (x_c - x_b), (I_c - m_I) \rangle$$

Now, we compute the scatter matrix of the three points, (define the scalar $m \triangleq m_I$).

$$S = \sum_{i=I}^{3} v_i v_i^T = \begin{bmatrix} (x_a - x_b) \\ (I_a - m) \end{bmatrix} \begin{bmatrix} (x_a - x_b) & (I_a - m) \end{bmatrix} +$$

$$\begin{bmatrix} o \\ (I_b - m) \end{bmatrix} \begin{bmatrix} o & (I_b - m) \end{bmatrix} + \begin{bmatrix} (x_c - x_b) \\ (I_c - m) \end{bmatrix} \begin{bmatrix} (x_c - x_b) & (I_c - m) \end{bmatrix} .$$

We assume unit spacing on the x axis, so $(x_c - x_b) = 1$ and $x_b - x_a = -1$, and S simplifies to

$$S = \begin{bmatrix} 2 & (I_c - m) - (I_a - m) \\ (I_c - m) - (I_a - m) & (I_a - m)^2 + (I_b - m)^2 + (I_c - m)^2 \end{bmatrix} .$$

Define $\zeta = (I_a - m)^2 + (I_b - m)^2 + (I_c - m)^2$ and $\Delta = I_c - I_a$, and simplify S some more yielding

$$S = \begin{bmatrix} 2 & \Delta \\ \Delta & \zeta \end{bmatrix}.$$

ORIGINAL PAGE IS
OF POOR QUALITY

In the example above $\Delta = I_c - I_a = 10-4 = 6$,

$m = \frac{10+8+4}{3} = 7.33$, and $\zeta = 18.665$.

The line which best fits the data will be in the direction of the principal eigenvector of S. Thus we must first find the principal eigenvalue of S. The characteristic equation is derived from

$$\det(S-\lambda I) = \begin{vmatrix} (2-\lambda) & \Delta \\ \Delta & (\zeta-\lambda) \end{vmatrix}$$

$$= \lambda^2 - (2+\zeta)\,\lambda + (2\zeta-\Delta^2).$$

Setting $\det(S-\lambda I) = 0$ yields

$$\lambda = 1 + \zeta/2 \pm \left(\sqrt{(2+\zeta)^2 - 4\,(2\zeta-\Delta^2)}\right)/2$$

$$= 1 + \zeta/2 + 1/2\,\sqrt{(\zeta-2)^2 + 4\Delta^2}\,.$$

Since we wish the largest eigenvalue, we will choose the positive sign.

Now solving

$$\begin{bmatrix} (2-\lambda) & \Delta \\ \Delta & (\zeta-\lambda) \end{bmatrix} \begin{bmatrix} e_1 \\ e_2 \end{bmatrix} = 0 \text{ for } e_1 \text{ and } e_2 \text{ will yield}$$

the principal eigenvector

The units we shall use for the gradient will be intensity units per pixel. Thus, we will define the horizontal axis to be measured in units of pixels. Therefore we can assume

$$x_a - x_b = -1 .$$

In this case, let $e_2=1$ then $e_1=\Delta/(\lambda-2)$. So the principal eigenvector is $[(\Delta/(\lambda-2)) \quad 1]$ and the best linear approximation to the 3 points is a line passing through $\bar{m}$ in the direction $[(\Delta/(\lambda-2)) \quad 1]$. The slope of that line is

$$(1) \qquad \text{slope} = \frac{\lambda-2}{\Delta}$$

which is the gradient in the x direction.

In the example:

$$= 1+ \frac{18.665}{2} + (2+18.665)^2 - 4(2(18.665)-36) = 20.602.$$

$$\frac{\partial I}{\partial x} = \frac{20.6 - 2}{6} = 3.1 .$$

We should observe at this point, that it is not necessary to assume the sampling rate (x_a-x_b) is equal to one. If instead, the sampling rate is $x_a-x_b = -k$, $x_b-x_c = -k$, for a positive real number k the solution yields:

(for $I_c = I_a$)

$$(2) \quad \text{slope} = \frac{\lambda_{max}-2k^2}{k\Delta} , \text{ where } \lambda_{max} = \frac{2k^2+\zeta+\sqrt{(2k^2-\zeta)^2+4k^2\Delta^2}}{2}$$

The solution (1) is degenerate in the case $I_c = I_a$, since the denominator then goes to zero. This case is best handled by explicity testing for $I_c = I_a$. In that case, one must then test if $\zeta<2$. If so, the gradient is zero, otherwise infinite.

Conclusion

This paper has derived a mechanism for computing image gradients from 3x3 neigborhoods. The method is based on fitting a straight line to the 3 points corresponding to a weighted average of the 9 points in the x (or y) direction. The method is optimal in the sense that it minimizes the summed squared perpendicular distance from the observed points to the line. It is not efficient computationally, since it requires calculation of a square root, but does provide highly accurate estimates of the image gradient in the neighborhood.

References

(1) Davis, L. "A Survey of edge detection techniques" Computer Graphics and Image Processing, Sept. 75.

(2) Duda, R., and Hart, P., Pattern Classification and Scene Analysis. Wiley, 1973.

(3) Pratt, W., Digital Image Processing, Wiley, 1978.

(4) Prewitt, T., "Object Enhancement and Extraction" in Picture Processing and Psychopictories (Lipkin + Rosenfeld, eds.) Academic Press, 1970.

(5) Rosenfeld, A., and Kab, A., Digital Picture Processing Academic Press, 1976.

Submitted to the 5th International Conference on Pattern Recognition
Miami, December 1980.

A full paper

AN ITERATIVE APPROACH TO REGION GROWING

by

Wesley E. Snyder
Dept. of Electrical Engineering
North Carolina State University
and
Deutsche Forschungs- und Versuchsanstalt fuer Luft- und Raumfahrt

and

Alan Cowart
Dept. of Electrical Engineering
North Carolina State University

This work was supported primarily by the National Aeronautics and Space Administration under Grant Number NSG 1535, and in part by the Deutsche Forschungs- und Versuchsanstalt fuer Luft- und Raumfahrt.

ABSTRACT

Region growing is often given as a classical example of the use of recursive control structures in image processing. Recursive control structures, however, are somewhat awkward to build in hardware, where the intent is to segment an image at raster scan rates. This paper describes a hardware structure capable of performing region labeling iteratively, at scan rates. Every pixel is individually labeled with an identifier signifying to which region it belongs.

The difficulties which often justify recursion ("U" and "N" shaped regions, etc.) are handled by maintaining an equivalence table in hardware, transparent to the computer, which reads the labeled pixels. The mechanism for updating the region map is explained in detail.

Actual implementation of such a system would require a content addressable read/write memory. Such memories are now feasible, using existing LSI fabrication techniques.

1. Introduction

One central problem arising in scene analysis is the partitioning of an image into a set of meaningful regions. These regions are composed of all pixels that have similar attributes (such as grey level) and that are connected to each other. One of the most powerful approaches to this task is "region growing" <3>. This technique is initiated by choosing a pixel which meets the criteria for inclusion in a region. It then proceeds by examining all adjacent neighbors of the pixel and incorporating into the region all neighbors meeting an acceptance criterion. This acceptance criterion is based upon the similarity between the pixel and the neighbor in question. Typical measures of similarity include the magnitude of the neighboring pixel's grey level or the relative contrast between the pixel and its neighbor under consideration for inclusion in the region. This process is repeated recursively for all newly accepted pixels until no new pixels can be added to the region. Then a new region is "grown" around a pixel which has not been assigned to a region. When all pixels have been assigned to a region, the algorithm terminates. Since the region growing technique always results in closed regions, this techniques is sometimes preferable to other techniques which are based on edge detection or line fitting.

Numerous variations and applications of the basic region-growing technique have been proposed in the past. A critical component of the region growing technique is the selection of appropriate acceptance criteria. For example, if the acceptance criteria are based on the thresholding of the contrast between a pixel and a neighboring candidate pixel, situations can arise in which a final region does not meet the

original acceptance criteria <4>.

Brice and Fennema <1> overcame this problem with a region merging procedure. An image is initially partitioned into a large number of regions each of uniform grey level. A pair of simple heuristics is then used to merge similar neighboring regions. These heuristics are based on the strength of boundaries separating regions and the rate at which the boundaries of merged regions increase. Yakimovsky and Feldman <2> described a similar procedure in which the region merging criteria are founded on Bayesian decision theory and heuristic techniques.

Milgram et. al. <11> have described the super-slice algorithm for region extraction based on edge detection and variable thresholding. First, a thinned edge picture of an image is obtained. Next, a range of thresholds is chosen, and the regions resulting from each threshold are computed. The threshold resulting in the highest percentage of border points which coincide with the thinned edge points is chosen to define the regions. This method bears similarity to the work of Krakauer <8>.

Although region growing has proved to be an integral part of scene analysis, its use can quickly become computationally prohibitive, particularly for high resolution images. This has prompted some researchers to consider alternative methods of region partitioning.

2. An Algorithm for Region Partitioning

The algorithm for region partitioning presented in this section is a region growing technique based on the concept of equivalence relationships between the pixels of an image. Two pixels a and b are

defined to be equivlaent (designated R(a,b)) if they belong to the same region of an image. This relationship can be shown to be reflexive (R(a,a)), symmetric (R(a,b)<=>R(b,a)), and transitive (R(a,b)&R(b,c)=>R(a,c)).

The transitive property enables all pixels in a region to be determined by considering only local adjacency properties. In this algorithm, each pixel will be compared first with the pixel to its left and then with the pixel above itself in a left-to-right, top-to-bottom raster scan fashon. The assignment of a region label to a pixel results from this comparison operation. Figure 1 demonstrates the situation tha can arise as a result of this comparison. Pixels in a simple binary image are being labeled in raster scan order. The region labeling proceeds in a straightforward manner until the equivlaence relation R(1,2) is discovered at the pixel designated by the question mark.

The system proposed in this paper employs hardware to assign regio labels to pixels and to maintain a table of equivalence relationships. Figure 2 shows that this hardware resides between the image memory and host computer. Functionally, this hardware is transparent to the host computer. For the example of Figure 1, all pixels will be perceived by the host computer as belonging to region 1 (the lower numbered region label takes precedence in an equivalence relationship).

The operation of the hardware is described in the flowchart of figure 3. In order to understand this flowchart, the following notation is introduced:

I(x,y) is the grey scale value of the (x,y) pixel in the image memory.

M(x,y) is the region label number corresponding to the (x,y) pixel in the image memory.

K(i) is the contents of the i-th element in the equivalence memory. This memory is a content-addressable memory.

K*(i)<=K(j) implies the sequence:

1) read K(j)
2) search K using i as a search key (i.e. determine all l such that K(l)=i
3) write K(j) to all positive responders of the search.

T is a threshold

P is the highest numbered region label

The algorithm is illustrated in Figure 4 for an arbitrary region adapted from Milgram et. al. <11>.

As the region partitioning proceeds in real-time (i.e. synchronously with the raster scan), two activities must be performed. First, the M memory must be loaded with the region label number of each pixel under consideration, and second, the K memory must be updated with all equivalence relationships discovered. For example, if region 4 is actually identical to region 2, then both K(2) and K(4) will contain 2 (the lower numbered region label takes precedence). Hence, when the host computer interrogates pixel (x,y) of the M memory, the interface/processor interprets M(x,y) in terms of the K memory and returns K(M(x,y)) to the computer. For example, pixel (10,10) in Figure 4 would be returned as belonging to region 1 since K(M(10,10))=K(4)=1.

The difficulty generally encountered in this type of procedure is the problem of chaining. That is, if R(2,4) and R(3,4) have been determined, then R(2,3) must also be deduced. However, to require that the computer search out all such possiblilities after the image has been processed defeats the original objective of performing region partitioning during the scan time. Chaining is avoided in this algorithm by ensuring that K(2)=K(3)=K(4)=2. However, this merely transfers the chaining problem to the scanning and labeling process. Block 4 of the algorithm flowchart shown in Figure 3 resolves the chaining problem. Whenever an equivalence relationship is detected, all locations in the K memory containing the larger region label number are loaded with the smaller region label number. While the execution of this step in real-time is patently absurd for conventional random access memories, it

is within the capability of the content-addressable memories discussed in the next section.

3. An Architecture for Implementation

The architecture proposed to implement the algorithm of section 2 is shown in Figure 2. This hardware is intended as a special purpose processor for an existing computer-based image processing system. Mori et. al. <9> have also described a special purpose region labeling module used with the Toshiba Pattern Information Cognitive System. However, their system was not intended to operate at real-time (i.e. video) rates.

The architecture of Figure 2 contains four major components: image memory (I), region label memory (M), equivalence memory (K), and an interface/processor. The grey scale values of the image reside in the image memory. Typically, the I memory would contain 512x512 bytes. The region labels assigned to individual pixels are contained in the region label memory. However, the contents of the M memory also include all intermediate region labels for which equivalence labels were determined. Therefore the contents of the M memory must be interpreted in terms of the contents of the equivalence memory. The size of the M memory is directly related to the bit length required to represent the region label (including intermediate region labels) associated with each pixel. This problem is further discussed in Section 4.

The I memory and the M memory are both conventional random access memories. However, the equivalence memory is a content-addressable memory. A content-addressable memory has the property that memory cells

can be accessed or (in this application) loaded by their contents <5,6,7>. In this application, the content-addressable capablilites of the K memory are used as follows: The K memory is considered to have a data bus into it, an address bus, and several control wires. (see Figure 5). In conventional read or write operations, the address is placed on the address bus, the A control line set to zero, and the data will appear on the data bus synchronously with the θ1 clock line. In this mode, it performs as a conventional RAM. In the associative mode (A=1), the data bus is interrogated during θ1, and all memory locations in K are compared with the contents of the data bus at that time. For each memory location, there is a flag flip flop (F) which will be set or cleared according to whether or not the contents of that cell matched the contents of the bus. Then, during θ2 time, the data which is then on the data bus (presumably the new equivalence) will be read into all cells whose flags are set. Thus, the operation $K^*(i)=j$ can be performed in two cycles of a fast clock.

High cost and technical limitations on the size of these memories have traditionally hindered their widespread acceptance and use. Hardware implementations of content-addressable memories have employed a wide variety of performance. However, the development of VLSI and VHSI technologies should stimulate the use of such memories.

Critical design specifications for the K memory are the access speed, memory size, and the ability to write an arbitrary number of memory locations simultaneously. Most techniques for building content-addressablememories emphasize one of these parameters at the expense of the other two. For example, the multi-dimensional access (MDA) memory of the STARAN computer <10>, with its bit-slice structure,

achieves the last two specifications at the expense of memory speed. The parameter most crucial to the development of a satisfactory memory, and hence a system capable of operating in real-time, is the memory size. A near real-time system will result if the memory size necessitates a compromise in the access speed. The memory size is discussed further in Section 4 where simulations involving real images are investigated.

The final component of the proposed architecture is the interface/processor. The primary purpose of this unit is to execute the algorithm described in Section 2 and flowcharted in Figure 3. Additionally, it must be capable of

1) processing the video signal input into grey scale values for storage in the I memory, and
2) interpreting the M memory in terms of the K memory.

The ability to process images in real time will undoubtedly necessitate a design based on special purpose, high speed, bipolar components. However, if near real-time performance is satisfactory, currently available bit-slice microprogrammable microprocessors should prove adequate.

4. Simulation

The algorithm was applied to a 128x128 television image of moderate complexity. Three cases were evaluated:

1) I1=I2, 4 connectedness
2) ABS(I1-I2)<3, 4 connectedness
3) ABS(I1-I2)<3, 8 connectedness

In case 1), for two pixels to be considered in the same region, they had to have identical intensities, whereas in cases 2) and 3), two pixels were considered in the same region if their grey values differed by less than 3 (in a 6 bit image).

In case 3, the center pixel was compared, not only to the pixels above and to the left, but in addition, to the pixel above and to the right (northeast). This is not true 8-connectedness, since the upper left pixel was not tested, but it demonstrates the reduction in complexity which results from this rather simple change.

Three parameters are of interest.

1) the number of elemental regions (those whose labels are stored in M), since this affects the word width of M and K, and the length of K
2) the number of equivalence relations detected, since that indicates amount of data reduction which occurs, and
3) the number of regions perceived by the computer, since that determines the amount of further processing which the computer must do before useful information can be gleaned from the image.

The results are summarized below:

Case 1) 1589 elemental regions (requires 128x128x11 M and 2048x11 bit K)

42 equivalences

1554 perceived regions

Case 2) 392 elemental regions (requires a 128x128x9 bit M and 512x9 bit K)

315 equivalences

291 perceived regions (a significant reduction)

Case 3) 318 elemental regions

412 equivalences

241 perceived regions

The simulation required 4 seconds in PLI on an Amdahl computer.

Figure 6 shows the contents of the M memory for a small area of the image, and Figure 7 shows the same region as perceived by the computer. It is interesting to note that most of the small regions occur because edges have widths greater than one pixel, a fact that is reasonably easy to detect and filter out (in this image).

5. Conclusion

In this paper we have addressed the issue of performing image analysis operations in real time on television scanned data. We have shown that it is possible to design hardware which can perform the operation of region growing in this way. The concept of using equivalence relations to partition an input set is fundamental to the algorithm. And the use of content-addressable read/write memories is essential to the implementation of such equivalence relation processing in real time. This raises the question of whether or not such associative lookup structures cannot also be used for other traditionally recursive computations.

