NASA CR-132463
ERIM 195800-25-F

MIDAS, PROTOTYPE MULTIVARIATE INTERACTIVE DIGITAL ANALYSIS SYSTEM — PHASE I

Volume I: System Description

by

F. J. Kriegler, et al.
Infrared and Optics Division

ENVIRONMENTAL RESEARCH INSTITUTE OF MICHIGAN
FORMERLY WILLOW RUN LABORATORIES, THE UNIVERSITY OF MICHIGAN

August 1974

prepared for

NATIONAL AERONAUTICS AND SPACE ADMINISTRATION

Langley Research Center
Hampton, VA 23665
Contract No. NAS1-11979

TECHNICAL REPORT STANDARD TITLE PAGE

1. Report No.	2. Government Accession No.	3. Recipient's Catalog No.
NASA CR-132463		

4. Title and Subtitle	5. Report Date
MIDAS, PROTOTYPE MULTIVARIATE INTERACTIVE DIGITAL ANALYSIS SYSTEM — PHASE I VOLUME I: SYSTEM DESCRIPTION	August 1974
	6. Performing Organization Code
7. Author(s) F. J. Kriegler et al.	8. Performing Organization Report No. 195800-25-F
9. Performing Organization Name and Address Infrared and Optics Division Environmental Research Institute of Michigan P. O. Box 618 Ann Arbor, MI 48107	10. Work Unit No. 11. Contract or Grant No. NAS 1-11979
12. Sponsoring Agency Name and Address National Aeronautics and Space Administration Washington, D.C. 20546	13. Type of Report and Period Covered Final Report, October 1972 through February 1974 14. Sponsoring Agency Code

15. Supplementary Notes

Volume I of III

16. Abstract

The MIDAS System is a third-generation, fast, multispectral recognition system able to keep pace with the large quantity and high rates of data acquisition from present and projected sensors. MIDAS, for example, can process a complete ERTS frame in forty seconds and provide a color map of sixteen constituent categories in a few minutes. A principal objective of the MIDAS Program is to provide a system well interfaced with the human operator and thus to obtain large overall reductions in turn-around time and significant gains in throughput. This goal is elaborated in this report as an objective of the Phase II program.

This report describes the hardware and software generated in Phase I of the overall program. The system contains a mini-computer to control the various high-speed processing elements in the data path and a classifier which implements an all-digital prototype multivariate-Gaussian maximum likelihood decision algorithm operating at 2×10^5 pixels/sec. Sufficient hardware has been developed to perform signature extraction from computer-compatible tapes, compute classifier coefficients, control the classifier operation, and diagnose operation.

Volume I describes the MIDAS System in detail; Volume II contains the diagnostic programs used to test MIDAS' operation; Volume III displays the MIDAS construction and wiring diagrams.

17. Key Words	18. Distribution Statement
Wirewrap Real-time processing A/D-D/A conversion MIDAS Multispectral recognition system Multivariate-Gaussian maximum likelihood decision algorithm	Initial distribution listed at the end of this report Unclassified-Unlimited

19. Security Classif. (of this report)	20. Security Classif. (of this page)	21. No. of Pages	22. Price
UNCLASSIFIED	UNCLASSIFIED	172	

PREFACE

A comprehensive multispectral program devoted to the advancement of state-of-the-art techniques for remote sensing of the environment has been a continuing program at the Environmental Research Institute of Michigan (ERIM), formerly the Willow Run Laboratories of The University of Michigan. The basic objective of this multidisciplinary program is to develop remote sensing as a practical tool to provide the user with processed information quickly and economically.

The importance of providing timely information obtained by remote sensing to such people as the farmer, the city planner, the conservationist, and others concerned with problems such as crop yield and disease, urban land studies and development, water pollution, and forest management must be carefully considered in the overall program. The scope of our program includes: (1) extending the understanding of basic processes; (2) discovering new applications; (3) developing advanced remote-sensing systems;(4) improving fast automatic data processing systems to extract information in a useful form; and also (5) assisting in data collection, processing, analysis and ground truth verification. The MIDAS program applies directly to No. (4) with its improved data processing capability.

This document is the final report for Phase I of the MIDAS program under NASA Contract NAS1-11979 and covers the period from October 1972 through February 1974. The contract effort was monitored by Mr. William Howle of NASA-Langley. The overall program is guided by Mr. R. R. Legault, a Vice President of ERIM and Director of the Infrared and Optics Division. Work on this contract was directed by J. D. Erickson, Head of the Multispectral Analysis Section and by F. J. Kriegler, Principal Investigator. The ERIM number for this report is 195800-25-F.

ERIM personnel who contributed to this project and who co-authored this report are Dempster Christenson, Michael Gordon, Roland Kistler, Seymour Lampert, Robert Marshall, and Rowland McLaughlin. In addition to providing the text, their individual contributions were as follows: Dempster Christenson and Michael Gordon provided system programming and diagnostic software; Roland Kistler and Seymour Lampert provided the detailed design and performed system checkout; Robert Marshall aided in overall system configuration; Rowland McLaughlin organized this report. The authors wish to acknowledge the direction provided by Mr. R. R. Legault and Dr. J. D. Erickson. Outstanding contributions were made by the following persons: John Baumler, Clyde Connell, William Juodawlkis, Robert Pierson, Cary Wilson, and Nancy Wilson for their efforts in system construction.

CONTENTS

FIGURES

TABLES

MIDAS, PROTOTYPE MULTIVARIATE INTERACTIVE DIGITAL ANALYSIS SYSTEM — PHASE I

Volume I: System Description

1 INTRODUCTION

1.1 SUMMARY

MIDAS, which stands for Multivariate Interactive Digital Analysis System, represents a breakthrough in the field of multispectral scanner image analysis by providing a low-cost capability for user-oriented, interactive, near-real-time, digital analysis to produce thematic mapping with instantaneous or multi-temporal data. MIDAS accepts data from multispectral scanners in the form of high-density digital tape, computer-compatible tape, or analog tape, and makes use of proven multispectral processing techniques (including signature extension) within an innovative hardware approach resulting in a cost-effective, user-controlled system for multispectral analysis and recognition. Its hardware and software are intended to require a minimum amount of instructional training for successful operation. MIDAS is intended to provide multispectral analysis for applications in disciplines such as agriculture, urban planning, forestry, geology, pollution detection, hydrology, and others. Features may be extracted that are spectral, spatial, temporal, and (possibly) polarization-dependent, thus giving a very general and powerful capability.

This report describes the status achieved at the end of the first phase of a two-phase, 2-1/2 year program to demonstrate the unique advantages of a special-purpose multispectral processor. In this machine the parallel digital implementation capabilities of a low-cost processor are combined with a mini-computer to achieve near-real-time operation of a complete processing system that includes multiple, user-selectable, preprocessing functions and color displays. The following sections describe the organization of the processor, its design details, its software design and implementation, and the further steps necessary in Phase II to make the system more complete. In addition, Phase II will demonstrate the effectiveness of this innovative approach in contrast with that of using large general-purpose computers presently employed in the Earth Observations Program.

Two additional volumes are considered to be a part of this final report, Volume II describing the diagnostics used with the system to maintain error free operation and Volume III giving hardware design details.

1.2 A RATIONALE FOR MIDAS

A particularly important and easily overlooked aspect of applying a remote sensing, multispectral system to aid in mapping crops, detecting pollution, or locating some ecological disturbance is that of processing the data to provide the proper information to someone in a time

short enough to meet his needs. Unfortunately, ongoing programs do overlook this aspect of the system design problem, for reasons which are rarely clear.

Knowing the utility of these techniques, one would think that the data, gathered as it is at very high rates and over very large areas, must be processed before it can be useful. And when the time allowed to produce such results is relatively short, as it invariably is, then it becomes clear that a major problem exists and requires a solution.

The magnitude of the discrepancy between the ability of a sensor to gather this data and the ability of a general purpose computer to process it becomes the next aspect of the problem to be assessed. This enormity can probably be best appreciated by considering a brief numerical example. An airborne scanner will, typically, gather data over a 20- to 30-mile flight line in about 15 minutes on one reel of magnetic tape. A general-purpose digital computer can classify this data in a time about 1000 times as long. Thus, the data collected in 15 minutes will require 15,000 minutes of processing. This would amount to 6 weeks of processing, assuming a 40-hour week. Given one such computer to process this data, this aircraft could only be used for eight 15-minute sorties per year. Clearly, the discrepancy in capabilities is unacceptable.

At this point, one must examine the problem more carefully. We should point out that the above discrepancy of 1000 to 1 is actually rather conservative, possibly by as much as a factor of 10 but editing of the data should alleviate the problem to this extent. We should also point out that this form of processing yields a map containing the color-coded identification of each element of the scene at the time of the overflight, but also containing errors in this color-coding and geometrical errors resulting from the sensor and motions of the sensor platform. And it is important to note that the scene is not at all unchanging as one flies along: ground conditions vary, the atmosphere varies, the sensor stability varies and, in short, there is a spatial-temporal uncertainty about the data. Processing, as it becomes better understood, will require even more computation to remove these errors and to improve the results finally provided. A current rationale for this is given by Erickson [1].

From another point of view, the objective of providing greater speed also includes much lower cost. This is true not only for the operational situation but also for research and development. Given the limited resources for research, more investigations can be conducted. It has been estimated that processing costs in an operational system can be reduced by about a factor of 20 or more from some processing costs based on present day feasibility. For example, the typical data set described earlier could be processed at a cost of less than $400 instead of present-day typical costs of about $8000.

Another important objective is to facilitate control by the user. Classification of remotely sensed data is an interactive process in which the man and machine must, in fact, be considered

as the real processing system. This is not apparent in some processing systems since the machine is so slow that an operator is easily able to keep pace with the task. In a new system in which the processor is substantially faster, the time required by an operator is three or four times longer than that taken for processing. Increases in processor speed will then provide little improvement in throughput. It thus becomes evident that well designed, interactive display and control subsystems will, in reality, offer the greatest gains in throughput. As should also then be evident, the development process must inherently involve an evolutionary refinement of the display and control interface. This, to some extent, is an experimental task initially addressed in the second phase of this development program. It is a task which can be bypassed only at the risk of seriously impaired throughput. This task also implies that the system should be considered as a prototype from which further refined systems may be derived for specific programs. To allow this user tailoring of the operational system he requires, modular design becomes an important aspect.

One final motivation exists in addressing this problem: Proper configuration of an operating system almost always requires experience in the use of a system similar to the one required.

One must, therefore, see this problem as one of great and, indeed, critical importance to the objective of making remote sensing a useful, cost-effective tool. It is thus necessary to pursue a course which, in the development of these methods, provides assurance that processing methods are available to meet the demands of those who would use them.

The present system was conceived and constructed as a demonstration that a better processing system will materially assist in practical, economic realization of the benefits of remote sensing.

1.3 CAPABILITY AT THE END OF PHASE I

Phase I has resulted in the generation of the classifier hardware, an elementary software system, and the integration of these with the general purpose digital computer. This system, although far from the system envisaged for Phase II, offers the capability of performing classifications at the rate of $5 \cdot 10^{-6}$ seconds per pixel vector for a 4-channel, 16-category (or 8-channel, 8-category) problem. This capability has been demonstrated using CCT input of ERTS data from Yellowstone National Park.

1.3.1 HARDWARE

The hardware system (Fig. 1) includes the classifier, the controller, and the diagnostic subsystem. The classifier is that portion of the system which accepts the input data-vector and performs the pipeline computations ending in the classification of the vector into one of 16 classes.

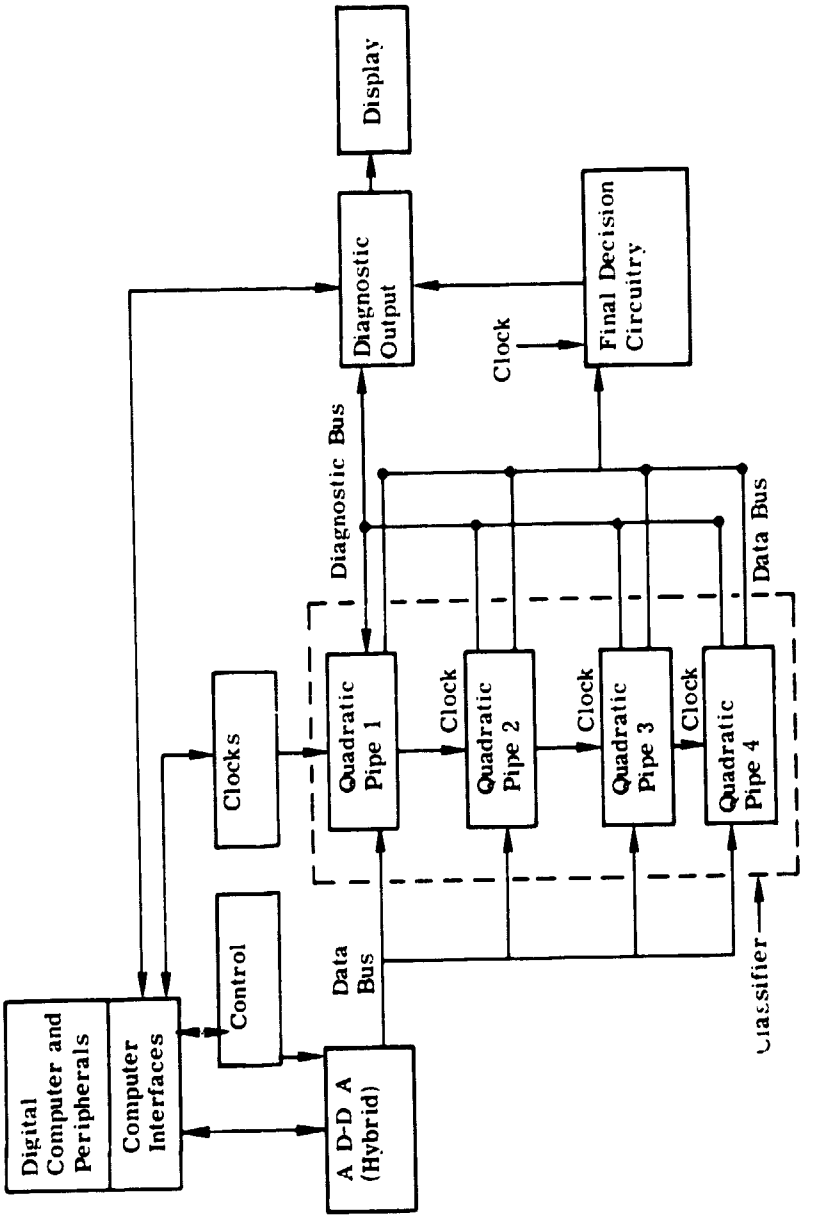


FIGURE 1. BLOCK DIAGRAM OF THE PHASE I MIDAS SYSTEM

The controller subsystem selects the input channels to go to the classifier or to the computer, routes control and data vectors to the internal storage of the classifier from the computer, and routes data from the classifier to the computer or displays. This also includes computer interfacing and its control.

The diagnostic subsystem operates in conjunction with the classifier as a multi-port bus able to access several key modules in the pipeline to allow this data to be provided to test equipment or to the computer. Using this subsystem, debugging and troubleshooting can be done under software control, allowing fault isolation to one or two cards.

1.3.2 SOFTWARE

Software capability at this time includes a set of programs operating under the DEC (Digital Equipment Corporation) DOS (Disk Operating System) monitor; among these are wire-wrap and diagnostic programs, input and output control programs, and analysis programs.

The wire-wrap programs have been written to allow card input control of a Gardner-Denver wire-wrap machine and are currently in the form of the original machine wiring, as yet unchanged to conform to final wiring after debugging.

Diagnostic programs include card test programs and programs both to exercise the arithmetic circuitry of MIDAS and to isolate faults in MIDAS.

The card test programs are employed to isolate component faults in the event of failure or to test a card after wiring and before insertion to verify its correct operation. Test jigs are used to interface these cards to the computer.

The diagnostic programs include subsystem or subassembly diagnostics and an accuracy analysis package. These programs allow overall or subassembly test and fault isolation.

The analysis programs allow data to be input to the computer and analyzed into multivariate parameters for the determination of classifier setup. These also properly scale the parameters to suit the arithmetic circuitry of the classifier.

Finally, the required parameters must be output to the classifier in properly sequenced tables along with addresses of memory locations in the classifier.

1.3.3 SYSTEM INTEGRATION

Interfacing with the DEC PDP-11 45 computer in the MIDAS System has been accomplished via unibus interface cards standard with DEC. The system as such has been tested using simulated data and actual ERTS data. Simulation tests verify the overall correct operation of the system as an arithmetic device controllable by its software. Tests using ERTS data have verified that the MIDAS will accept ERTS data and properly classify it. Only a scaling problem to

optimize the dynamic range of the fixed-point arithmetic remains to be corrected. This correction will serve to improve the performance for some unusual multivariate distributions.

The MIDAS is, of course, not a full system at this point and could not be used in processing except in special cases: it presently requires the use of other (non-contract) equipment for display, printing, location and editing of training sets. These deficiencies are to be eliminated in Phase II.

2
CONCLUSIONS AND RECOMMENDATIONS

2.1 CONCLUSIONS FROM PHASE I

(1) The principal objective of Phase I was to demonstrate that an all-digital classifier was a feasible and economical device capable of being controlled by a mini-computer and used to process remotely sensed data from multispectral sources. Phase II is intended to make the system more complete and provide the means both to process such data in a short time employing close matching of the system and the human operator and to verify that the design is effective for this purpose.

(2) The basic hardware and software elements of MIDAS have been designed, assembled, tested and operated. The system performs as specified, accepting data in a burst mode at a rate of 5×10^{-6} seconds per pixel vector and providing classified output at that rate. Somewhat higher performance may be obtained, if necessary, but design limits have not yet been explored. As stated above, the present rate is 200×10^3 pixels/second and could probably be raised to about 300×10^3 pixels/second. Higher rates would require more parallelism or greater length in the pipelines. These approaches could allow operation, using the same components in a redesigned architecture, out to about 10^6 pixels/second. Above this level it appears that the parallelism, the wiring techniques, and the MSI technology would begin to create limiting problems requiring a different technology and greater cost in terms of the product of dimensions and classification rate. Thus the approximate limit using the above technologies seems to be about 16×10^6 channels/second, or a bit rate of 128×10^6 ($8 \times 16 \times 10^6$) bits/second entering the preprocessor, or 64×10^6 ($8 \times 8 \times 10^6$) dimensions/second. As a result, this technology seems well suited to process data from more advanced sensors, now anticipated, such as EOS, etc, operating in the 100×10^6 bit/second range. The processing rate of MIDAS will be $25.6 \cdot 10^6$ ($16 \times 8 \times 2 \times 10^5$) bits/second in Phase II.

It is of interest to compare this technology with others by way of an example as a means of assessing the performance of representative systems and the importance of some aspects of these systems. The times required to process an ERTS frame into 12 categories is shown in Table 1. The times shown are for the five major steps in the processing of data. The first step is the production of a graymap from the data. This can be done quickly if the data source is fast such as a high density tape (HDT) and a special printer such as an ink jet printer is used rather than a conventional line printer. The second step, that of field location can be the most time consuming, and this is where the largest gain in throughput can be made. By making proper use of an HDT, CRT display, and ink jet display, the time required may be reduced from the present forty hours to four hours. The HDT can quickly load a disk with an ERTS frame and can display it at low resolution on a color CRT. Zoom capability and fast interactive color enhancement can be used as presently set up for producing an enhanced color hard copy with an ink jet

TABLE 1. COMPARISON OF PROCESSING TIMES FOR VARIOUS SYSTEMS

PROBLEM: CLASSIFY 12 CATEGORIES OVER AN ERTS FRAME (4 TAPES CCT)
(Times in Hours)

	Initial Map	Field Location	Signature Calculation	Classification	Map	Total (Less Location)
7094	2.0	40.0	2.0	8.0	2.5	14.5
MIDAS (CCT)	2.0	40.0	2.0	0.3	2.5	6.8
MIDAS (HDT)	0.1	40.0-4.0	0.3	40 sec	0.1	~0.5
360/57 (Est)	1.0	40.0	1.0	4.0	1.25	7.25

printer. This color hard copy should greatly aid the user in field location, thus reducing location time to four hours. The remaining three processing steps can be seen as being accomplished very quickly provided that the data source (HDT) is available.

(3) Diagnostic methods, for both software and hardware, assume significant importance in successful design and operation of such a system. This facet of system design was not assigned great importance at the beginning of the program but became progressively more important as design and assembly proceeded. As a result, as assembly progressed from card to subassembly to full assembly, software and hardware design changes were found desirable to facilitate automatic or semi-automatic testing of these levels of assembly. The final system, then, should be readily maintainable. This same approach should be continued in Phase II, to facilitate debugging and maintenance, and also to make available to the operator intermediate computational results at high speed if he desires.

(4) Operation of MIDAS during its debugging and test phases has made it evident that a display in color is an essential item in the system. In order to verify and estimate the performance of the system in processing ERTS data, a simple C-scan black and white display was assembled from various laboratory components. This was able to effectively display output from one class at a time; but, for the multimodal (or multiclass) signatures, the display was clearly a marginal device. The moving-window color display planned for Phase II will be much more effective, especially when integrated into the software-controlled system.

In summary, the objectives of the first phase of the program have been met. This includes demonstrating the feasibility of the digital classifier using readily available and economical components within the state of the art. The capability of employing a medium scale minicomputer system (DEC PDP-11/45) to control and set up such a classifier has been established. The applicability of current technology, including an advancement such as the present MIDAS, to projected space-borne sensors such as EOS seems clear. The problem of maintaining a special-purpose system employing a combination of hardware diagnostic buses and software has been adequately solved.

2.2 RECOMMENDATIONS

(1) By reason of the established feasibility of the design approach for the classifier, the design of the preprocessor should employ the same technology, continuing the present design approach of fault diagnostic access. This is elaborated upon in Section 5.

(2) Color display of on-line results in the form of a moving window seems necessary. There appears to be no other way of allowing an operator to monitor the results effectively during analysis and test of the processing steps.

(3) Also, for any high resolution data sources — especially when natural scenes or agricultural scenes containing small fields are of interest — there is a need for a high-resolution, color, hard copy to allow an operator to go off-line for training set selection and analysis to afford efficient use of the system. Analysis of current and ongoing programs to process ERTS data indicates that times as long as a week are sometimes required for the analyst to perform this function. In the best cases, this operation may require as little as one or two hours. This time, if spent on-line, would seriously reduce the system throughput since the operations of analysis and classification for typical batches of data would require only ten to fifteen minutes each. For many problems, however, the operator could remain on-line and obtain turn-around of a processing problem in times as short as one hour or as long as four hours. Determination of the basis for choice between an on-line and a mixed, on- and off-line mode will have to be made experimentally. In any event, final output of imagery will require such a hard-copy output.

(4) Operation of the system as presently configured has indicated that system control using the DEC DOS system is an adequate minimum. Better access times should be sought, however, as a goal to obtain better interaction. This may be accomplished by switching to the DEC key-CRO (VT05) terminal as the console device, by employing a line printer for output and, ultimately, by adding the RAMTEK display system.

(5) To provide wider access to the MIDAS, some consideration should be given to remote access to the system. This may be considered in the context of MIDAS operating in a time-share mode using data stored locally and providing outputs remotely or locally as a function of the quantity of data and the transmission costs and rates available. Time-share operating systems under development by DEC may be applicable.

(6) Present studies on the inclusion of spatial context and signature mixtures should be followed and incorporated, if possible, during Phase II design.

(7) Documentation of the system design in APL (probably APL/360) should be seriously considered as the means of defining, modeling, and testing the system. This could, conceivably, be extended to provide wire-wrap information directly. However, the latter may be beyond the scope of Phase II.

2.3 RESULTS

The MIDAS Phase I system has been constructed and tested in two ways. First, each arithmetic circuit has been operated and tested from input to output to ascertain that an output is correctly computed from an input over the range required. Secondly, an ERTS data set from Yellowstone National Park has been processed to provide recognition maps of 16 categories. These outputs were then compared by inspection with similar maps obtained from the IBM 7094 system.

The first tests have been successfully completed and verify the arithmetic design. The MIDAS processing of ERTS data is verified by comparison with the results obtained as described in Appendix A, Performance Test of MIDAS. These results are shown in Fig. 2. The colors and categories used are:*

Green - Lodgepole Pine 1 & 2, Spruce Fir
Brown - Grasslands 1, 2, 3; Shrubs; Grass-Brush; Brush
Orange - Light Rock 1, 2, 3
Magenta - Dark Rock
Black - Thermal Deposits
Blue - Water

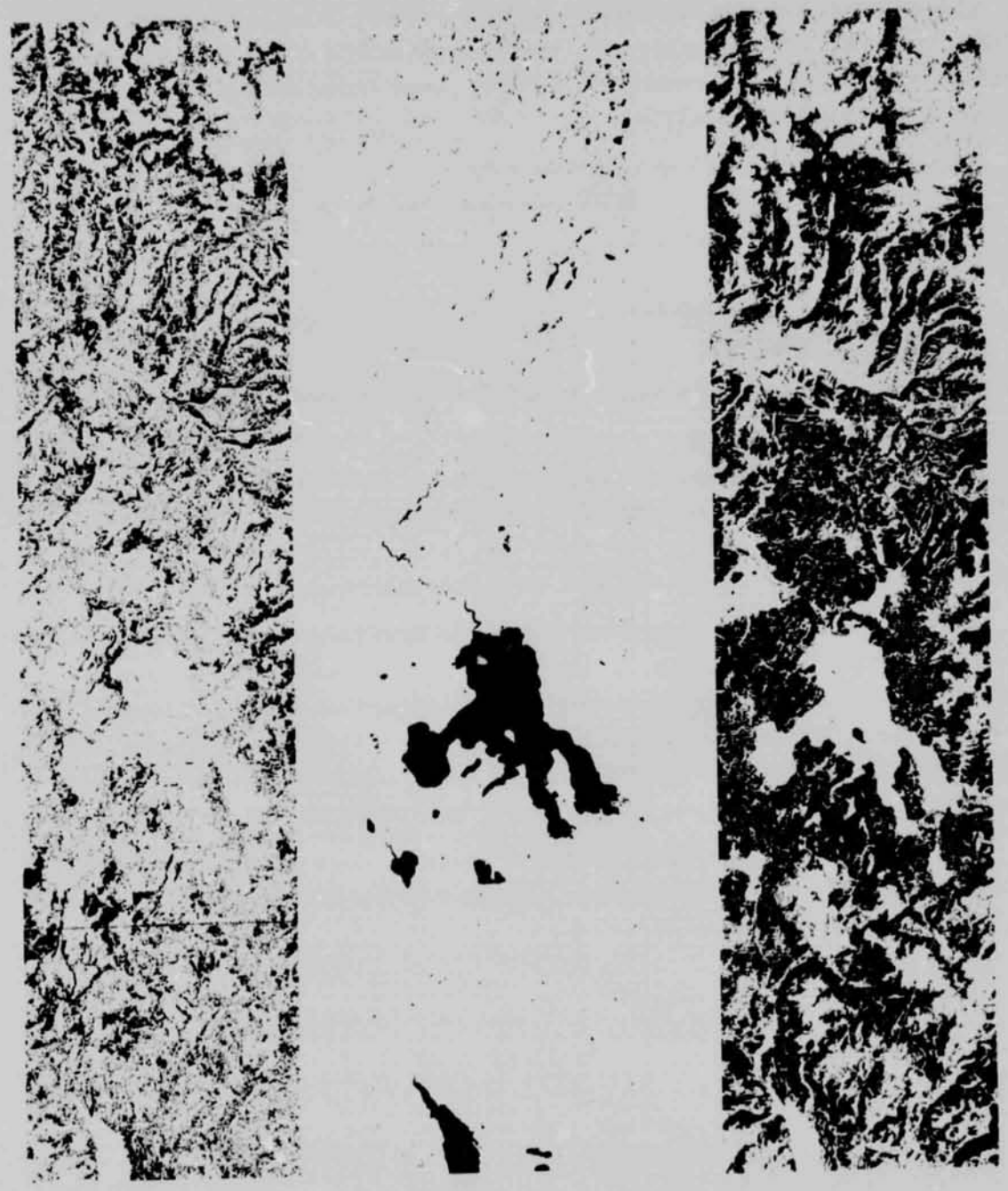
These may be compared with the results obtained from the 7094 shown in Fig. 3 using the colors listed in Table 2.*

The classification results have been examined and show good agreement among the categories with the exception of "unrecognized." This category does not exist in the MIDAS map because the threshold was different than for the 7094, forcing recognition of all data vectors, and the oversight was not detected until after the classification run and printout.

Furthermore, the 7094 color map was printed on the Mead Technology (formerly Data Corp.) color printer in Dayton, Ohio, and hence has a color range which we cannot duplicate locally. In consequence, interpretation of the results by inspection is difficult. The two results do, however, appear to be substantially the same. Though an element-by-element comparison with the 7094 results done on the 7094 computer or the PDP-11/45 would have been desirable, this would have required more time for software testing than was available.

To sum up, the MIDAS hardware and software system has been completed and tested. Comparison with the 7094 recognition results using the same signatures indicates that MIDAS recognition results are substantially the same.

*Not all copies of this volume include color.



(a) Light Rock (b) Water (c) Lodgepole Pine

FIGURE 2. MIDAS RECOGNITION MAP

FIGURE 3. YELLOWSTONE NATIONAL PARK, AREA 3

TABLE 2. YELLOWSTONE CLASSIFICATION MAP COLOR LEGEND*

Map Color	Terrain Class
Violet	Amorphous and shadowed rock
Blue	Water
Cyan	Coniferous forest >45% crown closure (Lodgepole, Ponderosa Pine, Douglas fir, and Spruce-fir)
Green	30-45% Conifer cover over grass
Yellow-green	15-30% Conifer cover over grass
Yellow	Grasslands
Yellow-orange	Grass/brush
Orange	30-45% Conifer cover over felsic rock-rubble
Orange-red	15-30% Conifer cover over felsic rock-rubble
Magenta	Felsic rock
White	Thermal deposits/snow
Black	Unclassified

*Not all copies of this volume include color.

3
MIDAS USE AND SOFTWARE

3.1 GENERAL CONSIDERATIONS

In this section, we describe the use of the MIDAS system through the programs employed on the mini-computer to provide the user with information about and control of the processing. Our approach has been to keep the programs as simple, as easy to use, and as easy to change as possible in this first phase of development. For example, the system issues a parameter-input request to the user, and the responses possible to obtain the desired result at each step are explained.

3.1.1 DOS MONITOR

The operating system we use on the PDP-11/45 computer is the Disk Operating System (DOS). This is the standard Digital Equipment Corp. (DEC) operating system provided to all PDP-11/45 users who purchase a disk peripheral.

The DOS Monitor supports the PDP-11/45 user through development and execution of programs by providing the following:

(1) convenient access to system programs and utilities such as the FORTRAN-IV Compiler, the MACRO-11 assembler, a linking program (LINK-11), an on-line debugging package (ODT-11), an editor (EDIT-11), a file-utility package (PIP-11), and other system utility programs
(2) input/output transfers at four levels, ranging from direct access of device drivers to full formatting capabilities, while providing the convenience of complete device independence when possible
(3) a file system for management of secondary storage
(4) a versatile set of keyboard commands for use in controlling the flow of programs

The user communicates with the DOS Monitor in two ways: through keyboard instructions (COMMANDS) and through programmed instructions (REQUESTS).

Keyboard commands enable the user to load and run programs; assign I/O devices or files; start or restart programs at specific addresses; modify the contents of memory locations; retrieve system information such as time of day, date, etc, and dump core. A user can utilize programmed requests, which are macros assembled into the user's program and through which he specifies the operation to be performed via the monitor. Some programmed requests are used to access I/O transfer facilities, and to specify data location, its destination, and its format. In these cases, the monitor brings drivers in from disk, performs the data transfer, and informs the user of the status of the transfer. Other requests access Monitor facilities

to query system variables, such as time of day, date, and system status information, and to specify special functions for devices.

The DOS Monitor currently in use on the MIDAS PDP-11/45 system is version 8.8. Version 9.0 will be used in Phase II of the program.

A detailed description of the DOS Monitor services can be found in the "Disk Operating System Monitor Programmer's Handbook" (DEC-11-OMONA-A-D) [2].