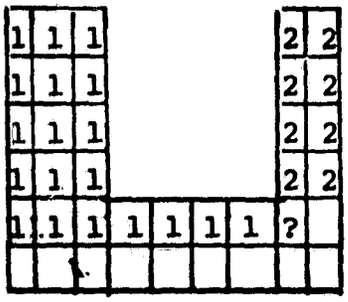


Figure 1

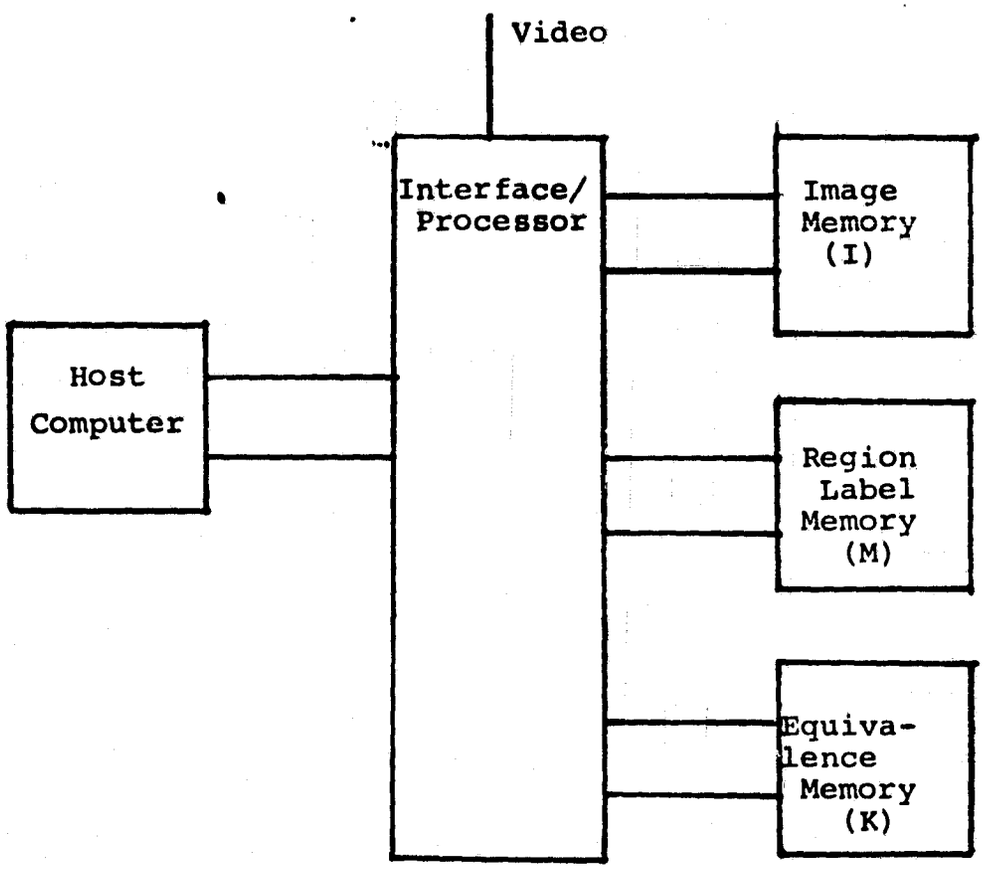


Figure 2

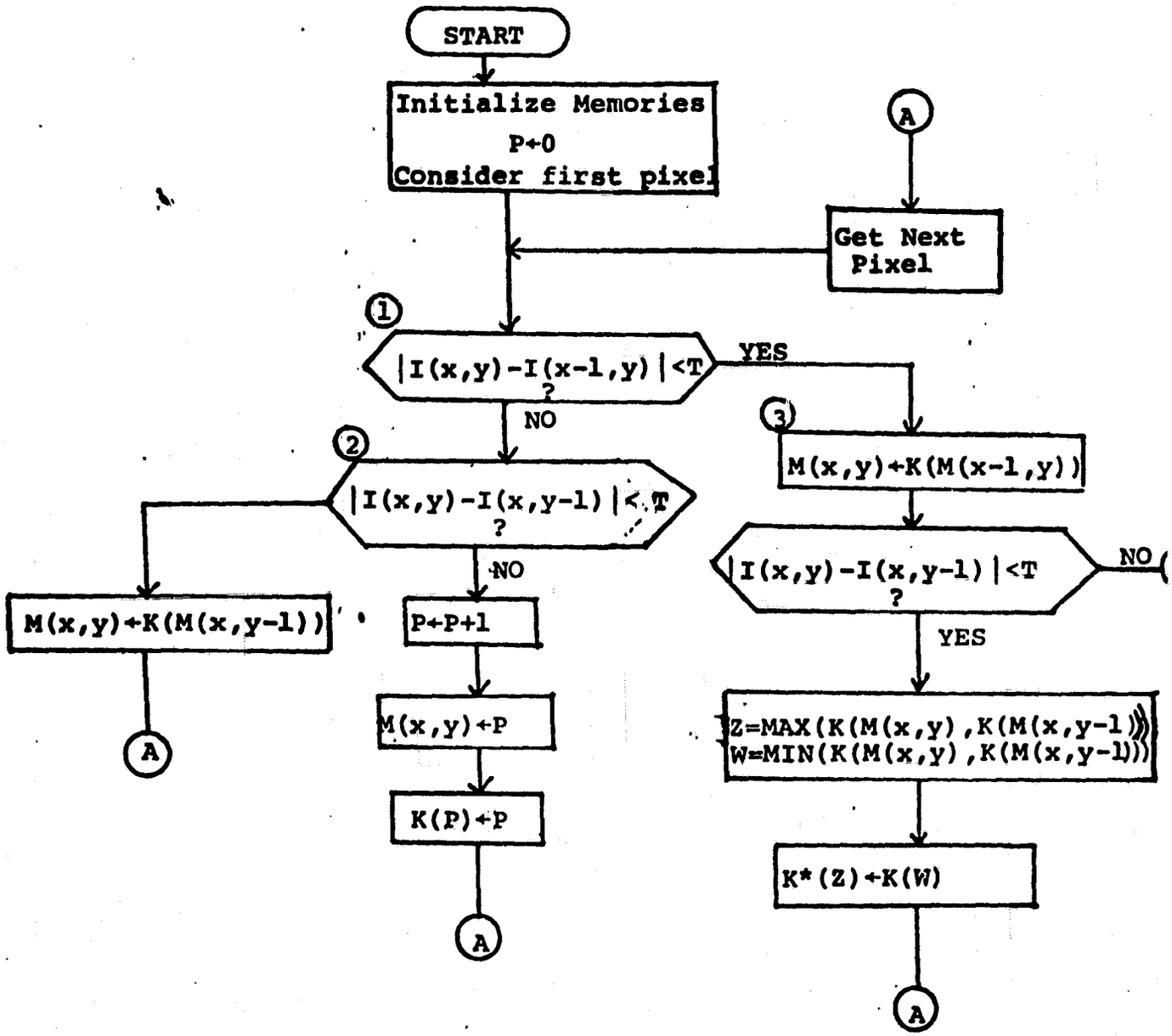
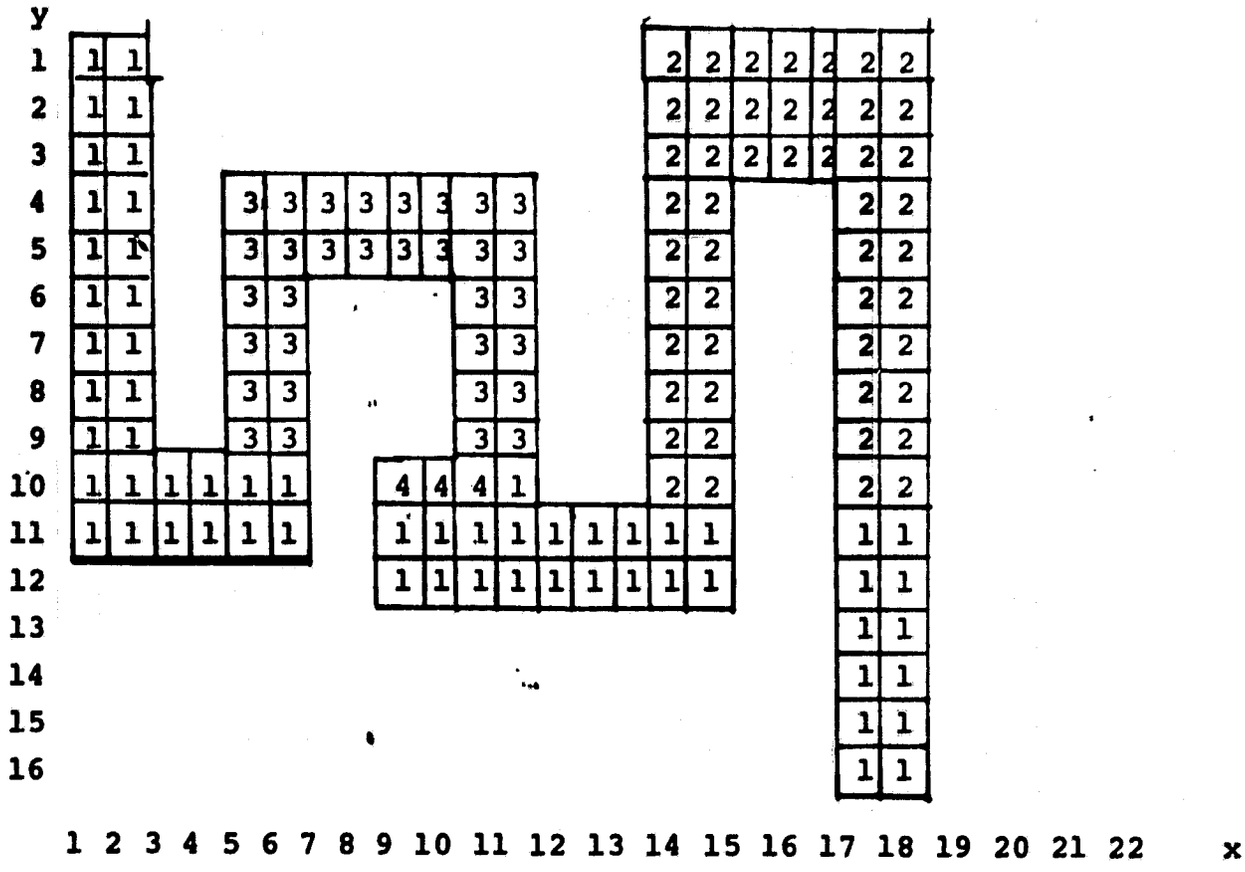

START
Initialize Memories
P←0
Consider first pixel
A
Get Next Pixel
1
|I(x,y)-I(x-1,y)|<T ?
YES
NO
3
M(x,y)←K(M(x-1,y))
2
|I(x,y)-I(x,y-1)|<T ?
NO
M(x,y)←K(M(x,y-1))
P←P+1
|I(x,y)-I(x,y-1)|<T ?
NO
YES
M(x,y)←P
Z=MAX(K(M(x,y),K(M(x,y-1))
W=MIN(K(M(x,y),K(M(x,y-1))
K(P)←P
K*(Z)←K(W)
A
A
A
A

Figure 3



M Memory

Pixel		K Memory Address			
x	y	1	2	3	4
1	1	1			
16	1	1	2		
5	4	1	2	3	
5	10	1	2	1	
9	10	1	2	1	4
11	10	1	2	1	1
16	11	1	1	1	1

K Memory Updates

ORIGINAL PAGE IS OF POOR QUALITY

Figure 4

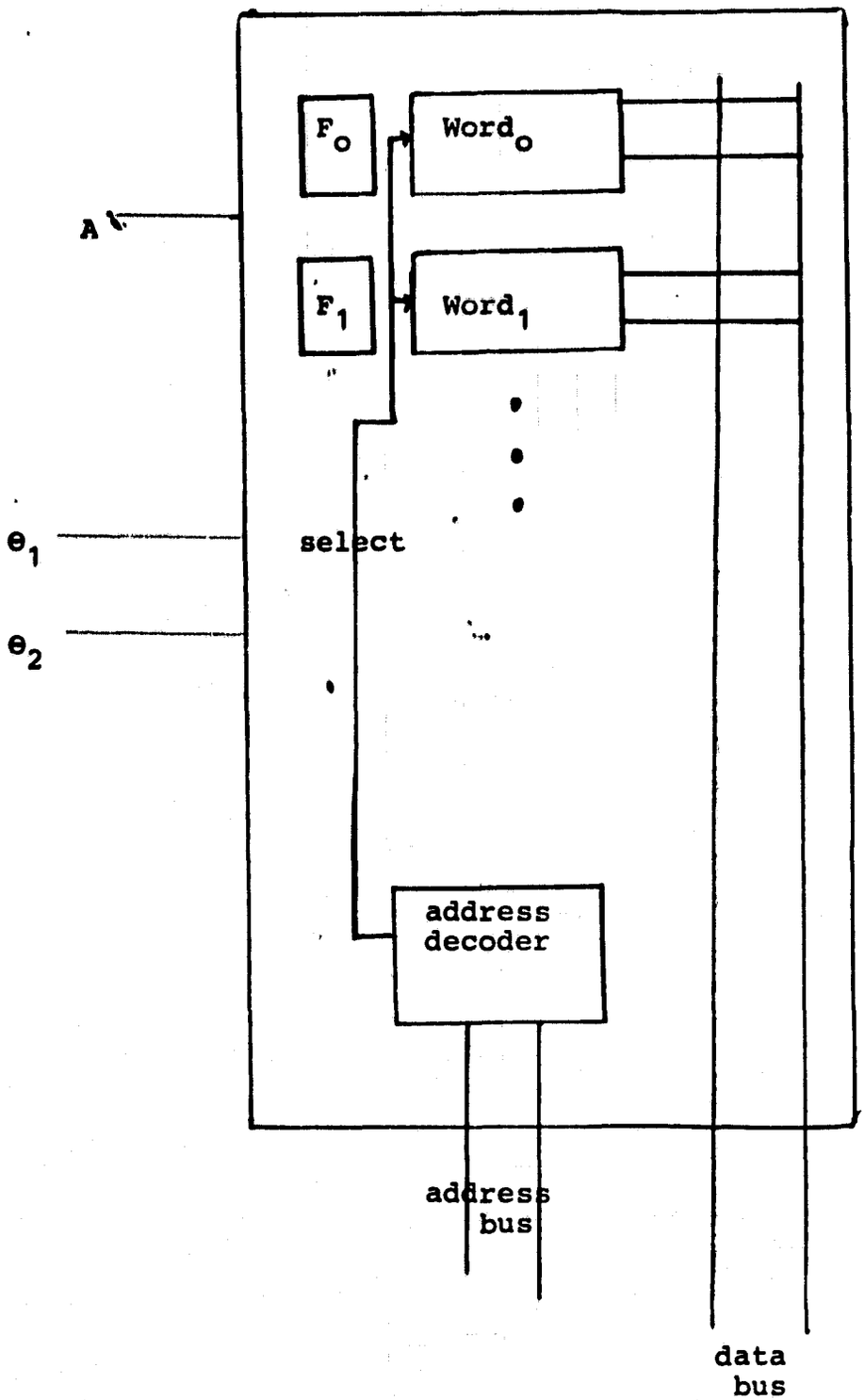


Figure 5

Organization of the K memory

THIS IS THE PRINT OF THE M MATRIX

ZEILENLAENGE = 128 ZEILENZAHL = 128

1*|	0	0	0	1	2	2	2	2	0	3	3	3	3	0	4	5	5	6
2*|	0	0	1	1	1	2	13	13	14	14	14	14	14	14	14	14	14	0
3*|	0	1	1	1	1	1	1	1	14	14	14	14	14	14	14	14	0	0
4*|	0	21	21	21	21	21	21	21	21	21	14	22	22	23	14	0	0	0
5*|	0	24	21	21	21	21	21	21	21	21	21	21	21	21	0	0	0	0
6*|	0	0	0	0	27	21	21	21	21	21	21	21	0	0	0	0	0	0
7*|	0	0	0	0	0	31	21	21	21	21	21	0	0	0	0	0	0	0
8*|	0	32	32	0	0	31	31	31	31	31	31	0	0	0	0	0	0	0
9*|	0	32	33	33	33	33	33	33	34	35	36	37	0	0	0	0	0	0
10*|	0	32	32	32	33	33	33	39	40	41	41	42	43	0	0	0	0	0
11*|	0	32	32	32	32	32	33	45	41	41	41	41	46	47	0	0	0	0
12*|	0	32	32	32	32	50	51	41	41	41	41	41	41	52	53	0	0	0
13*|	0	32	32	32	32	32	51	41	41	41	41	41	41	52	57	0	0	0
14*|	0	0	58	58	58	58	59	60	41	41	41	41	41	61	62	0	0	0
15*|	0	58	58	58	0	0	0	63	64	65	65	41	66	67	0	0	0	0
16*|	0	58	58	58	0	0	0	0	68	69	70	70	67	0	0	0	0	0
17*|	0	0	58	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Figure 6.
Contents of the region label memory
Areas of high intensity (saturated sensor) were set to zero

ORIGINAL PAGE IS
OF POOR QUALITY

THIS IS THE PRINT OF THE REDUCED M MATRIX

ZEILENLAENGE = 128 ZEILENZAHL = 128

1*|	0	0	0	1	1	1	1	1	0	3	3	3	3	0	1	1	1	1
2*|	0	0	1	1	1	1	13	13	1	1	1	1	1	1	1	1	1	0
3*|	0	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	0	0
4*|	0	1	1	1	1	1	1	1	1	1	1	22	22	1	1	0	0	0
5*|	0	24	1	1	1	1	1	1	1	1	1	1	1	1	0	0	0	0
6*|	0	0	0	0	27	1	1	1	1	1	1	1	0	0	0	0	0	0
7*|	0	0	0	0	0	1	1	1	1	1	1	0	0	0	0	0	0	0
8*|	0	1	1	0	0	1	1	1	1	1	1	0	0	0	0	0	0	0
9*|	0	1	1	1	1	1	1	1	34	35	36	37	0	0	0	0	0	0
10*|	0	1	1	1	1	1	1	39	40	41	41	42	43	0	0	0	0	0
11*|	0	1	1	1	1	1	1	45	41	41	41	41	46	47	0	0	0	0
12*|	0	1	1	1	1	1	51	41	41	41	41	41	41	52	53	0	0	0
13*|	0	1	1	1	1	1	51	41	41	41	41	41	41	52	57	0	0	0
14*|	0	0	58	58	58	58	59	60	41	41	41	41	41	61	62	0	0	0
15*|	0	58	58	58	0	0	0	63	64	65	65	41	66	67	0	0	0	0
16*|	0	58	58	58	0	0	0	0	68	69	70	70	67	0	0	0	0	0
17*|	0	0	58	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Figure 7
Contents of the region label memory as it appears to the computer, having been translated by the K mapping

REFERENCES

[1]. Brice, Claude R., and Fennema, Claude L., "Scene Analysis Using Regions," Artificial Intelligence, vol. 1, Fall 1970, pp. 205-226.

[2]. Yakimovski, Yoram, and Feldman, Jerome A., "A Semantics-based Decision Theory Region Analyzer," Proceedings of the Third International Joint Conference on Artificial Intelligence, August 1973, pp. 580-588.

[3]. Rosenfeld, Azriel, and Kak, Avinash C., Digital Picture Processing, New York: Academic Press, 1976.

[4]. Duda, Richard O., and Hart, Peter E., Pattern Classification and Scene Analysis, New York: John Wiley and Sons, 1973.

[5]. Thurber, Kenneth J., Large-scale Computer Architecture: Parallel and Associative Processors, Rochelle Park, NJ,: Hayden, 1976.

[6]. Yau, S.S., and Jung, H.S., "Associative Processor Architecture--A survey," Computing Surveys, vol. 9, no. 1, March 1977, pp. 3-26.

[7]. Parhami, Behrooz, "Associative Memories and Processors: An Overview and Selected Bibliography," Proceedings of the IEEE, vol. 61, no. 6, June 1973, pp. 722-730.

[8]. Krakauer, L., "Computer Analysis of Visual Properties of Curved Objects," Project MAC TR-82, 1971.

[9]. Mori, K., Kidode, M., Shinoda, H., and Asada, H., "Design of local parallel pattern processor for image processing," Proceedings of the AFIPS National Computer Conference, vol., 47, June 1978, pp. 1025-1031.

[10] Batcher, Kenneth E., "STARAN parallel processor system hardware," Proceedings of the AFIPS National Computer Conference, vol. 43, 1974, pp. 405-410.

[11] Milgram, David L., Rosenfeld, Azriel; Willett, Thomas; and Tisdale, Glenn, "Algorithms and Hardware Technology for Image Recognition," Final Report to U.S. Army Night Vision Laboratory, March 31, 1978.

Start second and subsequent pages

ABSTRACT

Computer tracking of moving objects using television imagery is discussed. Problems which are unique to this field are described, including camera motion, occlusion handling, and difficulties with correlations. These problems are described in the context of several solution approaches mentioned in the literature. A field-operational system is described which demonstrates the ability to deal with these problems with some limitations. Finally, future direction for research in this area are proposed.

1. INTRODUCTION

It was only a few years ago that even the simplest image processing operations strained state-of-the-art computers to their utmost limits. A 128x128 array took a lot of memory, and even the simplest operations required enormous amounts of time. Today, with faster computers and cheaper memory, image operations still take a lot of time, but it is no longer unthinkable to process 512x512 images. So now, with the technology catching up, imaging devices are expected to become an increasingly important type of sensor for closed-loop controllers in different kinds of "intelligent systems" such as occur in robotics, missile guidance, and automatic tracking of moving objects. These applications require real time processing of sequences of images, and once again the demands of image processing strain the technology to its limits.

We define Computer Analysis of Time-Varying Images (CATVI) to mean the processing of a time sequence of images in order to extract some property which changes in time. There are two types of uses for motion information:

1) Where the motion information itself is what is important. Trackers typify such systems.
2) Where the motion information is used to provide additional features to scene analysis systems.

Since in order to classify a target one must first locate and track it, systems whose primary goal is of type 2) generally incorporate components of type 1). Such systems can also be decomposed (as Martin and Aggrawal [13] did) into components which have the tasks of a) motion detection, b) attention directing, and c) cognition. For practical real time applications these process should run in parallel.

Under the category of trackers, we distinguish two types of systems, background trackers and target trackers. Background trackers identify some feature in a background (e.g., a bridge), and track that feature, even though the camera may move. Background trackers have found applications such as remotely-piloted vehicles, "smart bombs", which must home in on a target, and LANDSAT-type satellites which must identify and track ground control points to increase their pointing accuracy. In all these applications, the intensity function can usually be considered to be continuous and differentiable at all points, with non-zero derivatives at most points. (As we will discuss in a moment, these assumptions are important to the design of the tracker).

A target tracker may have to deal with an entirely different situation, for example, an airplane against a daytime sky. In such applications, the sensor is often saturated, resulting in an image with high contrast and low dynamic range. The region(s) in the image which describe the target are then homogeneous in intensity, and motion information can be extracted only at the edges of the regions.

Gilbert [9] extends our definition of trackers slightly to distinguish between contrast trackers, which threshold the intensity, edge trackers, which threshold the magnitude of the gradient, and correlation trackers, which correlate shapes, based on intensity.

Under our category 2) above, we can make use of motion information either for image segmentation or for target classification. Segmentation implies grouping of similar pixels. For example, velocity information could be used as a similarity measure to direct segmentation. In addition to grouping of pixels of a similar velocity or acceleration together and thus aiding segmentation, that same velocity or acceleration data can be used to provide features for a classifier.

The area of CATVI can be subdivided in many ways, according to objectives of the system, as we have done, or by the methods used, as Nagel [15] has done. In this survey paper, Nagel points out that the basic problem of finding relationships between images in a sequence can be broken down into several steps:

1) Derive a description (or descriptions) of the image substructure,
2) Establish a (dis-) similarity function of these descriptors,

ORIGINAL PAGE IS
OF POOR QUALITY

Start second and subsequent pages

3) Use this function to establish compatibility or incompatibility between substructures, and

4) As a result of the comparison, describe the pertinent aspects of the image sequence.

Nagel then goes on to survey the literature according to this schema.

Our purpose in this paper is not to provide a survey of the literature, but to discuss tracking. The reader is directed to the aforementioned survey by Nagel, and to the survey by Martin and Aggarwal [13] for a thorough literature review. We would mention one interesting item at this point, however. As Martin and Aggarwal point out, the fundamental problem in CATVI is the problem of occlusion. This observation brings us to discuss what we perceive as the principal research problems in this area.

2. DIFFICULT SITUATIONS FOR CATVI SYSTEMS

Although this is a very new area (the military has been in the business a bit longer than the University community), there have been nonetheless a considerable number of papers published, and it is instructive to note the simplifying assumptions which people have made in order to develop systems which work. The most common assumptions are:

1) The camera does not move

2) The target is never, or never totally, occluded

3) Image gradients are well defined

Item 3) is often an implicit assumption and seldom stated outright in this form. We will discuss these three items briefly.

2.1 Camera motion

The most straightforward means for tracking moving objects is to work with difference pictures. This entails computing the absolute value of the difference between two successive frames, on a pixel-by-pixel basis. Then, with a stationary background, the background pixels subtract away to zero, and the moving target stands out clearly.

But consider now a TV tracker with a servo on the pan and tilt of the TV camera. If the tracker works perfectly, the target will be in the center of the screen at all times. Frame to frame differencing then subtracts out, not the background, which changes with camera angle, but the target! If we then add a servo controlled zoom lens to the system, the problems become really interesting!

Of course, if all these effects are precisely known, they can be computationally taken into account. However, it is doubtful whether these effects will ever be so precisely known that difference methods on a pixel by pixel basis alone will work in real situations. Any sophisticated algorithm must therefore ultimately be tested against real world conditions. Robustness, the ability to adapt to unexpected circumstances, is a major requirement.

2.2 Occlusion

The target may occasionally be either partially or totally occluded. In the case of partial occlusion, a number of approaches are possible, including subtracting the occlusion to find the non-occluded portion(s) of the target, and then matching those parts to the model of the target, to get an updated target location. This matching problem can be quite difficult in the general case.

When the target disappears behind an occlusion, tracking is, of course, impossible. One clue for handling this situation may be derived by observing people. Confronted with a moving target which flys behind an occlusion, people tend to extrapolate the velocity of the target and predict its point of reemergence from behind the occlusion. They then direct their attention to that point. A backup mechanism in the form of periferal vision seems to be provided in case the target does not emerge as predicted. Thus, to emulate the occlusion handling capabilities of people, a computer system needs prediction, attention directing, and "peripheral vision" coupled with backup capabilities.

Yet another class of problems occurs when the occlusion is itself a moving object. In this case, some classification is necessary to distinguish the two objects when they separate. Among the more recent work along these lines is that of Martin and Aggarwal [14], extended by Roach and Aggarwal [16], where the edges of objects are segmented into a set of approximately straight lines and circular arcs. While this segmentation again appears artificial or "man-made", the idea behind it, i.e., matching edge segments of partially occluded objects using motion information to "analyze an apparent object into its constituent objects"- represents an important step in the direction we think occlusion analysis should go. In section 3, we discuss a very simple but effective form of occlusion handling in an operating tracker system.

2.3 The Problem with Edges/preprocessing

Digital images, particularly images of man-made objects, tend to have areas which are homogeneous in intensity. If one places a small window or local operator about a pixel in the center of a homogeneous region, and then looks at that same area in the next frame, no change will be observed unless an edge moves through that window. This situation is most extreme in the case of binary images and contrast trackers. Motion information can be derived, not from the motion of regions or objects, but only from the motion of edges.

We should point out here that our use of the word "edge" implies an area where there is a significant rate of change of intensity with respect to distance in picture coordinants. Edges and object boundaries are not synonymous. While object boundaries usually give rise to edges, the existence of an edge does not necessarily imply a boundary, since objects may have internal detail which gives rise to internal edges.

We have observed the fact that motion information comes from edges occuring several times in current literature; we will point out two examples.

First, consider the velocity estimator of Fennema and Thompson [5]. They point out that motion can be represented by a velocity vector

$$v = \frac{ds}{dt} ,$$

and that if ds is an incremental translation of the surface, and di an incremental variation in intensity due to this motion, then

ORIGINAL PAGE IS
OF POOR QUALITY

Start second and subsequent pages

$di = G.ds$, or $di = G.V\,dt$,

where G is the gradient at the evaluation point.

Since the frame-to-frame intensity change at a point is known, and the gradient computable, then it should be possible to find the velocity of each pixel.

Another paper [19], shows that Fennema and Thompson's formulation can be derived from a first order truncation of the Taylor's series expansion of the intensity function.

For this method to work, the gradient at the pixel being examined must be the same in both frames, which implies dG/dt must be very small or zero. Fennema and Thompson accomplish this by blurring the edges to insure that the width of the edge is greater than the frame to frame displacement, so that, on the average, G does not vary from frame to frame.

Analysis of time-varying images looks at first like radar signal processing, since the objective is to pick out a known signal from some background (noise). However, due to the 2-dimensional nature of the signal, the probability of homogeneous areas, and the likelihood of occlusions, classical correlation techniques based on intensity alone are not likely to be successful. As Dreschler and Nagel [3] point out,

> It might be argued that crosscorrelation between object candidates extracted from two consecutive frames should allow tracking the image of a moving object... There are several weak points to such an approach... Unless a tight subsection around the image of a moving object can be determined fairly accurately... the crosscorrelation of such subsections from consecutive frames will depend heavily on the structure of the stationary image components in the selected subsection. Tracking of the image of a moving object thus tends to become unreliable. On might attempt to circumvent this difficulty by crosscorrelating difference pictures obtained with respect to a reference frame. This will only succeed if the image of exactly one moving object dominates the difference picture- an assumption which might be difficult to check automatically during this stage of the analysis.

For these reasons, we feel it is necessary in general that processing of images of moving objects be based on "matching" of features other than simple intensity functions. However, with the correct assumptions, the concepts of correlation and matched filtering can be used, although such a system will inevitably encounter this same problem with edges.

One such system, developed by Fitts [6] encounters the problem with edges in a slightly different way. He determines that the displacement of a point δ_x can be computed with only two one-dimensional correlations, by using

$$\hat{\delta}_x = \frac{1}{c}\int_a^b -\frac{\partial s}{\partial x}\,[y(x)-S(x)]\,dx$$

$\hat{\delta}_x$ is the estimate of δ_x, y(x) is the measured intensity function, (assumed to be $y(x)=s(x-\delta_x)+n(x)$), and S is the known (tracked) signal. Again, one observes that information exists only in the edges, for to have an non-zero integrand, the tracked signal must have a non-zero derivative at the same point that it has a difference due to motion. It is thus more suitable to background tracking than contrast tracking. Fitts justifies his algorithm by matched filtering theory, however it is interesting to note that his tracker can also be derived from first order Taylor's series assumptions. This derivation is given in the appendix.

The point of these two examples drawn from the literature is simply this; motion information exists (locally) only in the edges, and this fact must be kept in mind when designing a CATVI system. Since it is clear that motion information is so intrinsicly involved with the structrure of intensity edges, one might then ask if it does not make sense to handle the extraction of edges as a separate operation. Furthermore are there not other, similar operations which could be handled at a low, preprocessing levels. There is a great deal of evidence that organic systems perform such preprocessing functions.

For instance, Gaardner [8] has pointed out that the human visual system heavily depends on an intrinsic feedback system involving eye jumps even in the fixation case. It is indeed easy to show that when the eye is fixed on a certain object without movement, the image of the object blurs due to adaptation. However, small eye jumps produce refreshed images resulting in new information only at the edges. There are other hints that in mammalian visual systems there is a lot of feature and edge enhancement. From Hubel and Wiesels's work [11], we know that in the cat's visual cortex there are neurons functioning as edge or line detectors, with special sensitivity to orientation. Some of these functions are even independent of translational shifts of these lines.

There have been different artificial approaches to preprocessing pictures for feature enhancement. Such preprocessing serves one of two basic purposes: to reduce the dimensionality of the feature space, or to increase the orthogonality of the feature vectors [12], both of which potentially lead to better performance of a matcher. One could, for example, preprocess to extract a segment of the target contour, and then match that segment.

Another form of preprocessing results from the use of image transforms. The general area of transforms has been studied extensively, to the point that such transforms are textbook material [1, 17], but only some work has been done on transforms in the context of CATVI [2, 4, 18, 20], particularly with respect to transforms of segments of an image with the intention of removing or identifying translational effects. For example, at first glance, the properties of the Fourier transform seem very striking, since the amplitude response is insensitive to translational shifts, while the phase response immediately shows up the amount of such shifts. But in reality there never is a whole picture which is completely shifted, but only a small part in it which is not identified a-priori. Thus the Fourier transform simply has a different appearance, without a discernable correspondence between the shift of the target and the response of the transform. Consequently, one must also temper his enthusiasm when considering "obvious" applications of transforms.

We have mentioned some of the difficulties with analysis of time varying images and some of the problems which one encounters when attempting to apply conventional tools to those problems. We will now describe one operational system which functions reasonably well in this environment.

Start second and subsequent pages

3. THE DFVLR TV-TRACKER

In this section, we will describe the TV tracker system which is currently operational at DFVLR [10]. It is a contrast tracker, in which the camera pan and tilt are servoed automatically to keep the target centered in the field of view. It can accomodate occlusions, (with some limitations), and rotations and size changes in the target.

The system is built around a critical design philosophy: it is more important to be robust than to be accurate. That is, small inaccuracies in computed target location, velocity, and description of target shape can be tolerated, but the tracker must not ever lose the target.

This section describes the DFVLR level 1 tracker, which is operational in the field. Our level 2 tracker provides greatly improved ability to distinguish target shapes, but is currently implemented only in simulation, and will not be discussed here.

3.1 The overall system

Fig. 1 shows a simplified block diagram of the tracking system. The video signal is supplied by a black and white gimballed camera and displayed on a color monitor. (The color functions of the monitor are used to overlay the window location and treshold boundaries). At the same time the analog video signal is compared in the signal processing logic with an adjustable grey value level. Whenever the amplitude of the video signal crosses this threshold, either rising or falling, the corresponding line number and pixel number in this line are registered from counters synchronized with the frame and line sync pulses. At the end of the line these values, having been accumulated in a fast memory, are transmitted to a process computer.

Signal evaluation (from the hardware) occurs only within a window which is normally adapted to the object size automatically by the computer. The computer then has the task of calculating the center and extensions of the target by the given contour values (including window calculation) and servoing the camera so as to null the deviation between object center and screen center.

The level 1 tracker is characterized by an enormous reduction of information flow. It uses only the first and last threshold values in one line (inside the window) and it actually does not process all values of the object contour, but only the extrema in the horizontal (x) and vertical (y) directions. So only four values are actually encoded in each frame and used for decision making. These four extrema may be due to the target contour, or they may stem from occlusions entering the window. It is the task of the logical decision part of the computer program to cancel implausible values and replace them by estimated ones.

3.2 Structure of the camera control loop

In this section, the closed loop for camera control (in x and y) is briefly outlined in its dynamical behavior. An essential implication is the temporal relation between scene perception, processing and realization of camera motion. (table 1).

acceptance of contour coordinates per area line in frame k; determination of extrema (max, min) for frame k	concurrent →	processing of the extrema from frame k-1; computation of window and camera position for frame k+1

Table 1. Software activities while frame k is recorded

With each frame starting pulse, the same operations are initiated, which is analogous to a normal sampled data system. While frame k is recorded from the video signal, the computer is processing the contour extrema of frame k-1. This program is concurrently interrupted by the transmission of the contour values of the running line of frame k. Coupled with this transmission is a min-max search so that at the end of frame k the corresponding extrema are known. Processing of frame k-1, being a background job is also finished at the end of frame k. The computer has by the end of frame k checked the suppositious target motion for plausibility, corrected it if necessary, predicted the target position two frames in advance, and calculated window coordinants and camera position from frame k+1.

Consequently, by the time data transmission from the window in frame k+1 starts, the camera has been commanded to move (by a command initiated after analyzing frame k-1), and will be in the correct position.

In a control theoretical sense, the plant here consists of a delay made up by two sampling periods (i.e. frame periods). In the z-transform domain often used with sampled data systems, a delay of 2 periods is characterized by a factor z^{-2}, while a prediction of two periods means multiplication by z^2 (see figure 2).

As is displayed on the screen, only the relative position between target position and camera position is measured by the computer. To deal with occlusions, it is necessary to get the corrected target position by adding the camera motion. The problem solving and processing program (see figure 2) tries to solve this task in "problem" cases too, and then predicts the ("true") target position two frames ahead in order to catch up the inherent delays. After processing occlusions and estimating velocities (if necessary), the effects of camera motion are re-inserted to provide an error signal for the camera servo. A digital low pass filter processes this error signal first. This filter is charged with:

a) smoothing the small contour jumps from frame to frame which are due to the interlaced scanning techniques used in television,

b) supplying sufficient phase reserve ahead of the position-error integrator in the camera control circuit.

The dynamics of the closed loop are adjustable with the gain V (from fig. 2), so that the closed loop poles yield sufficient stability. As the system contains only one integration, an object moving with constant velocity generates a stationary position error that has no major influence on the tracking capability.

3.3 Treatment of problem cases (e.g. occlusions)

The most critical part of the control program has the task of checking the measured (apparent) object

Start second and subsequent pages

extrema for plausibility. For this purpose it uses:

a) the past, digitally filtered speed values of the contour extrema xmax/min, ymax/min and, computed from these values, the overall speed of the target in x and y.

b) the object extension averaged over several frames.

Processing a frame implies checks for two kinds of disturbances.

a) a jump disturbance, which occurs when suddenly one or several contour extrema (in the following abbreviated as CE) coincide with window edges; in this case it can be assumed that an occlusion has entered the window; of course, for this to work, the real target size never should exceed the edges of the window. The window size is continously monitored to insure that this assumption is true.

b) a velocity disturbance, which occurs when the measured interframe change of a contour extremum (CE) does not have the same sign as the computed overall velocity.

Taking account of the typical structures of occlusions (trees, poles, horizon), different algorithms were developed for the x and y directions, which, in condensed form, are as follows:

x-direction
In case of a jump disturbance or a velocity disturbance the velocity of the disturbed CE is not updated; if both CEs are disturbed, pure prediction is performed, i.e., window size and speed keep their old values. Moreover, in the case of a velocity disturbance in one CE, the disturbed CE is estimated from the latest object size and the other (undisturbed) CE. Object size is updated only in undisturbed cases.

y-direction
Here only jump disturbances are checked. When a disturbance is recognized, the corresponding CE is again estimated from the undisturbed one and updating of CE velocity is omitted. If both CEs are disturbed, prediction is used.

If the growth of the y-extension taken over several frames exceeds a threshold, then that CE which is the leading one with respect to object velocity is used to briefly (i.e. for one frame) draw after it the other CE. Then (in the next frame) it will be determined whether the object size actually increased so much or whether it must overcome an occlusion stationary with respect to y.

Several typical cases as they are recognized and solved by the program are shown in fig. 3. For example, it takes care that in case of a target merging horizontally into an occlusion the window is fixed in the y direction, but the window edge entering to occlusion tries to move to the end of this occlusion and to wait for the target there. Independent of whether meanwhile the target reverses its direction or not, the window is finally detached from the occlusion.