3.1.2 MIDAS OPERATING SOFTWARE OVERVIEW

The software provided for Phase I of the MIDAS project is designed to exhibit the accuracy and speed of the MIDAS digital classifier hardware. Since development of the MIDAS Classifier hardware was of primary importance in Phase I, the software development was limited in available resources and concentrated mainly on developing the most effective methods of data transfer between the PDP-11/45 and the MIDAS hardware. Because development of the software and hardware took place simultaneously, the time necessary to fully implement many useful facets of control software was not available. In addition, many of the control functions not implemented were omitted as a result of lack of adequate hardware peripheral I/O devices (e.g., interactive CRT, large-size fast-access disk, etc.). Since these drawbacks will not be present during the implementation of Phase II, a major software thrust will be made in the area of human-engineering control. However, the software developed in Phase I does provide a basic structure for interactive multispectral classification procedures.

All programs for the MIDAS software system were written for the PDP-11/45 with the following available devices:

(1) 24K 16-bit core memory
(2) FP-11 floating point processor
(3) LA-30S serial DEC writer
(4) RK-11C disk controller and RK-Ø5 disk drive
(5) VT-Ø5B alphanumeric CRT
(6) TM-11 magnetic tape drive controller, one TU-1Ø 7-track magnetic tape drive, and two TU-1Ø 9-track magnetic tape drives
(7) KW-11P programmable real-time clock
(8) two DR-11B general device interfaces (for transfer of data to and from MIDAS hardware)
(9) one DR-11C general device interface

All program source code was entered on-line to the PDP-11/45 via the console keyboard and subsequently stored both on disk and magnetic tape. Most of the routines and programs were written for and assembled in the system assembler program, MACRO-11, although some

statistics routines and several diagnostic main programs were written for and compiled by the FORTRAN IV compiler. Routines written in FORTRAN were not employed in any real-time data-acquisition processes because of the inefficient code generated by the FORTRAN compiler.

Current versions of MIDAS operating programs (Fig. 4) are available under user-identification code (UIC) 100,100 in the following files:

1. DISPLAY.LDA - graphic display program
2. SIGCVT.LDA - signature conversion program
3. SIG.LDA - signature extraction program
4. SCALER.LDA - signature coefficient scaler
5. RAMLD.LDA - MIDAS RAM loader
6. CLAS1.LDA - classification control program
7. CLAS2D.LDA - control program

The general flow of operation for the MIDAS operating software is as follows:

Initially, the classification process must obtain the spectral signatures of various types of ground species. In order to isolate areas from which to calculate these statistics, the user may use some type of grayscale display for selection of boundaries. The program DISPLAY, using 9-track ERTS input CCT, displays a single-channel either on the VT-Ø5B CRT or the line printer for perusal by the user. From this output, we extract line and point selection for training areas.

At this point, the program SIG allows the user to calculate the statistics of a signature for a given area, obtain via a hard-copy output device (either the console or the line printer) the results of the statistical calculations, test the validity of the training set statistics by employing these statistics to classify the raw data that generated them, and decide from this displayed information whether to use a particular training area or to modify the areas to obtain better recognition. This iterative, interactive signature extraction process is critical in terms of classification accuracy on an unknown dataset.

After the signature set has been selected, the program SCALER is run on the output dataset from SIG to convert the calculated statistics to coefficients in a format suitable for loading into MIDAS RAMs. The magnitude of signature coefficients is error-checked at this level.

With the output file from the SCALER program, the user may call upon the program RAMLD to select a subset of signatures for actual loading into the MIDAS Classifier RAMs. The RAMLD program performs the inter-signature normalization procedures necessary for the particular subset of signatures chosen for loading. Also, at this juncture, a maximum exponent value is chosen as an upper bound necessary for classification accuracy.

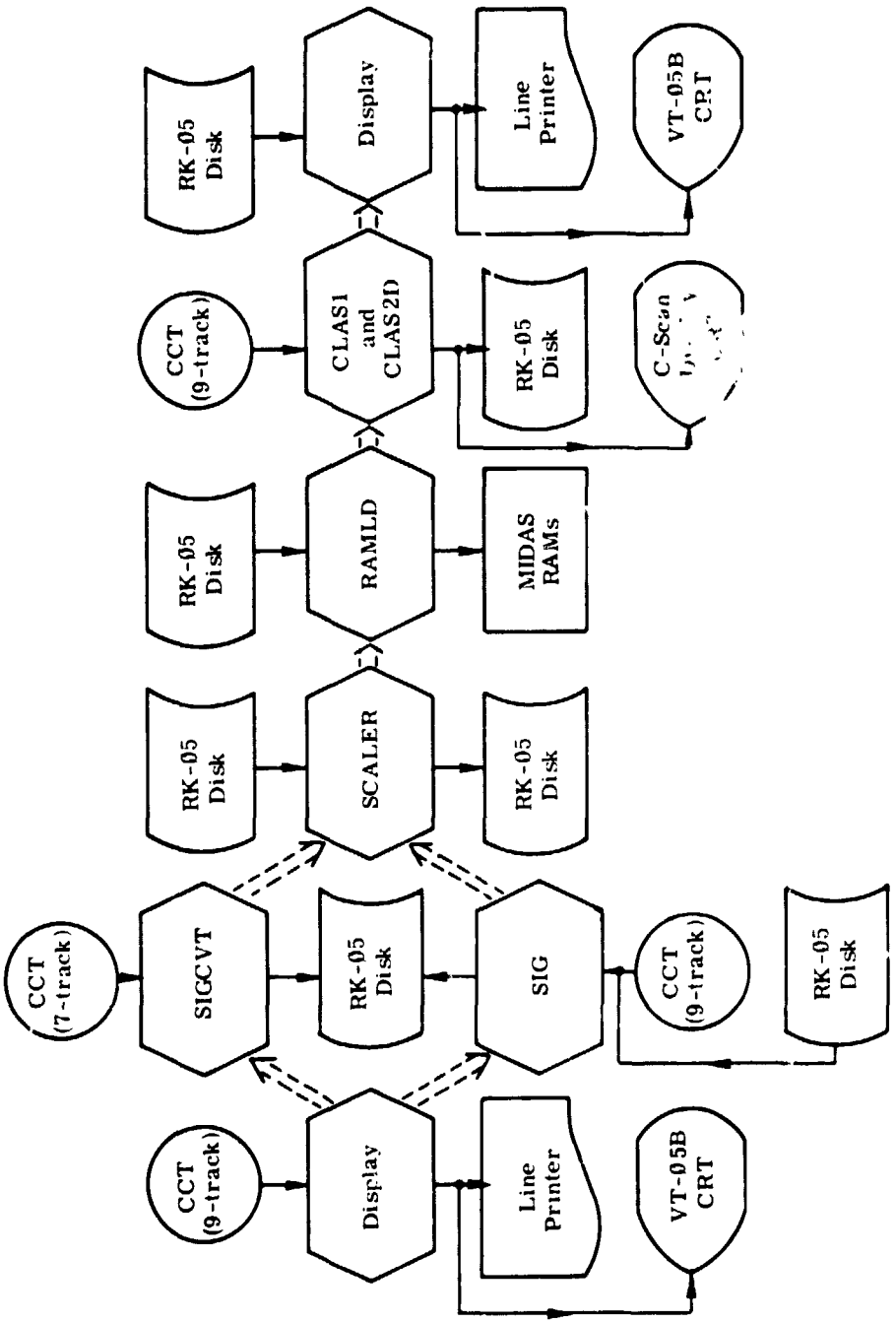


FIGURE 4. MIDAS SOFTWARE SYSTEM. Includes I O media.

Now that the classifier coefficients have been loaded into the MIDAS hardware, the user normally calls the program CLAS1. CLAS1, via the console keyboard, interrogates the user about the classification run about to be made (e.g., output devices to be used, scene boundaries, etc.). This information is written into a temporary disk file for use by the program CLAS2D which is loaded into core over the CLAS1 program, and given control. CLAS2D uses the information in the temporary file, created by CLAS1, to set up the MIDAS control logic, and then to actually control the transfer of data between the MIDAS Classifier, core memory, and various peripheral output devices.

Ordinarily, the output of a classification run is stored on disk and also sent to the C-scan CRT display in a real-time mode. However, some post-analysis of the classification results is almost always necessary. Currently, DISPLAY provides the capability for grayscale display of recognition results (stored on disk) to either the VT-05E alphanumeric CRT or to the line printer.

This system obviously does not include some of the fine points in interactive digital analysis of multispectral data, such as post-analysis, pre-classification linear combination of channels, or channel selection, etc.; but the currently implemented programs we discuss below do provide a framework for future inclusion of these features.

3.2 MIDAS OPERATING SOFTWARE DESCRIPTIONS

3.2.1 DISPLAY

TITLE: DISPLAY - Graphic Display Program

CSECT: None

PURPOSE: This program allows the user to obtain a graphic display (both soft- and or hard-copy output) of the various types of multispectral data encountered in recognition processing. (See Fig. 7.)

CALLING SEQUENCE: None

PARAMETERS: Operating parameters are input via the console keyboard in response to the following requests:

1. INPUT FILE =

This request asks for the file specification of the input data set. The string input must conform to CSI (Command String Interpreter) format as specified in Appendix D. In general, this format is

DDD:NNNNNN.EEE

where DD refers to the physical device name (in this case, either MT0 or VT, designating magtape unit 0 or the VT-05 CRT, respectively).

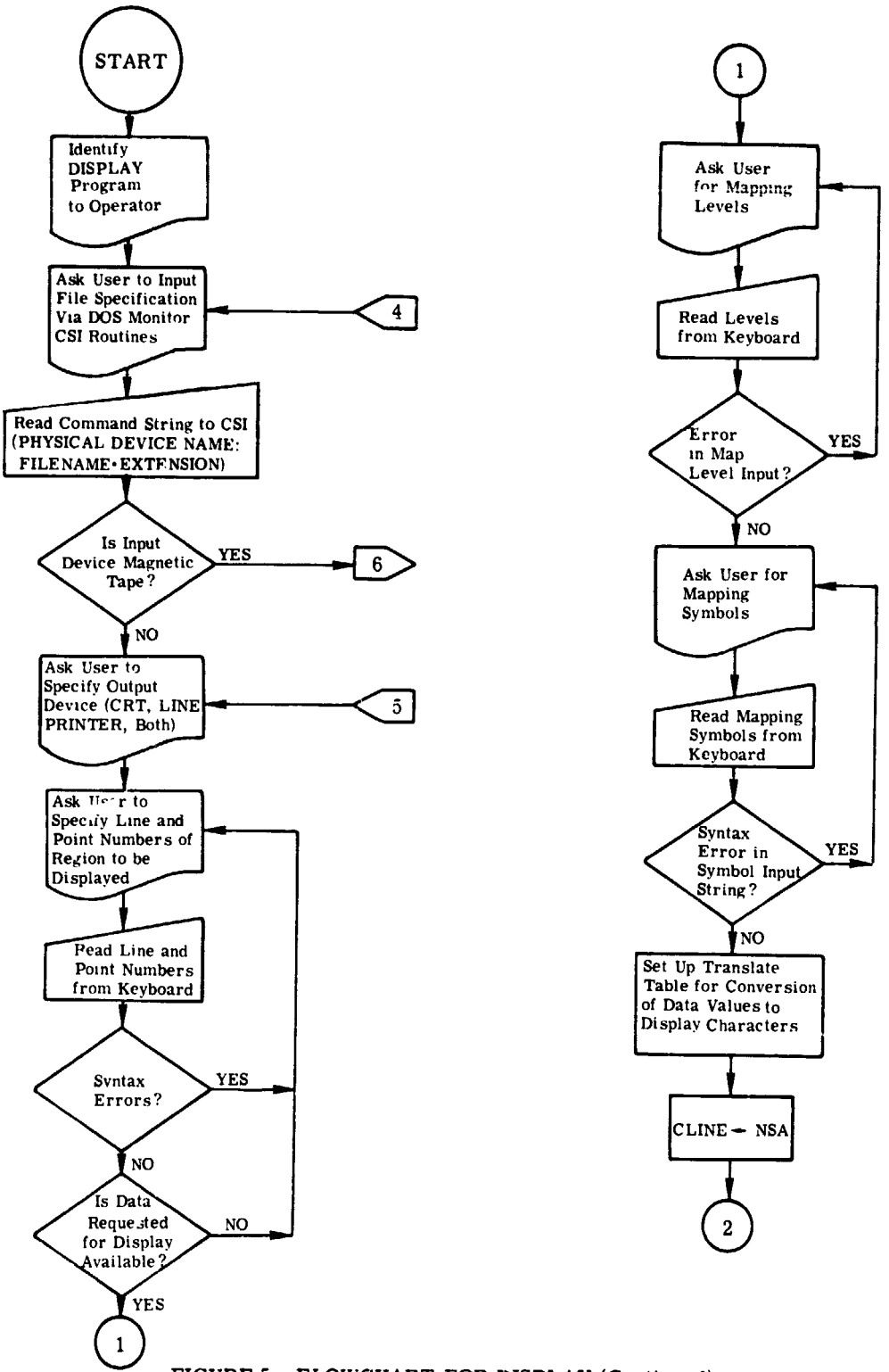


FIGURE 5. FLOWCHART FOR DISPLAY (Continued)

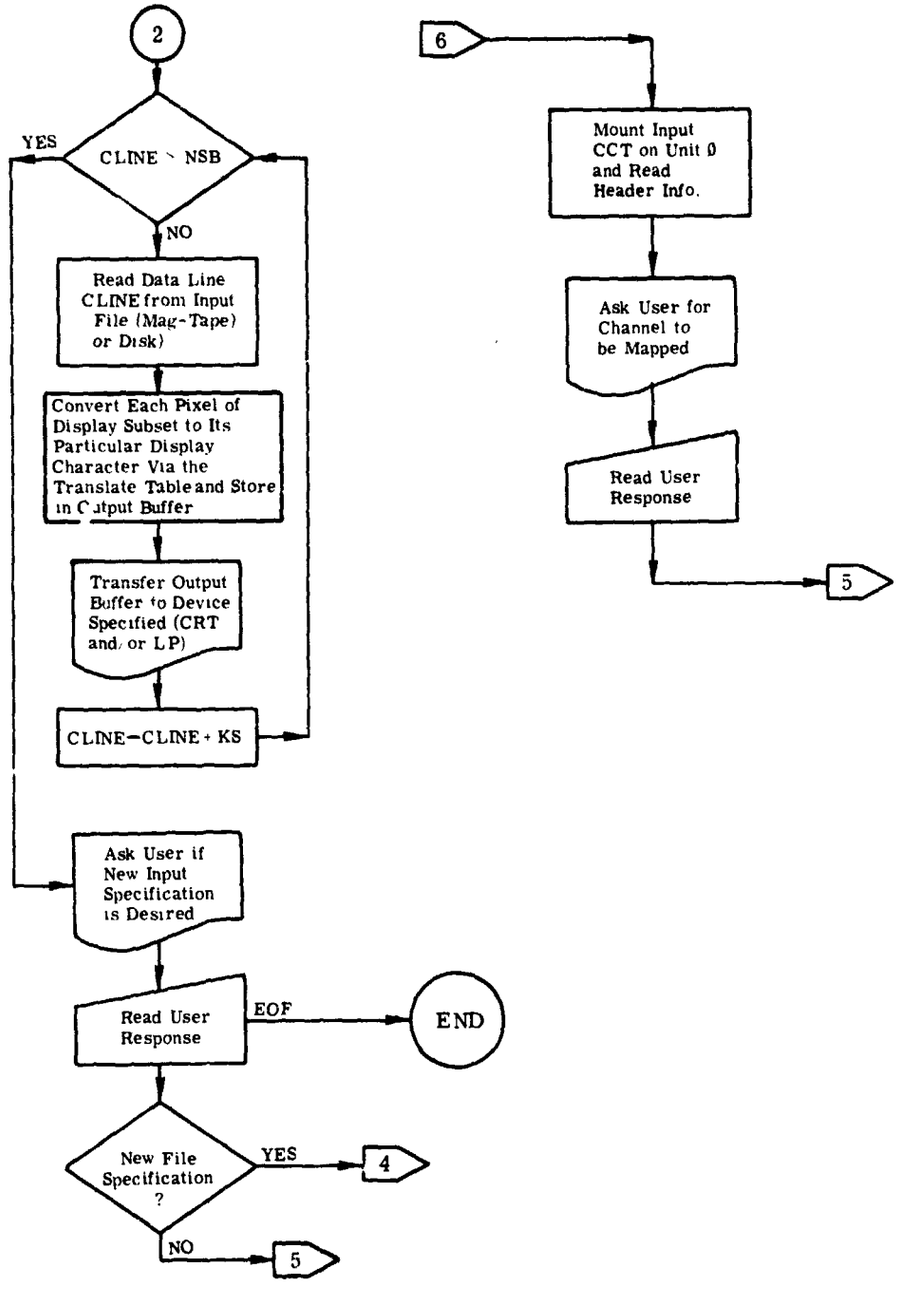


FIGURE 5. FLOWCHART FOR DISPLAY (Concluded)

2. MAP CHANNEL =

This request asks for the channel to be displayed by the DISPLAY program. This request is made only if the input device is magnetic tape (thus, the assumption is that raw data is being displayed).

Proper response is a decimal digit specifying the channel to be mapped followed by a carriage return. Any other response causes request #2 to be reprinted on the console until a proper response is made.

3. OUTPUT DEVICE(S) =

This request asks for the output device(s) to be used in the display run.

Proper responses are 'VT,' 'LP,' or 'BOTH.' Any response other than the above causes request #3 to be reprinted on the console device until a proper response is made.

4. LINE AND POINT NOS. =

This request asks for the line and point numbers of the region to be displayed. If the region entered in response to this request does not exist, the request is reprinted on the console and available line and point numbers must be entered.

Proper response to this request is a string of six decimal numbers, separated by commas, with no embedded blanks allowed, followed immediately by a carriage return. The six numbers have the following significance:

QNAS - starting line number of display region
QNSB - ending line number of display region
QKS - line increment
QNA - starting pixel number of display region
QNB - ending pixel number of display region
QKP - pixel increment

5. MAP LEVELS =

This request asks that the map levels be set for the output display. Proper response to this request is a series of not more than 20 decimal numbers in the range Ø-255. These numbers must be separated by commas, must be monotonically increasing, and the input series must be terminated with a carriage return.

6. MAP SYMBOLS =

This request asks for the map symbols to be used in the output display.

Proper response to this request is a set of characters, separated by commas. The format of this line must be

C,C,C,C, .. , C <CR>

where C is any printing ASCII character.

7. NEW INPUT FILE ?

This request determines the program branch to be taken after completion of a display run for a given region.

Proper responses are 'YES' or 'NO,' with the first character of each sufficient. If neither of the above responses (or allowable abbreviations) is made, an affirmative response is assumed.

DESCRIPTION: This program provides the user with the ability to display two types of multispectral data in the initial implementation of MIDAS software: (1) raw multispectral data and (2) classification results.

Input is either through magnetic tape (assumed to be raw multispectral data) or from disk storage (assumed to be classification results).

The actual display is performed on either an alphanumeric CRT (the DEC VT-05B) or on the line printer, or both simultaneously.

The correspondence between input data and display character is by way of a translate table set up from the two vectors (which) the user input: (1) the level vector and (2) the symbol vector.

The display algorithm assumes an input data value M to be displayed (where $\emptyset \leq M \leq 255$). Also assumes a set of N map levels L (such that $Li < Li + 1$, $\emptyset \leq Li \leq 255$ and $Ln = 255$) and a set of N symbols S. Then the display character to be used is Si, such that $L_{i-1} < M \leq Li$.

STORAGE REQUIREMENTS: 4300_8 bytes

SUBROUTINES REQUIRED: GETARG
GETFIL
VTW

3.2.2 SIGCVT

TITLE: SIGCVT - 7Ø94 to 11/45 SIG. CONVERSIONS

CSECT: None

PURPOSE: This program provides a software interface between the IBM 7Ø94 general-purpose digital computer and the PDP 11/45 - MIDAS system for the purpose of using signatures which were calculated and extracted on the IBM 7Ø94 to operate the MIDAS system. (See Fig. 6.)

CALLING SEQUENCE: None

PARAMETERS: Operating parameters are input via the console keyboard in response to the following requests:

1. NAME OF FILE TO BE CREATED =
This request asks for the name of the signature file to be created on disk. This file will be contiguous and conform to the standard format of a signature file as outlined in the STAT write-up. The format of this name must be 'NNNNNN.EEE' where N is a character in the filename and E is a character of the extension.

2. NUMBER OF CHANNELS =
This request asks for the number of channels from which these signatures have been calculated. Proper response is a one- or two-digit decimal number followed immediately by a carriage return.

3. NUMBER OF SIGNATURES =
This request asks for the number of signature sets contained in the file on the 7-track input tape. Proper response is a decimal number followed by a carriage return.

DESCRIPTION: This program takes BCD card images of signatures generated on the IBM 7Ø94 computer (on 7-track 8ØØ BPI magnetic tape), converts these to ASCII and then, by using the FORTRAN I/O conversion routine DECODE, to binary floating-point representation. A file is created on disk and these converted signatures are stored in the signature file format (see Appendix C).

ERRORS: The following error message is the only one possible in the SIGCVT program:
'UNIT Ø IS NOT 7-TRACK' means that the drive which is currently identified as unit Ø is not a 7-track magnetic tape drive. The

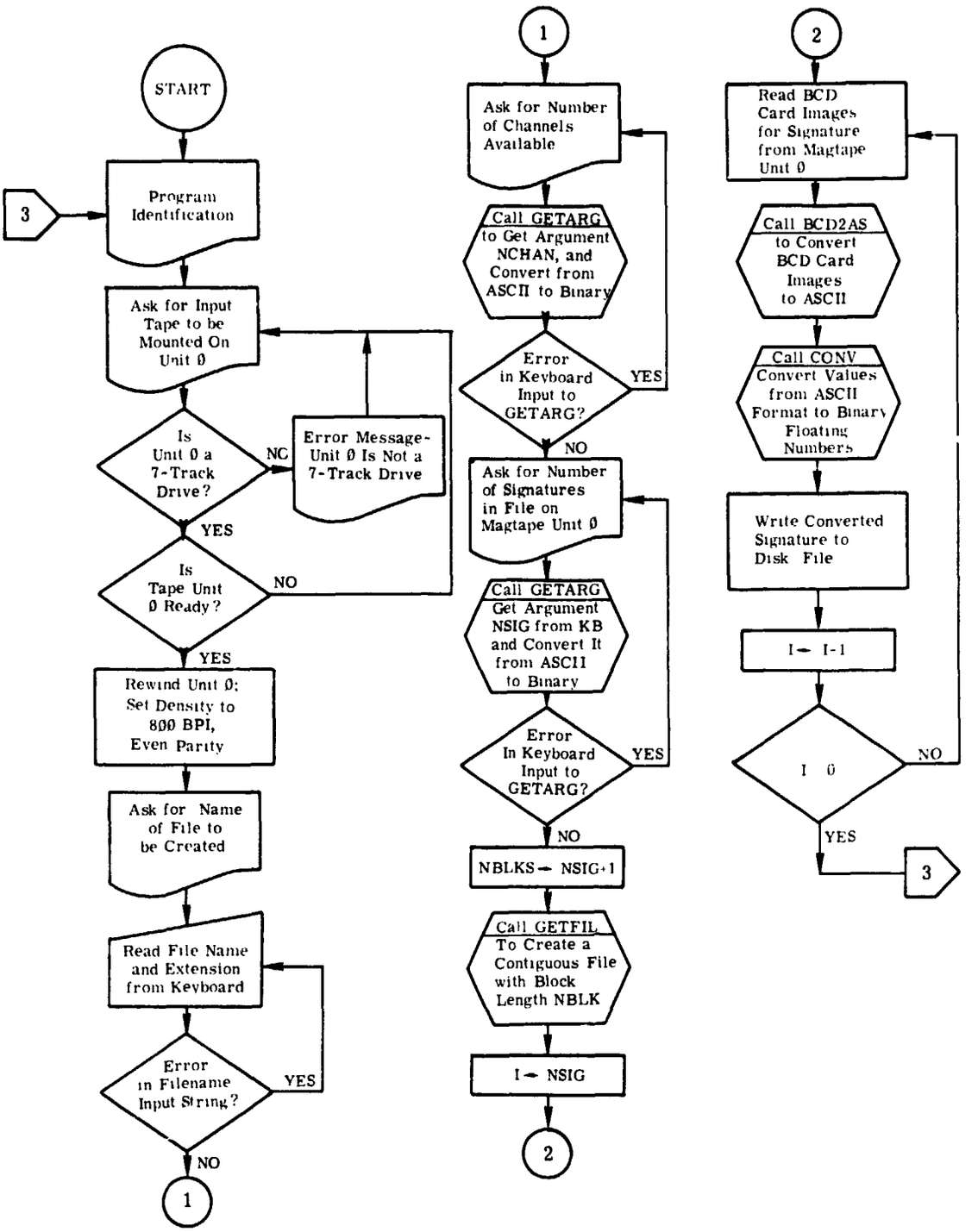


FIGURE 6. FLOWCHART FOR SIGCVT

magnetic tape drive with the 7-track tape on it should be switched to unit ∅ and placed on-line. All other units should be switched off-line.

STORAGE REQUIREMENTS: 13062_8 bytes

SUBROUTINES REQUIRED: GETARG
GETFIL
CONV
BCD2AS

3.2.3 SIG

TITLE: SIG - Signature Extraction Program

CSECT: None

PURPOSE: The program provides the overhead functions to transfer raw data from digital inputs to a subroutine STAT which calculates the signature statistics. This program then displays the actual statistics and the results of classification on the raw data using the calculated signature coefficients. (See Fig. 7.)

CALLING SEQUENCE: None

PARAMETERS: Operating parameters are input via the console keyboard in response to the following requests:

1. GENERAL REGION LINE AND POINT NOS. =

This request asks for the line and point numbers associated with a region which, in general, contains all the data to be used in the signature extraction. This data will be stored on fast randon-access disk storage for quick retrieval. Proper response to this request is a string of six decimal numbers separated by commas, the string being terminated by a comma. The string arguments are as follows:

GNSA - starting line number of general signature extraction (GSE) region
GNSB - ending line number of GSE region
GKS - line increment
GNA - starting pixel number of GSE region
GNB - ending pixel number of GSE region
GKP - pixel increment

NOTE: In the current implementation, both GKS and GKP are set equal to 1 to avoid complications in the pixel-addressing algorithm.

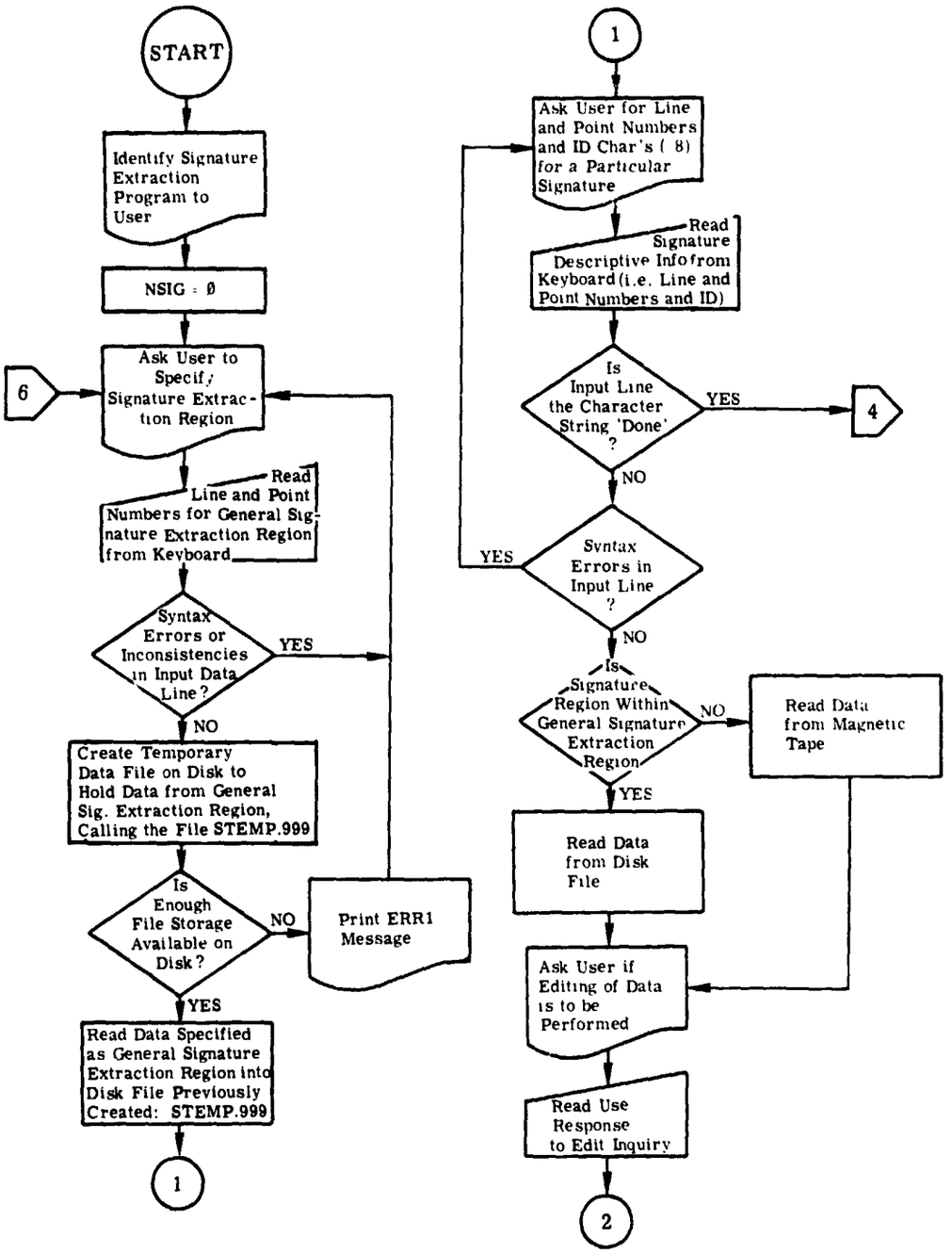


FIGURE 7. FLOWCHART FOR SIG (Continued)

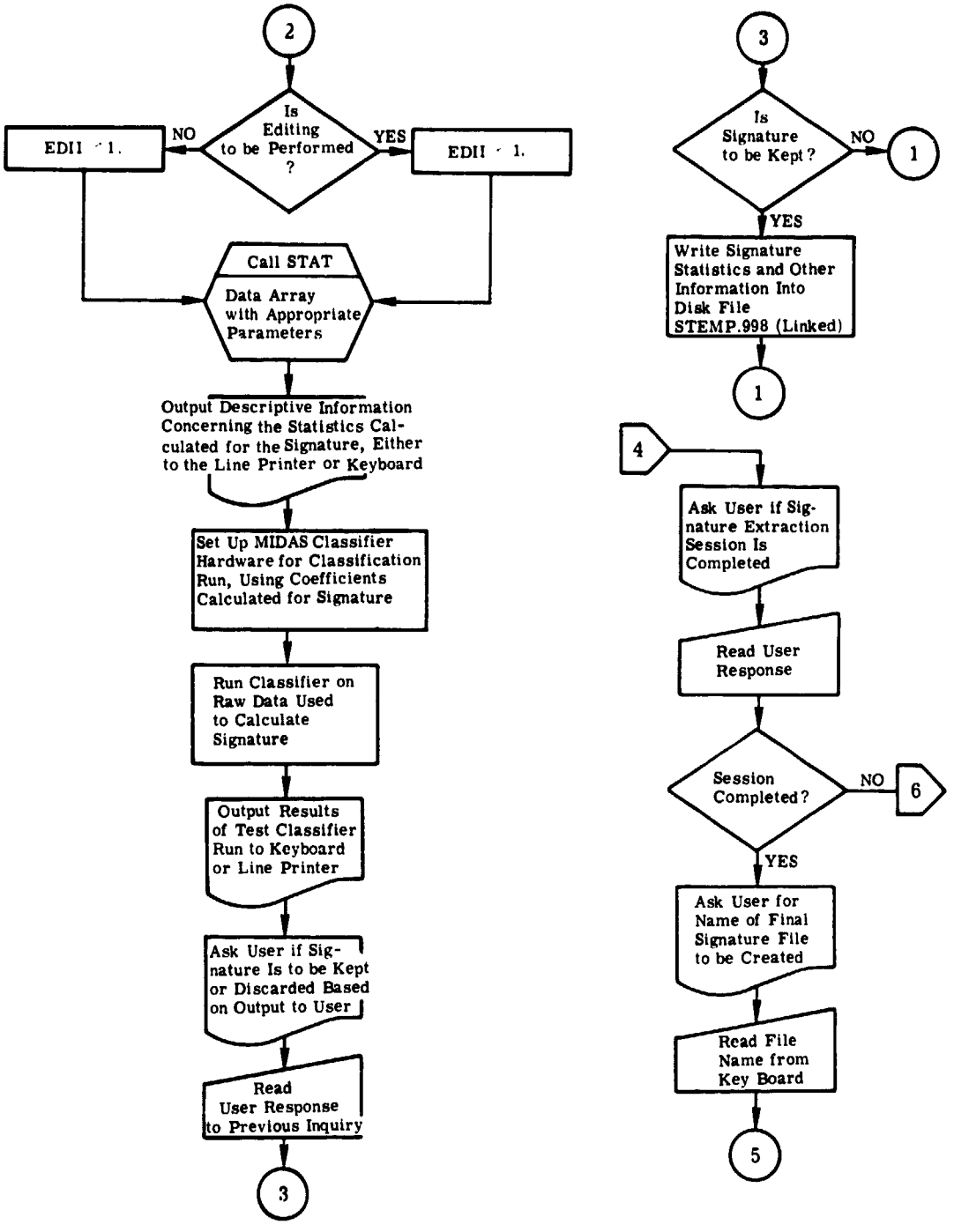


FIGURE 7. FLOWCHART FOR SIG (Continued)

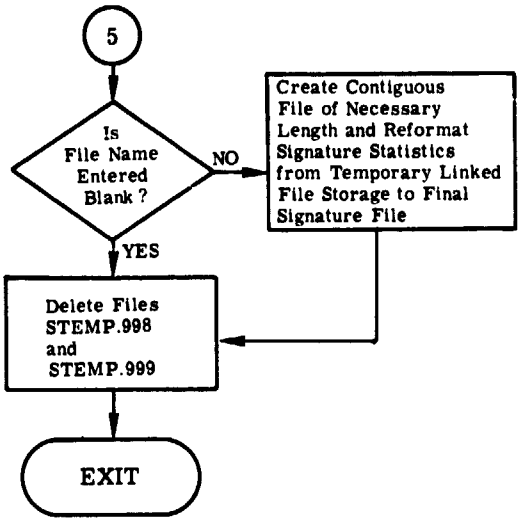


FIGURE 7. FLOWCHART FOR SIG (Concluded)

2. SIG. LINE AND POINT NOS. AND ID =
This request asks for the line and point numbers and ID (≤ 8 ASCII characters) for a specific signature.
Proper response to this request is a string of six decimal numbers separated by commas, followed by a string of ASCII characters which are assumed to be the ID associated with the signature. Any characters past the 8th one are ignored. The first six numbers describe the signature region in the same way as the GSE region line and point numbers do.
The alternate proper response to this request is the simple character string 'DONE.' This response instructs the program to proceed to request #5.
3. EDITING ?
This request asks if editing of the raw input data is to be performed. The editing algorithm to be employed is described in the STAT and EDIT subroutines.
Proper response to the request is either 'YES' or 'NO' with the first character sufficing. If neither proper response (nor equivalent allowable abbreviation) is made, an affirmative response is assumed.
4. RETAIN SIGNATURE ?
This request asks whether or not the signature just calculated is to be stored on disk for later use. It is the user's responsibility to make this decision based upon the printed output provided by the program.
Proper response to this request is either 'YES' or 'NO' with the first character of each sufficing. If neither of the proper responses (nor equivalent allowable abbreviation) is made, an affirmative response is assumed.
5. SESSION COMPLETE ?
This request asks if the signature extraction session is completed. If a negative response is made, the program returns to request #1. Otherwise, the program continues to request #6.
Proper response to this request is 'YES' or 'NO' with the first character of each sufficing. If neither of the proper responses (nor their allowable abbreviations) is made, an affirmative response is assumed.