Though the tracker initially was designed only for ground-air tracking with good contrast, it proved in field experiments to show good tracking capabilities in ground-ground tracking with occlusion situations too. Thus it can be shown that by intelligent processing of dramatically reduced information (4 values per frame) needing only 30-40% of real time on the computer, surprisingly good results could be achieved.

However it is clear that with using only 4 numbers per frame it is impossible to adequately distinguish between occlusions and targets in all cases. The system has particular problems where an occlusion is oriented in the direction of target motion. Therefore we are currently developing software which will be able to better discern occlusions by making use of all threshold transitions within the window rather than only the extrema.

4. FUTURE TRENDS

Most work in the area of computer analysis of images to date has used "synthetic" algorithms. That is, the tracking techniques are based on ideas of how we intuitively perceive that a computer could be directed to perform these tasks. Our thinking in developing these algorithms is likewise directed by the assumption that we will be using a digital computer. The methods we develop may be well formulated mathematically, but they are, in some sense, not "natural", being based as they are on the principles of today's serial computers.

A major weakness of these programs is their inability to generalize. Just as one cannot write a program which is a "little bit wrong", such programs are incapable of generalizing to slightly different situations. Thus, such trackers are highly dependent on assumed situations, for example, on the assumption that acceleration of the target is less than some threshold.

But how is it possible to build machines wich generalize?

In the late 50s and early 60s, the theory of threshold-logic devices as computational structures representing neural models was highly touted as the solution to all pattern recognition and cognition problems. These threshold logic devices seemed so similar to basic computer operations that the main problem was believed to be how such elementary classification decisions could be combined in multilevel or hierarchical processes in order to construct intelligent and learning systems.

These theories declined in popularity rapidly as it became virtually impossible to find support for related research. This decline can be attributed to several causes:

1) The remarkable advances in silicon technology with corresponding drops in price and increases in capacity of conventional (von Neumann) computers.

2) The observation that von Neumann machine implements a Turing machine and consequently can compute any computable function, at least in principle.

3) Early overenthusiasm on the part of the neural model researchers, with corresponding claims of potential which far surpassed actual results.

4) A significant underestimation of the complexity of the brain.

So people became more and more interested in the use of digital computers, particularly as the technological breakthroughs of the 60s occurred. Most of the work in artificial intelligence then concentrated on formulating problems so that they were conveniently treatable by digital computers. The enormous advances in technology prevented people from asking the critical questions as to whether computer development based on the von Neumann machine might turn out as a tremendous sidetrack.

C-8

Start second and subsequent pages

Nevertheless, by forthcoming neurophysiological research in the late 60s and early 70s, it became obvious that

a) the human brain is not composed of an aggregate of single threshold devices, but of highly parallel, analog, feedback-type and even linear connections that represent distributive "memory traces" which are learned during an experience and which allow the association of stored patterns from incomplete input keys. Reconstruction of patterns seems to be the main point of cognitive operations, and not classification of single items; and

b) todays computers show very little similarity to the functioning of the brain. They are, as already mentioned, unable to get along with even slightly erroneous input information unless that particular case has been anticipated by the programmer; i.e., the are unable to associate.

Thus, after several years of neglect, the concept of brain models is arising again, with a better understanding of neural structure and much more sophisticated models [7]. Work currently underway is also well founded mathematically [12]. Such systems have demonstrated the ability to recall a sequence of previosly taught images when presented with a single element from the sequence [22, 7]; to recall an entire image when presented with only a portion of it [12]; and under certain constraints, to generalize.

As yet, there is no clear way to structure such an associative machine so that it can recall an entire image (say of a target) when presented with only a portion of the image (an occluded target), which is rotated and displaced in space. This property of handling displacement is essential to the use of associative machines as trackers.

We feel that this area of research deserves more attention now that a better theory is seen to be evolving, as it could potentially provide solutions with the necessary parallelism to attain the speeds required for real time image analysis.

REFERENCES

[1] Andrews, H. and Hunt, B
Digital Image Restoration, Prentice-Hall 1977

[2] Arking, A., Lo, R. and Rosenfeld, A.
"An Evaluation of Fourier Transform Techniques for Cloud Motion Estimation", Computer Science Technical Report TR-351, University of Maryland, 1975

[3] Dreschler, L. and Nagel, H.,
Using Affinity for Extracting Images of Moving Objects from TV-frame Sequences" Technical Report Ifi-HH-B-44/78, University of Hamburg, 1978

[4] Dukhovich, I. and O'Neal, J.,
"A Three-Dimensional Spatial Non-linear Predictor for Television", IEEE Transactions on Communications, COM-26, May, 1978.

[5] Finnema, C. and Thompson, W.
"Velocity Determination in Scenes Containing Several Moving Objects", Computer Graphics and Image Processing, 9, 1979.

[6] Fitts, J.
Video-Correlation Tracker, US Patent 4133004, 1979.

[7] Fukushima, K., and Miyake, S.
"A Self-Organizing Neural Network with a Function of Associative Memory", Biological Cybernetics 28, 1978.

[8] Gaardner, K.
Eye Movements, Vision and Behavior, Hemisphere Press, Washington, 1975.

[9] Gilbert, A., and Giles, M.
"Concepts in Real-time Tracking", TR STEWS-ID-78 1, White Sands Missile Range, New Mexico.

[10] Hirzinger G., and Landzettel, K.
"A TV-tracking System Based on Computer Intelligence" AGARD Panel Technical Meeting on Image and Sensor Data Processing, for Target Acquisition and Recognition, Copenhagen, 1980

[11] Hubel, D., and Wiesel, T.
"Receptive fields and Functional Architecture in two non-striate visual areas of the Cat", Journal of Neurophysiology, 1965.

[12] Kohonen, T.
Associative Memory, Springer Verlag, 1978.

[13] Martin, W., and Aggarwal, J.
"SURVEY, Dynamic Scene Analysis", Computer Graphics and Image Processing, Vol 7 no 3, June, 1978.

[14] Martin, W., and Aggarwal, J.
"Dynamic Scene Analysis: the study of Moving Images" TR NO. 184, Information Systems Research Lab, Electronics Research Center, University of Texas at Austin, 1977.

[15] Nagel, H.
"Analysis Techniques for Image Sequences", 4IJCPR, Kyoto, Japan, 1978.

[16] Roach, J., and Aggarwal, J.
"Computer Tracking of Objects Moving in Space" IEEE Trans on PAMI, 1-1, No 2, 1979.

[17] Rosenfeld, A., and Kak, A.
Digital Picture Processing, Academic Press, 1976.

[18] Sawchuk, A.
"Space-variant Image Motion Degradation and Restoration," Proc. of the IEEE, Vol 60, No 7, 1972.

[19] Snyder, W., and Rajala, S.
"Analysis of Time Varying Images and the Problem with Edges", submitted to the IEEE Conference on Pattern Recognition and Image Processing, Miami, December, 1980.

[20] Stullet, J., and Netravali, A.
"Transform Domain Motion Estimation" Workshop on CATVI, Philadelphia, 1979.

[21] Thompson, W.
"Combining Motion and Contrast for Segmentation" TR 79-7, Computer Science Dept, Univ of Minnesota, 1979

[22] Willwacher, G.
"Fähigkeiten eines assoziativen Speichersystems im Vergleich zu Gehirnfunktionen", Biological Cybernetics 24, 1976.

Start second and subsequent pages

APPENDIX

Derivation of the Fitt's tracker from a first order Taylor's series expansion

Center this information within shaded area

Type title of paper, name and affiliation of author in this area

The measured signal $y(x)$ is assumed to have the form

$$y(x)=s(x-\Delta)+n(x)$$

where the expected signal is $s(x)$ and $n(x)$ is noise. For purposes of this derivation, we will ignore the noise term. Then expanding $y(x)=s(x-\Delta)$ in a Taylor's series we get

$$y(x)=s(x) - \frac{\partial s}{\partial x}\Delta + (\frac{\partial s}{\partial x})^2\Delta^2 - \ldots$$

$$\approx s(x) - \frac{\partial s}{\partial x}\Delta, \text{ or}$$

$$(1)\Delta \frac{\partial s}{\partial x} = s(x)-y(x)$$

We can make this equation independent of x by integrating over the non-zero domain of x. However, since

$$\int_a^b \frac{\partial s}{\partial x}\,dx = s(b)-s(a)$$

may be expected to be zero, we must first multiply both sides of eq (1) by $\frac{\partial s}{\partial x}$ (assuring a non zero integral), and then integrate

$$\Delta \int_a^b (\frac{\partial s}{\partial x})^2 dx = \int_a^b \frac{\partial s}{\partial x}\,(s(x)-y(x))\,dx$$

$$\rightarrow \quad \Delta = \frac{1}{c}\int_a^b \frac{\partial s}{\partial x}\,(s(x)-y(x))\,dx; \text{ where } c = \int_a^b (\frac{\partial s}{\partial x})^2 dx$$

Instructions for Typing

Conference papers will be reproduced for publications by the photo offset method.

1. Use a new black or photographic typewriter ribbon.
2. Make sure type is clean.
3. If you make a mistake, make a clean erasure or use Snopake correction fluid.
4. All copy must be within the two columns indicated by solid lines.
5. Position illustrations on sheets within solid margin lines.
6. Single space typing only.

ORIGINAL PAGE IS OF POOR QUALITY

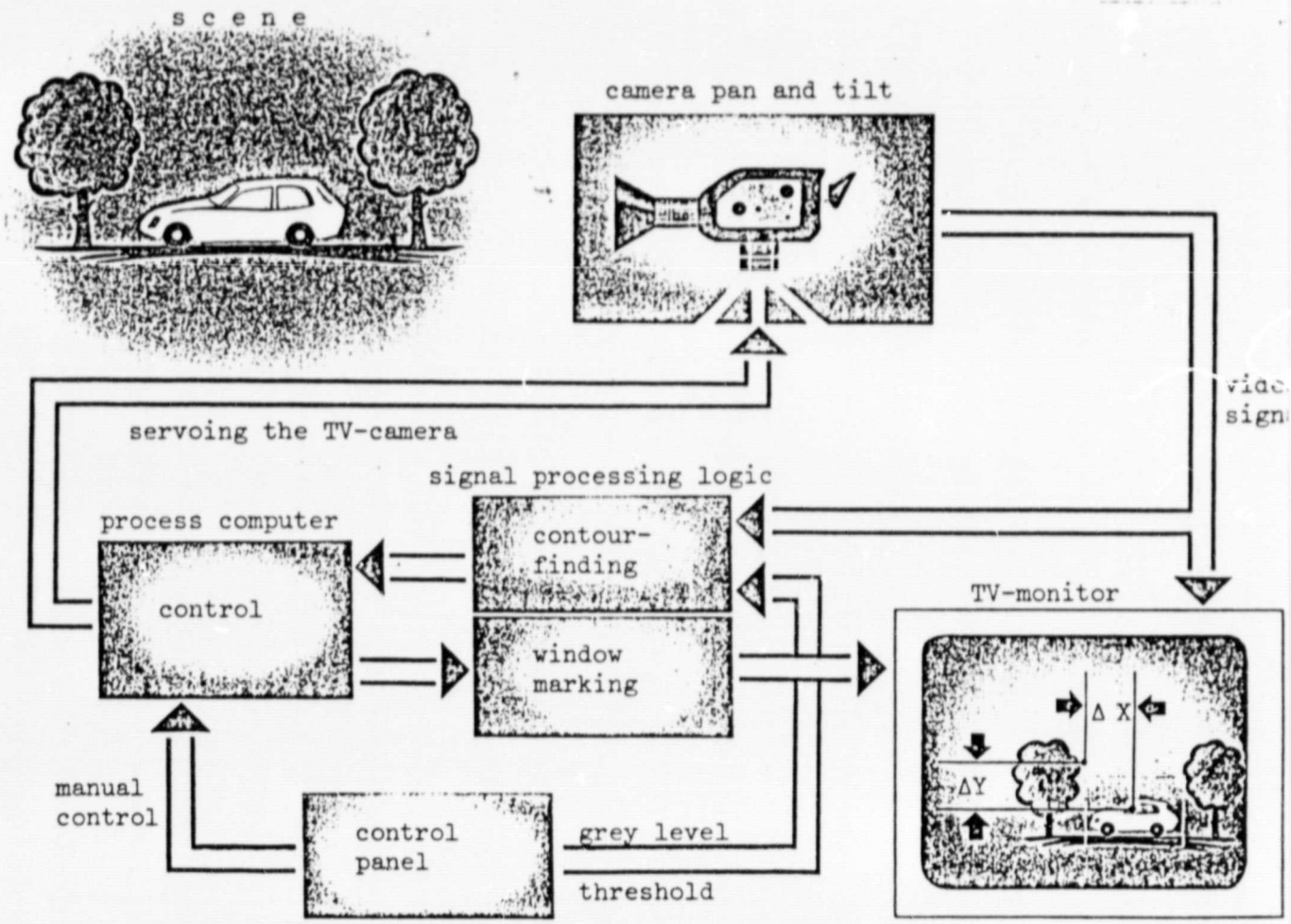


fig. 1 Block structure of the tracking system

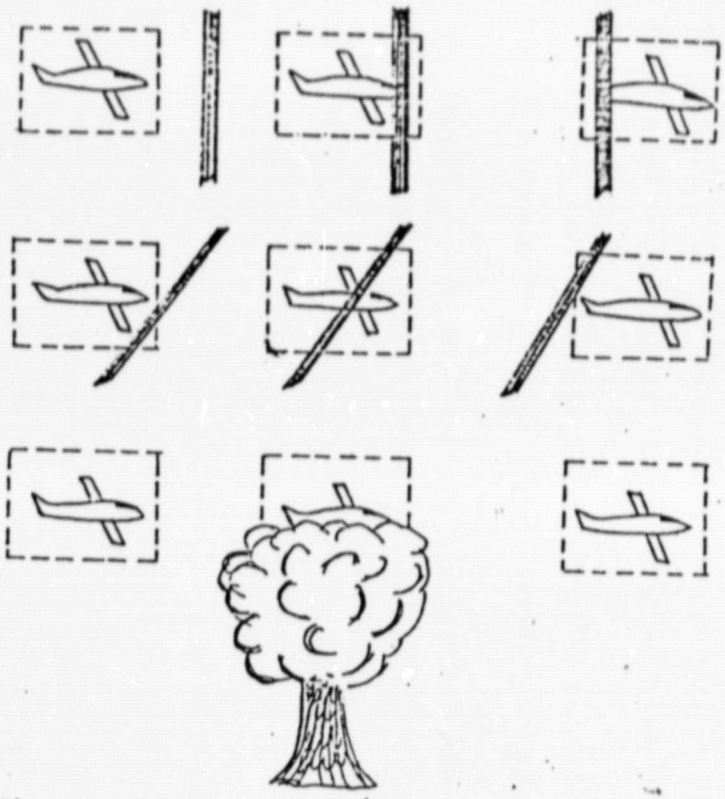
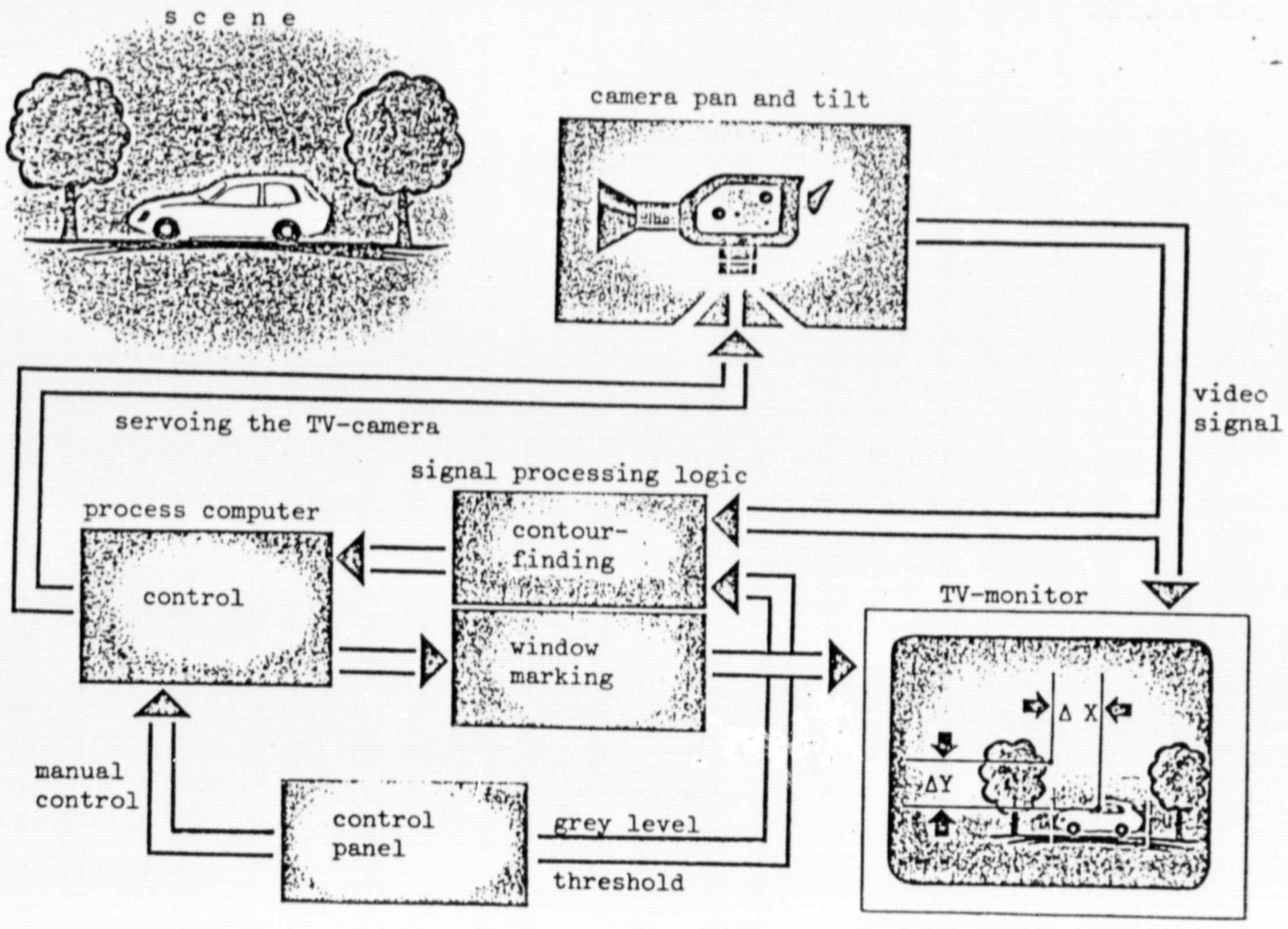