6. SIGNATURE OUTPUT FILENAME =

This request asks for the name of the output file to be created to contain the data and statistics calculated for the retained signatures.

Proper response is (1) a character string of format NNNNNN.EEE where N is a character of the filename and E is a character of the extension name, or (2) any number of blank characters followed by a carriage return. The latter indicates that none of the signature information is to be retained.

DESCRIPTION. This program allows the user to interactively extract signatures from raw multispectral data, analyze the data, and recalculate the signature if necessary.

The user is asked to supply a general signature extraction region (a subset of the entire set of input data), which will be stored on disk. This region should be the region most frequently accessed in the calculation of the various signatures, because data can be accessed very quickly from the disk. If data for signature calculation are requested from outside this region, the data will be retrieved from the input CCT, but the time required will be much larger than the time necessary for comparable disk accessing.

Once the data have been accessed, they are passed to a routine STAT for actual signature statistics calculation. The values returned to the calling program include: median vector, mean vector, standard deviation vector, covariance matrix, correlation matrix, upper and lower editing bounds, number of rejected values (based on editing criteria), eigenvalue vector, and the eigenvector matrix. These values are output to a hard-copy output device (either the console keyboard or the line-printer can be assigned via the DOS Monitor ASSIGN command - the keyboard is the default device).

The actual raw data used to calculate the signature are then classified using the MIDAS hardware, with the signature calculated as the only signature. The results of this test of signature reliability are also printed out by the assigned hard-copy output.

Once all the signatures to be extracted have been assembled, they may then be stored in their final form in a disk file specified by the user in request #6, (see Appendix B).

SUBROUTINES REQUIRED: STAT
(plus all subroutines required by STAT)

3.2.4 STAT

TITLE: STAT - Statistical Analysis

PURPOSE: This FORTRAN subroutine accepts a matrix of data values, edits out the extreme points (optional), and calculates the mean for each channel, the standard deviation for each channel, the median, covariance matrix, and the correlation matrix. If desired, the routine will calculate the eigenvectors and eigenvalues for the covariance matrix. Basically, this routine performs the actual signature extraction from a given set of data values. (See Fig. 8.)

CALLING SEQUENCES:

```
ASSEMBLER:  JSR     R5, STAT
            BR      .+46
            .WORD   NCHAN
            .WORD   NSS
            .WORD   DATA
            .WORD   EDII
            .WORD   AMED
            .WORD   AMEAN
            .WORD   STD
            .WORD   COV
            .WORD   COR
            .WORD   EDHI
            .WORD   EDLO
            .WORD   NREJ
            .WORD   NREJT
            .WORD   IGENUE
            .WORD   IGENTR
            .WORD   QDEV
            .WORD   WORK
            .WORD   ISW
```

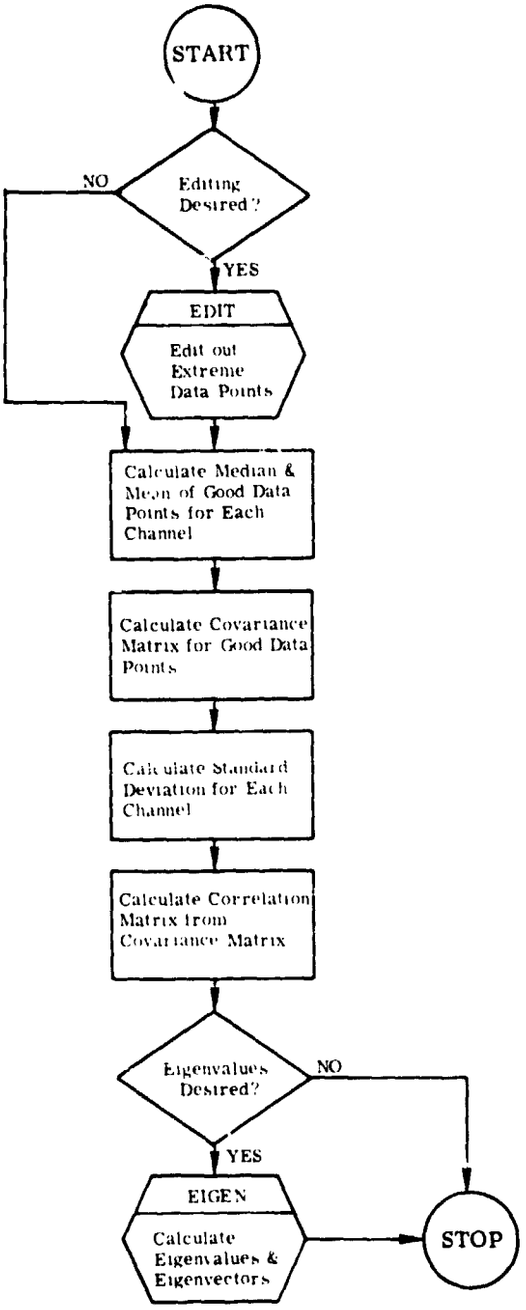


FIGURE 8. FLOWCHART FOR STAT

FORTRAN: CALL STAT (NCHAN, NSS, DATA, EDII, AMED, AMEAN, STD, COV, COR, EDHI, EDLO, NREJ, NREJT, IGENUE, IGENTR, QDEV, WORK, ISW).

PARAMETERS:

NCHAN - This is the address of the integer number of channels in the DATA array. (Input)

NSS - This is the address of the integer number of data values in each channel of the DATA array. (Input)

DATA - This is the address of the FORTRAN data point array containing the data values packed two integer values per word. The array is dimensioned NCHAN + NSS and is stored in column-by-column form. (Input)

EDII - This is the address of the floating point, single precision edit switch. If EDII < 1, then the data in the DATA array is to be edited. Otherwise, no editing is performed. (Input)

AMED - This is the address of a vector of the median for each channel as a single-precision, floating-point number. (Output)

AMEAN - This is the address of a vector of the mean for each channel as a single-precision, floating-point number. (Output)

STD - This is the address of a vector of the standard deviation for each channel as a double-precision, floating-point number. (Output)

COV - This is the address of the FORTRAN array containing the covariance matrix for the DATA array. It is stored in column-by-column form. The elements of the matrix are double-precision, floating-point numbers. (Output). There are NCHAN elements in any row or column.

COR - This is the address of the FORTRAN array containing the correlation matrix. Each element is a double-precision, floating-point number. The matrix is stored in column-by-column form with NCHAN elements in any row or column. (Output)

EDHI - This is the address of the vector of the upper limit for all data values in each channel. Each element is a single-precision, floating-point number. (Output)

EDLO - This is the address of the vector of the lower limit for all data values in each channel. Each element is a single-precision, floating point number. (Output)

NREJ - This is the address of the vector of the number of rejected data values in each channel. Each element is an integer. (Output)

NREJT - This is the address of the integer total number of data points rejected. (A data point is comprised of NCHAN data values, one data value in each channel.) (Output)

IGENUE - This is the address of the vector of the eigenvalue for each channel. Each element is a double-precision, floating-point number. (Output)

IGENTR - This is the address of the FORTRAN array of eigenvectors. The eigenvectors are in the columns of the matrix which is stored in column-by-column form. The matrix is dimensioned NCHAN*NCHAN. Each element is a double-precision, floating-point number. (Output)

QDEV - This is the address of the vector of NCHAN single-precision, floating-point elements representing the estimate used by EDIT for the standard deviation in each channel. (Output)

WORK - This is the address of the double-precision, floating-point vector that will be used as work space by the FIGEN subroutine. There should be NCHAN elements available. (Input)

ISW - This is the address of a switch indicating, upon the call to the program, whether eigenvectors and eigenvalues are to be calculated. If ISW = -1, no eigenvectors or eigenvalues will be calculated. Upon return from this program, ISW indicates the following.

If ISW = -1, eigenvalues and eigenvectors were not calculated.
= 0, eigenvalue and eigenvector calculations were successful.
= 1, eigenvalue and eigenvector calculations were not successful.
= 2, editing procedure edited out all data points except perhaps one.

DESCRIPTION: If editing of the data base is requested (EDII < 1), the upper and lower bounds for data values in each channel are chosen so that no more than one out of a thousand are rejected.

These editing bounds are based on the median (as an estimate of the mean) and the quartile value (as an estimate of the standard deviation). The actual standard deviation used for each channel appears in QDEV, and the upper and lower editing limits for each channel appear in EDHI and EDLO respectively. After the editing procedure, all the data points that were within the editing limits in all channels are the first NSS-NREJT data values in each channel. The remaining data values in each channel were rejected because at least one of the corresponding data values in another channel was not within the editing limits. If fewer than two points are considered good after the editing procedure, ISW is set to 2 and the subroutine returns. If there are two or more good points after the editing procedure, the mean and median are determined for each channel. Then the covariance matrix is calculated using the following equation:

$$COV(K, L) = \frac{NESS}{NESS-1}\left\{\left[\frac{1}{NESS}\sum_{I=1}^{NESS} DATA(K,I)*DATA(L,I)\right] - MEAN(K)*MEAN(L)\right\}$$

where COV = floating point, double-precision covariance matrix
NESS = the integer number of good data points
DATA = the integer array of data values
MEAN = the floating point, single-precision vector of the mean for each channel

The standard deviation for each channel is calculated by extracting the square root of the diagonal elements of the covariance matrix. The correlation matrix is determined next through use of the following equation:

$$COR(K,L) = \frac{COV(K,L)}{STD(K)*STD(L)}$$

if $STD(K)*STD(L) = 0$, $COR(K,L) = 1$.

where COR = correlation matrix; the diagonal elements will be all 1's and all off-diagonal elements are between 0 and 1
COV = covariance matrix
STD = vector of the standard deviation for each channel

If the user has requested the eigenvectors and eigenvalues for the covariance matrix (ISW ≠ -1), these are calculated and the subroutine returns control to the calling program.

ERRORS POSSIBLE: All possible error returns are described under the parameter definition for ISW.

SUBROUTINES REQUIRED:

EDIT - This routine removes all data points that have a value outside certain limits in any channel.

SEARCH - This routine is used to calculate the median data value for each channel.

LTI - This unpacks the data values from the data matrix.

EIGEN - This routine calculates the eigenvectors and eigenvalues of the covariance matrix.

FTNLIB - This library contains the FORTRAN utility subroutines.

STORAGE REQUIREMENTS - 2552 octal locations

3.2.5 SCALER

TITLE: SCALER

CSECT: SCALE

PURPOSE: This program calculates all the coefficients that need to be loaded into the MIDAS memory. The coefficients are calculated, scaled, and shifted so that MIDAS will receive the greatest number of significant bits. These coefficients are then written into a file with a directory so that the MIDAS memory loading program (RAMLD) can select the set of signature coefficients it requires. (See Fig. 9.)

CALLING SEQUENCES: None. This is a main program. All information needed comes in response to typed requests on the keyboard.

1. PLEASE TELL CSI1 YOUR OUTPUT FILE AND YOUR INPUT FILE(S)

This requests the user to type the name of the file where the scaled signature coefficients are to be put and the name of the file(s) where the signature statistics to be scaled are stored. In addition, the user must specify a switch telling the scaler

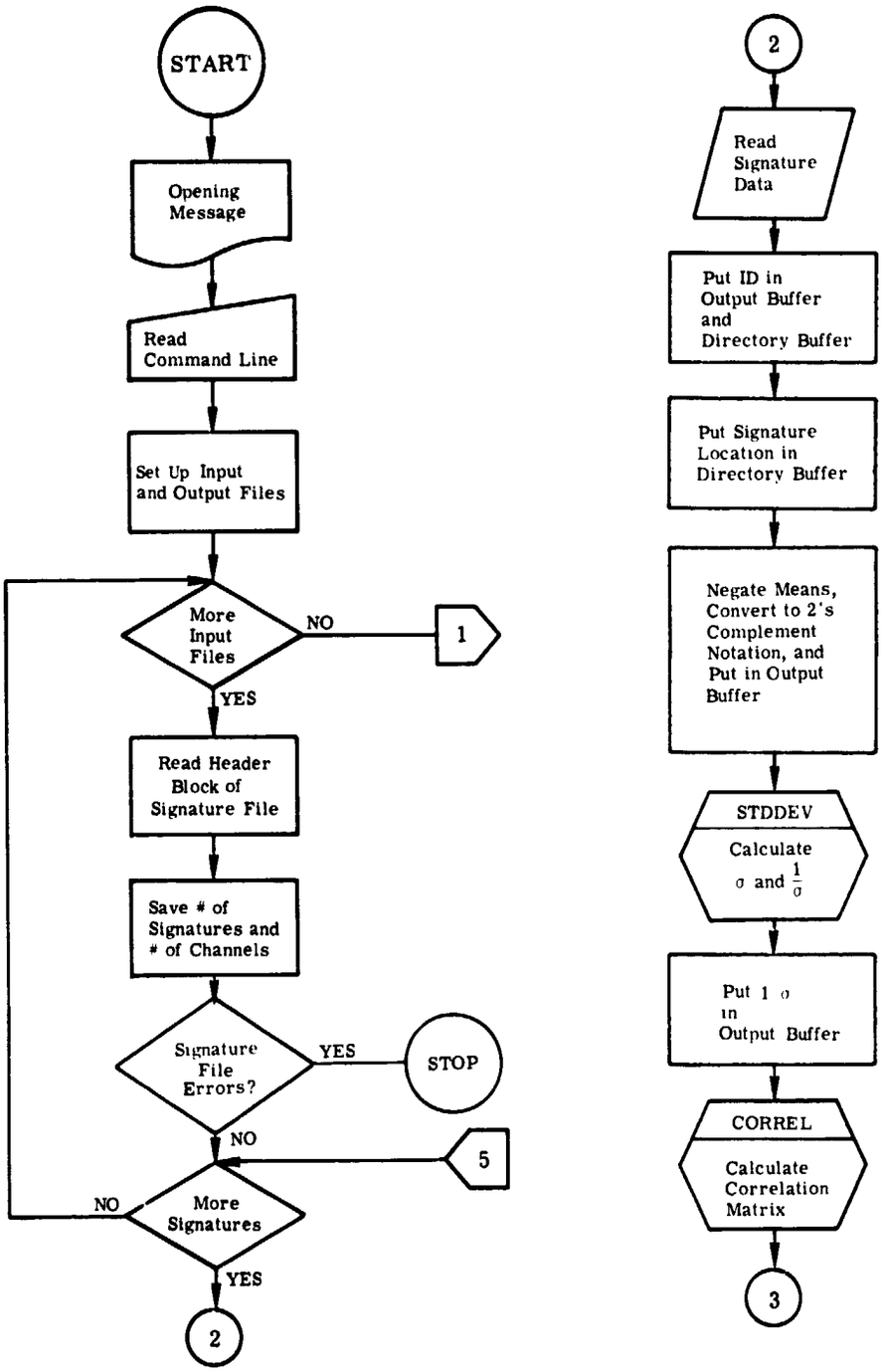


FIGURE 9. FLOWCHART FOR SCALER (Continued)

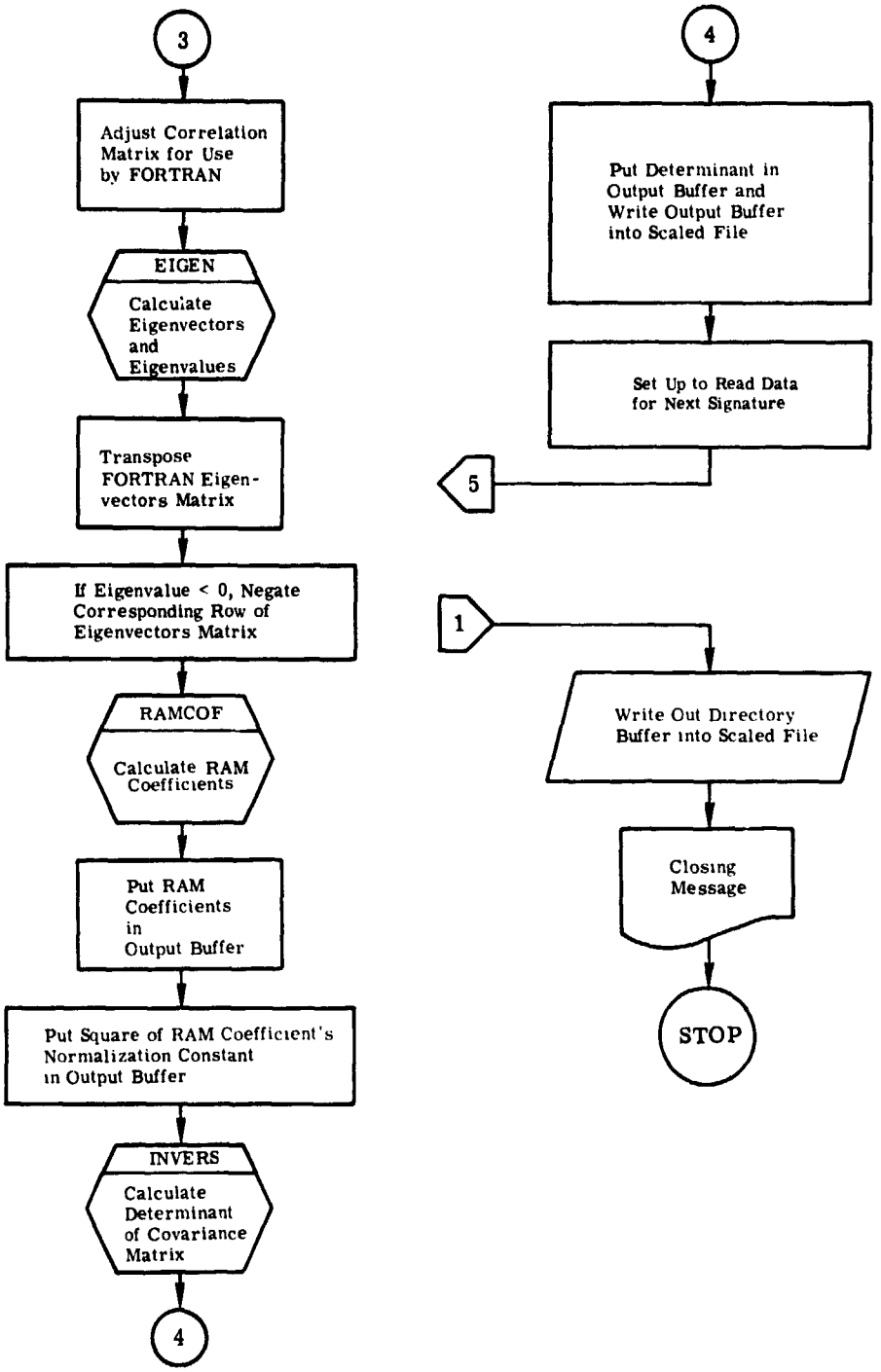


FIGURE 9. FLOWCHART FOR SCALER (Concluded)

what to do when nonfatal errors are encountered while scaling. The format of the response should be acceptable to the command string interpreter.

DV: OFILENAME. EXT [UIC] <DV: IFILENAME. EXT [UIC] /SW

where
- DV: = the device where the following file is stored. If not specified, it defaults to the disk.
- OFILENAME. = the six-character name of the output file.
- IFILENAME. = the six-character name of the input file. These files must be contiguous files with at least two 256-word blocks. The filename is followed by a period.
- EXT = the three-character extension name for the preceding filename.
- [UIC] = the sign-on ID (surrounded by brackets) under which the file was created. May be left out if the same ID as the user's.
- /SW = the two-character switch preceded by a slash which tells the scaler what action to take when nonfatal errors are encountered while scaling the input file(s).
- /SW=/CO = tells the scaler to continue processing whenever a nonfatal error is encountered. The directory entry for the bad signature is removed and the next signature statistics are obtained.
- /SW=/ST = tells the scaler to stop processing whenever a nonfatal error is encountered. The directory entry is removed for the bad signature.

2. SCALING COMPLETED

This indicates that all of the input files have been scaled and the directory has been written out for the scaled file.

Algorithmic Description:

Input Format

The format for each input file is as follows (disk file assumed):

BLOCK 1 -

WORD 1 - integer # of signatures in the file

WORD 2 - integer # of channels. This must be the same for all signatures in all files being scaled at one time (NCHAN).

WORD 3 - integer # of words per signature

INTEGER $\left[\frac{\text{WORD 3 + 255}}{256}\right]$ = # of blocks per signature in input file

BLOCK 2. -

WORD 1→ 4 - 8 ASCII character signature ID

WORD 5 → (W-1) - vector of signature means with NCHAN double-precision, floating point elements

WORD W→ - the upper triangle of the symmetric, square covariance matrix. Each element should be a double-precision, floating point number. There must be $\frac{\text{NCHAN + 1}}{2}$ • NCHAN elements. This format is repeated with each signature file.

where W = NCHAN*4 + 5

PROCESSING:

MEANS:

The means used by MIDAS must be negated and scaled so that no more than 8 bits in two's complement notation are used. That is, the high-order bit (bit 7) is used as a sign. Therefore, the smallest number possible is:

$10000000_{\text{base 2}} = -128_{10}$

and the largest number possible is:

$01111111_{\text{base 2}} = +127_{10}$

The numbers coming from the signature file are in offset binary notation with the smallest integer number represented by

$00000000_{\text{base 2}} = 0_{10}$

and the largest integer number represented by

$11111111_{\text{base 2}} = 255_{10}$

The incoming double-precision, floating-point mean is negated. The constant 128_{10} is then added to the result to convert it to two's complement notation. This number is converted to integer and put in the output buffer. If an input mean is ≥ 256, an error message is printed.

VARIANCES:

The standard deviation for each channel is determined by calculating the square roots of the diagonal elements of the covariance matrix. The standard deviation is inverted, scaled, and converted to integer.

MIDAS presently has only 12 bits available for this number. Therefore, to obtain as many significant bits as possible, the standard deviations are subjected to the following restriction:

$$1 < \sigma$$

With this restriction, MAX $[1/\sigma] = 1.\emptyset - 2^{-12}$. In binary this is:

$$0.111111111111$$

This provides the maximum number of significant bits with no bits constantly $\emptyset$ or constantly 1. If a standard deviation is less than 1, the standard deviation is altered to make it just enough greater than 1 so that

$$\frac{1}{\sigma} = 1.\emptyset - 2^{-12}$$

Consequently, the processing procedure is as follows:

1. A diagonal element of the covariance matrix is found, and its square root is calculated. If no more, quit processing.
2. If the result is greater than 1, go to step 3. Otherwise, multiply the diagonal element by

$$\left[\frac{1}{(1 - 2^{-12})*\sigma}\right]^2$$

   Then multiply the column of the upper diagonal covariance matrix (except the diagonal element itself) by

$$\frac{1}{(1 - 2^{-12})*\sigma}$$

   Then go back to step 1 and select the same diagonal element again. If $\sigma = \emptyset$, σ is set equal to $\frac{1}{1 - 2^{-12}}$

3. Multiply this number by 2^{12}

$$\frac{1}{\sigma} * 2^{12}$$

4. Convert the result to integer and save it.
5. Go to step 1.

See write-up on STDDEV.

RAM COEFFICIENTS:

To understand the reasons behind all the following mathematical gyrations, see the mathematical analysis later in this section (see also Section 4). The processing goes as follows:

1. Calculate the correlation matrix from the covariance matrix.
2. Transpose the correlation matrix. This matrix must be transposed because the next routine to use it was written in FORTRAN and expects all arrays to be stored in column-by-column form instead of row-by-row form.
3. Calculate the eigenvalues and eigenvectors using the FORTRAN subroutine EIGEN.
4. Transpose the FORTRAN-form eigenvectors' matrix.
5. Treat the eigenvalues as the diagonal element of a square matrix with all off-diagonal elements set to zero.
6. Calculate the square root of each diagonal element and invert it.
7. Multiply this transformed matrix by the eigenvectors' matrix and store the results back in the eigenvectors' matrix.
8. Find the largest element in absolute value in the new matrix, negate it, divide every element of the new matrix by it, and save the constant.
9. Multiply the new matrix by 128 and convert it to integer. The low-order 8 bits are the only significant bits that MIDAS can handle.
10. Take the constant saved previously, square it, and store it in the output buffer following the coefficient matrix (stored in row-by-row form).

See RAMCOF writeup for a more detailed explanation of RAM Coefficients.

DETERMINANT:

This program does no scaling on the determinant. The determinant is found for the covariance matrix by performing a Gauss-Jordan inversion with partial pivotal elimination. The double-precision, floating point natural logarithm is calculated and stored in the output buffer.

ERRORS: (See CSI parameter /SW above)

1. ERR1,1. INPUT FILE IS A LINKED FILE. TRY AGAIN.
   All files used by the scaler must be contiguous files.
2. ERR1,2. INPUT FILE DOESN'T EXIST. TRY AGAIN.

3. ERR2. NCHAN ≤ Ø
   NCHAN must be a positive integer number greater than Ø. [FATAL]
4. ERR3. NCHAN NOT THE SAME FOR ALL SIG FILES
   NCHAN was different for two input signature file header blocks. [FATAL]
5. ERR4. NUMBER OF SIGNATURES ≤ Ø
   The integer number of signatures entered in a signature file header block has a value ≤ Ø. [FATAL]
6. ERR5. MORE THAN 5Ø SIGNATURES FOR SCALED FILE.
   The program can put only 5Ø scaled signatures or less in the scaled file because only one block is allowed for a directory. The header information for 5Ø signatures takes up the entire directory block. The scaled file is complete to and including the 5Øth scaled signature. [FATAL]
7. ERR6. MEAN ≥ 256.
   Mean values must not be greater than or equal to 256. [NONFATAL]
8. ERR7. VARIANCE ≥ 1.Ø.
   The value of $1/(\sigma)$ was too large for MIDAS. The number has been scaled again and the covariance matrix has been altered. Processing always continues. This is just a warning.
9. ERR8. EIGEN SUBROUTINE DIDN'T CONVERGE.
   The routine for calculating the eigenvalues and eigenvectors of the correlation matrix was unable to process the data given it. [NONFATAL]
10. ERR9. EIGENVALUE < Ø
    If a negative eigenvalue is found, then the original correlation matrix was not positive definite. This might indicate a data input error. [NONFATAL]
11. ERR1Ø. COVARIANCE MATRIX IS SINGULAR.
    The determinant of the covariance matrix was found to be zero or almost zero. Therefore, its natural logarithm cannot be calculated. [NONFATAL]
12. ERR11. DETERMINANT < Ø.
    The determinant of the covariance matrix was less than zero. This is mathematically impossible if the covariance matrix is positive definite. [NONFATAL]

13. ERR12. SCALED FILE TOO SMALL FOR REQUESTED SIGNATURES.
    The scaled signature file is not big enough for all the signatures requested for scaling. The signature number -1 indicates the number of signatures actually scaled. [FATAL]
14. ERR13,1. KEYBOARD ERROR
    There was an IO error. The status word is printed from the line buffer header. [FATAL]
15. ERR13,2. SCALED FILE TRAN BLOCK ERROR
    There was an output error. The status word is printed from the scaled file tran block. [FATAL]
16. ERR13,3. SIG FILE TRAN BLOCK ERROR
    There was an input error. The status word is printed from the signature file tran block. [FATAL]
17. ERR14. SIGNATURE FILE TOO SMALL FOR REQUESTED SIGNATURES
    The number of signatures declared in the header block for a signature input file was larger than the file. The signature number printed minus 1, is the number of the signature scaled last. [FATAL]
18. ERR15. STANDARD DEVIATION = Ø
    A standard deviation was found equal to zero making it impossible to calculate a complete correlation matrix. [NONFATAL]
19. SIGNATURE 3 = XXXXX
    Signature number XXXXX is the number of the signature currently being processed. This is relative to the first signature of the first input file. This is printed after an error message.