fig. 3 problem cases

ORIGINAL PAGE IS
OF POOR QUALITY



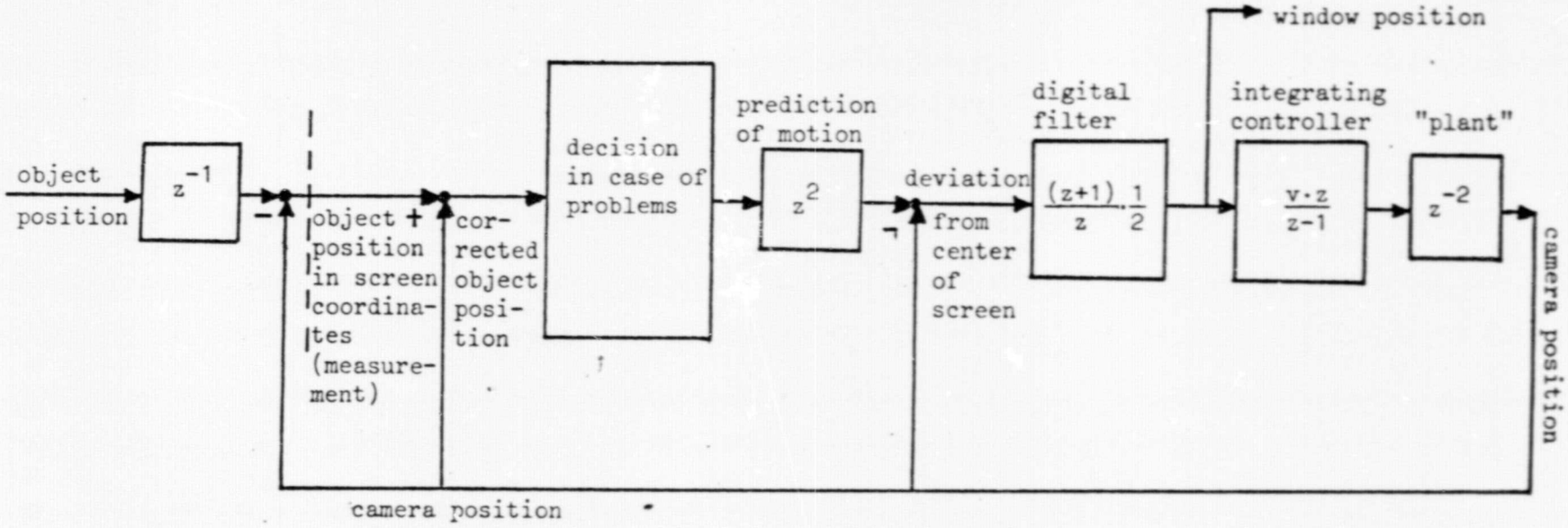


fig. 2 Structure of camera servo loop

APPENDIX C

USE OF THE AAC-32 IN A TRANSVERSAL FILTER APPLICATION

A PROGRAMMABLE TRANSVERSAL FILTER USING BUCKET BRIGADE DEVICE

2 REVIEW OF RELATED TOPICS

2.1 Digital Filter

A digital filter is a computational process in which the sequence of input numbers is converted into a sequence of output numbers representing the alteration of the data in some prescribed manner. The input-output relationship can be mathematically represented as

$$y(n) = \sum_{i=0}^{n} a_i x(n-i) - \sum_{i=1}^{n} b_i y(n-i) \tag{1}$$

where x(n), is the input, y(n) is the output and a_i's and b_i's are constants that determine the characteristics of the filter. If all values of b_i's are zero, then it is a transversal filter.

The realization of a digital filter may involve either hardware or software. In the hardware approach, it is necessary to build digital circuitry which can have the operations of delay, addition and multiplication with appropriate order. In the software approach, the end product might be a program for a computer.

The Z-transformation of equation (1) is

$$Y(Z) + \sum_{i=1}^{n} b_i Z^{-1} Y(Z) = \sum_{i=0}^{n} a_i Z^{-1} X(Z) \tag{2}$$

where Z is a complex variable.

The transfer function H(Z) of transversal filter is defined as

$$H(Z) = \frac{Y(Z)}{X(Z)} = \frac{\sum_{i=0}^{n} a_i Z^{-i}}{1 + \sum_{i=1}^{n} b_i Z^{-i}} \tag{3}$$

To examine the frequency response of the filter, it is just simply to substitute Z for $e^{j\omega T}$ [3]. T is the reciprocal of sampling frequency.

2.2 Bucket Brigade Device

The bucket brigade device is a type of charge transfer device. These devices can move quantities of electrical charge in a controlled manner across a semiconductor substrate by applying a sequence of clock pulses. A fully integrated MOS version of BBD was fabricated in 1970 [4].

The circuit shown in Figure 1 is known as a MOSFET BBD, because the activity of this circuit resembles a fire brigade of old.. Its basis is a chain of storage capacitors and charge-transfer circuits acting as an analog shift register with externally variable shift rate.

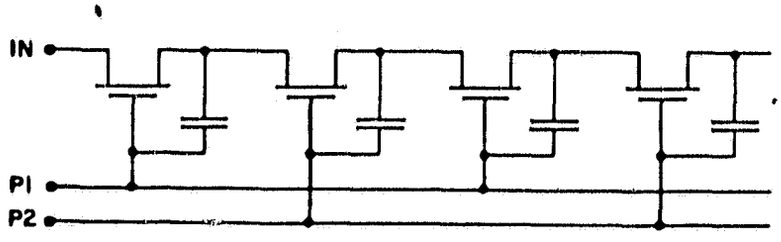


Figure 1 MOSFET BBD

3 THE RETICON AAC-32

The configuration of the AAC-32 is shown in Figure 2. There are two charge-transfer analog delay lines and each line has 32 individual taps. Each pair of corresponding taps is input to a four-quadrant analog multiplier. The outputs of the multipliers are summed on-chip, thus performing sum-of-product operations. Clocks control the shift rate of the sampled analog signal. Each delay line also needs a complementary clock. Two "dead" cells offer the convenience of examining the sampled signal on each delay line. This particular device was evaluated by Snyder [4]. Some useful conclusions are summarized below.

Clock rate. The AAC-32 will function well with clock rates as high as 100KHz.

Four quadrant multiplication test. In the four quadrant multiplication test, it was shown that the device had a tendency to clip in the third quadrant.

ORIGINAL PAGE IS
OF POOR QUALITY

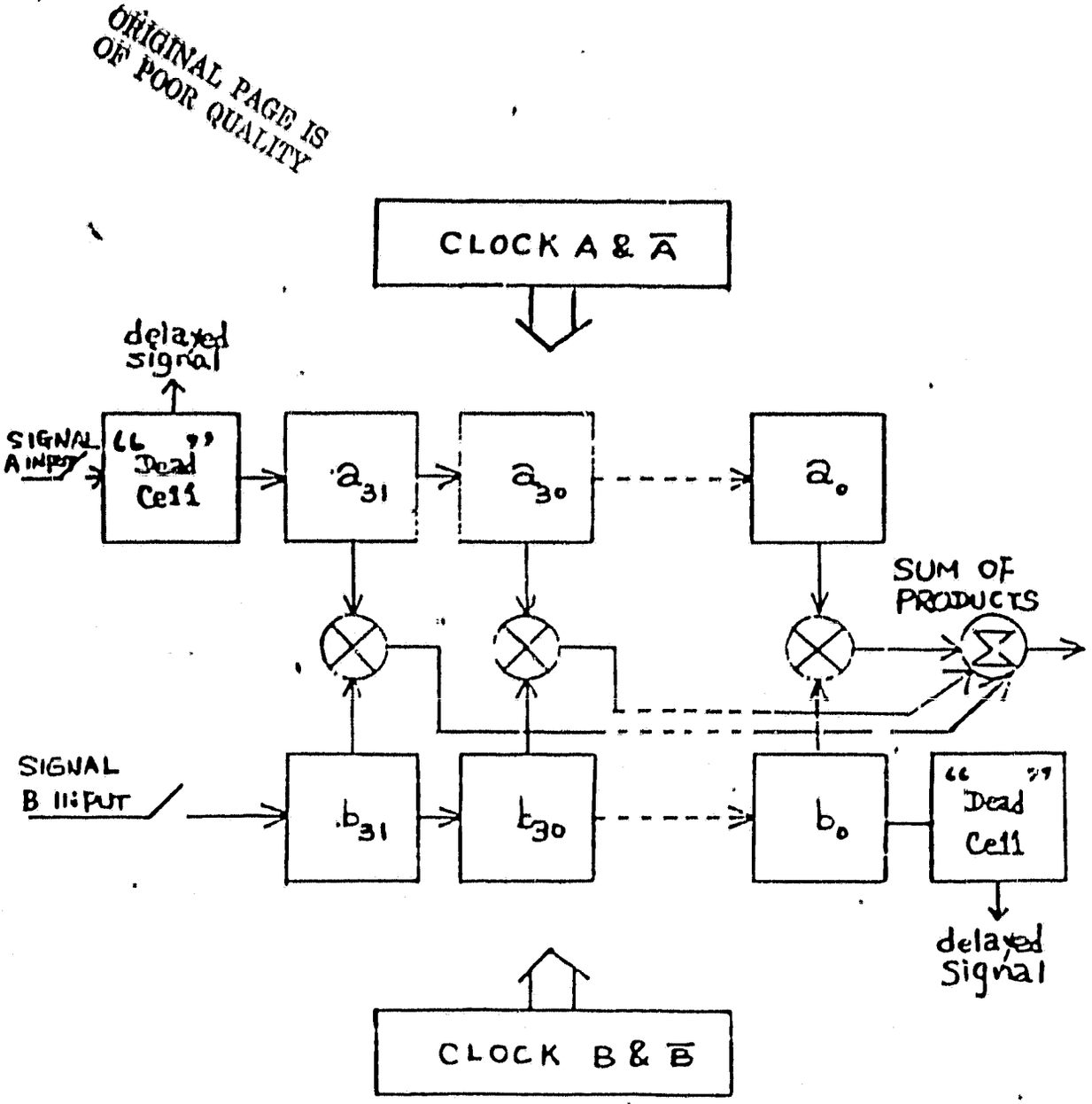


Figure 2. Configuration of the Reticon AAC-32

4 DESIGN AND IMPLEMENTATION

4.1 Design Considerations

4.1.1 Clock frequency

The clock frequency of AAC-32 is essentially the sampling frequency for the input signal. For a baseband input signal in filtering, lower sampling frequency has the better performance. This conclusion comes from the result of computer simulation. In Figure 3, transfer functions of different sampling rates at 60, 80 and 100KHz were simulated and plotted. The transfer function for a sampling rate of 60 KHz has the sharpest cutoff at stopband. It seems good to have a possibly low sampling frequency in this filter design. However, sampling frequency was set at 100 KHz in order to have minimum idle time (A second order distortion was observed to be a function of the time during which the clock was idle).

4.1.2 Tap weights

The transversal filter was designed to be a 10KHz low-pass filter. An 8-bit PROM was programmed to provide weighting coefficients. Computer simulations was completed to examine the effect of quantizing tap weights for this filter design. Since the Fourier series method [5] was used in determining the weighting coefficients, only an odd number of coefficients was required for a finite impulse response filter design. Thus the 32nd tap was set to have value zero. An examination was also made to pick up the first 32 weighting coefficients from the 33-coefficient filter design under the same specifications. A Hamming window [5] has been used in evaluating weighting coefficients in order to overcome the Gibbs' phenomenon [5].

ORIGINAL PAGE IS
OF POOR QUALITY

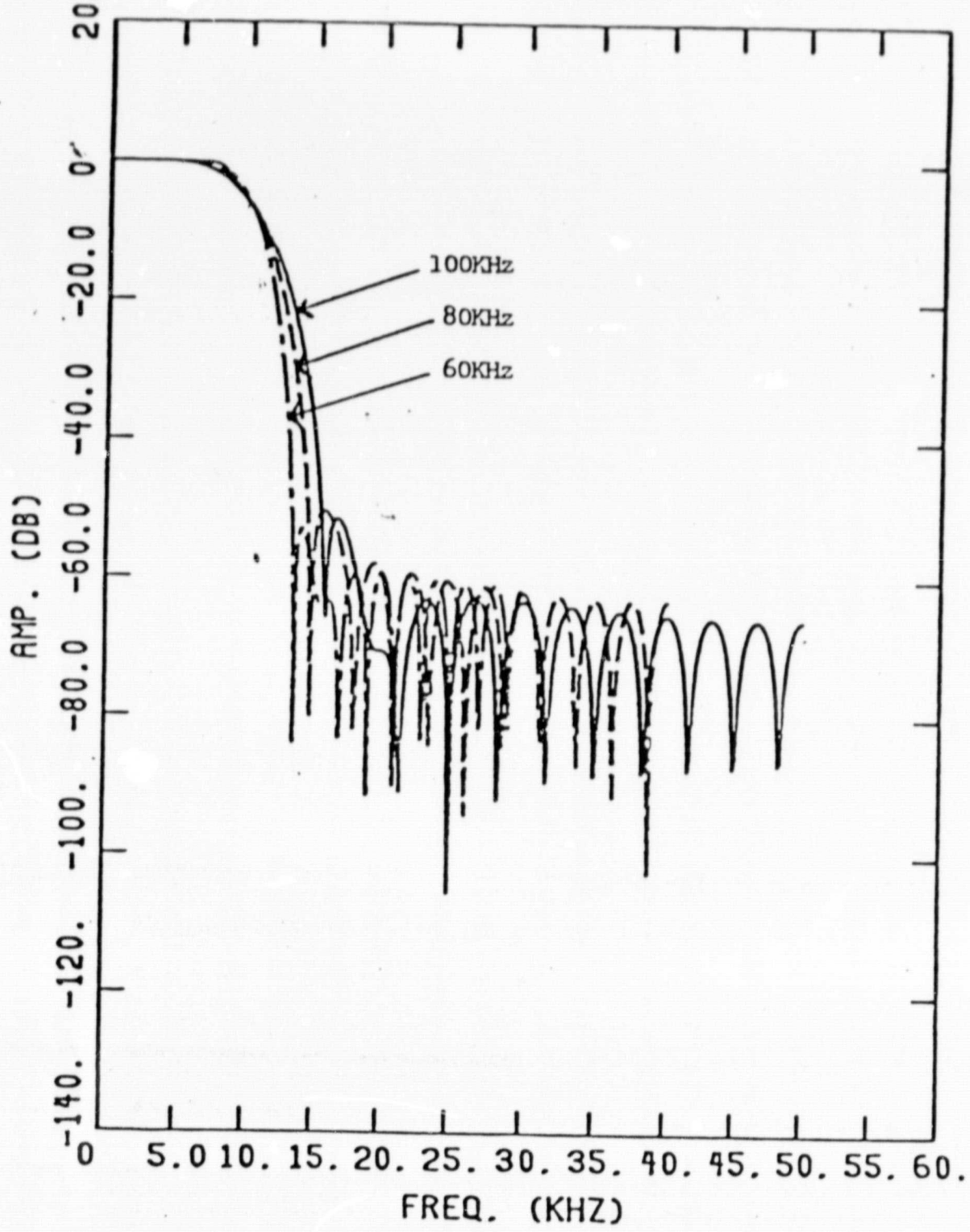


Figure 3. Frequency response of a low-pass filter with different sampling rate 60, 80, 100KHz

The results of simulations are summarized in the following:

The quantization of weighting coefficients has a significant effect on the stopband attenuation. There is about 13 dB difference between the maximum ripple in the stopband (Figure 4). Also the calculated response of the filter having exactly computed weighting coefficient has a sharper cutoff. These conclusions hold for 31- and 32-coefficient filters in this design. Figure 4 is the simulation of a 31-coefficient filter and Figure 5, for a 32-coefficient filter. The dashed line represents the frequency response of the filter with weighting coefficients quantized to eight bits. The solid line gives the frequency response with weighting coefficients computed 32 bits accuracy. Comparing the frequency responses of 31 quantized coefficients' filter with that of 32 quantized coefficients, 31 coefficients' filter has slightly better performance. The 31-coefficient filter will be used in the implementation of a practical filter. Figure 6 shows the frequency responses of these two filters.

4.1.3 Charge transfer inefficiency

The charge transfer inefficiency [6] is defined to be the fraction of charge left in one stage when charge is transferred to the next. The overall charge inefficiency is proportional to the number of stages if the fraction is constant and independent of signal charge. Typically, a device with less than 10000 stages is acceptable [1]. Transfer inefficiency did not appear to affect the performance of the filter.