EXTERNAL CONSTANTS DEFINED: (all double-precision floating point numbers):

DFLTØ - Ø.
DFLT1 - 1.
DFLT4 - 4.
DFLT8 - 8.
DFLT2 - 2.
DFLT27 - 128.
DFLT29 - 512.
DFLT99 - -1.ØE2Ø
DFLT18 - Ø.125
DFLT97 - 32768.
DFLT96 - -2.44140625E-Ø4

SUBROUTINES NEEDED:

- STDDEV - this calculates the standard deviations and the $1/\sigma$ vector
- EIGEN - this calculates eigenvectors and eigenvalues
- RAMCOF - this calculates the coefficient matrix to be loaded into the MIDAS RAMs
- INVERS - this routine inverts a matrix and calculates the determinant of the original matrix at the same time
- SEEKFL - this routine finds the starting block # for any given filename. It also indicates whether the file is contiguous or linked.
- DSUB - calculates the floating point, double precision displacement of any element in a square matrix
- DSUB2 - calculates the floating point, double precision displacement of any element in an upper triangular matrix
- CORREL - calculates the correlation matrix from a given covariance matrix and standard deviation vector

DEVICES REQUIRED: KEYBOARD input
KEYBOARD output
If not specified to CSI { Disk signature file input / Disk scaled file output }

MACROS REQUIRED: .INIT, .WRITE, .TRAN, .CSI1, .CSI2, .WAIT, .RLSE, .EXIT, .READ, .BIN2D, .BIN2O

STORAGE REQUIREMENTS: 21342 octal locations

Mathematical Analysis. This system performs a maximum likelihood decision and assumes a multimodal Gaussian multivariate distribution. The function computed in the classifier is

$$pr(X) = \frac{1}{2\pi^{N/2}} \frac{1}{|\vartheta|^{1/2}} \text{EXP}\left[-1/2(X-\mu)^T \vartheta^{-1}(X-\mu)\right]$$

where X = input data vector
μ = mean vector
ϑ = variance - covariance matrix
$|\vartheta|$ = determinant of ϑ

Taking the natural logarithm of both sides

$$\ln[pr(X)] = -1/2(X-\mu)^T \vartheta^{-1}(X-\mu) + \ln|\vartheta|^{-1/2} + \ln(2\pi^{-N/2})$$

The decision rule is

$$\frac{pr(X_A)}{pr(X_B)} \quad \text{if } < 1 \text{ choose B; } > 1 \text{ choose A}$$

or

$$\ln|pr(X_A)| < \ln|pr(X_B)| \quad \text{choose B}$$

Substituting on both sides yields

$$-1/2(X-\mu_A)^T \vartheta_A^{-1}(X-\mu_A) + \cancel{\frac{\ln}{(2\pi^{-N/2})}} + \ln|\vartheta_A|^{-1/2}$$

$$< -1/2(X-\mu_B)^T \vartheta_B^{-1}(X-\mu_B) + \cancel{\frac{\ln}{(2\pi^{-N/2})}} + \ln|\vartheta_B|^{-1/2}$$

Choose B.

After multiplying through by -2

$$(X-\mu_A)^T \vartheta_A^{-1}(X-\mu_A) + \ln|\vartheta_A| < (X-\mu_B)^T \vartheta_B^{-1}(X-\mu_B) + \ln|\vartheta_B|$$

The covariance matrices ϑ_A and ϑ_B are both available, but the values within the matrices vary over a wide range. This is particularly a problem with MIDAS because of its limited bit capacity. It would be desirable to compress the same information from the covariance matrix into a matrix whose values did not vary so widely and would require a minimum bit capacity to represent all but the extreme points of the matrix's full range.

This covariance matrix scaling can be achieved by use of the following substitution.

$$[\vartheta] = [\sigma][\rho][\sigma]$$

where σ = diagonal matrix of standard deviations

ρ = correlation matrix with 1's on the diagonal and numbers between Ø and 1 off the diagonal

Inverting ϑ yields

$$\vartheta^{-1} = \left[\frac{1}{\sigma}\right]\left[\rho\right]^{-1}\left[\frac{1}{\sigma}\right]$$

Substituting into $\ln[pr(X)]$

$$\frac{(X-\mu_A)^T}{\sigma_A}\rho_A^{-1}\frac{(X-\mu_A)}{\sigma_A} + \ln|\vartheta_A| < \frac{(X-\mu_B)^T}{\sigma_B}\rho_B^{-1}\frac{(X-\mu_B)}{\sigma_B} + \ln|\vartheta_B|$$

The $\frac{X-\mu}{\sigma}$ term still has a wide range, but the following coarse restriction will eliminate those X values too far from the mean to be considered for classification.

$$-8 \leq \frac{(X-\mu)}{\sigma} \leq 8$$

This equation can be simplified further by decomposing ρ^{-1}. This decomposition is achieved through eigenvalue analysis. The solution of the following equation produces the eigenvalues and their associated eigenvectors.

$$\rho X = \lambda X \tag{1}$$

where λ = an eigenvalue

X = an eigenvector

also

$$\rho^T Y = \lambda Y$$

where Y = an eigenvector

λ remains the same because the characteristic equation remains the same.

For example

$$\det [\rho^T - \lambda I] = \det [\rho - \lambda I]$$

The set of X vectors and Y vectors that satisfy the relations in Eq. 1 are called the U and V eigenvectors' matrices respectively.

U is associated with ρ

V is associated with ρ^T

λ is the same for ρ and ρ^T

The individual eigenvectors appear in the columns of both matrices. Since ρ is a symmetric matrix about its diagonal in our case, then U and V are the same matrix. Throughout the following analysis, V may be replaced by U with no change to the result.

It can be proven that

$$U^T V = I \text{ and } V^T U = I \tag{2}$$

Let us create a new matrix

$$\Lambda = \lambda I$$

where I = the identity matrix with 1's on its diagonal and zeros everywhere else.

Λ = a diagonal matrix with the eigenvalues on its diagonal and zeros everywhere else.

It can also be proven that

$$\rho U = U\Lambda \tag{3}$$

$$\rho^T V = V\Lambda \tag{4}$$

$$\Lambda^T = \Lambda$$

therefore, from Eq. (3)

$$\rho U V^T = U\Lambda V^T$$

From Eq. (2) this dissolves into

$$\rho = U\Lambda V^T \tag{5}$$

Also, from Eq. (4)

$$\rho^T V U^T = V\Lambda U^T$$

From Eq. (2) this dissolves into,

$$\rho^T = V\Lambda U^T \tag{6}$$

Rewriting Eq. (1)

$$\rho X = \lambda X \Rightarrow \rho U = \Lambda$$

$$\rho^{-1}\rho U = \rho^{-1}\Lambda U$$

$$U = \rho^{-1}\Lambda U$$

$$\rho^{-1}U = \Lambda^{-1}U$$

Therefore, U appears to be the eigenvectors matrix for ρ^{-1} as well as ρ. The only change is that the eigenvalues for ρ^{-1} are the reciprocal of the eigenvalues for ρ. Finally, the result appears for which we have all been waiting.

From Eq. (5),

$$\rho^{-1} = U\Lambda^{-1}V^T \tag{7}$$

and from Eq. (6),

$$[\rho^T]^{-1} = V\Lambda^{-1}U^T \tag{8}$$

Since Λ is a diagonal matrix, the inverse of Λ is simply the reciprocal of each diagonal element. Also

$$\Lambda^{-1} = \Lambda^{-1/2} \times \Lambda^{-1/2}$$

Therefore, from Eq. (7),

$$\rho^{-1} = U\Lambda^{-1/2}\Lambda^{-1/2}V^T \tag{9}$$

and from Eq. (8),

$$[\rho^T]^{-1} = V\Lambda^{-1/2} \times \Lambda^{-1/2}U^T \tag{10}$$

Substituting Eq. (9) back into $\ell n\,[pr(X)]$ and replacing the V matrix with the U matrix (because U and V are identical) yields

$$\left[\frac{X-\mu_A}{\sigma_A}\right]^T U_A\Lambda^{-1/2} \times \Lambda^{-1/2}U_A^T\left[\frac{X-\mu_A}{\sigma_A}\right] + \ell n\,|\theta_A|$$

$$< \left[\frac{X-\mu_B}{\sigma_B}\right]^T U_B\Lambda^{-1/2} \times \Lambda^{-1/2}U_B^T\left[\frac{X-\mu_B}{\sigma_B}\right] + \ell n\,|\theta_B|$$

The only difference between

$$\left[\frac{X-\mu}{\sigma}\right]^T U\Lambda^{-1/2}$$

and

$$\Lambda^{-1/2}U^T\left[\frac{X-\mu}{\sigma}\right]$$

is that the former yields a row vector and the latter yields a column vector. The respective elements are identical. Therefore, the same result can be achieved by calculating

$$\Lambda^{-1/2}U^T\left[\frac{X-\mu}{\sigma}\right]$$

The vector elements can then be squared. The sum of these squares yields the same result as performing the matrix multiplications indicated on both sides.

Therefore, all the scaling program has to do is calculate the following:

$-\mu$ = a vector of negated means

$\frac{1}{\sigma}$ = a vector of variances

U^T the transpose of the eigenvector matrix (the eigenvectors end up in the rows, assuming they were originally in the columns)

$\Lambda^{-1/2}$ = a diagonal matrix with the square root of the reciprocal of each eigenvalue on its diagonal with zeros everywhere else.

$\Lambda^{-1/2}U^T$ = the matrix of coefficients to be put into the memory of the MIDAS Classifier

For a more complete description of the eigenvalue analysis applied here, see Ref. [3].

3.2.6 RAMLD

TITLE: RAMLD - COEFFICIENT LOADER

CSECT: None

PURPOSE: This program allows the user to select a set of signatures from a disk file (generated by the SCALER program) and load the appropriate coefficients into the random-access memory (RAM) of the MIDAS Classifier hardware in preparation for classification of multispectral data. (See Fig. 10.)

CALLING SEQUENCE: None

PARAMETERS: Operating parameters are input via the console keyboard in response to the following requests.

1. ENTER INPUT SIGNATURE FILENAME =

This requests the filename and extension for the file to be used as input. This input file must be created by the SCALER program to insure proper data format. The proper response format is 'NNNNNN.EEE' where N represents a character in the filename (blanks are significant characters) and E represents a character in the extension. (See Figs. 11, 12, and 14.)

2. SIGNATURE NOS. TO BE USED =

This requests the selection of a subset of signatures from those present in the file specified by the response to Request 1.

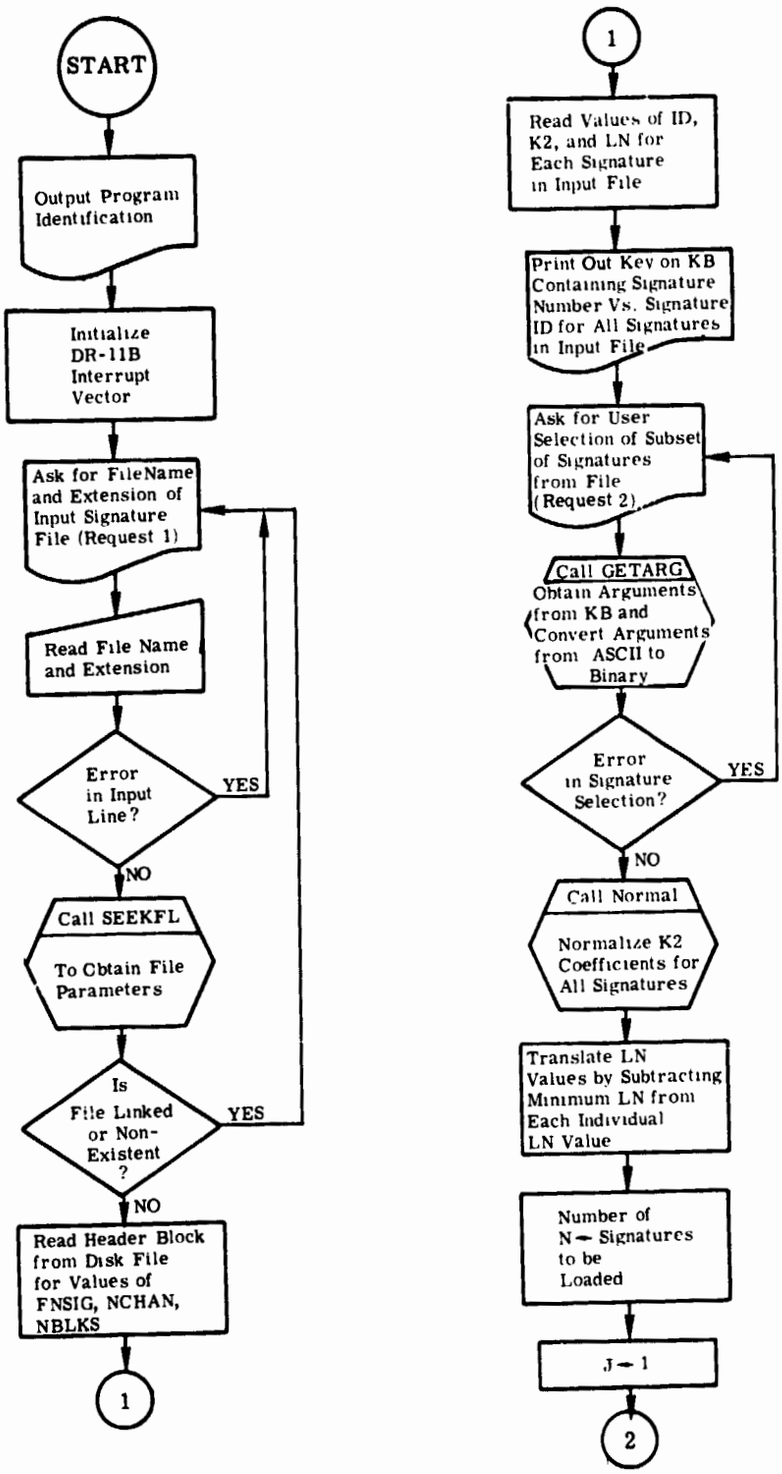


FIGURE 10. FLOWCHART FOR RAMLD (Continued)

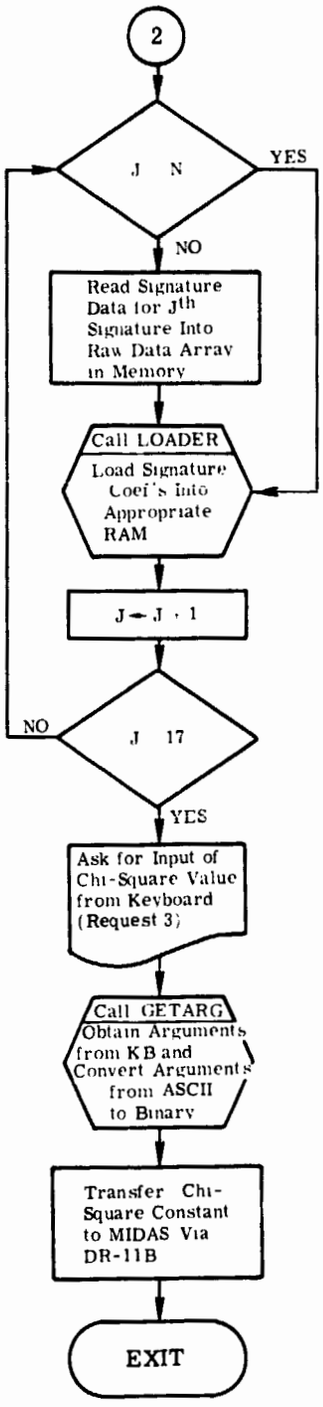


FIGURE 10. FLOWCHART FOR RAMLD (Concluded)

```
$LO 60,40
DATE:-22-FEB-74
TIME -00.03 21
$RUN RAMLD

CLASSIFIER COEFFICIENT LOADER ** V.1.0 ** MFG ** OCT 1973

ENTER INPUT SIGNATURE FILENAME = YNP    .SCL

SIG NO        ID
--------    --------
 00001      LODGE D1
 00002      LODGE D2
 00003      SP FIR
 00004      GR 3
 00005      LT RK 3
 00006      GR 2
 00007      LT RK 2
 00008      LO SHRUB
 00009      GRLANDS
 00010      GR/BR
 03011      BRUSH
 00012      DK RK
 00013      LT RK
 00014      THERMDEP
 00015      WATER

SIGNATURE NOS. TO BE USED = 1,2,3,4,5,10,11,13,15

ENTER CHI-SQUARE VALUE (0 - 1000) = 1000

$
```

FIGURE 11. COEFFICIENT LOADING OF A SUBSET OF SIGNATURES FROM FILE YNP.SCL
Note that in a computer printout, user-supplied responses will be underscored.

```
$RUN RAMLD

CLASSIFIER COEFFICIENT LOADER ** V.1.0 ** MFG ** OCT 1973

ENTER INPUT SIGNATURE FILENAME = YNP    .SCL

SIG NO.        ID
-------    --------
 00001     LODGE D1
 00002     LODGE D2
 00003     SP FIR
 00004     GR 3
 00005     LT RK 3
 00006     GR 2
 00007     LT RK 2
 00008     LO SHRUB
 00009     GRLANDS
 00010     GR/BR
 00011     BRUSH
 00012     DK RK
 00013     LT RK
 00014     THERMDEP
 00015     WATER

SIGNATURE NOS. TO BE USED = ALL

ENTER CHI-SQUARE VALUE (0 - 1000) = 10

$
```

FIGURE 12. COEFFICIENT LOADING OF ENTIRE SET OF SIGNATURES FROM FILE YNP.SCL USING "ALL" FEATURE
Note that in a computer printout, user-supplied responses will be underscored.

Input is made via a call to the subroutine GETARG, and thus the input string must follow the format necessary for input to the GETARG routine (see write-up on GETARG routine). This input will be a set of decimal-numeric strings (separated by commas) specifying the signatures to be loaded. (See Fig. 11.) If all of the signatures are to be loaded, a proper response to this request would be the 3-character string 'ALL.' (See Fig. 12.)

3. ENTER CHI-SQUARE VALUE (Ø-1ØØØ) =
This requests the input of a value for the chi-square test of the exponent in the classifier. A proper response to this request is a decimal number in the range Ø to 1ØØØ of 1 to 4 characters.

DESCRIPTION: This program is designed to use as input a signature file created by the SCALER program, such that a set of 4- or 8-channel signature coefficients are loaded into the MIDAS Classifier hardware RAMs.

There are five sets of coefficients to be loaded into MIDAS for each signature. (See Fig. 13.) (1) a mean vector μ, (2) a $(1/\sigma)$ pseudo-variance vector V, (3) a square eigenvector matrix U, (4) a normalization coefficient K2, and (5) an additive constant LN, corresponding to the natural logarithm of the determinant of the matrix U.

The functional flow of this program can be obtained from the accompanying flowchart. Certain aspects, however, will now be described in more detail.

(1) The normalization of the K2 values is performed by the subroutine NORMAL. This routine divides each K2 element by the maximum of all the K2 elements and then converts this floating-point value to a 12-bit unsigned integer, such that Ø.Ø corresponds to integer Ø and 1.Ø corresponds to integer 7777_8. These 12-bit integers are passed to the RAMs of the MIDAS hardware.

(2) The translation of the LN values is performed by subtracting the minimum LN from all of the LN values.

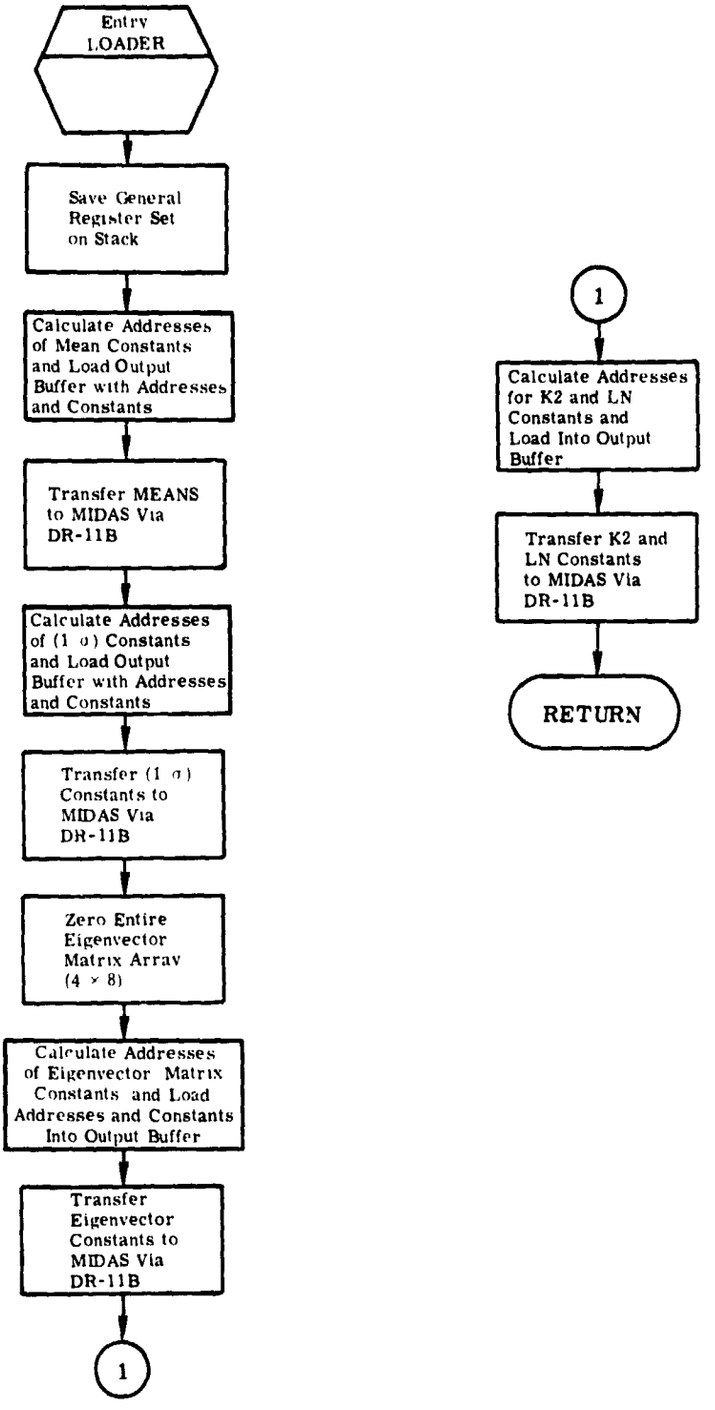


FIGURE 13. FLOWCHART FOR LOADER

(3) The addressing scheme which is used to pass coefficients to the proper RAMs for MIDAS is described in Appendix A.

ERRORS: The following error messages are possible:

(1) ***ERROR - INPUT FILENAME NON-EXISTENT

This error is caused by a non-existent filename being entered as a response to Request 1. Check the format of response to Request 1 or use PIP* to check for proper filename.

(2) ***ERROR - INPUT FILE LINKED

Since the output from SCALER is always a contiguous file, the filename specified in response to Request 1 must also be contiguous.

(3) ***ERROR - SIGNATURE NO. TOO LARGE

In the response to Request 2, a signature number greater than the number of signatures available from the input file was entered (Fig. 14). Signature numbers will have to be reentered.

(4) ***ERROR - ILLEGAL CHARACTER IN STRING

In the response to Request 2, the format necessary for input to the routine GETARG was not followed. Signature numbers will have to be reentered.

(5) ***ERROR - NO. OF SIGNATURES > 16.

More than 16 signatures were selected to be loaded into the MIDAS RAMs (Fig. 14). In the 4-channel case, 16 is the maximum number of signatures; whereas, in the 8-channel case, only 8 signatures may be loaded.

(6) ***ERROR IN DR-11B-DRST IN SWR

A hardware malfunction occurred in the DR-11B used to transfer data to the MIDAS hardware. The DR-11B status register is in the console display register. Diagnostics on the MIDAS and DR-11B hardware should be initiated.

STORAGE REQUIREMENTS: $1\emptyset376_8$ bytes

SUBROUTINES NEEDED: GETARG
SEEKFL
NORMAL

*PERIPHERAL INTERCHANGE PROGRAM - DOS system program for file manipulation and maintenance.

```
$RU RAMLD

CLASSIFIER COEFFICIENT LOADER ** V.1.0 ** MFG ** OCT 1973

ENTER INPUT SIGNATURE FILENAME = YNP    .SCL

SIG NO.       ID
-------    --------
 00001     LODGE D1
 00002     LODGE D2
 00003     SP FIR
 00004     GR 3
 00005     LT RK 3
 00006     GR 2
 00007     LT RK 2
 00008     LO SHRUB
 00009     GRLANDS
 00010     GR/BR
 00011     BRUSH
 00012     DK RK
 00013     LT RK
 00014     THERMDEP
 00015     WATER

SIGNATURE NOS. TO BE USED = 1,2,3,4,5,16,10,11

*** ERROR - SIGNATURE NO. TOO LARGE

SIGNATURE NOS. TO BE USED = 1,2,3,4,5,6,7,8,9,10,11,12,13,15,1,2

*** ERROR - TOO MANY SIGNATURE NOS.

SIGNATURE NOS. TO BE USED = 1,2,3,4,5,6,7,8,9,10,11,12,13,14,15

ENTER CHI-SQUARE VALUE (0 - 1000) = 1000

$
```

FIGURE 14. COEFFICIENT LOADING WITH INPUT ERRORS FLAGGED BY PROGRAM
Note that in a computer printout, user-supplied responses will be underscored.

3.2.7 CLAS1 AND CLAS2D

TITLE: CLAS1 - Pass 1. Classification

CSECT: None

PURPOSE: This main program establishes the parameters of the classification run to be made, performs the system setup necessary, creates an intermediate disk file describing the system setup performed, and turns control over to the module CLAS2D for the actual classification run. (See Figs. 15, 16.)

CALLING SEQUENCE: None

PARAMETERS: Operating parameters are input via the console keyboard in response to the following requests:

1. IS OUTPUT TO DISK DESIRED ?

This question establishes whether an output file of the classification results is to be created on the disk during the classification run. The only proper negative request is 'NO'; any other response is assumed to be affirmative. If the response is negative, Requests #2 and #7 (below) will not be made.

2. ENTER OUTPUT FILENAME =

This request establishes the name of the file to be created to contain the results of the classification. Proper format for the response is NNNNNN.EEE where N is a character in the filename proper, and E is a character in the file's extension.

3. ENTER TYPE OF CLASSIFY INPUT =

This request establishes the type of input which will be fed through the classifier. System setup is performed depending on the type of input mode. Proper responses are either 'DIGITAL' or 'ANALOG,' with the first character sufficing for either. NOTE: In Phase I, the analog data input mode has not yet been implemented; thus the digital data mode is assumed, after printing out the warning:

ANALOG CLASSIFY NOT IMPLEMENTED ** 8-OCT-73
DIGITAL INPUT ASSUMED

4. MOUNT DIGITAL INPUT TAPE ON UNIT Ø (9-TRACK)
ENTER <CR> WHEN INPUT READY

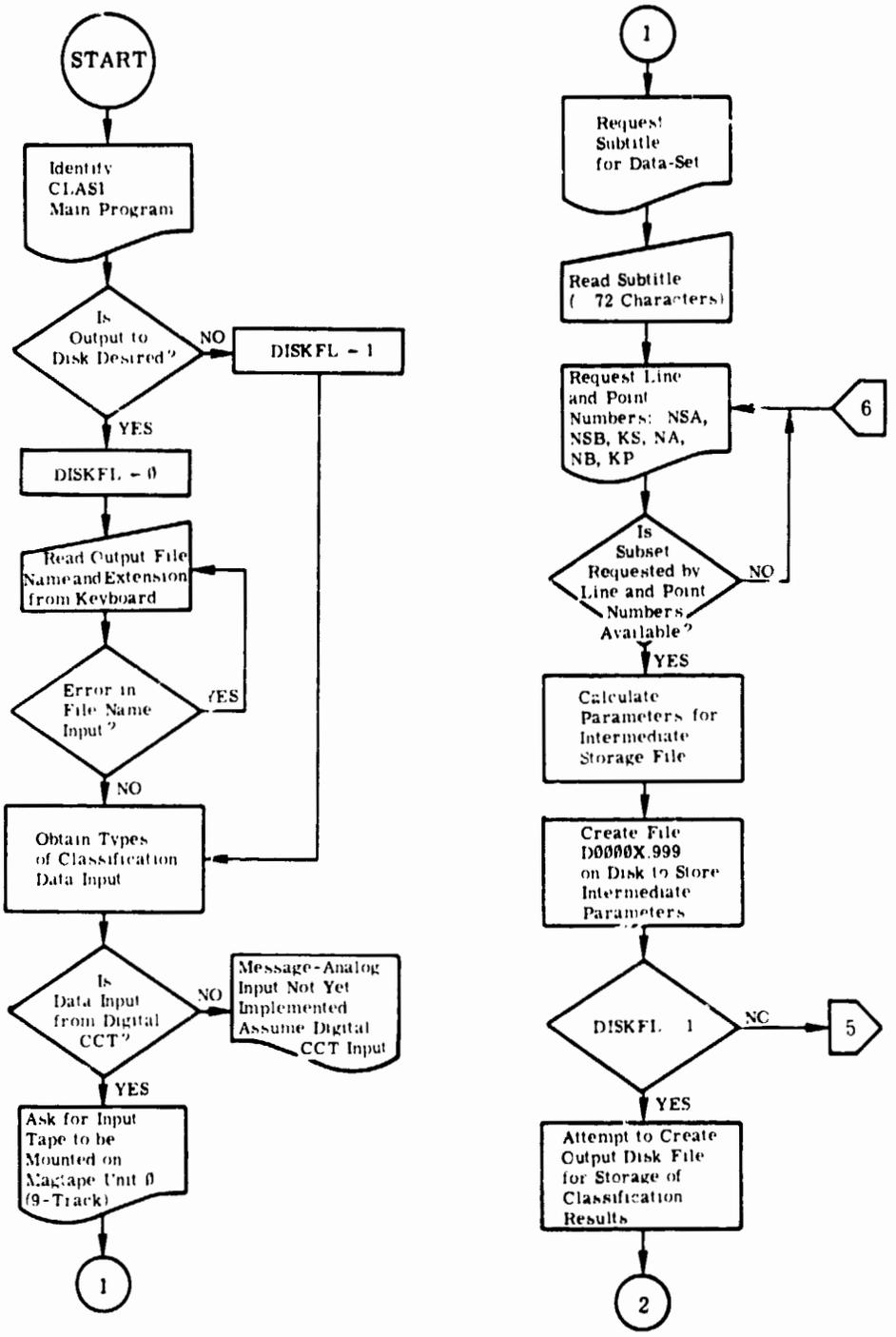


FIGURE 15. FLOWCHART FOR CLAS1 (Continued)

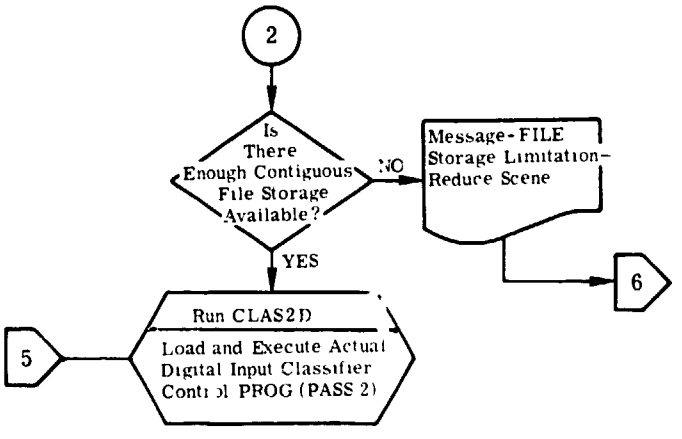


FIGURE 15. FLOWCHART FOR CLAS1 (Concluded)

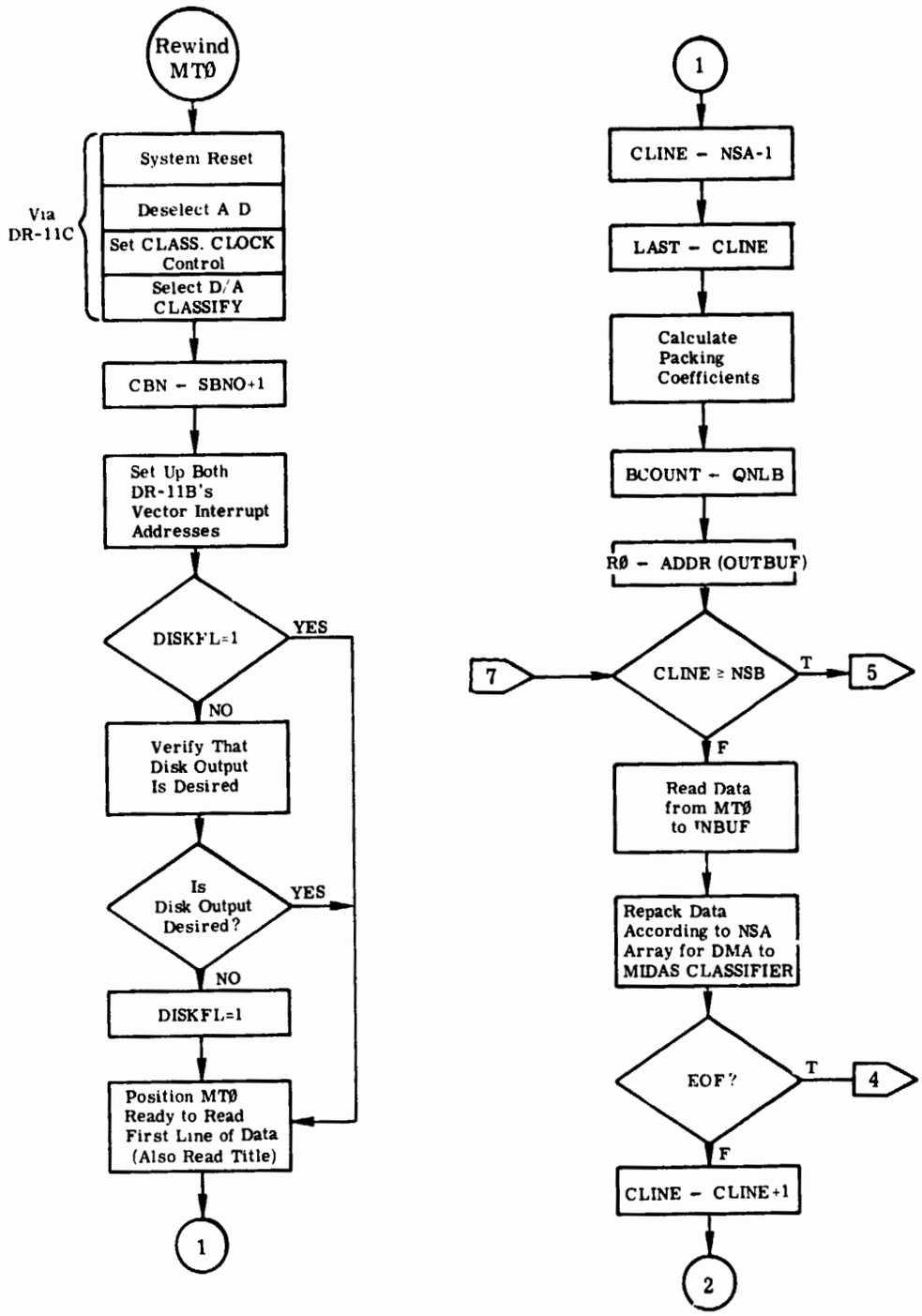


FIGURE 16. FLOWCHART FOR CLAS2D (Continued)

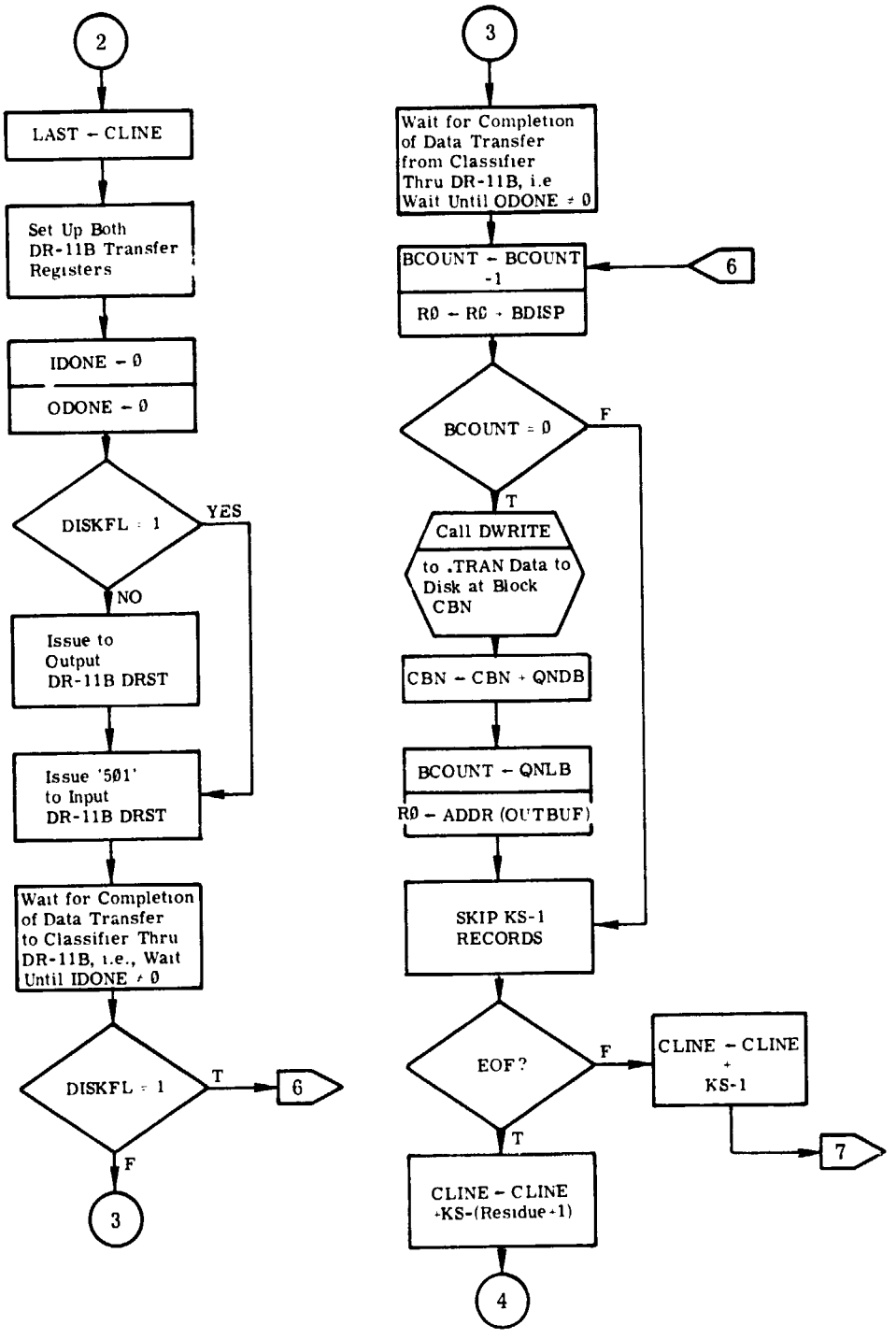


FIGURE 16. FLOWCHART FOR CLAS2D (Continued)

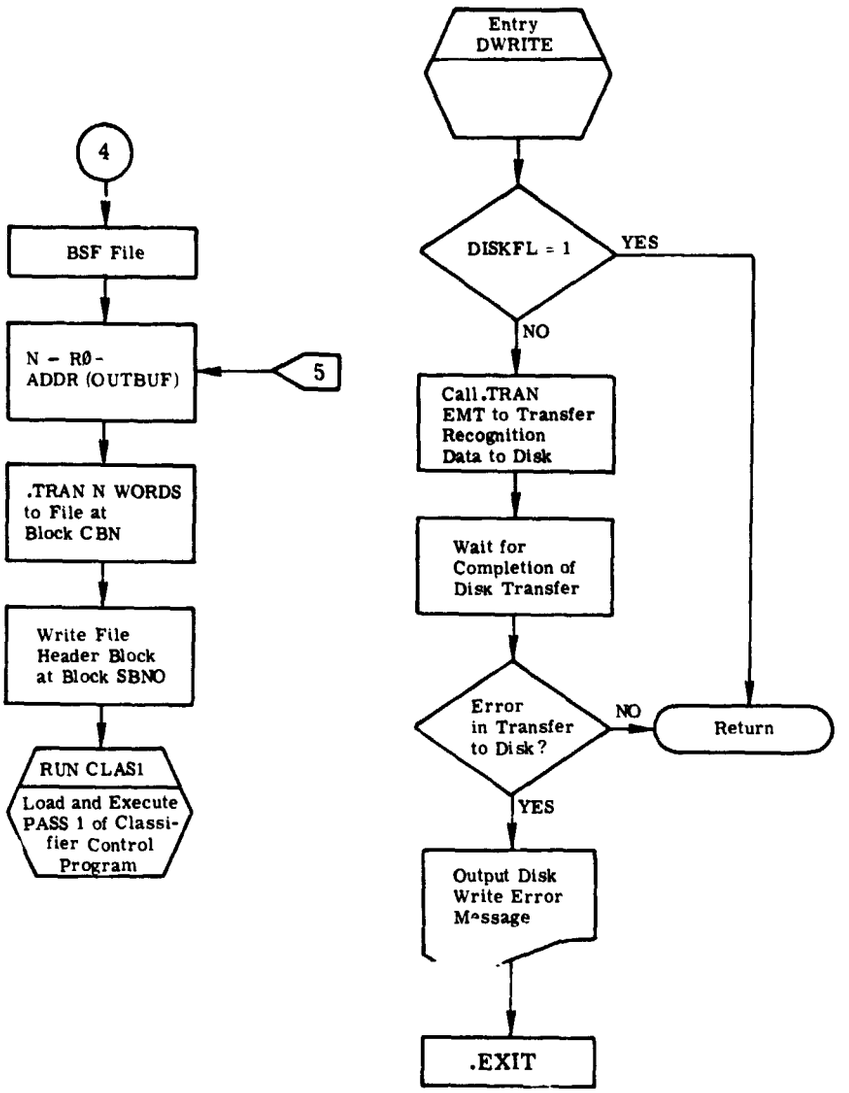


FIGURE 16. FLOWCHART FOR CLAS2D (Concluded)

This request informs the operator that the input tape should be mounted on magtape unit Ø, on a 9-track drive. Proper response when the tapes are ready is any string of characters followed by a carriage return (no preceding characters are necessary). If no drive Ø has been established, an 'AØØ2 XXXXX' error warning message will be printed out on the console. When the error condition has been corrected (i.e., the input tape mounted properly on unit Ø), the proper response to continue is 'CO' followed by a carriage return.

5. ENTER SUBTITLE =

This request establishes a subtitle or secondary descriptor for the particular data-set to be classified in the upcoming run. Proper response is a string of up to 72 characters terminated by a carriage return.

6. ENTER LINE AND POINT NOS. =

This request establishes the rectangular boundaries of the scene to be classified. This scene is usually some subset of the entire input data-set. Proper response to this request is a sequence of six numbers separated by commas, where the six numbers represent the following:

NSA - the starting line (record) number
NSB - the ending line (record) number
KS - the line (record) increment
NA - the starting pixel number
NB - the ending pixel number
KP - the pixel increment

For example, an input string of 1ØØ, 1ØØØ, 1Ø, 15Ø, 8ØØ, 3 would indicate a subset of data starting with the 1ØØth data record on the tape and ending with the 1ØØØth data record; reading only every 1Øth record, then, the records used would be 1ØØ, 11Ø, 12Ø, ... , 98Ø, 99Ø, 1ØØØ. Furthermore, for each data record, the pixels (resolution elements) used, start with number 15Ø and go through 798, i.e., 15Ø, 153, 156, 159, ... , 792, 795, 798. Note that 8ØØ was not used because, according to the increment KP, point 8Ø1 would be the next logical pixel—but this is greater than the upper point limit of 8ØØ.

If an improper response is given, no error message is printed out; however, the request line is reprinted on the console, and proper input should be made. Improper inputs consist of other than six numbers in the string or a non-numeric character other than a comma.

7. EXPONENT TO DISK DESIRED ?
This request establishes whether or not the exponent calculated for each pixel classification is to be stored along with the actual 5-bit classification result. If so, a full word (16 bits) will be stored on disk per pixel input and, therefore, classified; the high-order 11 bits will be the exponent value, whereas the low-order 5 bits will be the classification result. Otherwise, if the exponent is not desired, classification results will be packed 2 to a word, with bits Ø-4 representing the classification result for pixel n and bits 8-12 representing the classification result for pixel n+1.

Proper responses are 'YES' or 'NO' with the first character sufficing for either. A negative response is assumed if a response other than the two above (or an allowed abbreviation) is made.

ERRORS: An error is possible if output of classification results to disk storage is desired. If the amount of contiguous disk storage necessary to store the entire scene selected by the line and point number input is not available, a warning 'FILE STORAGE LIMITATION - REDUCE SCENE' will be printed on the console. A branch will be effected to the point in the program where request #6 is made. At this point, one may either reduce the size of the data-set by reducing the line and point numbers, or by not storing the exponent value.

DESCRIPTION: This program is used to establish the classification parameters of the run to be made using the MIDAS hardware. No actual hardware initializing is performed in the program; however a temporary disk file is created containing the descriptive parameters entered from the console by the operator. After these parameters have been placed in the disk file (currently always named DØØØØX.999), the module CLAS2D (or CLAS2A when

analog classification is fully implemented) is loaded into memory and executed, thus replacing CLAS1 as the controlling program.

STORAGE REQUIREMENTS: 3350_8 bytes

SUBROUTINES NEEDED: GETFIL
GETARG

TITLE: CLAS2D - Pass 2: Classification (Digital)

CSECT: None

PURPOSE: This program, using system descriptive information obtained from a disk file created by CLAS1, performs actual hardware setup, initiates the classification, and controls IO functions on a line-by-line basis for both input and output of data. This includes data transfers directly to the MIDAS Classifier either from or to memory, and also the transfer of data to memory from magnetic tape and transfer of data (classification results) to the disk.

CALLING SEQUENCE: None

PARAMETERS: General system descriptors obtained previously by CLAS1 are stored in a file D0000X.999, which is used only for transmission of information between CLAS1 and CLAS2D.

In addition, if disk output of classification results was requested during CLAS1, additional verification of disk output is requested by the following:

IS OUTPUT TO DISK DESIRED ?
Proper response to this request is either 'YES' or 'NO' with the first character sufficing in either case. An affirmative response is assumed if neither of the two proper responses (or an allowed abbreviation) is given in response to the request.

DESCRIPTION: This program provides all of the control of data transfer to and from the MIDAS Classifier, and between memory and peripheral data storage devices, i.e., magnetic tape and random-access disk.

Control of data transfer is performed on a line-by-line basis; no buffering of input data is performed because the tape drive is moving at its maximum rate with unbuffered tape input.

Data transfers to and from the MIDAS hardware are interrupt-driven—i.e., the conclusion of a transfer in or out of MIDAS via the two DR-11B general device interfaces causes an interrupt through a unique hardware location for each data transfer.

Output to disk is buffered into a region 9720_{10} 16-bit words long, making output to disk necessary much more infrequently than every line.

Initially, the MIDAS hardware is set up by the transfer of various appropriate command words via the DR-11C general device interface. These MIDAS commands are detailed in Section 6.1.

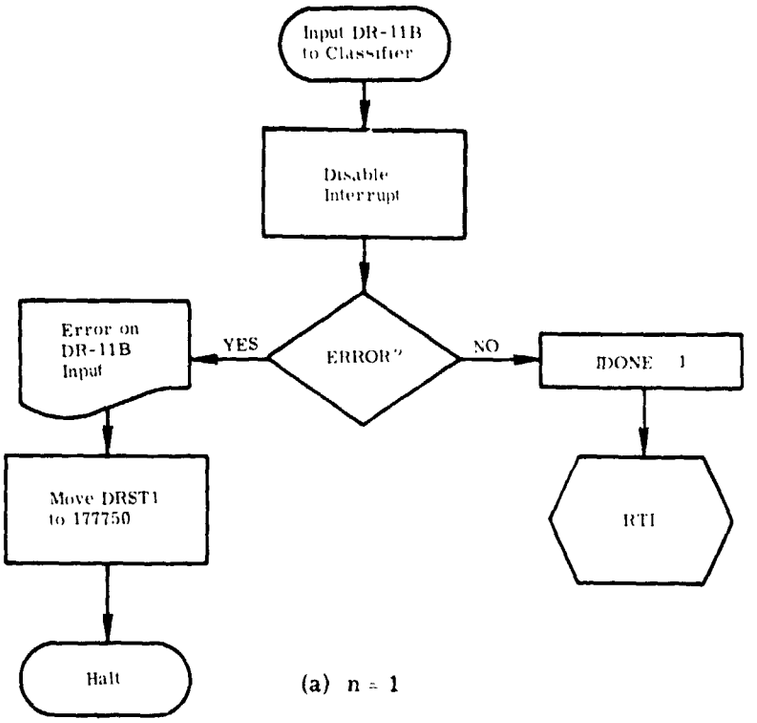
ERRORS: The following error returns are possible:

1. FATAL DISK ERROR - RETRY COUNT = 5 DISPLAY REGISTER CONTAINS DRST

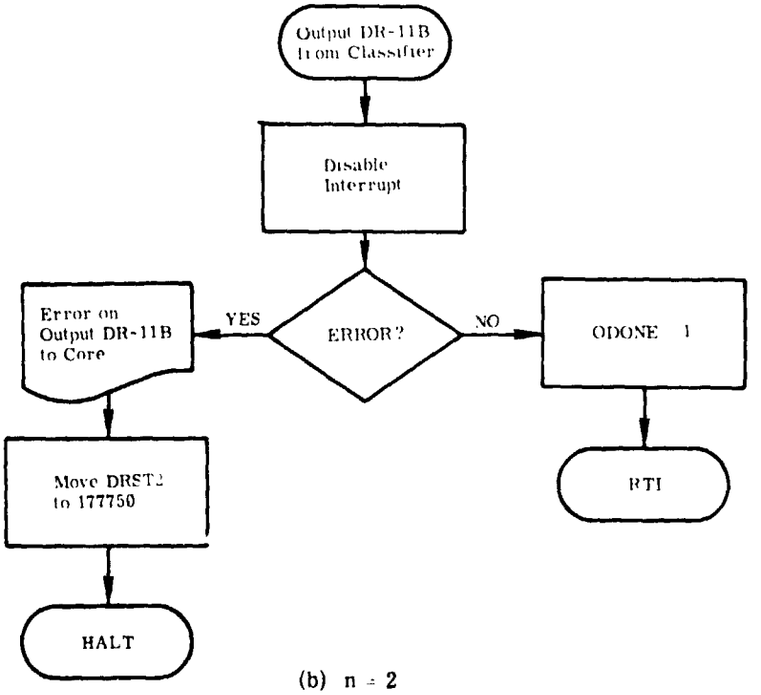
This error indicates that a hardware error on the RK-Ø5 disk drive makes impossible the correct transfer of data to disk. The write operation, though attempted 5 times, was unsuccessful. The initial disk block number of the attempt write operation is placed in the console display register for operator information. The classification run is then terminated and return is made to DOS MONITOR command mode via the .EXIT EMT call. Diagnostics on the RK-Ø5 disk drive should be initiated immediately by qualified software system personnel.

2. DR-11B #n ERROR - DISPLAY CONTAINS DRST

The contents of the appropriate DR-11B's status register (DRST) are placed in the console display register for operator information. The classification run is then terminated and return is made to DOS MONITOR command mode via the .EXIT EMT call. Diagnostics on the appropriate DR-11B should be initiated immediately by qualified software personnel. (See Fig. 17.)



(a) n = 1



(b) n = 2

FIGURE 17. DR-11B INTERRUPT ROUTINES

n = 1 indicates a malfunction in the DR-11B used to transfer input data to the MIDAS Classifier.

n = 2 indicates a malfunction in the DR-11B used to transfer classification results back to the PDP-11/45 core memory.

STORAGE REQUIREMENTS: 57612_8 bytes

SUBROUTINES REQUIRED: COMAND
BN2BCD

4
SYSTEM CONFIGURATION

4.1 APPROACH

A special-purpose computer used for recognition (classification) of remotely sensed objects may take several forms. In general, the system design indicates that a general purpose computer be used for statistical analyses and for control and monitoring of the special purpose machine. The special purpose machine could be implemented using analog or digital techniques.

Analog techniques for classifying objects using multispectral data have been well developed over the past few years and, until very recently, appeared superior in cost and complexity to digital techniques in which a special purpose mechanization would be used. One of the most significant cost-components, the multiplier, was previously estimated to cost on the order of several hundred to a thousand dollars, and would be replicated, in the worst case, about 300 times. Two significant changes have taken place: (1) the cost per 8-bit multiplier has declined substantially, and (2) the speed of such units has increased to a range of 70 to 200 nsec/product. As a result, the multiplier can be built to be time-shared in computing several products per input sample (5 μsec) so that a factor of cost reduction may be had ranging between 5 and 10. Labor costs for wiring the medium scale integration ALUs (Arithmetic Logic Units) rather than the small scale integration dual in-line packages (DIPs) can be reduced by significant factors also. The classifier in this configuration, then, has the same organization as the previously operated analog system (SPARC) but has its functional components implemented in time-shared digital circuitry. Each component acts as a computing element in a hard-wired sequence, which may be thought of as a pipeline or cascade of computational circuits. The desired processing rate is comparable with the SPARC system, that is, multispectral data are processed at a rate equal to that at which it was collected, i.e., 2×10^5 resolution elements per second.

An ERTS frame (8×10^6 resolution elements) can be classified, element by element, in about 40 seconds provided that the data is recorded in a form such tha' it can be supplied at a rate of 2×10^5 pixels/second. This is, of course, not the case for CCTs (Computer Compatible Tapes) presently supplied which can only allow rates one-seventh to one-twentieth of those possible with high density tape.

The computation this system performs is a maximum-likelihood decision, assuming a multimodal Gaussian multivariate distribution. This assumption has been well justified by more than 100 experiments using multispectral data at ERIM [4] by a similar number at LARS and, as time goes on, by more and more experience at NASA and other agencies. Although simpler algorithms can perform well for some data-sets, a significant percentage of

applications demand this powerful decision rule. No penalty in speed and only a small additional cost occurs in using this algorithm, hence it is employed.

Thus, the basic calculation to be performed is

$\ln\{pr(X)\}$

where X is the input data vector and the probability density function is a Gaussian density function:

$$\ln\{pr(X)\} = -\frac{1}{2}\left\{(X - M)^T \theta^{-1}(X - M) + \ln|\theta| + n \ln\left(\frac{\pi}{2}\right)\right\}$$

The

$$(X - M)^T \theta^{-1}(X - M) = Q \tag{11}$$

is a quadratic calculation in which M is the mean vector for each distribution and θ^{-1} is the inverse of the variance-covariance matrix. This computation must be performed for each object class included in the classification process and the number of computational steps increases as the square of the number of channels. Therefore, for one MIDAS Classifier configuration, the number of channels is limited to eight, and the number of object classes is limited to eight, while the same hardware switched to another configuration limits the number of channels to four but increases the allowable number of object classes to sixteen. This implementation uses about 40 digital multipliers. This same size classifier, if implemented as a hybrid processor, would have required about 210 multiplying D/A converters. Since the cost of each was comparable, the choice was clearly in favor of an all-digital classifier with characteristics as summarized in Tables 3 and 4.

4.2 ORGANIZATION

The Phase I MIDAS System can be visualized most readily if organized into subsystems as shown in the block diagram of Fig. 18. The system is under complete control of the Digital Equipment Corporation (DEC) PDP-11/45 computer system. All control inputs are made by an operator via the computer keyboard. All commands are translated by computer software into code words sent out over interface devices to set up the hardware registers in the special-purpose processor. These codes are decoded in three of the blocks shown in Fig. 18: (1) in the Control section, (2) in the Clock section, and (3) in the Diagnostic/Output section. The codes will be described in detail in subsequent sections.

High-speed mass data transfer to and from the computer takes place from (1) the A/D-D/A (hybrid) section, (2) the Clock section, and (3) the Diagnostic/Output section. High-speed

TABLE 3. ADVANTAGES OF DIGITAL PARALLEL CLASSIFIER

Uses current state-of-the-art digital techniques
Less costly than current hybrid techniques
Complete repeatability in setup and performance
Computer-controlled diagnostics easily implemented for error-free operation
Throughput equal to current analog hybrid techniques

TABLE 4. CHARACTERISTICS OF PROTOTYPE PROCESSOR

All-digital parallel classifier under computer control
Classification rate of 200,000 data vectors/second
Classification of data vector into one of eight stored categories for eight-channel data
Classification of data vector into one of sixteen stored categories for four-channel data
Potential of expansion to sixteen-channel capability

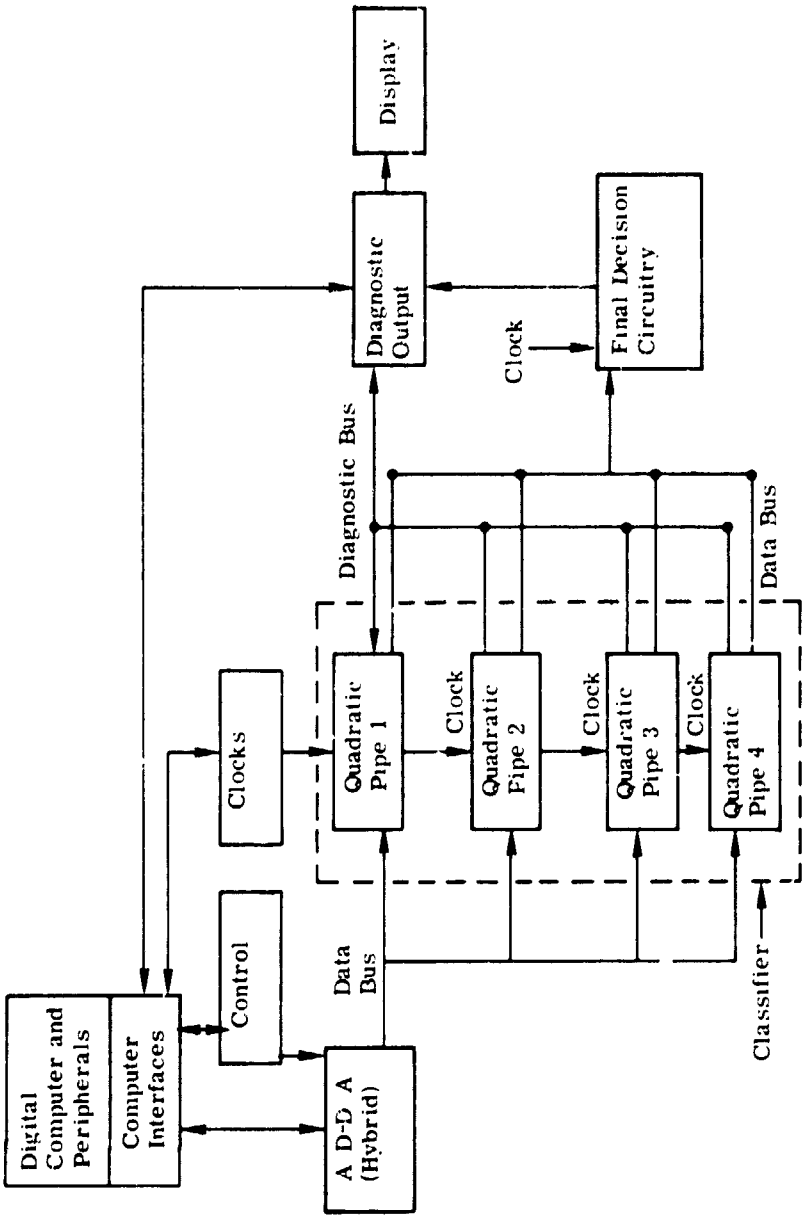


FIGURE 18. BLOCK DIAGRAM OF THE PHASE I MIDAS SYSTEM

multichannel data transfer takes place through the hybrid section and the quadratic computation hardware.

4.2.1 QUADRATIC PIPE COMPUTATION

The quadratic pipe computers perform the quadratic calculation given by Eq. (11):

$$Q = (X - M)^T \theta^{-1} (X - M)$$

This calculation can be expressed in a number of ways to optimize the computation. Since the number of bits in the special-purpose classifier is limited, it is desirable to express the quadratic calculation such that the calculated result has a very limited range. The variance-covariance matrix θ can be expressed as

$$[\theta] = [\sigma][\rho][\sigma] \tag{12}$$

where $[\sigma]$ is a diagonal matrix of the standard deviation, and $[\rho]$ is the correlation matrix with all 1's on the diagonal and values of 0 to 1 off the diagonal (in some cases negative values may occur). Taking the inverse of (12) yields

$$[\theta]^{-1} = \left[\frac{1}{\sigma}\right][\rho]^{-1}\left[\frac{1}{\sigma}\right] \tag{13}$$

Substitution of Eq. (13) into (11) results in

$$Q = \left[\frac{X - M}{\sigma}\right]^T [\rho]^{-1} \left[\frac{X - M}{\sigma}\right] \tag{14}$$

The terms $(X - M)/\sigma$ can have a very wide range. However, if the range

$$-8 \leq (X_i - M_i)/\sigma_i \leq 8 \tag{15}$$

is exceeded, the value of X for that channel is too far from the mean to be considered for classification. Truncation of significant bits will occur and a flag is set indicating this condition. This indication is used to reject a decision that the sample is from the particular class.

The computation of Eq. (14) could proceed in a straightforward manner, but can be simplified somewhat due to the symmetry of the correlation matrix and its inverse. This simplification can be accomplished in more than one way. One method is as follows:

$$Q = [Y]^T[Y] = \left[\frac{X - M}{\sigma}\right]^T [B]^T[B]\left[\frac{X - M}{\sigma}\right] \tag{16}$$

where B is an upper triangular matrix formed by the decomposition of the inverse ρ matrix. By calculating

$$[Y] = [B]\left[\frac{X - M}{\sigma}\right] \tag{17}$$

the final matrix operation is simply

$$Q = [Y]^T[Y] = \sum_{i=1}^{n} y_i^2 \tag{18}$$

where the y_i are the elements of the [Y] vector.

There are four steps implied by Eqs. (16-18). These are:

(1) Subtract the mean from each channel.

(2) Multiply each result by $1/\sigma$.

(3) Perform the Y matrix multiplication on each result of Step to get Y's.

(4) Square each resulting Y and add the results together.

Another method for calculating I_{-1} (14) is to express the inverse of the correlation matrix ρ^{-1} in terms of its eigenvalues and eigenvectors. From Lanczos [3] we can express the correlation matrix as:

$$\rho = U\Lambda U^T \tag{19}$$

where the U matrix is comprised of eigenvectors arranged in columns, and U^T is its transpose. The Λ matrix is the set of eigenvalues on the diagonal. Taking the inverse of the correlation matrix, it can be shown that

$$\rho^{-1} = U\Lambda^{-1}U^T \tag{20}$$

which is to simply take the reciprocals of the eigenvalues and multiply by the two original eigenvector matrices. One further decomposition brings us to the desired form

$$\rho^{-1} = [U(\Lambda^{-1})^{-1/2}]\cdot[(\Lambda^{-1})^{-1/2}U^T] \tag{21}$$

Substitution of Eq. (21) into Eq. (14) and defining $(\Lambda^{-1})^{-1/2} = \Lambda^{-1/2}$ yields

$$Q = \left\{\left[\frac{X - M}{\sigma}\right]^T U\Lambda^{-1/2}\right\}\cdot\left\{\Lambda^{-1/2}U^T\left[\frac{X - M}{\sigma}\right]\right\} \tag{22}$$

In this case, if a vector Y is defined as

$$Y = \Lambda^{-1/2}U^T\left[\frac{X - M}{\sigma}\right] \tag{23}$$

and computed as such, then the final matrix operation can be performed in the same manner as in Eq. (18). The hardware required for this second calculation must perform more multiplication than in the first method. However, in order to implement the first method, a more

elaborate switching scheme is needed to avoid multiplying by a large number of zeros. The details of this switching scheme were not worked out and only general consideration was given to it. Since it appeared desirable to have the flexibility and capability to do the matrix multiplication required by either Eq. (17) or (23), the hardware was designed to do full matrix multiplication. Tests were run on over 100 signatures to see if the resultant set of coefficients for either method appeared better suited to a limited-word-length multiplier. From these results it appeared that a slight advantage might be gained by using the second method.