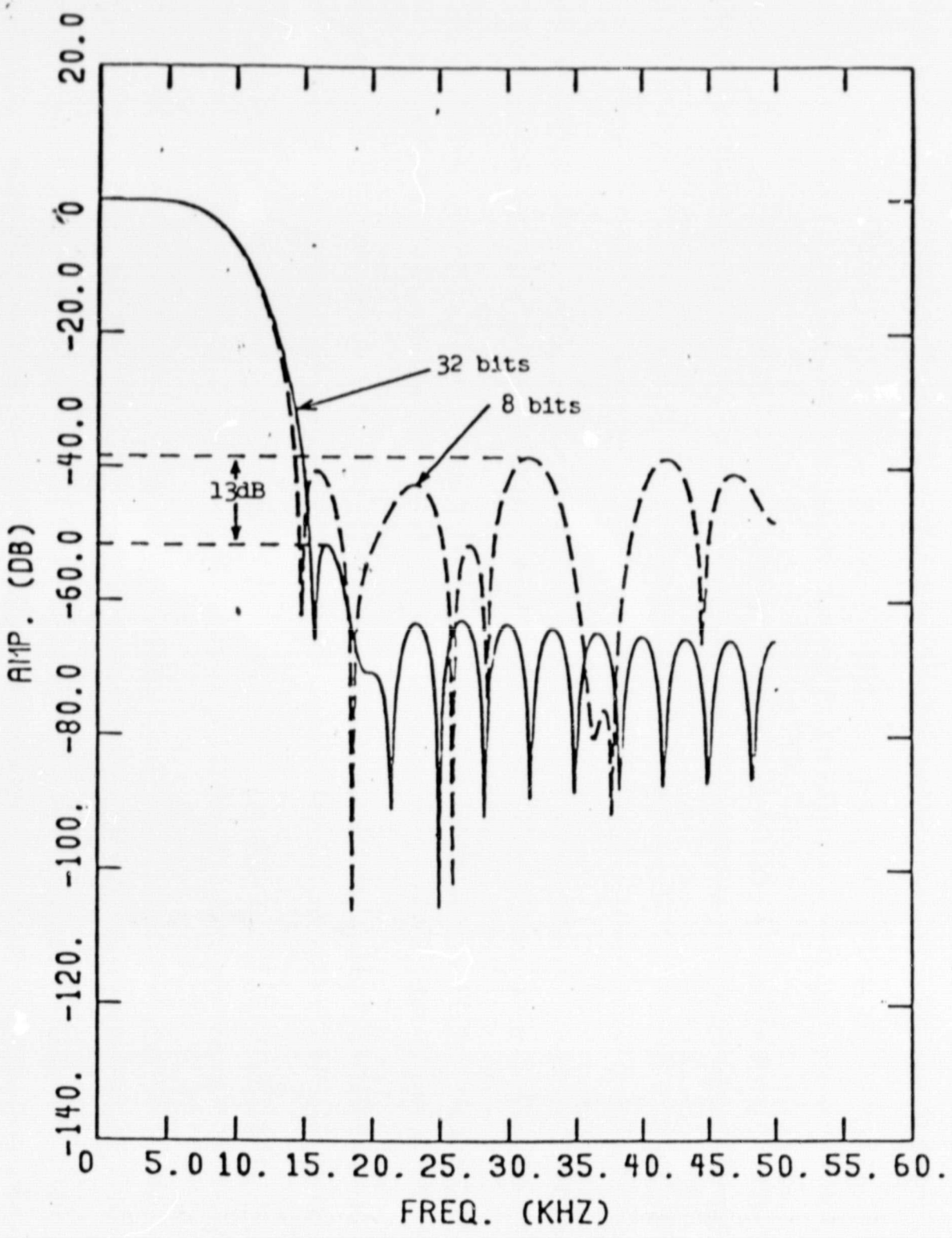


Figure 4. 31-Coefficient low-pass filter

ORIGINAL PAGE IS
OF POOR QUALITY

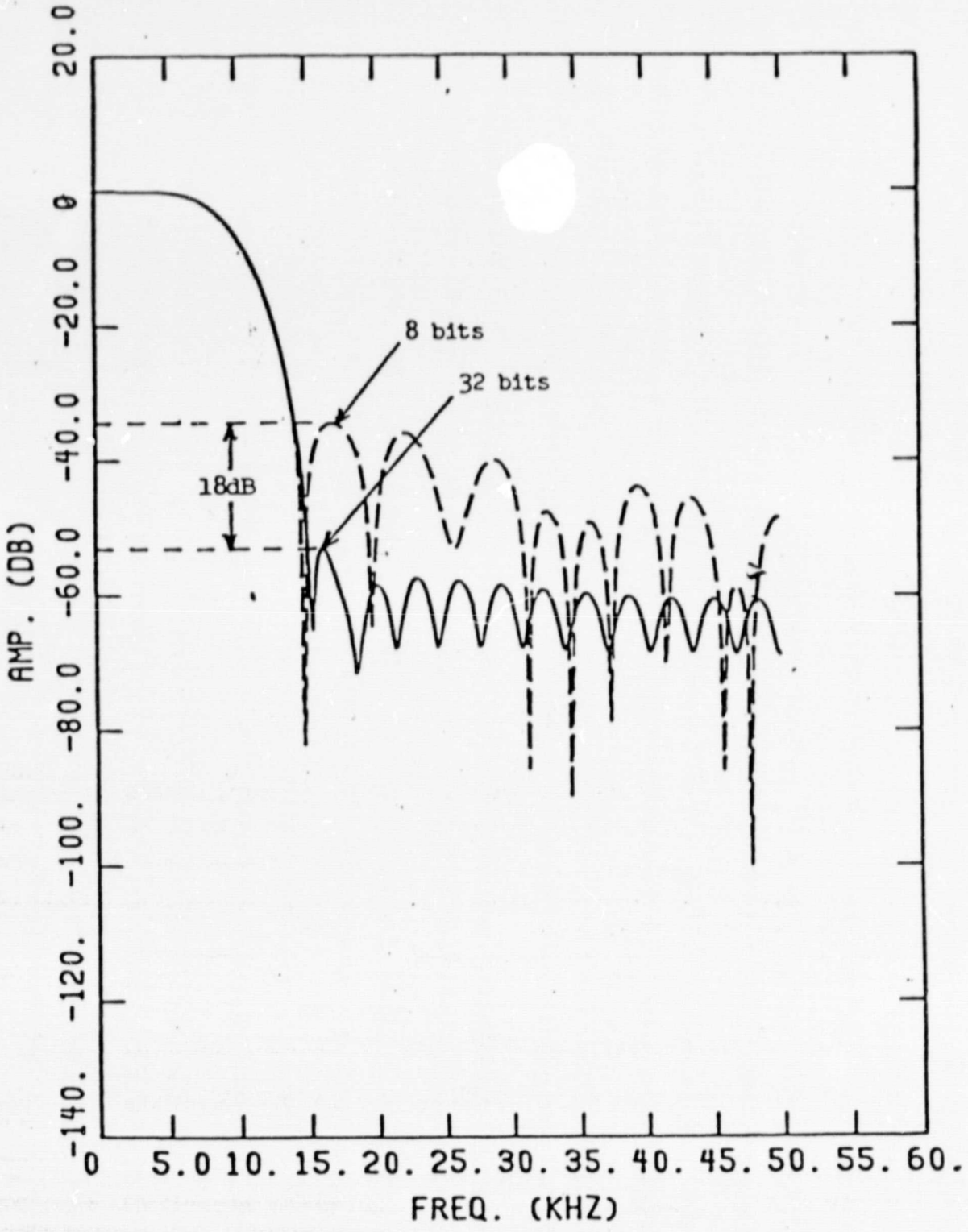


Figure 5. 32-Coefficient low-pass filter

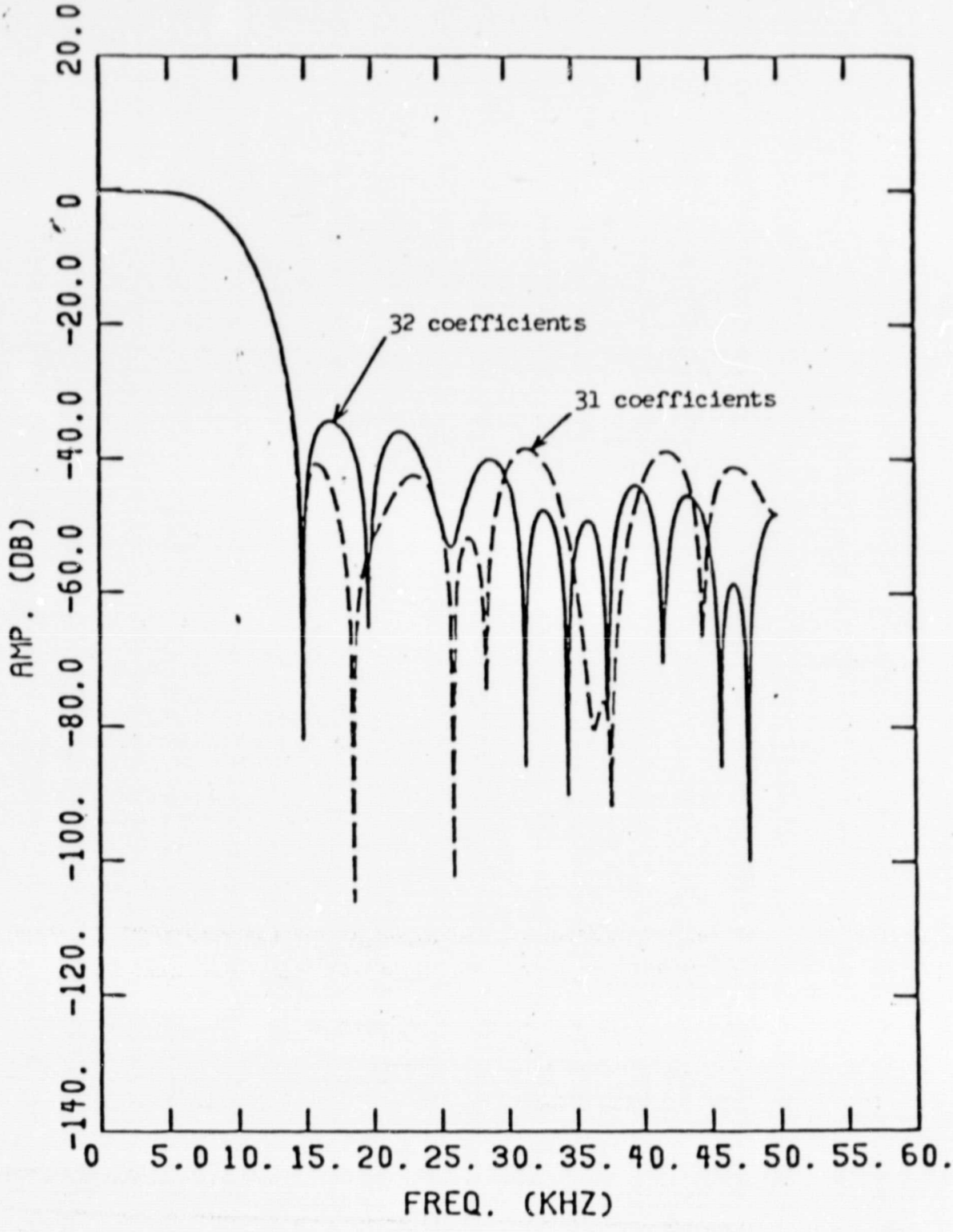


Figure 6. 8-bit transversal filter

4.2 Implementation

4.2.1 Loading weighting coefficients

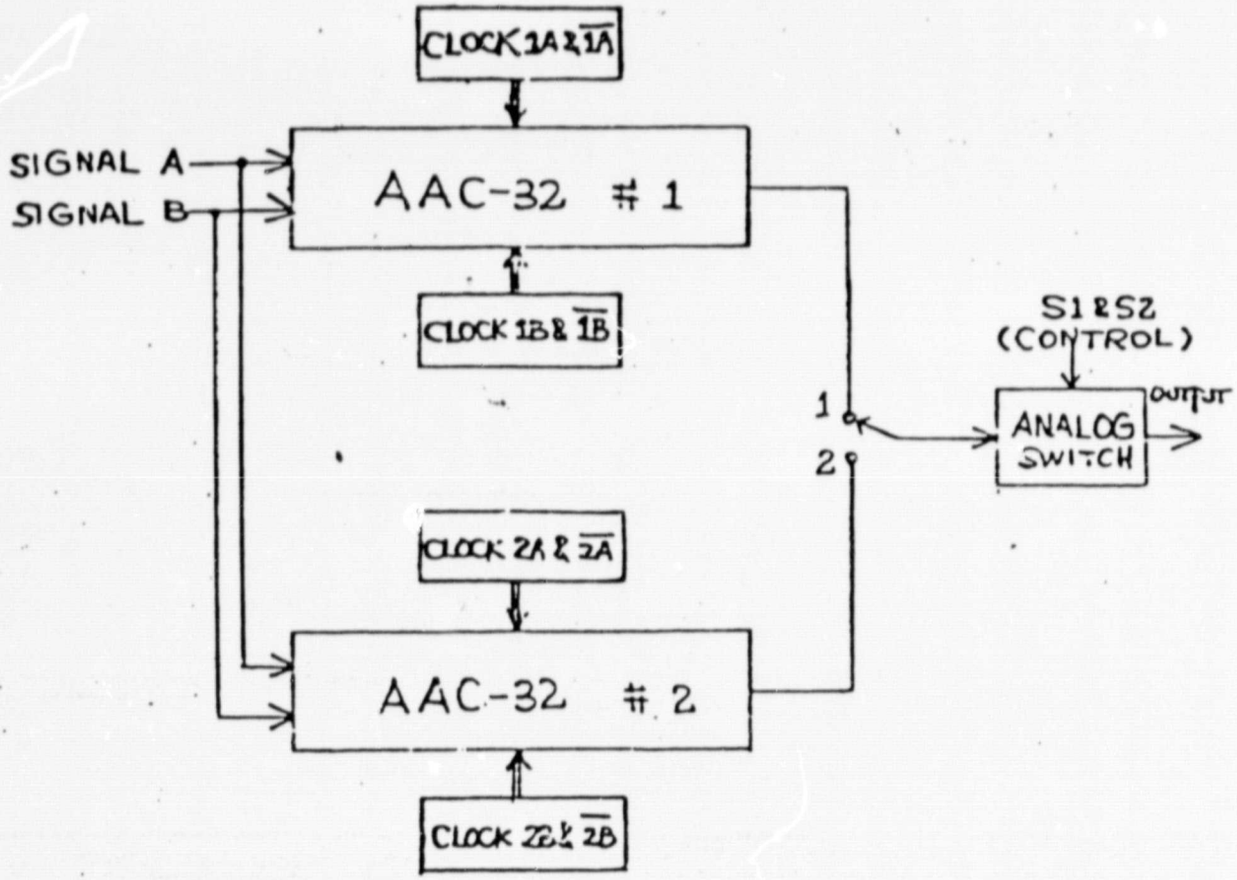
An AAC-32 can compute a sum of products of two sampled analog signals. The output is

$$y = \sum_{i=0}^{31} a_i x_i \tag{4}$$

where y is the analog output and the a_i's and x_i's are the samples of the analog input signals controlled by the external clock. The transversal filter has the following input-output relationship:

$$y(n) = \sum_{i=0}^{n} a_i x(n-i) \ . \tag{5}$$

Here y(n) and x(n) are samples of the input and output signals respectively. The a_i's are weighting coefficients which determine the characteristics of the filter. To use the AAC-32 for the computation of Equation (5), we must keep the a_i's constant while the x_i's are being clocked through the delay line. It would be desirable to simply load the coefficients into the CTD, and leave them alone, while passing the data through the filter. Unfortunately, the coefficients are stored in the form of charge on capacitors, and will decay if not refreshed periodically. A method of refreshing which is general to this class of devices is shown in Figure 7, with timing diagrams in Figure 8. The general concept is one of time multiplexing. The coefficients of one BBD are refreshed while the other is computing, and the roles are reversed every 32 clock cycles. All a_i's are clocked into one delay line of one of these devices and held 320 μsec; that cycle then repeats.



ORIGINAL PAGE IS
OF POOR QUALITY

Figure 7. Configuration of refreshing

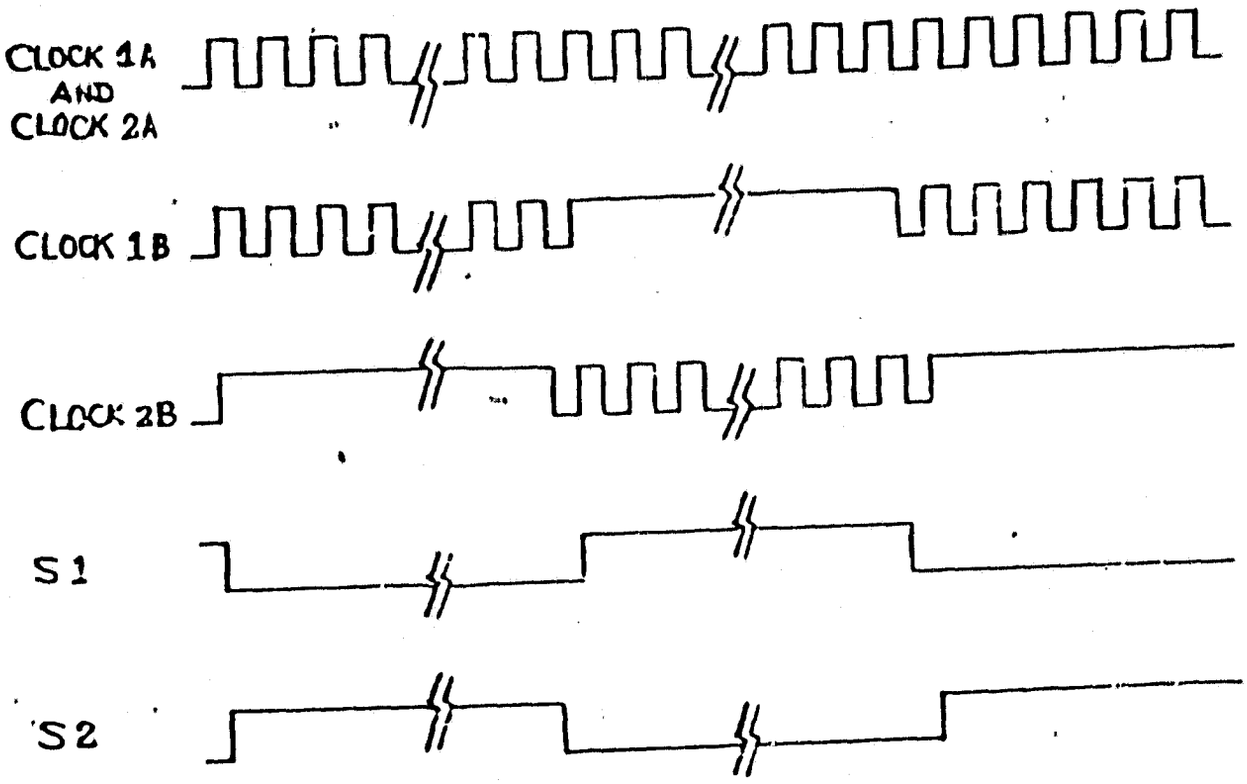


Figure 8. Timing diagrams

(At a sampling frequency of 100KHz, 32 delays results in 320 μsec delay.)

4.2.2 Configuration

The overall configuration of this implementation is illustrated in Figure 9. It can be divided into three main portions. These are control, data processing and data acquisition. Each part will be discussed separately.

4.2.3 Control

AAC-32 requires a two-phase complementary square-wave clock drive. A free-running variable frequency clock provides the time base. Two-phase complementary square-wave clock and the clocking waveforms which control the loading of coefficients in Figure 8 are implemented by using combinational logic as shown in Figure 10.

4.2.4 Data processing

The input signal with 1 volt swing passes through a level shifter to have 'zero' at 5.6v. Since the multipliers have biasing problems in the third quadrant, we are forced to have all signals higher than 5.6v after passing the level shifter. Thus we are actually computing

$$y(n) = \sum_{i=0}^{31} a_i x(n-i) + \sum_{i=0}^{31} a_i c \qquad (6)$$

where c is the input DC offset with respect to 5.6v.

Two AAC-32's were used to process input signals. Since we need 320μsec to refresh weighting coefficients, only 320μsec of processed data of every 640μsec output is valid for each AAC-32. The output circuit is a differential amplifier which is a current-to-voltage converter (Figure 11).