A block diagram of the quadratic pipe is given in Figure 19. The actual computations are performed by a set of time-shared arithmetic units arranged in a sequence allowing a set of 16 operations to be executed in less than 5 μsec before the outputs are latched. Each stage supplies its results to a subsequent stage for further processing. Precision varies from 8 to 14 bits as the data progresses through the quadratic pipe and acquires greater significance.

Internal arithmetic operations are currently performed at less than 50% of manufacturer's rated component speeds. For example, an 8×8 bit multiplication is rated at one product every 135×10^{-9} sec, but is currently being employed to produce a product every 270×10^{-9} sec. This rate may be increased to one product in less than 200×10^{-9} sec while maintaining adequate derating if the system operates at higher speeds in later design stages.

4.2.2 SCALING

The calculation of the quadratic form given in Eq. (11) above can result in a very large number. Fortunately, we are not interested in the exact result for large values and, therefore, can restrict the range, provided that an overflow condition is detected. The scaling of the quadratic pipe is summarized in Figure 20. All arithmetic operations up to the input of the squaring circuit are 2's complement arithmetic. The input data is an 8-bit word having the range of values

$$-128 \leq x \leq 127$$

We are interested in small values of $X - \mu$, but an overflow condition can be detected which allows us to keep the resultant to eight bits. Following addition, a multiplication by $1/\sigma$ is performed. A restriction on the range

$$-8 < \frac{X - \mu}{\sigma} < 8$$

is placed on the output of this multiplication. This in effect limits any data channel value to less than eight standard deviations from the mean. It is to be noted here that a lower limit on σ is required and consequently an upper limit on $1/\sigma$ also. The limit selected is

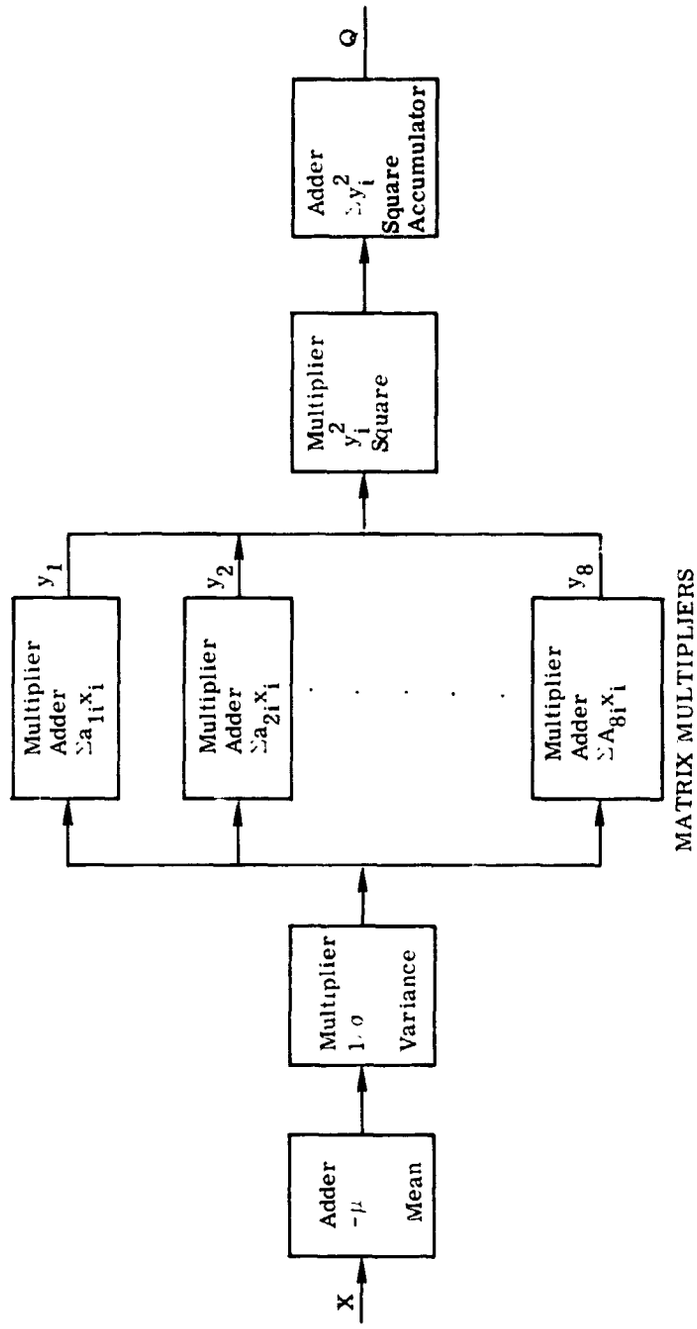
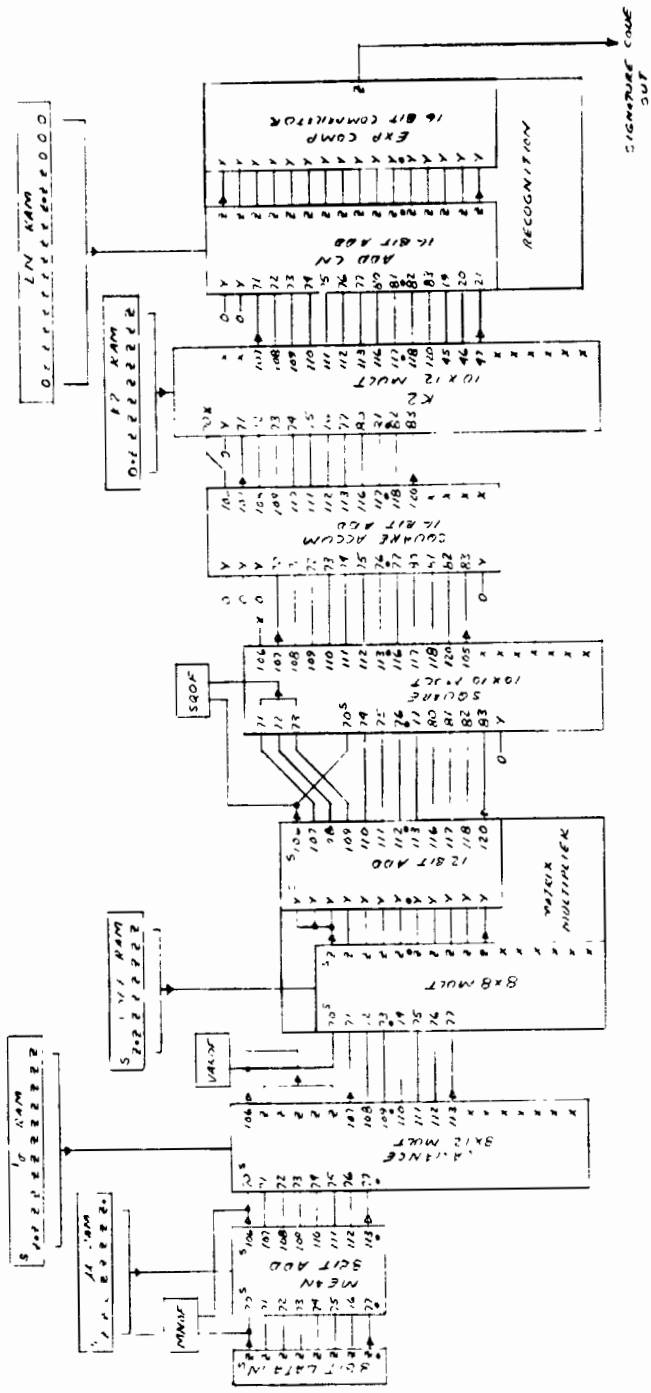


FIGURE 19. BLOCK DIAGRAM OF THE QUADRATIC PIPE



NOTES:
1. [illegible]
2. [illegible]
3. X IS UNUSED BIT
4. Y IS USED IN BUT BIT
5. Z IS USED [illegible]
6. • IS BINARY POINT
7. [illegible]

FIGURE 20. BLOCK DIAGRAM OF CLASSIFIER SCALING

$$\sigma \geq 1 + (1.2048)$$

or

$$1\ \sigma \leq 1 - (1/2048)$$

which appears to be a reasonable choice based upon examining a number of signatures. Also, it would seem reasonable that variation in the data for a given object class would show changes in the two least significant bits. However, for ERTS data there is some recent evidence that this is not always the case.

The above two restrictions determine which high-order bits can be tested for overflow and discarded at the output of the $1/\sigma$ multiplication. The five bits from 2^3 to 2^7 are tested for an overflow condition. If an overflow is detected, an overflow bit is put into and clocked down a shift register, which is parallel to the pipe, in step with that data value, so that at the final test for recognition that calculation is discarded.

The input to the eight matrix multiplier cards is comprised of the sign bit and the seven bits from 2^2 to 2^{-4} from the variance card. It is to be noted that this quantity $(X - \mu)/\sigma$ has zero mean and variance of one when the input data (X) originates from the same distribution as that used to calculate μ and σ. Similarly, after the matrix multiplication is performed in the eight matrix multipliers, the y outputs have zero mean and a variance 1/k where the constant k is the largest value in the $\Lambda^{-1/2}U^T$ matrix. This normalization of the matrix is done in order to get maximum utilization of the 8-bit RAMs and their associated multiplier input. Empirical study of more than 100 signatures indicates that the value of k is from a little over 1 to about 5. The output of the matrix multiplier is tested for overflow on bits 2^3, 2^4, and 2^5 of the 12-bit adder used to accumulate the matrix multiplication result, but these bits are not used as input to the squaring circuits.

The constant k which was factored out in the matrix multiplication is reintroduced in a final multiplication stage. Since there is a different k with each object class, the set of k's can be normalized with largest k factored out. This again makes maximum use of the limited-bit multiplier and RAM.

4.3 COMPUTER SUBSYSTEM

The general-purpose machine is a disk-based Digital Equipment Corporation (DEC) PDP-11/45 configured with 24K of core, three tapes, serial printer and keyboard-CRT. The system software is built around the disk operating system to provide fast access to the operating programs and available languages.

The DEC PDP-11/45 computer system is a fast, medium-size minicomputer, whose architecture allows extremely flexible interfacing with non-standard I/O devices. This ease

of interfacing and speed were the primary reasons for use of the PDP-11/45 computer as a controller for the input and classifier hardware used in multispectral data recognition processing.

The hardware peripherals available in the ERIM configuration are, in detail:

(1) The PDP-11/45 central processing unit (CPU), including the floating-point processor.

(2) An RK-11C disk controller and an RK-Ø5 disk with interchangeable cartridges, each containing up to 1.2 million 16-bit words. Average total access time is 90 milliseconds. Data transfer rate is 1'.1 microseconds per 16-bit word.

(3) 24K of 16-bit core memory with a cycle time of 850 nsec, access at 350 nsec (450 nsec at the UNIBUS).

(4) An LA-3Ø DECwriter data terminal, with a character set of 64 symbols at speeds up to 3Ø cps. Output is generated by a 5 × 7 matrix.

(5) A VT-Ø5 alphanumeric display terminal with a CRT display and communications hardware capable of data transmission at rates up to 3ØØ baud in full or half duplex modes.

(6) A TM-11 magnetic tape drive controller, two 9-track TU-1Ø magnetic tape drives, and one 7-track TU-10 magnetic tape drive (read/write speed of 45 ips). Densities available are 2ØØ, 556, and 8ØØ bpi for the 7-track and 8ØØ bpi only for the 9-track drive.

(7) Two DR-11B direct memory access devices for transmission of data between an external device and memory via the UNIBUS, without a need for continuous control by by CPU.

(8) One DR-11C device interface for transfer of data between a user device and memory via the UNIBUS.

(9) One KW-11P programmable real-time clock, providing programmed real-time interrupts and interval counting in several modes of operation.

Figure 21 shows the configuration in use at ERIM.

Software available on the PDP-11/45 system comprises the Disk Operating System (DOS), Version 8.8, supplied by DEC. Under this operating system, MACRO-11 (a Macro assembler), FORTRAN IV, EDIT-11 (a file editing package), ODT-11R (an on-line debugging package), PIP (a file modification and transfer package), and several other utility packages are provided.

All output is either through the LA-3Ø DECwriter, on magnetic tape, or printed with a line printer. Input for large quantities of data is via magnetic tape or through the keyboard devices.

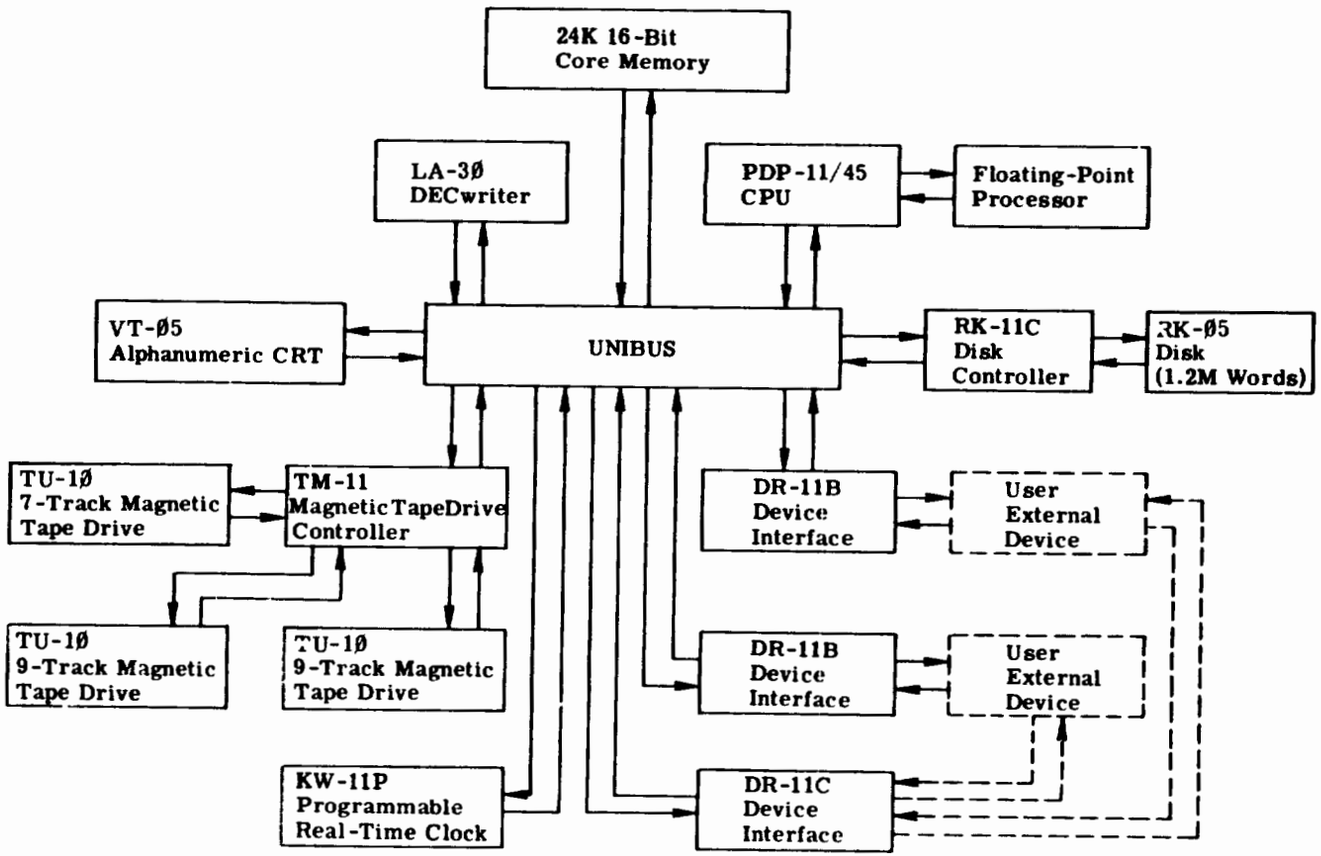


FIGURE 21. PDP-11/45 CONFIGURATION

5
SUMMARY OF PHASE II SYSTEM DESIGN

The MIDAS System is to be developed further during the second phase of the two-phase, 2-1/2-year program by accomplishing several principal objectives: (1) the addition of a pre-processor; (2) adding more peripherals, such as a color, moving-window display, a high density tape interface and, if possible, a hard-copy color output; (3) developing control and operating software to provide a simple interface to the user; and (4) study of several possible uses for the system in order to evaluate its overall potential.

This portion of the report describes the projected system, complete with both hardware and software, as planned for the end of the second phase.

5.1 SYSTEM DESCRIPTION

The MIDAS System, at the end of the first phase of the program, is a minimal operating system for demonstrating the feasibility of on-line, near-real-time digital multispectral analysis and recognition using a parallel, low-cost processor combined with a minicomputer. These objectives have been met.

The second phase is intended to further develop the system by adding hardware devices and a software operating system to obtain a powerful, flexible, and easily controlled system for analyzing and processing multispectral data at high data rates from a variety of sources. The data rate may be as high as 200,000 data vectors per second if the data source permits. Since a large number of functions (or operations) are to be performed by the system, these functions may perhaps be best summarized by a narrative presentation describing individual functions as a user would normally employ them in the process of operating the system with real data. There are, of course, many different sequences a user may wish to follow in processing multispectral data through the system. These cannot all be described in detail, but the more important or more likely sequences will be discussed or mentioned in passing as the sequence is elaborated.

5.1.1 A TYPICAL PROCESSING SEQUENCE

The user normally enters the processing sequence with a set of data (some ERTS tapes for example), has some imagery and collateral information about the scene, and wants to obtain additional imagery showing the location, identification, and possibly the spatial extent of certain natural or cultural features of interest to him. He will also be interested in the reliability of the classification process, the feasibility of separating and analyzing various features in the scene, and, perhaps, in obtaining some summary statistics about the spatial or spectral characteristics of particular features.

His problems may range from finding answers for research questions to accomplishing some quasi-production task. Thus he may be interested in the feasibility of determining the volume of water in the Everglades, on the one hand, to obtaining the projected yield of corn in the state of Iowa, on the other. The MIDAS System is flexibly designed to do useful processing in a multiplicity of such applications.

5.1.1.1 Setup

Two things are done in this initial mode. First, the user must perform a routine signal conditioning of the data. Since the classifier input is only eight bits, the accuracy of classification may be affected if the dynamic range of the signals is too small. Thus the signal conditioning will consist of a search through a representative set of the data to pick off, with hardware, the minimum and maximum value in each channel (assuming that the data is in straight binary format). The minimum value is stored (negatively) as one input to an adder on each channel. The maximum value is used to compute an integer value 1 through 7 for input to a multiplier. These operations on the data can be performed with simple hardware and will condition the data so that the 0 to 255 dynamic range available is well used. Although such operations may be handled in conjunction with angle corrections, it appears desirable to keep them separate (to avoid a bookkeeping problem) as explained below. It is planned for Phase II that all data-sets stored in the computer will be raw data. All signal conditioning constants and preprocessing constants, either computed previously or to be computed, will be stored with the data. These can then be selected and used to transform the raw-data set (a training set, for example) into data input to the signature analysis.

As a second operation in the setup mode, the user must locate features and areas of the scene about which he has information; this allows their designation as training or test areas for the system. He may do this off-line with printouts which enable him to designate the spatial address of his data and then supply this address to the system. Or, beginning with no printout, he may call for a color display of the complete area, zoom down to a full resolution subarea of the original frame, and then designate to the system the training and test areas via an interactive display and appropriate cursor placement. To perform this search and locate such areas in minimum time requires quick search capability for a large amount of data (about 30 million bytes). The quickest way would be to have the data stored on a large disk; the next quickest is to have the data stored on high-density digital tape (HDT). However, if the data are stored on computer-compatible tape, a high-speed tape drive (125 ips) should be available to read the data onto the disk.

But what if the unenhanced display does not provide enough contrast to allow the operator to locate these areas? In this case, the operator may choose functions of the input data, combining data and colors to enhance certain differences among the features of interest. Upon selection,

this data may be supplied to memory as training sets. Optionally, the operator may make a color copy of this enhanced imagery as a record for reference or off-line study.

If its quantity is large, the data which an operator desires to use for further analysis or classification would, at this point, normally be placed on high density digital tape (HDT). (The use of HDT as input in all digital data cases may be desirable to lighten the raw data handling load on the computer by placing this load in the MIDAS front end.)

5.1.1.2 Analysis Mode

When the user has isolated data from areas of the scene as described above, he must then begin an analysis to define the methods and parameters necessary in pre-processing to obtain the parameters needed for classification. If we assume, just to illustrate the options, that he requires all the analysis capability of the MIDAS, the complete setup can be described as follows: (1) calculation of scan-angle correction functions, additive and multiplicative; (2a) selecting and calculating channel ratio transforms, or (2b) calculating a linear dimension reduction transform; and (3) calculation of classifier parameters. These options or steps are performed in the sequence given above since each step provides both improved data for the next step and successively enhanced data for classification. Not all of these steps need be performed. Note also that it is not possible to take linear combinations of ratios.

The calculation of scan-angle correction functions (step 1, above) is normally required in processing aircraft data and, for spacecraft data, may be needed to remove path radiance. (Processing of S-192 data from Skylab indicates 10-20% variations in data amplitude as a function of scan angle for some channels.) The calculation may be done in either of two ways. In one, the complete data-set is analyzed in strips of scan-angle increments to obtain averaged variations of data as a function of scan angle for multiplicative corrections and to look for darkest objects at various scan angles in order to correct for path radiance. The other way is to choose the training set data in such a manner that it is distributed over the scan angle range of the scene. From these sets of data, a set of correction functions may be calculated. The training set data is then transformed with these functions before the next step in processing—i.e., that of calculating ratio transforms.

Ratio transforms (see step 2a above) are used in pre-processing when one wants the data to be relatively insensitive to illumination variations, where transference of signatures from one frame to another is desired, or where spectral features can be enhanced. There is no clear-cut means of examining a data-set to decide, in an a priori manner, which of several possible transforms is needed for a particular scene. The method normally used would be to perform each of several transforms on the training sets and then test the resulting data for its optimal probability of correct classification; here the training set and test set data enable selection of the transform to be employed. When this selection has been made, the properly transformed data is used for the next step - calculation of a dimension-reducing linear transform.

The purpose of the linear transform (step 2b, above) is to provide a rearranged set of data in which the spectral information is combined in such a manner that the transformed data has a greater discriminability per dimension for the classes of interest to the user. The classifier can then perform a classification operation using the smaller number of dimensions with an accuracy of classification equivalent to that obtainable with a larger number of untransformed dimensions. The classifier may then make more classifications on the data-set than it does on the untransformed data. The operator can, in effect, use a single transform on every data vector to obtain a set of transforms of smaller size within the classifier, thus increasing the classifier's net capacity. (Because there are several approaches to calculating such a transform, a specific method has not as yet been selected; this uncertainty, however, in no way affects the design of the hardware.)

Finally, after these transformations have been effected on the training data, the classifier parameters are calculated (step 3) and supplied to the classifier. This completes the MIDAS setup for classification.

5.1.1.3 Verify Mode

Often the user will wish to defer an immediate start on the classification of a large area of data, preferring instead to evaluate the classification operation on the basis of a set of smaller test set areas. These are stored in the computer early in the setup operation.

The data from these smaller test set areas are called up, supplied to MIDAS, and then, after classification, input to the color display. Here they are displayed in one of two modes: single-field or checkerboard. The single-field mode shows the classified results for one field at a time as a true image of the field in which anomalies can be detected based on their shape and location in the field. The checkerboard mode presents several fields on the display at one time; these are located in a matrix of sub-areas arranged to make optimum use of the display area.

The display that performs this task should have its own storage for refreshing the CRT display. The alternative of loading classified results at 200,000 decisions per second into the computer over the Unibus, while at the same time refreshing the display, does not appear possible. Much lower data transfer rates can be achieved by having the memory in the display, along with a scrolling capability. In this way display updating can be achieved by simply reading in the new scan line, thus keeping display-data transfer rates low and greatly helping to maintain the fast classification-result transfer rate.

The operator-user may examine the results to verify that the machine is correctly set up and that the discrimination among classes is adequate. Possibly he may call for a color print of these displays to examine the results off-line before releasing the system.

5.1.1.4 Classify Mode

A user normally enters this mode by supplying his bulk data into MIDAS on an HDT, although he may optionally supply data from analog tape, computer-compatible tape (CCT) or from disk core. As data enters the front-end, proper channels are selected and gated into the preprocessor. The data is then corrected by the preprocessor functions and transformed as specified. The data stream out of the preprocessor is immediately input to the classifier where it is classified and displayed as rapidly as it enters the system. The user may pause at any time and examine a portion of the area on the color display to obtain an intermediate evaluation of the classification process. The output codes may be logged into the computer in three ways: element-by-element, summary by class for each line, or totalized results for each class for the entire run. These codes may be examined at any time either via printout or on the display.

The color display may be dumped to a color printer at any time during classify mode operation—though normally, the user would examine the full area once before obtaining a color print. Printing takes about 50 seconds per frame.

5.1.1.5 Diagnostic Mode

The preprocessor and processor contain a bus system which allows the computer to access the data being processed at principal points in the cascade of arithmetic operations. This feature—which has been included to facilitate fault isolation, particularly in bringing the system up after manufacture—allows examination of intermediate results which are very often of interest to a user operator during the processing sequence.

These intermediate results are available as an image displayed or printed out in color and can be selected at any time under computer control. Thus, the user is afforded considerable flexibility in taking advantage of any subsets of arithmetic operations done in the system.

5.1.1.6 Operator Log

All of the steps taken by an operator user in this sequence of operations are maintained in a system log which becomes his record of the processing operation. Items to be included are all analytic results, training and test area data, a record of operator choices, the sequence of steps taken, and any pertinent ancillary data. This record is to be kept on the operator's tape by the user operator.

5.2 SYSTEM HARDWARE

The overall system hardware is diagrammed in Fig. 22. The MIDAS System consists of several principal subsystems: the general purpose computer (DEC PDP-11/45) (see Fig. 21), the classifier (see Fig. 19), the control subsystem, a hybrid subsystem, and the preprocessor. Associated peripheral equipments are the color display, HDT tape recorder, analog tape recorder.

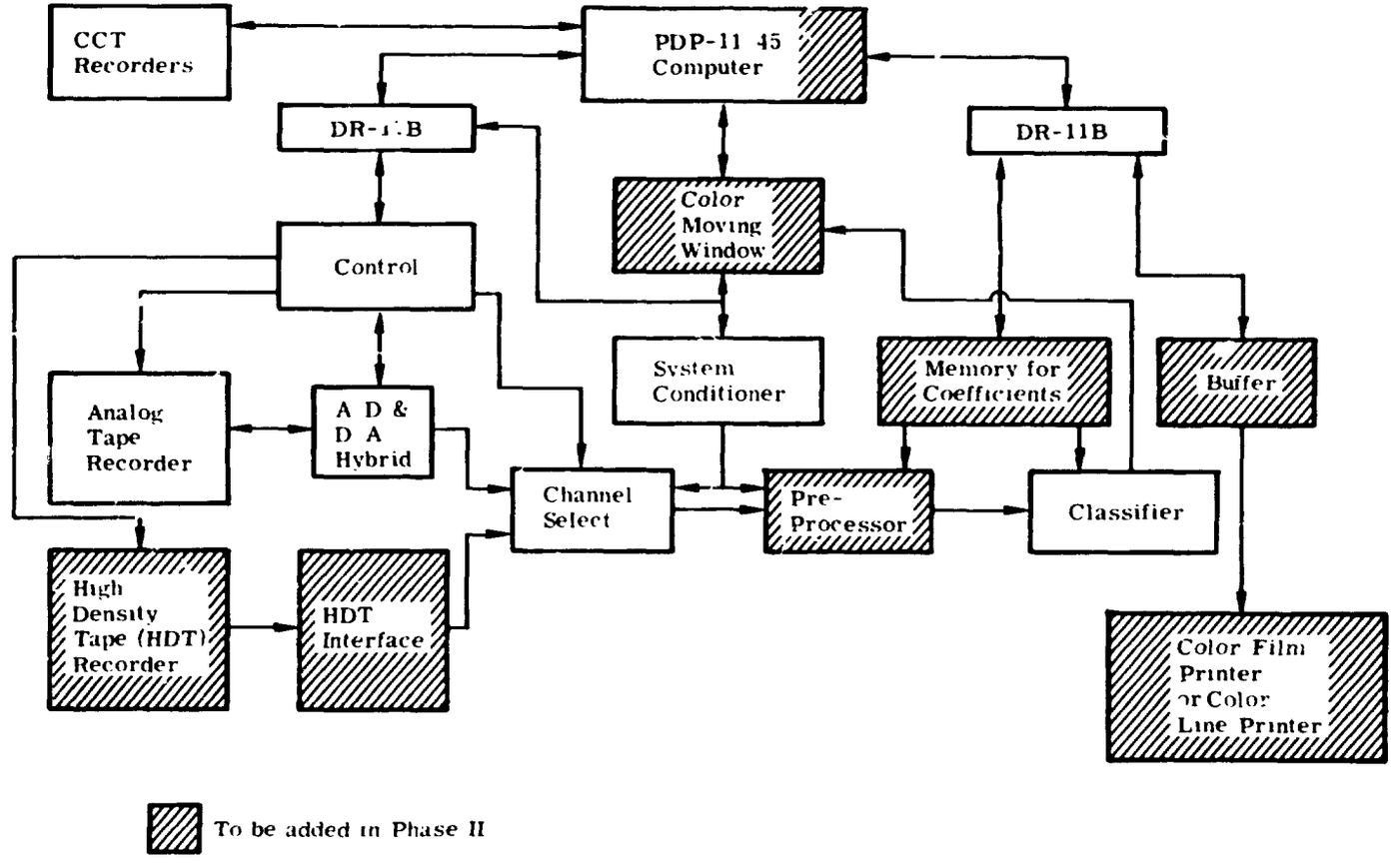


FIGURE 22. OVERALL MIDAS SYSTEM SHOWING PHASE II ADDITIONS

and a color printer. Of these, the PDP-11 45, the classifier, the control subsystem, and the hybrid subsystem have been designed, fabricated, and tested during Phase I of the program. The remaining units to be fabricated, procured and interfaced during Phase II are indicated in Fig. 22 as cross-hatched whole or partial blocks. Among these are the preprocessor, color moving-window display, HDT and its interface, the color printer, and some computer peripherals such as a disk and a line printer. A more detailed description of these subsystems and peripherals is given below.

5.2.1 CLASSIFIER

The classifier will be expanded from the present eight-channel, eight-class configuration to an eight-channel, sixteen-class configuration. The hardware implementation of this expansion will be essentially identical to the present classifier except for possibly the final output circuitry. The expanded classifier will operate at the present rate of 200,000 decisions per second. One change from the present design may have to be made in the computation of the subtraction of the mean. Currently this operation is an 8-bit integer subtract, but may have to be modified to a floating point operation to allow dynamic range expansion in the divide process of the preprocessor described below.

5.2.2 PREPROCESSOR

The preprocessor first performs the several functions described in Section 5.1.1.2: correction of data as a function of angle, calculation of ratios of spectral bands, and calculation of a linear transform of the corrected data. A block diagram of the subsystem is shown in Fig. 23.

Data are passed to the angle correction port as a data vector, X. For each angular increment, this vector is multiplied, element by element, by a function of the scan angle. This function may be supplied in one of two ways: (1) by computing each function of each angle using a fifth-order polynomial calculation in which the coefficients alone are stored and the functions of θ for a given angular increment are calculated; or (2) by pre-computing all functions of angle and supplying these into a local memory, thus requiring only access to these stored constants followed by multiplication. The sequence of multiplications will occur at a rate of 300 nsec per multiply and, for 16 input signals, require 5 μsec to complete.

The corrected data then enter the ratio calculator which consists of a fast divider and a set of 16 adder-subtracters. These arithmetic elements operate in a sequence controlled by microprogram vectors stored in a semiconductor random access memory (RAM). Any generalized sum and difference functions of sixteen channels can be performed. In the block diagram of Fig. 23 the a_i and b_i take on the values zero, one, or minus one. Eight of these calculations are placed on a bus to form the numerator input to the divider; the remaining eight are bussed to the denominator input from which eight ratios will be formed. A division technique that appears

C-2

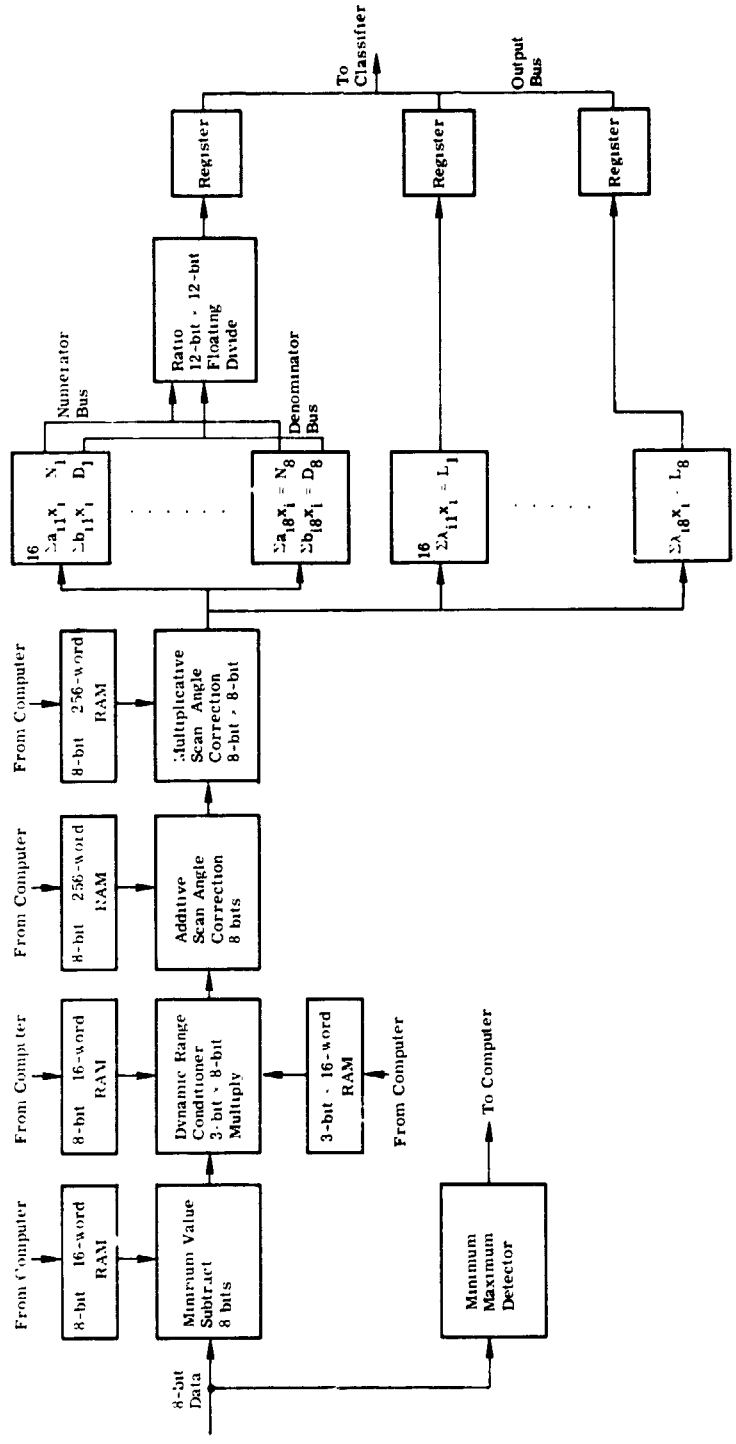


FIGURE 23. BLOCK DIAGRAM OF PREPROCESSOR

promising for this function will consist of a table-lookup and subtraction process done in a pipeline manner. An unnormalized, floating-point numerator and denominator are entered into the divider and updated every 312 nsec. The quotient of the first pair appears 5 μsec (16 clocks) later as a 12-bit floating-point number.

The linear transformation will consist of eight of the multiplier-summer cards operating on each of the 16 channels to provide eight transformed vector elements. A certain amount of further study is needed in this connection to determine the effectiveness of the transformed vector as a classifier input vector. Preliminary studies [5] indicate that for some representative recognition problems, no more than 4 transformed channels may suffice to describe data from 10 spectral bands. In this case the preprocessor could be restricted to a 4-dimension output and the classifier used in the 4-channel, 16-class mode. The preprocessor as now planned will supply an 8-dimension output and accept a 16-spectral-band input.

The preprocessor, then, can accept 16 or fewer spectral bands from sources such as Skylab (S-192), ERTS, Bendix M^2S, NASA-Houston-MSDS scanner, or the ERIM M-7 scanner. The 24-channel Houston-MSDS scanner will require two passes to enter training sets, followed by selection of 16 out of 24 channels for MIDAS processing. In each case the output will be an 8-dimension vector to the classifier.

It is to be noted that the preprocessor will not perform linear combinations of ratioed data. Since the ratio is a floating-point number, hardware will have to be designed and developed for this more complex arithmetic. However, by working with 8-bit integers coming from the angle correction, the classifier circuitry developed in Phase I can be used directly. This limitation is not too severe when one considers that ratioing of sums and differences is a dimensionality-reducing operation, as is the case with linear combinations. Furthermore, since the outputs of these two preprocessing operations are bussed together, a mix of ratioing and linear combinations can be obtained, thus increasing preprocessor flexibility.

5.2.3 COLOR MOVING WINDOW DISPLAY

This is a three-color video display using a shadow-mask CRT and MOS storage for display refresh. The present choice is the RAMTEK GX-100/200 series configured to hold a 5-bit color vector with 480 × 640 elements on the display. The unit will require modification to allow direct data input from the MIDAS as well as normal input from the PDP-11/45.

The unit offers image display, alphanumerics, vector generation, moving window display, and table look-up of predetermined colors. It provides the major means for a fast man-machine interface. A scene or portions of a scene can be displayed almost immediately upon user request, provided the computer can access the data quickly. Once the MOS memory is loaded, the

computer can be performing other operations. Using the display memory as a source, the unit appears capable of driving external hard-copy devices while the user is viewing the CRT.

5.2.4 HIGH-DENSITY DIGITAL INPUT

In the present design of the MIDAS System, digital data are presented to the classifier by reading data from standard computer compatible tapes (CCTs) into the PDP-11/45 computer and from there to the classifier through a DR-11B interface. The maximum rate at which data can be processed in this mode depends on the throughput of the input tape units. The present MIDAS configuration uses tape units that operate at 45 ips and offer a maximum byte density of 800 per inch. With these units an average byte rate of approximately 35,000 per sec can be presented to the classifier.

In order to increase the rate at which digital data can be processed, a data source offering higher throughput than the present tape units would be required. If a currently available high-density digital tape system (HDT) were designed into the MIDAS system, data rates in the order of 150,000 data vectors per second could be realized. The HDT system uses a wide-band (with a 1.5 MHz cutoff) multichannel tape recorder to encode and record the digital data in serial form (NRZL). A special interface unit would be required to read the data from this type of recording system into the MIDAS classifier. Figure 24 shows a block diagram of the HDT interface unit.

Multichannel digital information is recorded on parallel tracks of the HDT system, one track for each channel. The data from one track comes off the HDT system in serial form (NRZL). This serial bit stream is converted into parallel 8-bit words in the serial-to-parallel converter. These words are then stored in the deskew memory until the data from all the tracks are in their respective memory units. The multichannel information for a resolution element is then stored in output three-state registers where it is multiplexed into the MIDAS System.

5.2.5 COLOR PRINTER

An extremely important device for the MIDAS System is a fast color printer. A number of methods have been proposed for this function including CRT-color filter camera devices, multi-laser film devices, LED color array systems for printing on film, and colored-ink jet printers. Of these, the most attractive method is the ink-jet printer since it immediately produces a usable output, with no delay, for color film processing. This printer has been developed by several sources at present and appears capable of producing a picture containing about 10^7 elements in 15 minutes or less. An alternative possibility is a multi-LED scanning film printer using a Polaroid or Eastman fast color development method. Its feasibility depends, however, on adequate brightness in experimental LEDs as well as on the resolution and color fidelity obtainable.

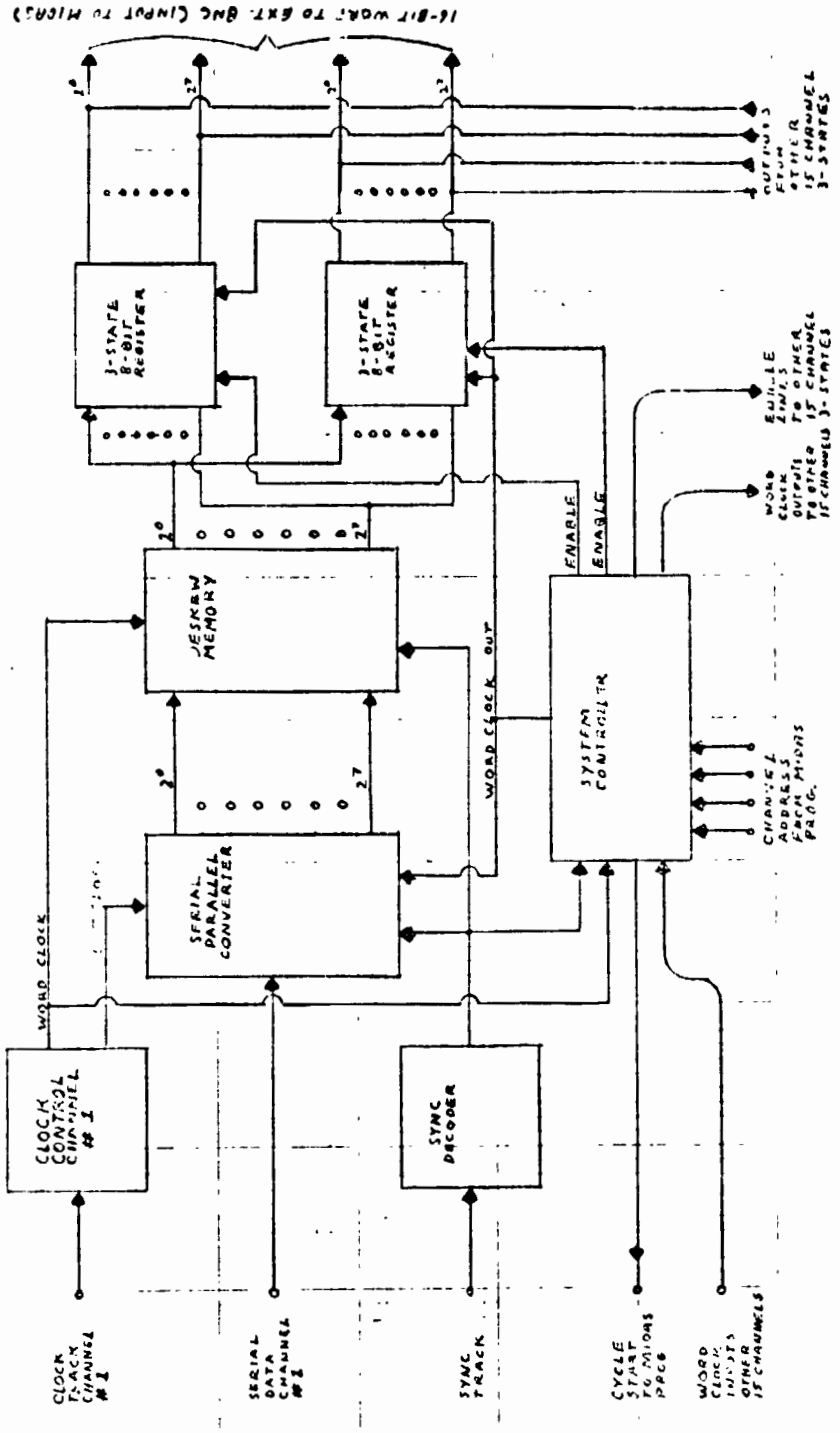


FIGURE 24. HDT INTERFACE UNIT. One channel shown.

Some effort will be devoted in Phase II to study and evaluation of suitable color printing techniques.

5.2.6 COMPUTER PERIPHERALS

Other computer peripherals to be added to the PDP-11/45 system during this period include: (1) a mass storage (58×10^6 byte) disk, (2) additional core memory (32K words), (3) memory-management hardware, (4) a fast line printer (300 to 600 lines per minute), and (5) fast tape units (150 ips, 9 track, 800 bpi). Principal reasons for these additions are: (1) to facilitate Phase-II software development wherein a real-time-sharing executive-system for all operating modes will be developed that will normally require the simultaneous involvement of three programmers; (2) to speed up data handling for framed data such as ERTS; and (3) to improve the operating speed of programs using large quantitites of data by increasing core residency. These reasons are discussed below as they relate to each peripheral.

5.2.6.1 Line Printer (Printer/Plotter)

The acquisition of a medium- to high-speed line printer is necessary if the development of the complete set of operating software proposed for Phase II is to be completed. In Phase I, difficulties were encountered in obtaining timely hardcopy listings when a two-man software effort was being made. With the full-time effort of three programmers and with the increased amount, scope, and complexity of the Phase II software, the effectiveness of the programming staff would be severely curtailed if it were necessary to rely on the non-dedicated use of a low-speed line printer (approximately 120 cps) or the console keyboard/printer capability (approximately 30 cps). Turnaround times currently approximate several hours during prime working hours for moderate-size software listings (~20 to 30 pages), whereas this could be reduced by a factor of 10 with the acquisition of a line-printer capability dedicated to the PDP-11/45 MIDAS System.

In addition, an electrostatic printer/plotter seems an increasingly desirable acquisition. While this type of printer/plotter system would fulfill the printing requirements admirably (approximately 600 lines per minute), the inclusion of a hardcopy plotting capability at a small additional charge is extremely attractive in view of recent price decreases by the leading manufacturers of this type of system. This plotting capability would be extremely helpful in providing hardcopy of graphs, histograms, cluster-plots, etc., generated in both pre- and post-classification analysis of the data.

5.2.6.2 Large-Capacity Disk Drive

In the Phase I implementation of MIDAS, the secondary storage capacity via random-access disk was limited to 1.2×10^6 words of data. This number severely limits the amount of data

that can be handled in a single pass of the input data source. Since one of the primary objectives of Phase II is the interactive communication between user and the MIDAS Classifier in a relatively short time frame, it is necessary to be able to access large amounts of data in an extremely brief period of time—i.e., in less than ten seconds, rather than in a period of minutes. If the data to be accessed resides on the disk (with average access times of approximately 30 msec), input data can be obtained in a time frame of less than 1 sec, and subsequent data transfers (once the read heads are positioned) can take place at rates on the order of 3.12×10^6 bytes/sec.

In addition, if we were to attempt the post-classification analysis necessary to establish accuracy, acreage totals, etc., it would be necessary to store the classification results on a secondary (i.e., noncore memory) storage medium. No peripheral device other than a disk of this capacity fulfills the following necessary requirements: (1) the ability to store data at a rate comparable to the speed of the input data source as well as to that of the MIDAS Classifier hardware, and (2) the capacity for storage of data on the order of several megabytes (10^6 bytes).

The disk configuration currently being investigated consists of a dual-spindle drive, and thus contains two 29-megabyte disk cartridges with separate moving heads. Thus, if one drive were dedicated to input and one to output, no alternating head repositioning would be necessary; and the system could be driven at close to the maximum rate.

Another proposed feature of the MIDAS System for Phase II is the capability to expand or decrease the scale of the display generated on the color CRT. To do this in a mode which allows quick response to user commands, random access to stored data is likewise necessary in a time frame of seconds in order to maintain an interactive system—and this is possible only with a large-capacity, high-speed, random-access disk system.

If a large capacity disk drive is not included for Phase II, interactive uses of the system will be severely crippled. With only a low capacity disk, all the programs and data sets necessary for interactive use can not be kept on disk. Instead, magnetic tape will have to be used as storage for some high-use programs and data-sets. Then in order to run a program, the tape will have to be searched and the program loaded, all at low data-transfer rates. If a series of programs is to be executed repetitively or a data-set used repetitively, the MIDAS user will have to wait a few minutes for each program to be found, a few minutes for each program to be loaded and started, and a few minutes for each data-set to be accessed. This defeats the purpose of the Phase II MIDAS—namely, the ability of the hardware to interact with the user in a short time frame, hopefully less than a few seconds.

5.2.6.3 Expansion of Memory and Addition of KT-11C Memory Segmentation Unit

During the Phase I implementation of MIDAS, programming difficulties arose through lack of adequate amounts of core memory storage. In particular, limitations were necessary on the

number of pixels which could be used in signature extraction via the SIG and STAT programs. In Phase II, in order to increase system throughput of data, multi-level buffering of I/O will be necessary in some instances, depending on the peripherals in use (e.g., magnetic tape CCT input and disk output will require buffering both on input and output to efficiently use the capabilities of the MIDAS Classifier hardware).

The amount of additional core memory necessary for Phase II cannot be absolutely determined; however, 32K words seems the logical choice for the following reasons:

(1) The allocation of 32K into 8K for additional program storage, 16K for input buffers, and 8K for output buffers is appropriate in terms of quantity of data input vs. quantity of data output. For ERTS data, in particular, four channels (and thus four bytes) of input data will usually generate two bytes of output data (recognition result and exponent), hence the two-to-one ratio in the allocation of I/O buffers.

(2) 16K boards are currently available; from past experience these are more reliable (by virtue of fewer ICs, connectors, etc.) than 8K modules and also more cost-effective in terms of bits per dollar. (Note: It is not possible to mix 8K modules and 16K modules because of chassis requirements.)

5.3 SOFTWARE

The software effort for Phase II will address two principal objectives: (1) the generation of a well-designed operating system for the user/operator to control and use MIDAS; and (2) generation of the new programs to perform calculation and setup of the preprocessor, to operate the display and print subsystems, and to control format and sampling of data input.

The modes of operation described in Section 5.1 assume the presence of the Phase II hardware and software and illustrate the manner in which the Phase II software will be used. The Phase-I software may be thought of as a skeletal version of the software needed for the system; all of it is, indeed usable in Phase II. There has, however, not been enough time or funds available in the Phase I effort to attempt to manage the elements of the software system with an efficient operating system or, moreover, to project the kind of software operating system a user may need when the MIDAS system becomes more operational.

An important task of Phase II, then, is to project the framework of an operational system within which to develop and assemble the Phase II software system. Several environments may be considered likely for a MIDAS System as processing technology moves more toward operational use. A time-shared, multi-terminal version of MIDAS may prove, for example, to be an efficient means of providing more output for more users by taking advantage of the speed of MIDAS versus the slower interactions of the human user. In this version, it is important to

define the software modes and modules carefully and assess the execution time and operator time for each. From this, we can determine how many users with what kinds of terminals could effectively use MIDAS.

On the other hand, it may be desirable to provide remote time-sharing of the system by having various modular remote terminals. Estimates of communication loads and cost savings over conventional computer usage are needed to verify the utility of such an operation. Such a mode may well offer considerable economy.

Because of the above considerations, the Phase-II software modules and operating system will first be examined in the light of subsequent use as parts of more advanced systems and then developed, insofar as possible, so that they may be employed in such systems with little or no modification. This constitutes the initial design step in the development of the software.

Next, given these constraints and also an overview of MIDAS operational modes as discussed in Section 5.1, a MIDAS executive system will be designed to allow an operator/user to employ the MIDAS System for simple tasks or with considerable flexibility, as needed. This executive system may be overlaid on available multiprocess systems or could be designed to replace such a system.

Briefly then, the software design will include the principal tasks shown in Tables 5 through 10 and discussed in the following task description:

(1) Investigate an operating system to be used to control loading, linking, assembling, etc., for MIDAS software system:

(a) feasibility of using the newly released (August 1973) DOS/BATCH operating system. This is the standard Digital Equipment Corporation software system supported by DEC.

(b) possibility of using a multi-user, multi-tasking system with MIDAS System. If this approach is implemented, possible candidates are RSX-11D or RSTS-11E, both being software products of DEC.

(c) feasibility of writing our own operating system incorporating system programs available from DEC, (e.g., the MACRO-11 assembler, the Editor, etc.) tailored to our own specific needs with the MIDAS System, e.g., possible multi-user graphic CRT consoles, or several single-user-dedicated disk units.

(2) Determine the software functions to be implemented in Phase II of the program. These will include programming in the following areas:

(a) data format and/or storage media conversion
Examples:

TABLE 5. EXECUTIVE SYSTEM (MIDAS EXEC)

Function	Requirements	Software	Hardware
System Startup and Instruction	Provide initialization of all modules, headers, etc.	Documentation	Display/Printer
	Provide operating instruction for user	Text	
User Log	Provide as desired log of user operations, training sets, test sets, MIDAS parameters, and results—output to tape/printer	File Generator	Disk/Tape and Printer
Command Interpreter	Accept commands to initialize and run in all operating modes; normal setups to be default mode	Interpreter-Overlaying DOS/Batch, RSTS, RSX	Special Terminal(?)/Keyboard Display/Printer

TABLE 6. MODE I - SETUP

Function	Requirements, Controls	Software	Display Hardware
Data Input	No. of Channels Ident. of Channels No. of Samples Location of Samples 16 Locations — 8SS, 8RS (SS = Scan Sync RS = Roll Sync)		I/O Interface for DR-11C and DR-11B Indicators Printout
	Sampling Rates 2C, C, C 2, C/3 . . . (C = clock) Line Count - (2 Words-Line Count) (1 Word-Angle) Enabled, Disabled Internal, External Line-to-Line Update Line Search — Fwd, Rev Data Formats Byte Pack (2 samples/word) Word Pack Single Sample-I.D., No I.D. Averaged Sample	Software Control of DR-11C Interface	
	Time Data $\Delta t(SS\text{-}RS)$ $\Delta t(SS_j\text{-}SS_{j+1})$	Input	Indicators/Printout
Tape Control	Speed; Fwd, Rev; Filter Count/Zero Cross; Record; Stop	Control	
Display	Multichannel (for Analog Tapes) Address & Data-Decode to Channel	Control	
	Display Input (Digital) $(X - \mu)/\sigma$ Exponent Classified Output (Class # Selectable) Mixed Input and Class Output Linear Combinations of Inputs	Bus Select I/O Control	CRT-P7; Moving Window, Color
	Syncs for Recording (RS, SS) Dead Time Control	Control	

TABLE 7. MODE II - ANALYSIS

Function	Requirements	Software	Hardware
Preprocess	Examine data/training sets for systematic disturbance	ACORN IV, U-V	Preprocessor
	Select ratio transform	Ratio	
	Calculate linear transform	LXFORM	
	Correct data in training sets		
Channel Select	Calculate channel ordering using average probability (misclassification)	Channel select: Calculate $P(MC)_{avg}$	
Statistics	Calculate means, variance-covariances matrix determinants, scaled inverses of matrices, merged training sets	STAT PAC with printouts	
Setup	Calculate and prepare tables for classifier setup; output setup to classifier	DSR for DR-11B, Scaling, RAMLD	Interface Definition

TABLE 8. MODE III - VERIFY

Function	Requirements	Software	Hardware
Input training-set data and test-set data	Enter training/test set data into classifier from disk and compare output with data.	Route data to and from classifier	
Analysis	(1) Determine no. of element counts classified into various target classes.		
	(2) Display recognition results:		
	(a) in checkerboard form	Determine geometric format	[illegible]play format
	(b) in single training-set form	Output control for CRO & printer display (VT05)	Display training set and print results LA-30/VT05 (graymap or film) Color display and printer

TABLE 9. MODE IV - CLASSIFY

Function	Requirements	Software	Hardware
Input Data	Set up A/D, D/A channels and Control	DSR for DR-11C	I/O Interface for DR-11C
	Output format for CCT/Disk Data	Format establishment and operation	
Input Data from Classifier	Record classified results on disk	File structure DSR for DR-11B	Buffering to 1/2 byte, line buffer, run buffer
Output Classifier Data	Record results on color display	Display control, color printer control	Color printer, C-scope, color display

TABLE 10. MODE V - DIAGNOSTIC

Function	Requirements	Software	Hardware
Preprocessor			
Input Normalization	Access and test	Select bus, compare calculation	Bus and port
Angle Correction	Access and test	Select bus, compare calculation	Bus and port
Ratio Transform	Access and test	Select bus, compare calculation	Bus and port
Linear Transform	Access and test	Select bus, compare calculation	Bus and port
Classifier			
Overall Test	Input data from two distributions, examine each output, test overflow	Input dataset for test and list outputs	
D/A-A/D	Link D/A to A/D and input data; test output	Data-set output	Switch D/A to A/D
$(X - \mu)/\sigma$	Access and input data from bus; compare with internal computation	Comparison of output with internal setup plus port	Bus availability, port
Y_i	Access matrix; multiply output and compare	Do matrix operations internally and compare. Select port for Y	Bus and port
ΣY^2	Access ΣY^2 and compare	Do calculation and compare. Select ΣY^2 port	Bus and port
EXPONENT	Access exponent data and compare	Do calculation of exponent and compare. Select EXP port	Bus and port

- conversion of data from various formats to one more compatible with the MIDAS System

- transfer of data from CCT (i.e., slow input data source) to HDT (high-density digital tape, i.e., a fast input data source) for multiple-pass input applications

(b) improved signature extraction and analysis

(c) pre-classification analysis

1 determination of angle effects, both additive and multiplicative

2 selection of a ratio transform

3 selection and calculation of a linear transform

(d) MIDAS hardware set-up programming—i.e., preparing the MIDAS control logic, loading up RAM coefficients for use in a classification run, preparing proposed pre-processor with appropriate functional parameters

(e) control of classifier in actual classification operation, using various input and output media

(f) display of data

1 grayscale and/or color display of either multispectral data or classification results with zooming and scrolling capability

2 scatter plots of classification results

3 histograms

(g) post-classification analysis of data

1 area totalling (with histogram capability) for classification results

2 ground-truth comparison

(3) Define standard I/O formats acceptable to the system for the following peripheral devices:

(a) industry-compatible digital magnetic tape (7- and 9-track)

(b) high-density digital tape

(c) disk-file formats

(4) Generate documentation for all programming to be implemented prior to actual coding. This will permit evaluation of the MIDAS System as an integrated whole before any programming has been performed.

(5) Determine the ramifications of a possible Phase III which would implement a multi-user time-shared system for use of MIDAS hardware. In order to avoid the necessity of recoding programs in a possible Phase III, it will be necessary to specify a multi-user operating system which will incorporate Phase-II program implementations after relatively minor modifications.

(6) Evaluate software system performance in terms of human/machine interaction. Since the primary emphasis of Phase II is to increase the efficiency of the user in producing a final product, this evaluation will take place at all stages of the software system development and include interaction with various existing multispectral processing user groups. It will also include determination of the execution and user times for all software modules as a basis for multi-processing or time-shared operations.

5.4 STUDIES

5.4.1 GEOMETRY CORRECTION

Previous investigations have revealed various requirements for spatial resolution and location accuracy in different user applications. Over a wide range of such applications there appears to be a strong correlation between the resolution required and the location accuracy needed: on the average, a difference factor of four seems to be the upper limit. In other words, if a given application requires a resolution of 50 ft to see and classify a certain object, then the classification map should show the same object located no farther than about 200 ft from its true location. This is not to imply that the locations shown need be this accurate with respect to geodetic or earth coordinates, but rather that on a classification map, objects of interest should be positioned to at least this accuracy with respect to local landmarks. A study of the possible geometrical correction approaches that could be used to meet the above criteria should be undertaken.

5.4.2 SYSTEM OPERATION

As part of the operation of the system in Phase II, a number of studies and evaluations should be made. In using the system it should be possible to predict the improvements in performance and processing costs that will be realized by the interactive man/machine capability. The time that is saved by this capability will contribute to lower processing costs.

Although MIDAS is designed basically as a complete, stand-alone system with a mini-computer performing all the control functions, a study of how this system could interface to a large general-purpose computer system is desirable. Conceivably, for example, an operational system having an extremely large data base might well feature a large general-purpose computer and one or more MIDAS Systems as peripheral processors.

The operational MIDAS System can accept data from analog tapes or CCT tapes and will be upgraded in Phase II to handle data from high-density digital tapes. This should enable the system to accept and process data from many scanner sources—for example, from a specified scanner sponsored by AAFE (Advanced Applications Flight Experiment). In processing these and other data, system performance can be stated in terms of data throughput. Then knowing

the limitations of the system developed under Phase II, we should be better able to specify the special-purpose digital processing concept which would process data at much higher data rates. Current data rate through the MIDAS Classifier is about 12 megabits/sec. The investigation should examine data rates on the order of 100 megabits/sec.

In evaluating the next generation system, some thought should also be given to a more advanced processing concept. The present system uses a single hardwired quadratic decision-rule. While it is possible to design a generalized quadratic decision-rule which includes a linear decision-rule, the hardware may prove prohibitive in cost. Thus a study of hardware and cost impacts and trade offs is desirable.

6
HARDWARE DESCRIPTION

6.1 MIDAS CONTROL SECTION

The MIDAS System has three distinct modes of operation:

(1) Classifier mode

(2) System A/D conversion

(3) System D/A conversion

In Mode 1, the MIDAS System accepts multichannel data in either analog or digital form and synchronizes the conversion and flow of this data to the classifier.

In Mode 2, the MIDAS System accepts multichannel analog data, converts it to digital form, and synchronizes the transfer of the resultant data to the PDP-11/45 computer.

In Mode 3, digital data is transferred from the PDP-11/45 to the hybrid D/A subsystem where it is converted to analog form.

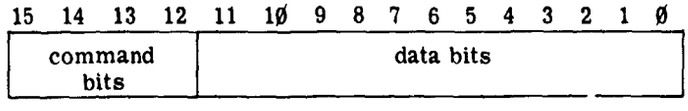
The section of the MIDAS System that establishes the operating mode and controls the flow of data from source to destination is referred to as the control section. A block diagram of the control section is shown in Fig. 25. In the following paragraphs we describe the blocks that make up the control section design and detail their interrelations.

6.1.1 SYSTEM CONDITIONER

The conditioning of the MIDAS System to one of its operating modes is accomplished in the system conditioner of the control section. This is achieved by sending codes that define the mode parameters from the PDP-11/45 through a DR-11C interface to the system conditioner. The defining codes are then stored in the conditioner where they are decoded and transferred to the appropriate sections of the system. The parameters that define the system mode and their corresponding codes are detailed in the section below.

MIDAS CONTROL COMMANDS (TO DR-11C)

A. Command Format



Bits 15-12 (highest-order bit = 15) are used to specify one of the 16 commands.

Bits 11-Ø are used to pass command information to the processor control logic.

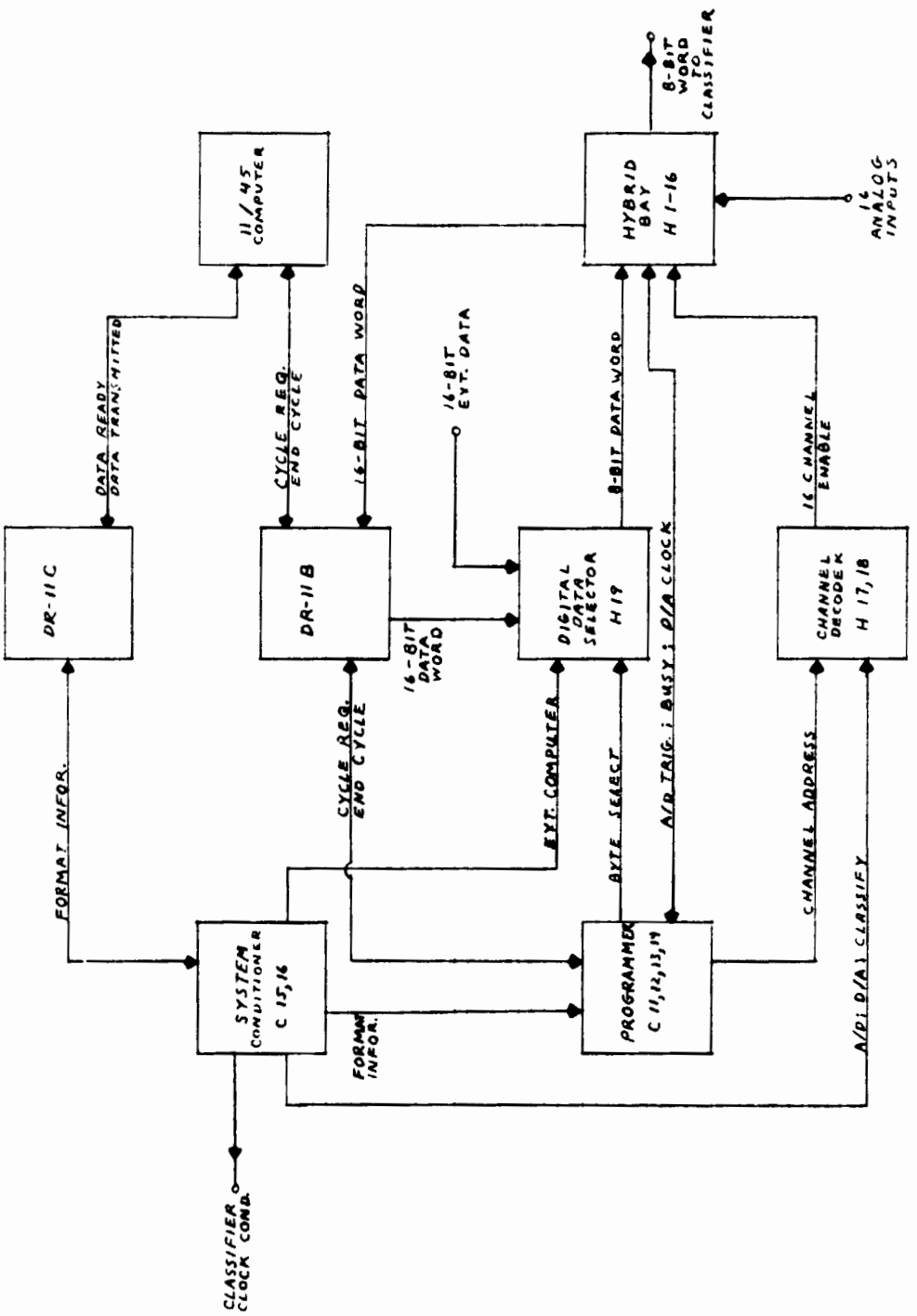


FIGURE 25. BLOCK DIAGRAM OF THE CONTROL SECTION

B. Commands (to DR-11C)

The commands to the system conditioner follow, listed in order of binary command.