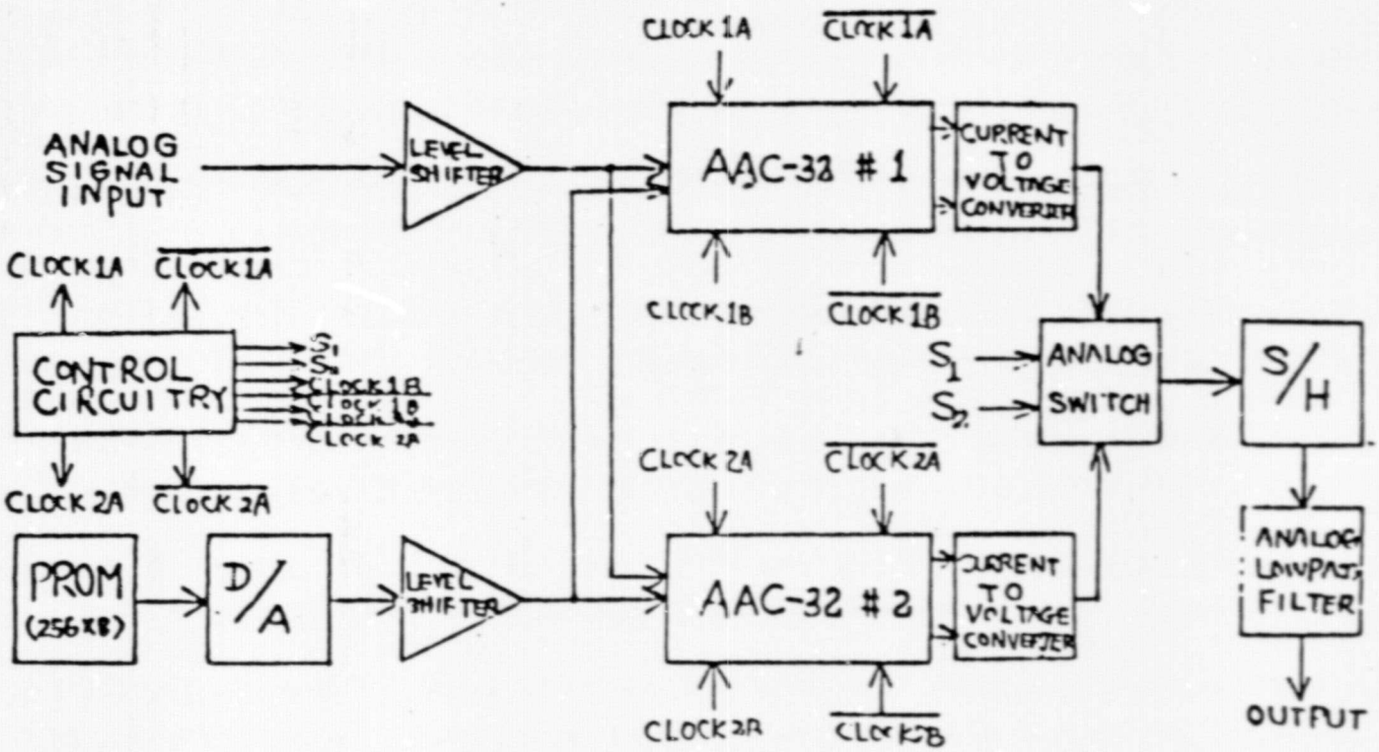


Figure 9. The configuration of overall system

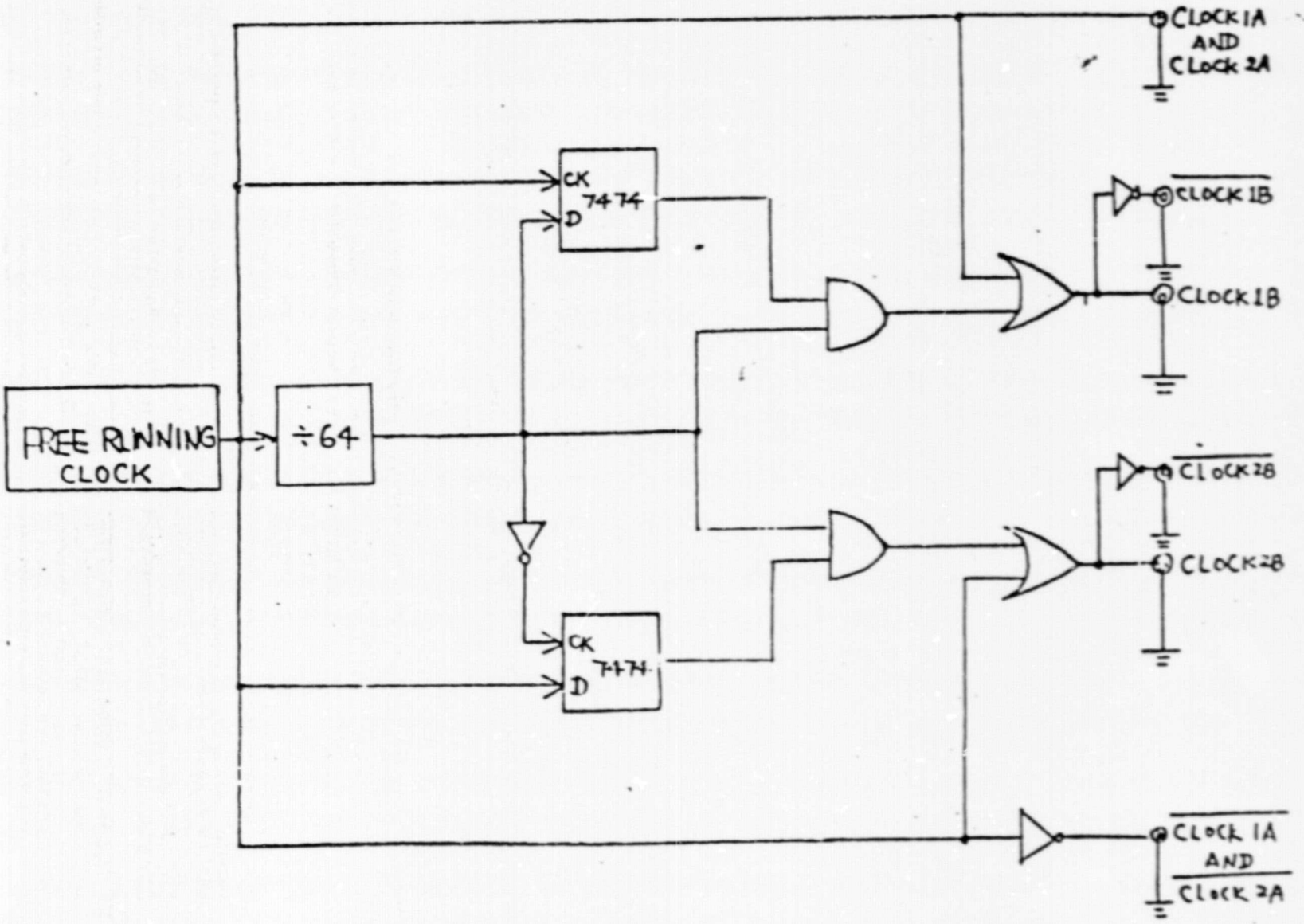


Figure 10. Control circuitry

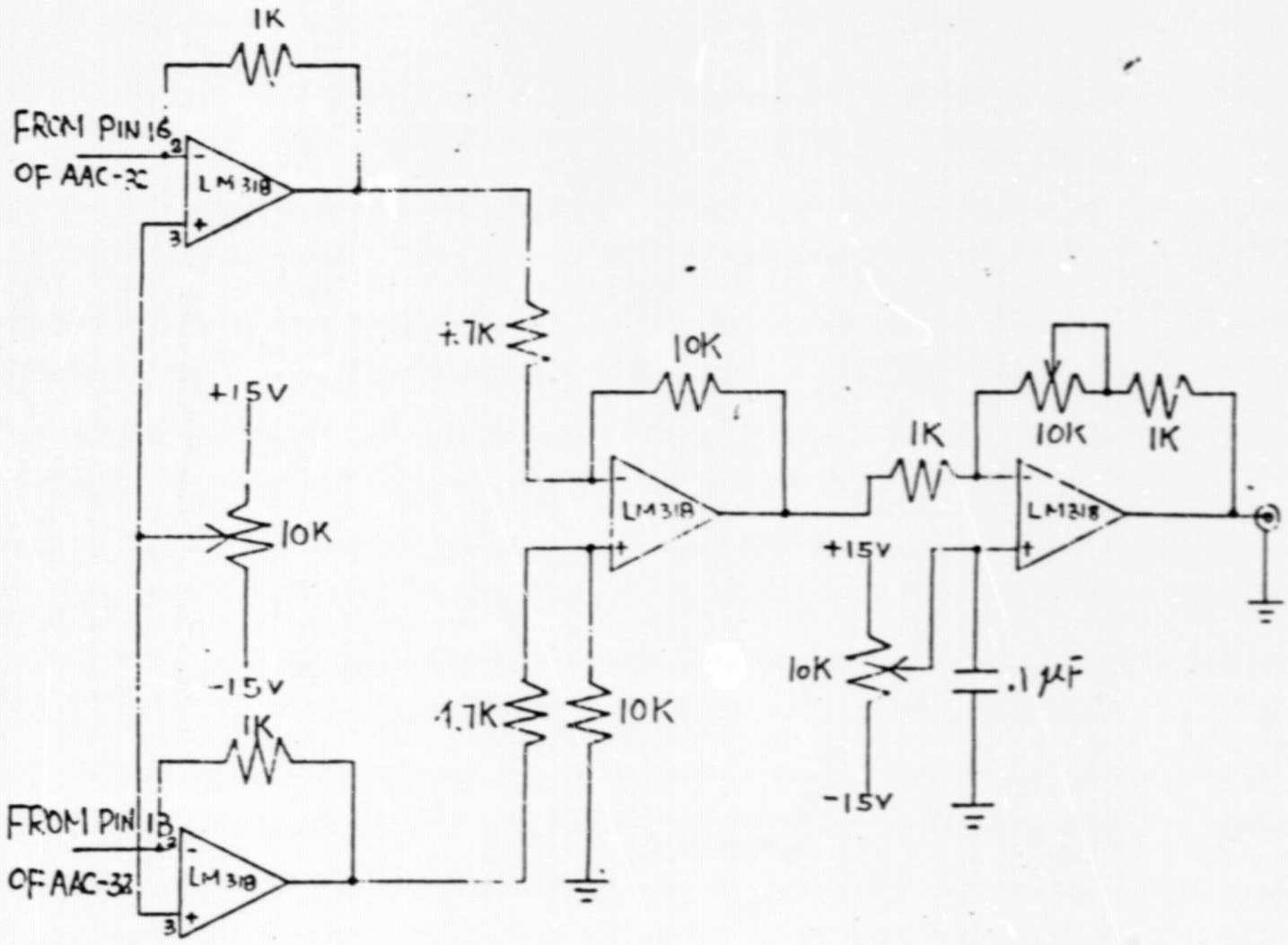


Figure 11. Current-to-voltage converter

4.2.5 Data acquisition

An analog switch is used to select the valid data from the outputs of the current-to-voltage converters. Every 320μsec, the analog switch passes one output and cuts the other. Following the analog switch, a sample and hold circuit is used to hold valid data, because only when both of the driving clocks of the AAC-32 are high is the data valid (refer to Figure 7). The sample pulse is delayed about 3μsec to allow the data to settle. A simple low-pass filter can be connected to the output of the sample and hold circuit to eliminate high frequency component due to clocking.

4.3 Results

Figure 12 shows the frequency response of a low-pass transversal filter compared to the calculated filter response from DC to 14KHz. These two sets of data are quite consistent.

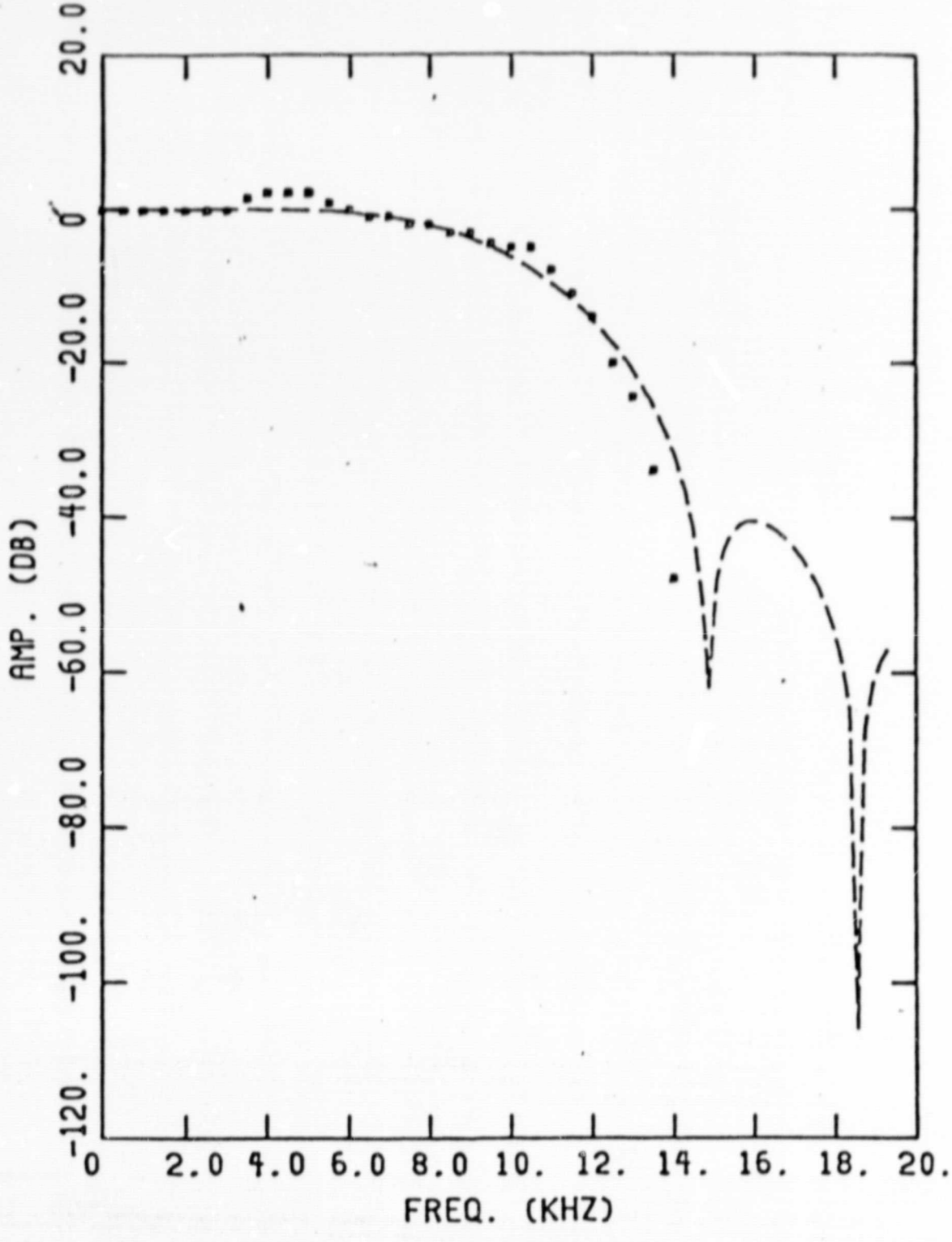


Figure 12. Frequency response plots for 8-bit accuracy 31-coefficient low-pass filter. Boxes represent experimental values for the frequency response

5 SUMMARY AND CONCLUSIONS

The overall operation of a transversal filter can be described as a cross-correlation of the sampled input signal and weighting coefficients. If the weighting coefficients can be treated as another sampled input signal, then a transversal filter is a remarkably versatile instrument for processing signals with different characteristics. A different response can be obtained by simply changing the signal source which generates the weighting coefficients.

This thesis discusses the implementation of a programmable 32-tap charge-transfer transversal filter. The AAC-32 was used as a primary device to implement a low-pass filter with tap weights controlled externally. An 8-bit PROM was programmed to store the tap weights. It was used to generate the coefficients for the designed filter, and the coefficients may be changed simply by replacing the PROM. The amplitude response of this filter is consistent with theoretical prediction.

Since both the weights and signal are stored in charge transfer devices, the weights are volatile and must be refreshed periodically. Additional circuitry is required to accomplish this refreshing.

The complexity and cost in design and implementation of this transversal filter can be reduced significantly if one can have an on-chip static binary shift register or Programmable Read-Only Memory to hold the weights. We understand that semiconductor manufacturers are now developing such a product.

6 LIST OF REFERENCES

1. Buss, D. D., R. W. Brodersen, C. R. Hewes and A. F. Tasch, Jr. 1975. Communication applications of CCD transversal filters. IEEE Press, Charge-coupled devices: Technology and applications, pp. 366-370, 1977.

2. Baertsch, R. D., W. E. Engeler, H. S. Goldberg, C. M. Pockette, IV and J. J. Tiemann. 1976. The design and operation of practical charge-transversal filters. IEEE Trans. Electron Devices, Vol. ED-23, pp. 133-142, Feb. 1976.

3. Rabiner, L. R. and B. Gold. 1975. Theory and application of signal processing. Prentice-Hall, Englewood-Cliffs, NJ.

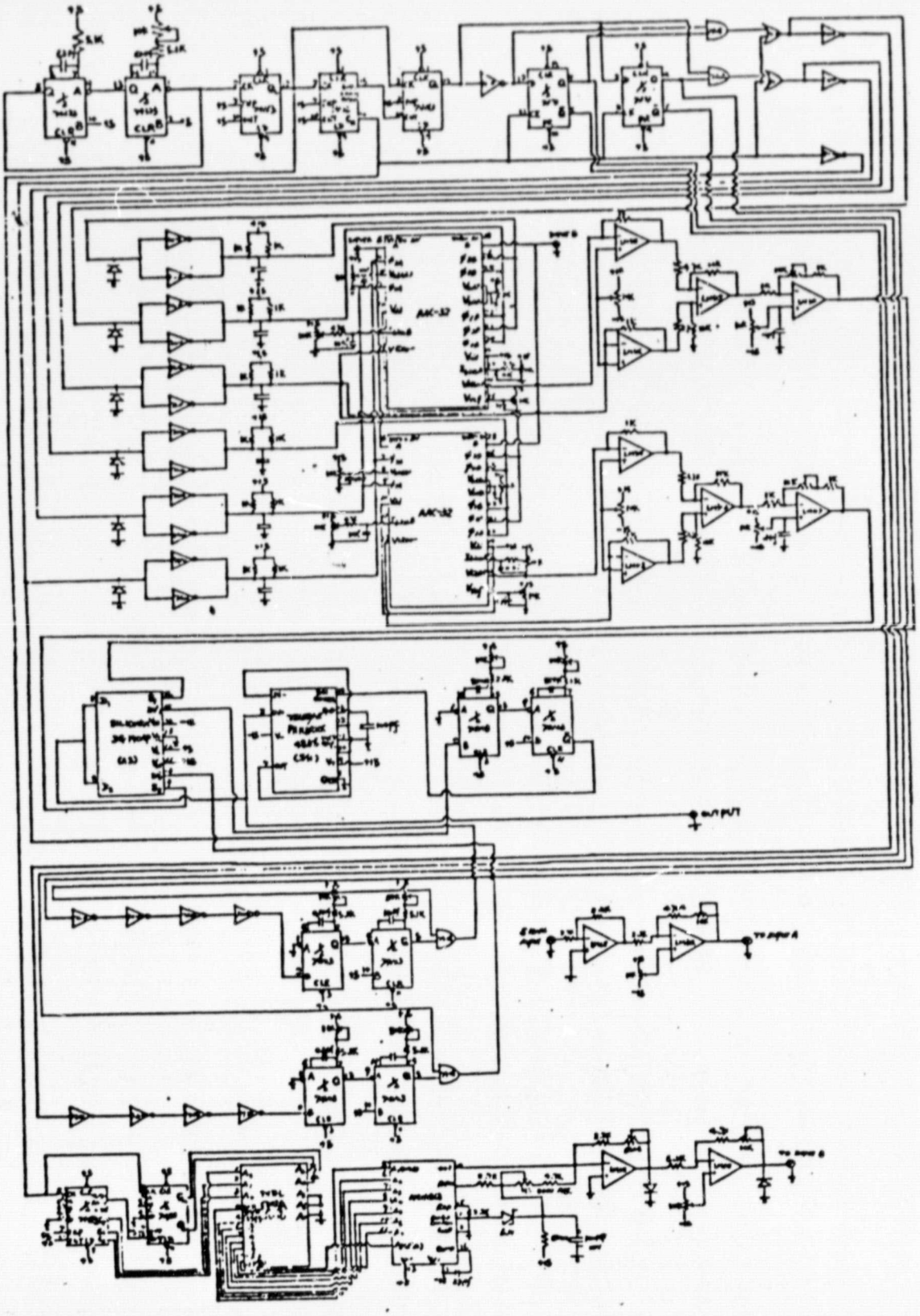
4. Snyder, W. E. 1977. Charge transver devices for remote sensing application. Annual report to NASA. Grant No. NSG 1353.

5. Stanley, W. D. 1975. Digital signal processing. Reston, Virginia.

6. Sequin, C. H. and M. F. Tompsett. 1975. Charge transfer devices. Academic press, NY.

7 APPENDICES

ORIGINAL PAGE IS
OF POOR QUALITY



7.1 Circuitry of the System

7.2 The weighting coefficients of 31-tap low-pass filter for loading Intel 1702A in implementation

Address	Data (Hexadecimal)
00	80
01	80
02	80
03	7F
04	7F
05	7F
06	80
07	82
08	85
09	87
0A	85
0B	80
0C	76
0D	68
0E	5C
0F	51
10	4D
11	51
12	5C
13	68
14	76
15	80
16	85
17	87
18	85
19	82
1A	80
1B	7F
1C	7F
1D	7F
1E	80
1F	80

APPENDIX D

PROGRAM LISTINGS

ORIGINAL PAGE IS
OF POOR QUALITY

```
        .TITLE RAMTSB.MAC VERSION 87

        .MCALL .PRINT, .EXIT

        CSR1=167770             ;DEFINE LOG. NAME FOR CONTROL
        CSR2=167760             ;STATUS REGISTERS
        OUT2=167762             ;DATA INPUT ASSIGNMENT
        IN2=167764              ;DATA OUTPUT ASSIGNMENT
        OUT1=167772             ;OUTPUT CARD 1

START:  MOV     #.,SP           ;INITIALIZE SP
        BIC     #140,@#CSR1     ;CLEAR AFFECTED BITS OF CSR1
        BIC     #140,@#CSR2     ;CLEAR AFFECTED BITS OF CSR2
        BIC     #177777,@#OUT2  ;RESET THE DATA OUTPUT
        CLR     R1              ;INITIALIZE TO ZERO ALL DATA
        BIS     #1,R1           ;SET TO 1 THE 0 BIT OF THE DATA OUTPUT
        JSR     PC,TEST         ;INITIALIZE FOR OUTPUT BITS
        BIS     #2,R1           ;SET TO 1 THE 1 BIT OF THE DATA OUTPUT
        JSR     PC,TEST         ;INITIALIZE FOR OUTPUT BITS
        BIS     #4,R1           ;SET TO 1 THE 2 BIT OF THE DATA OUTPUT
        JSR     PC,TEST         ;INITIALIZE FOR OUTPUT BITS
        BIS     #10,R1          ;SET TO 1 THE 3 BIT OF THE DATA OUTPUT
        JSR     PC,TEST         ;INITIALIZE FOR OUTPUT BITS
        BIS     #20,R1          ;SET TO 1 THE 4 BIT OF THE DATA OUTPUT
        JSR     PC,TEST         ;INITIALIZE FOR OUTPUT BITS
        BIS     #40,R1          ;SET TO 1 THE 5 BIT OF THE DATA OUTPUT
        JSR     PC,TEST         ;INITIALIZE FOR OUTPUT BITS
        BIS     #100,R1         ;SET TO 1 THE 6 BIT OF THE DATA OUTPUT
        JSR     PC,TEST         ;INITIALIZE FOR OUTPUT BITS
        BIS     #200,R1         ;SET TO 1 THE 7 BIT OF THE DATA OUTPUT
        JSR     PC,TEST         ;INITIALIZE FOR OUTPUT
        CLR     R1              ;CLEAN THE DATA OUTPUT
        .EXIT                   ;END OF PROGRAM

TEST:   MOV     R1,@#OUT2       ;OUTPUT THE DATA SET
        MOV     #OUT1,R4        ;STORE DATA OUTPUT ASSIGN INTO R4
        BIC     #100000,@R4     ;ENABLE THE BUS DRIVERS
        CLR     R3              ;INITIALIZE TO ZERO EIGHTS COUNTER
        CLR     R2              ;INITIALIZE TO ZERO UNITS COUNTER
        BIC     #3407,@R4       ;CLEAR ADDRESS BITS
IN1:    JSR     PC,CHECK        ;CALL CHECK SUBROUTINE
        INC     @R4             ;INCREMENT IN 1 THE LOW ADDRESS BITS
        INC     R2              ;INCREMENT THE UNIT COUNTER
        CMP     R2,#7           ;TEST FOR COUNT TO 7
        BLE     IN1             ;BRANCH TO ROUTINE
;
        CMP     #7,R3           ;CHECKS TOP OF HIGH ADDRESS BITS
        BEQ     NEXT            ;EXIT THE PROGRAM IF COUNT IS FINISHED
        BIC     #7,@R4          ;CLEAR THE LOW BIT ADDRESS
        ADD     #400,@R4        ;INCREMENT 1 UNIT OF HIGH BIT ADDRESS
        INC     R3              ;INCREMENT THE EIGHTS COUNTER
        CLR     R2              ;INITIALIZE TO ZERO UNITS COUNTER
        JMP     IN1             ;INITIALIZE COUNT WITH LOW ADDRESS
```

```
NEXT:   .PRINT  #MES2           ;INDICATE END OF TESTING OF ONE BIT
        CLR     R1              ;RESET VALUES OF THE OUTPUT BITS
        RTS     PC              ;RETURN TO ROUTINE TESTING THE BITS
;
CHECK:  BIC     #50000,@R4      ;WRITE TO THE RAM
        BIS     #10000,@R4      ;READ THE RAM
        CMPB    R1,@#IN2        ;COMPARE THE INPUT AND ECHOED BITS
        BEQ     RET             ;IF DIFFERENT WRITE MESSAGE
        .PRINT  #MESSAG
        BIC     #177777,@#OUT2  ;RESET THE OUTPUT DATA
        .EXIT
RET:    RTS     PC              ;RETURN VIA PC
;
MESSAG: .ASCIZ  /APPARENT ERROR IN RAM/
MES2:   .ASCIZ  /FINISHED ONE BIT/
;
        .END    START           ;MAKE PROGRAM SELF STARTING
```

ORIGINAL PAGE
OF POOR QUALITY

```
;
;
;       RAM TEST PROGRAM
;
;
        .MCALL .PRINT, .EXIT
;THIS IS A SHORT VERSION OF RAM TESTING PROGRAM.
;IT PASS ALL 1'S TO RAM AND CHECK THE RAM OUTPUTS.
;THEN IT PASS ALL 0'S TO RAM AND CHECK THE RAM OUTPUTS.
;INITIALIZATION:
        .PSECT
        OP1=167772
        OP2=167762
        IP2=167764
START:  MOV     #.,SP           ;LOAD STACK POINTER
        BIC     #140,@#167770   ;SET CSR OF PORT 1
        BIC     #140,@#167760   ;SET CSR OF PORT 2
        MOV     #50000,R3       ;SET CONTROL FORMAT
        MOV     #50000,R4       ;SET CONTROL FORMAT
        CLR     @#OP1           ;CLEAR OUTPUT PORT 1
        CLR     @#OP2           ;CLEAR OUTPUT PORT 2
BEGIN:  JMP     TEST            ;BEGIN TESTING
CK:     CMP     #54000,R4       ;COMPARE THE REQUIRED NO. OF TEST
        BNE     BEGIN           ;BRANCH IF NOT FINISH
RETURN: .PRINT  #TXT2           ;INDICATE FINISH TEST
        .EXIT                   ;EXIT TO MONITOR
TEST:   CLR     @#OP1           ;CLEAR OUTPUT PORT 1
        CLR     @#OP2           ;CLEAR OUTPUT PORT 2
        MOV     R3,@#OP1        ;PUT CONTROL FORMAT AT OUTPUT PORT 1
        MOV     #377,@#OP2      ;LOAD DATA TO OUTPUT PORT 2
        BIC     #40000,@#OP1    ;ENABLE RAM
        BIC     #10000,@#OP1    ;PUT RAM AT WRITE MODE
        BIS     #10000,@#OP1    ;DISABLE RAM AT WRITE MODE
        BIS     #40000,@#OP1    ;DISABLE RAM
        MOV     R4,@#OP1        ;PUT CONTROL FORMAT AT OUTPUT PORT 1
        BIC     #40000,@#OP1    ;ENABLE RAM
        CMPB    #377,@#IP2      ;COMPARE THE ECHOED DATA
        BEQ     FIRST           ;BRANCH IF NO ERROR
        .PRINT  #TXT            ;OTHERWISE INDICATE ERROR TYPE 1
FIRST:  CLR     @#OP1           ;CLEAR OUTPUT PORT 1
        CLR     @#OP2           ;CLEAR OUTPUT PORT 2
        MOV     R3,@#OP1        ;PUT CONTROL FORMAT AT OUTPUT PORT 1
        MOV     #0,@#OP2        ;LOAD DATA TO PUTPUT PORT 2
        BIC     #40000,@#OP1    ;ENABLE RAM
        BIC     #10000,@#OP1    ;PUT RAM AT WRITE MODE
        BIS     #10000,@#OP1    ;DISABLE RAM AT WRITE MODE
        BIS     #40000,@#OP1    ;DISABLE RAM
        MOV     R4,@#OP1        ;PUT CONTROL FORMAT AT OUTPUT PORT 1
        BIC     #40000,@#OP1    ;ENABLE RAM
        CMPB    #0,@#IP2        ;COMPARE THE ECHOED DATA
        BEQ     SECOND          ;BRANCH IF NO ERROR
        .PRINT  #TXT1           ;OTHERWISE INDICATE ERROR TYPE 2
```

```
SECOND: CMPB    #7,R3           ;CHECK LOWEST 3 ADDRESS BITS
        BNE     GO              ;BRANCH IF NOT FULL
        JMP     THIRD           ;OTHERWISE JUMP TO THIRD
GO:     INC     R3              ;NEXT TEST LOCATION
        INC     R4              ;
        JMP     CK              ;BACK TO THE MAIN PROGRAM
THIRD:  BIC     #7,R3           ;RESET LOWEST 3 ADDRESS BITS
        BIC     #7,R4           ;
        ADD     #400,R3         ;INCREMENT HIGER 3 ADDRESS BITS
        ADD     #400,R4         ;
        JMP     CK              ;BACK  TO THE MAIN PROGRAM
TXT:    .ASCIZ  /ERROR: TYPE 1: CHECK RAM/
TXT1:   .ASCIZ  /ERROR: TYPE 2: CHECK RAM/
TXT2:   .ASCIZ  /FINISH TESTING/
        .END    START
```

```
;
;
;       ZERDA PROGRAM
;
;
;THIS IS A PROGRAM TO ADJUST THE ZERO OF A SIDE OF BBD.
;S/H STROBE KEEPS HIGH ALL THE TIME.
;THE SIGNAL INPUTS TO B SIDE OF BBD HAS SAWTOOTHED PATTERN.
;THE PROGRAM CAN BE TERMINATED VIA CONSOLE BY USE 'THE COMMAND CONTROL C.
;
;INITIALIZATION:
;
        .PSECT
        OP1=167772
        OP2=167762
;
;LOADING RAM WITH '0' AT LOCATION 0:
;
START:  MOV     #.,SP           ;LOAD STACK POINTER
        BIC     #140,@#167770   ;SET CSR OF PORT 1
        BIC     #140,@#167760   ;SET CSR OF PORT 2
        MOV     #150000,R2      ;STORE THE INITIAL CONTROL FORMAT FOR LODING
        MOV     R2,@#OP1        ;CONTROL FORMAT FOR LOADING RAM
        MOV     #376,@#OP2      ;LOAD RAM WITH '0'
        BIC     #40000,@#OP1    ;ENABLE RAM
        BIC     #10000,@#OP1    ;INPUT DATA TO RAM
        BIS     #10000,@#OP1    ;
        BIS     #40000,@#OP1    ;DISABLE RAM
;
;INPUT DATA TO BBD AND ADJUST THE ZERO OUTPUT ON A/I BOARD:
;
LOOP:   MOV     #130330,R2      ;INITIAL CONTROL WORD TO LOAD DATA TO BBD
        MOV     #177000,@#OP2   ;LOAD DATA '0' TO D/A CONVERTER
        JSR     PC,DATA         ;OUTPUT DATA FROM RAM TO D/A CONVERTER
                                ;AND LOAD DATA TO BBD
        MOV     #1400,@#OP2     ;LOAD DATA '+1/2' TO D/A CONVERTER
        JSR     PC,DATA         ;
        MOV     #177400,@#OP2   ;LOAD DATA '+1' TO D/A CONVERTER
        JSR     PC,DATA         ;
        MOV     #1400,@#OP2     ;LOAD DATA '+1/2' TO D/A CONVERTER
        JSR     PC,DATA         ;
        MOV     #177000,@#OP2   ;LOAD DATA '0' TO D/A CONVERTER
        JSR     PC,DATA         ;
        MOV     #1000,@#OP2     ;LOAD DATA '-1/2' TO D/A CONVERTER
        JSR     PC,DATA         ;
        MOV     #0,@#OP2        ;LOAD DATA '-1' TO D/A CONVERTER
        JSR     PC,DATA         ;
        MOV     #1000,@#OP2     ;LOAD DATA '-1/2' TO D/A CONVERTER
        JSR     PC,DATA         ;
        BR      LOOP            ;CONTINUE THE PROCESS
;
```

```
;SUBROUTINE TO LOAD DATA TO BBD:
;
DATA:   BIS     #10030,R2       ;PULL UP THE CLOCK 1A & 1B
        MOV     R2,@#OP1        ;
        BIS     #40,@#OP1       ;PULL UP THE MASTER CLOCK
        BIC     #40,@#OP1       ;PULL DOWN THE MASTER CLOCK
        BIC     #30,R2          ;PULL DOWN THE CLOCK 1A & 1B
        MOV     R2,@#OP1        ;
        BIS     #40,@#OP1       ;PULL UP THE MASTER CLOCK
        BIC     #40,@#OP1       ;PULL DOWN THE MASTER CLOCK
        RTS     PC              ;RETURN VIA PROGRAM COUNTER
;
        .END    START
```

ORIGINAL PAGE IS
OF POOR QUALITY

```
;
;
;       ZEROB PROGRAM
;
;
;THIS IS A PROGRAM TO ADJUST THE ZERO OF B SIDE OF BBD.
;S/H STROBE KEEPS HIGH ALL THE TIME.
;THE SIGNAL INPUTS TO A SIDE OF BBD HAS SAWTOOTHED PATTERN.
;THE PROGRAM CAN BE TERMINATED VIA CONSOLE BY USE THE COMMAND CONTROL C.
;
;INITIALIZATION:
;
        .PSECT
        OP1=167772
        OP2=167762
;
;LOADING RAM:
;
START:  MOV     #.,SP           ;LOAD STACK POINTER
        BIC     #140,@#167770   ;SET CSR OF PORT 1
        BIC     #140,@#167760   ;SET CSR OF PORT 2
        MOV     #150000,R2,     ;CONTROL FORMAT FOR LOADING RAM
        MOV     R2,@#OP1        ;
        MOV     #376,@#OP2      ;LOAD RAM LOCATION ZERO WITH '0'
        JSR     PC,LOAD         ;
        INC     R2              ;NEXT LOCATION
        MOV     R2,@#OP1        ;
        MOV     #374,@#OP2      ;LOAD RAM WITH '+1/2'
        JSR     PC,LOAD         ;
        INC     R2              ;NEXT LOCATION
        MOV     R2,@#OP1        ;
        MOV     #0,@#OP2        ;LOAD RAM WITH '+1'
        JSR     PC,LOAD         ;
        INC     R2              ;NEXT LOCATION
        MOV     R2,@#OP1        ;
        MOV     #374,@#OP2      ;LOAD RAM WITH '+1/2'
        JSR     PC,LOAD         ;
        INC     R2              ;NEXT LOCATION
        MOV     R2,@#OP1        ;
        MOV     #376,@#OP2      ;LOAD RAM WITH '0'
        JSR     PC,LOAD         ;
        INC     R2              ;NEXT LOCATION
        MOV     R2,@#OP1        ;
        MOV     #375,@#OP2      ;LOAD RAM WITH '-1/2'
        JSR     PC,LOAD         ;
        INC     R2              ;NEXT LOCATION
        MOV     R2,@#OP1        ;
        MOV     #377,@#OP2      ;LOAD RAM WITH '-1'
        JSR     PC,LOAD         ;
        INC     R2              ;NEXT LOCATION
        MOV     R2,@#OP1        ;
        MOV     #375,@#OP2      ;LOAD RAM WITH '-1/2'
        JSR     PC,LOAD         ;
```

```
;
;INPUT DATA TO BBD AND ADJUST THE ZERO OUTPUT ON A/I BOARD:
;
        MOV     #150300,@#DP1   ;DISABLE RAM
        MOV     #400,@#DP2      ;LOAD (X-U)'S D/A WITH '0'
        MOV     #130330,R2      ;INITIALIZE CONTROL WORD TO LOAD DATA TO BBD
LOOP:   BIS     #10030,R2       ;PULL UP THE CLOCK 1A & 1B
        MOV     R2,@#DP1        ;
        BIS     #40,@#DP1       ;PULL UP THE MASTER CLOCK
        BIC     #40,@#DP1       ;PULL DOWN THE MASTER CLOCK
        BIC     #30,R2          ;PULL DOWN THE CLOCK 1A & 1B
        MOV     R2,@#DP1        ;
        BIS     #40,@#DP1       ;PULL UP THE MASTER CLOCK
        BIC     #40,@#DP1       ;PULL DOWN THE MASTER CLOCK
        INC     R2              ;INCREMENT RAM ADDRESS
        CMPB    #310,R2         ;CHECK LOWEST 3 ADDRESS BITS
        BNE     LOOP            ;CONTINUE TESTING
        BIC     #10,R2          ;ELSE RESET LOWEST 3 ADDRESS BITS
        BP      LOOP            ;CONTINUE TESTING
;
;SUBROUTINE TO LOAD RAM:
;
LOAD:   BIC     #40000,@#DP1    ;ENABLE RAM
        BIC     #10000,@#DP1    ;INPUT DATA TO RAM
        BIS     #10000,@#DP1    ;
        BIS     #40000,@#DP1    ;DISABLE RAM
        RTS     PC              ;RETURN VIA PROGRAM COUNTER
;
        .END    START
```