(1) ØØØØ - Register Interrogate & Misc.

15	14	13	12	11	1Ø	9	8	7	6	5	4	3	2	1	Ø
Ø	Ø	Ø	Ø	not used				register interrogate				misc.			

Register Interrogate -

bit 7: Ø - register interrogate disable (disregard bits 6-4)
1 - register interrogate enable

bits 654:
ØØØ - least-significant byte (LSB) of line count register (1Ø bits)
ØØ1 - most-significant byte (MSB) of line count register (1Ø bits)
Ø1Ø - scanner period (12-bit binary)
Ø11 - scanner/roll sync interval (12-bit binary)

Miscellaneous -

bits 321Ø:
ØØØØ - no action taken
ØØØ1 - reset
ØØ1Ø - start
ØØ11 - gate reset
Ø1ØØ - clock strobe
Ø11Ø - transfer
Ø111 - A/D strobe
1ØØØ - start reset
1ØØ1 - reset CSRØ (on DR-11C) and ATTN bit (on DR-11B)
1Ø1Ø - spare
1Ø11 - n.a.
11ØØ - n.a.
11Ø1 - n.a.
111Ø - n.a.
1111 - n.a.
} no action taken

(2) ØØØ1 - A/D Channel Select RAM Load

15	14	13	12	11	1Ø	9	8	7	6	5	4	3	2	1	Ø
Ø	Ø	Ø	1	not used				address				channel data			

bits 7-4: RAM address for channel number to be stored

bits 3-Ø: channel number to be stored in RAM addressed by bits 7-4

(3) ØØ1Ø - Video Gate RAM Load

15	14	13	12	11	1Ø	9	8	7	6	5	4	3	2	1	Ø
Ø	Ø	1	Ø	resolution # to be loaded into video gate RAM											

bits 11-Ø: Load up Video Gate RAMs in order. Addressing is performed by the hardware starting at address Ø after a gate reset command is issued. Maximum of 16 video gate limits can be loaded (2 limits per gate).

(4) ØØ11 - Calibration Gate RAM Load

15	14	13	12	11	1Ø	9	8	7	6	5	4	3	2	1	Ø
Ø	Ø	1	1	resolution # to be loaded into calibration gate RAM											

bits 11-Ø: Load up calibration gate RAMs in order. Addressing is performed by the hardware starting at address Ø after a gate reset command is issued. Maximum of 16 calibration gate limits can be loaded (2 limits per gate).

(5) Ø1ØØ - A/D Condition Load

15	14	13	12	11	1Ø	9	8	7	6	5	4	3	2	1	Ø
Ø	1	Ø	Ø			A/D mode		# of gates				# of channels			

line start/stop enable
data format switch

bits 3-Ø: total number of channels to be selected from analog tape for digitization

bits 7-4: total number of gates per scan line (video gates plus calibration gates)

bits 9-8: A/D mode select

bit 98:

ØØ - deselect A/D

Ø1 - A/D analog mode (i.e., data transfer to computer memory via DR-11B)

1Ø - A/D classify mode (i.e., data transfer directly to classifier hardware via hybrid circuitry from analog tape)

11 - A/D external mode (i.e., data transfer directly to classifier hardware via hybrid circuitry from an external source)

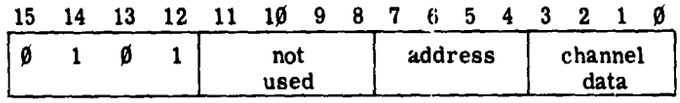
bit 1Ø: line start/stop enable

- value of Ø—disable line start/stop hardware
- value of 1 — able line start/stop hardware

bit 11: data form t witch (relates only to A/D analog mode transfer via DR-11B to core)

- value of Ø—one 8-bit data value per 16-bit bus transfer
- value of 1—two 8-bit data values per 16-bit bus transfer

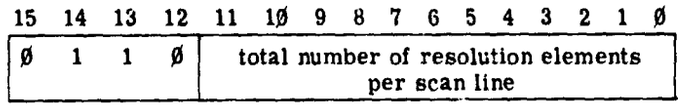
(6) Ø1Ø1 - D/A Channel Select RAM Load



bits 7-4: RAM address for channel number to be stored

bits 3-Ø: channel number to be stored in RAM addressed by bits 7-4

(7) Ø11Ø - D/A Duty Cycle



bits 11-Ø: total number of resolution elements to be generated per scan line (i.e., between generated sync pulses)

(8) Ø111 - D/A Video Resolution Cycle

15	14	13	12	11	1Ø	9	8	7	6	5	4	3	2	1	Ø
Ø	1	1	1	total number of video res'n elements per scan line											

bits 11-Ø: total number of video resolution elements to be generated per scan line

(9) 1ØØØ - D/A Condition Load

15	14	13	12	11	1Ø	9	8	7	6	5	4	3	2	1	Ø
1	Ø	Ø	Ø		D/A mode			not used				# of channels			

└→ data format

bit 11: data format switch (relates only to the D/A mode of transfer from core directly to analog tape via the DR-11B)

- value of Ø — 1 8-bit data values per 16-bit bus transfer
- value of 1 — 2 8-bit data values per 16-bit bus transfer

bit 1Ø-8: D/A mode select

bits 1Ø98:

ØØØ - not assigned
ØØ1 - digital classification (from core)
Ø1Ø - digital classification (from external source)
Ø11 - digital transfer
1ØØ - diagnostic check of three-states (digital word from computer to computer)
1Ø1 - diagnostic of A/D-D/A (from computer to D/A thru A/D back to computer)
11Ø - digital-to-analog conversion
111 - not assigned

(10) 1ØØ1 - Clock Control

15	14	13	12	11	1Ø	9	8	7	6	5	4	3	2	1	Ø
1	Ø	Ø	1	D/A line count clock			A/D resolution rate			A/D calib. gates			A/D video gates		

bits 11-9: D/A line count clock
bits 8-6: A/D resolution rate
bits 5-3: A/D calibration gates
bits 2-Ø: A/D video gates

Clock Control Schedule (for 1ØØ1 Command)

Gate Clock - bits 5-3 (calib.) of 2-Ø (video):

Sample Rate (f = resolution rate)	543 210
4f	ØØØ
2f	ØØ1
f	Ø1Ø
f/2	Ø11
f/4	1ØØ
f/8	1Ø1
f/16	11Ø

A/D Resolution Rate - bits 8-6:

Tape Speed (ips)	876
6Ø	ØØØ
3Ø	ØØ1
15	Ø1Ø
7-1/2	Ø11
3-3/4	1ØØ
1-7/8	1Ø1
15/16	11Ø

(11) 1Ø1Ø - Not assigned

(12) 1Ø11 - Port Select

15	14	13	12	11	1Ø	9	8	7	6	5	4	3	2	1	Ø
1	Ø	1	1	data											

data: to be assigned

(13) 11ØØ - Classifier Clock

15	14	13	12	11	1Ø	9	8	7	6	5	4	3	2	1	Ø
1	1	Ø	Ø	BURST N-1									freq. divide		

Burst stop/reset
Free Run/Burst mode

bits 2-Ø: FREQUENCY DIVIDE

Clock Rate (f = 3.2 MHz)	21Ø
f	ØØØ
f/2	ØØ1
f/4	Ø1Ø
f/8	Ø11
f/16	1ØØ
f/32	1Ø1
f/64	11Ø
f/128	111

bit 3: Ø—BURST/STOP
1—BURST/RESET

bit 4: Ø—FREE RUN MODE
1—BURST MODE

bits 11-5: BURST N (STORE N-1)

(14) 11Ø1 - Line Count and Start/Stop Control

15	14	13	12	11	1Ø	9	8	7	6	5	4	3	2	1	Ø
1	1	Ø	1	function		data									

bits 11-1Ø: function to be performed depends on current processor mode (i.e., either A/D or D/A):

(1) When processor is in D/A Mode

bits 11 1Ø
Ø Ø - load LSB of line count (10-bits BCD)
Ø 1 - load MSB of line count (10-bits BCD)

(2) When processor is in A/D mode

bits 11 1Ø
Ø Ø - load LSB of line start (10-bits BCD)
Ø 1 - load MSB of line start (10-bits BCD)
1 Ø - load LSB of line stop (10-bits BCD)
1 1 - load MSB of line stop (10-bits BCD)

Modes (1) and (2) Bits 9-Ø: data to be loaded

(15) 111Ø - not assigned

(16) 1111 - not assigned

6.1.2 PROGRAMMER

That part of the control section which synchronizes the flow of data from one of the input ports of the MIDAS System to an output data bus is called the programmer. It is made up of two separate components; one of these, the digital synchronizer, is used to control digital input data to the system; the second component, the A/D synchronizer, controls analog input data. In the following sections, we discuss the way the two components of the programmer operate in the different MIDAS System modes.

Digital Input Synchronizer. This section of the programmer receives start commands from a clock source that starts the transfer of data from the PDP-11/45 and finally controls the latching of the data into the appropriate register on the hybrid cards. The source of the initializing signal is dependent on the operating mode of the system. If the system is performing a classification of the input data, the start command originates in the classifier clock control. When the system is in the D/A mode, the request command is generated in the D/A clock generator. A block diagram of the digital synchronizer is given in Fig. 26.

The start command generates a request for data from the computer in the request logic of Fig. 26. Upon receiving the request for data in the DR-11B interface, the computer initiates an output data transfer cycle. When the transfer of data to the output bus is complete, the DR-11B sends an end data signal to the digital synchronizer. The end data signal starts the unpacking of the 16-bit word on the output bus into two 8-bit data samples which are loaded into registers on the appropriate hybrid cards. After a data word has been unpacked and stored, the current channel address is compared with the final address (loaded from the system conditioner into a 4-bit latch). If the final hybrid card has not been loaded with data, a request for data is again sent to the computer. This sequence of transferring data from the computer is continued until all the channels are loaded. When the cycle is complete, the digital synchronizer is reset, and the data in the holding registers on the hybrid cards are ready for classification or D/A conversion.

Digital Output Synchronizer. This section of the programmer controls the operation of the MIDAS System when the input data are multichannel analog. With this type of input data, the system has two separate modes. In one mode, the multichannel analog is digitized, stored, and finally presented to the classifier for processing. The components of the system that perform these functions are all located on the hybrid card; the timing signals that control them are generated in the classifier clock. A signal from the clock logic (A/D strobe) starts the A/D converters on the hybrid cards. Another signal from the classifier clock generator (A/D latch) latches the converted data into registers and from there the data are presented to the output three-state gates of the hybrid cards. These multichannel data are then multiplexed to the classifier by controlling the enable lines of the output three-state gates with signals derived from the address lines of the classifier clock.

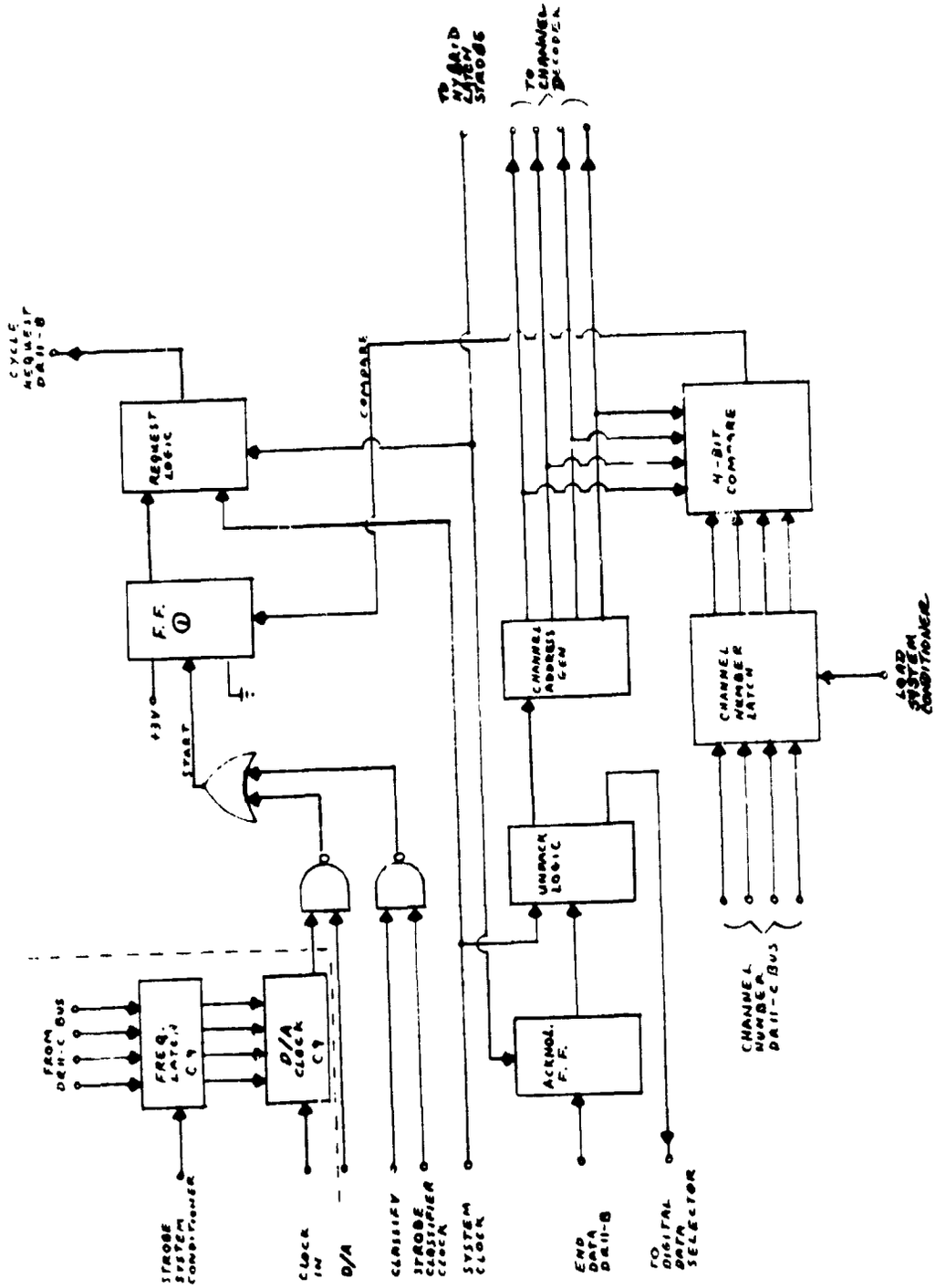


FIGURE 26. BLOCK DIAGRAM OF THE DIGITAL INPUT SYNCHRONIZER

The second mode of operating the system with analog input data is the conversion of the analog data to digital data, and the transfer of the data to the PDP-11/45 for analysis. A block diagram of the A/D synchronizer that performs the data transfer is presented in Fig. 27. For this mode of operation, the A/D converters are controlled by an A/D timing generator which determines the sampling of the analog data. When an A/D conversion has been completed, the end of the A/D busy command starts the A/D synchronizer to pack and transfer words to the computer. The words are packed by enabling the three-state gates of two data channels for each word transfer. The trailing edge of the A/D busy signal sets the start flip-flop in the synchronizer. When the start flip-flop is set and the cycle logic is in a reset state, the cycle logic generates three signals that pack and transfer one word to the PDP-11/45. After these signals have been generated, the cycle logic remains in a set condition. The end cycle from the DR-11B, which tells the synchronizer that the word has been accepted, resets the cycle logic. If the start flip-flop is still in the set state, the cycle logic initiates another word transfer. This sequence continues until the start flip-flop is reset and the cycle is complete (i.e., all channels for the A/D sample pulse have been transferred).

The first signal out of the cycle logic checks the comparator to see if the cycle is complete. If all the channels have been transferred, the signal out of the comparator resets the start flip-flop. If the cycle is not complete, a second signal is generated in the cycle logic which loads the address counter into the lower byte register and then updates the address counter. A third signal which loads the address counter into the upper byte register and request a data transfer to the DR-11B is then generated. The byte registers are decoded in the channel decoder which, in turn, enables appropriate three-state gates on the hybrid cards.

6.1.3 DIGITAL DATA SELECTOR

The digital data selector of the control section performs the following functions:

(1) Accepts two 16-bit input words —one from the computer unibus, and the other from the external digital input port.

(2) Selects one of the two input words.

(3) Sequentially unpacks the selected word into 8-bit samples.

A block diagram of the digital data selector is given in Fig. 28.

The two 16-bit input words to the data selector are divided into 8-bit bytes by feeding the lower bit of the two words to one 8-bit word selector (A) and the higher order bits to another (B). The input data source can then be selected by appropriately controlling the select lines of the word selector.

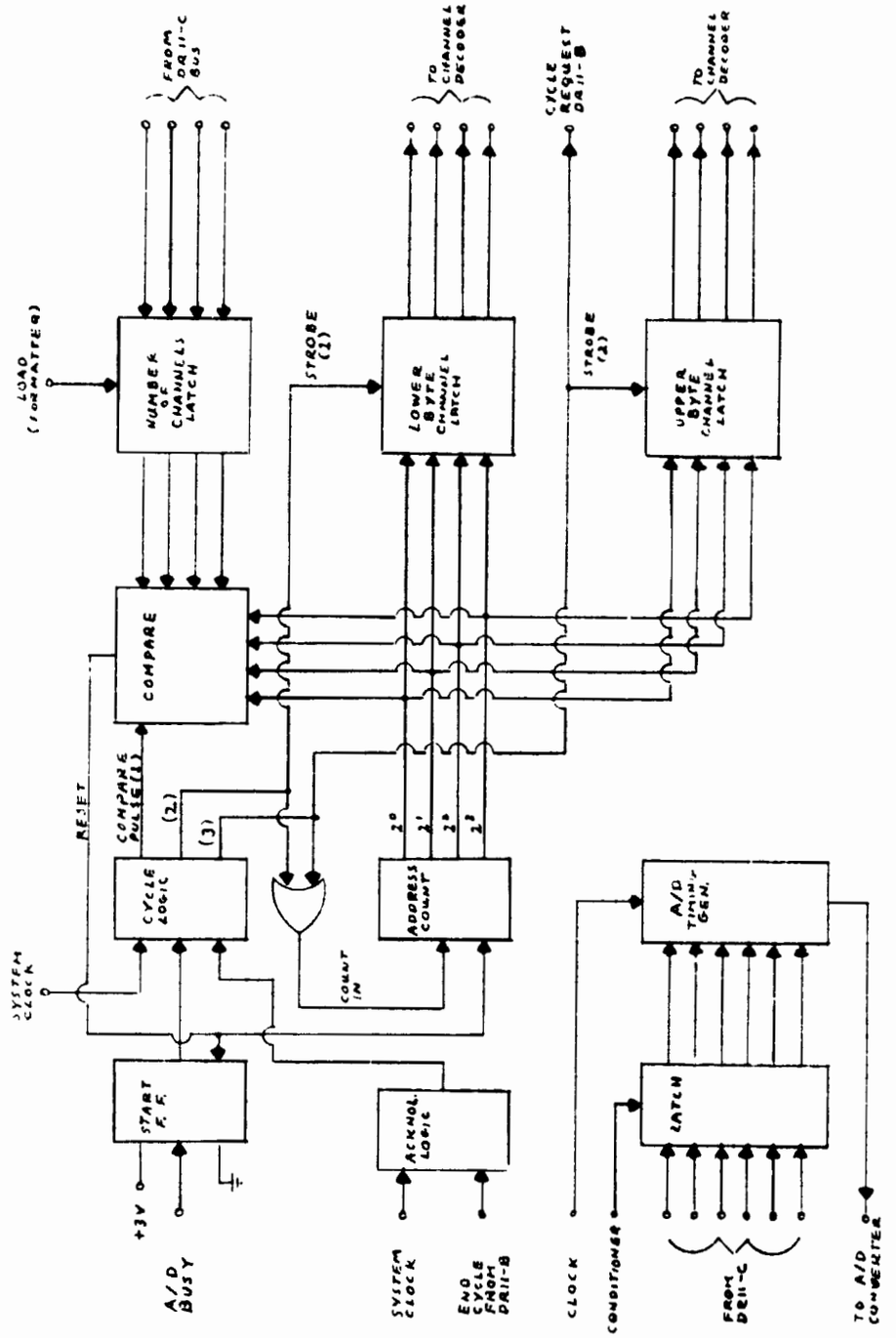


FIGURE 27. BLOCK DIAGRAM OF THE DIGITAL OUTPUT SYNCHRONIZER.

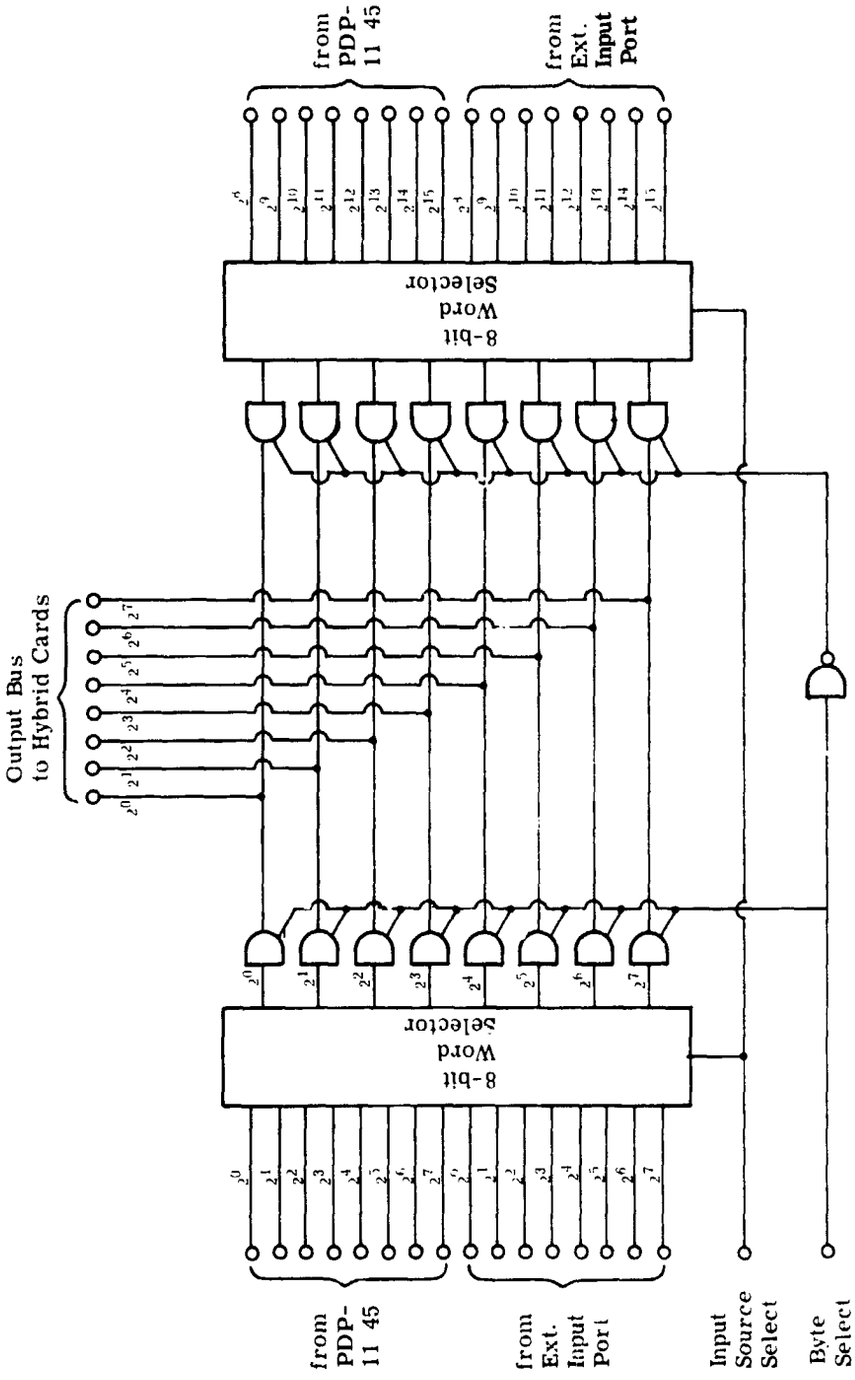


FIGURE 28. BLOCK DIAGRAM OF THE DIGITAL DATA SELECTOR

The outputs of the word selectors are terminated in three state gates. The gate outputs of the corresponding bits from the lower and higher order bytes are tied together. That is, the 2^0 three-state gate is tied to the 2^8 three-state, the 2^1 to the 2^9, etc.

In this way the 8-bit output lines of the data selector are formed. The unpacking of the 16-bit input word is achieved by alternately enabling the three-states of the lower and higher order bytes. The control of the three-state enable lines is generated in the programmer of the control section.

6.1.4 CHANNEL DECODER

The function of the channel decoder is to decode the channel address supplied by the programmer and to generate a channel enable signal to the hybrid cards.

When the system is operating with analog input data, the channel enable signal selects the appropriate three-state gate of the hybrid cards. This insures that the correct channel is presented to either the classifier or the PDP-11/45 under the control of the programmer.

For the system operating with digital input data, the channel decoder not only enables the output three-state gates of the hybrid cards, but it also generates the strobe pulse for latch A of the hybrid cards. This strobe is timed by the programmer to insure that the data are latched into the correct hybrid card.

6.2 HYBRID CARDS

The MIDAS System can accommodate multichannel analog or digital data. The source for this data may be the 16 analog inputs of MIDAS, the 16 external inputs of the system, or the 16-bit data lines from the PDP-11/45 (through a DR-11B interface). The particular data source is selected through the system conditioner of the control section. Irrespective of source, all the data are presented to separate hybrid cards within the control section where it is converted (where required), stored, and eventually sent to the selected destination.

In MIDAS there are 16 hybrid cards, one for each data channel; a block diagram of the cards is given in Fig. 29.

If the selected source for input data is multichannel analog, the separate signals are brought from the 16 analog input jacks to their assigned hybrid cards. The analog signals are converted on the cards to digital form and stored for eventual presentation to the selected destination. This destination is the 8-bit input lines of the classifier when the system is operating in the classifier mode. When the system is operating in the system A/D mode, the outputs of the hybrid cards are transferred to the PDP-11/45 computer.

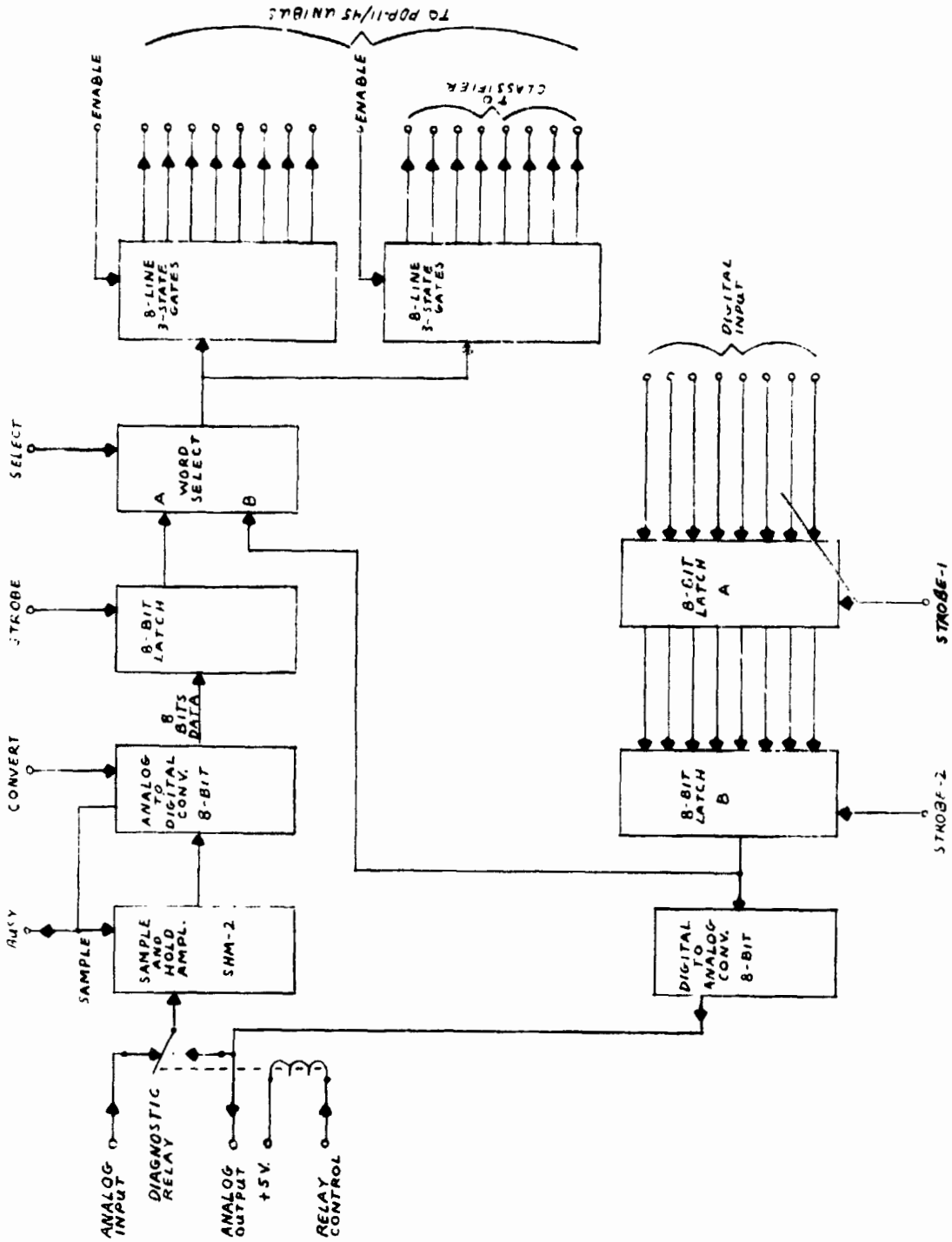


FIGURE 29. BLOCK DIAGRAM OF THE HYBRID CARD

For analog data the hybrid card is operated in the following manner: The analog signal is fed to a sample-and-hold amplifier located on the hybrid card where the signal is held during the time it is being converted to digital form. The A/D conversion is performed by an 8-bit converter on the card and the control of the converter is generated in the programmer of the control section. Upon completion of the A/D conversion the digital data is stored; then it is presented to the output three-state gates through a word selector. The word selector acts as a data switch; it selects word A when the system is operating with analog input data and selects word B when the input data are digital.

If the selected input data are digital, the source for this data may be the 16-bit word from the PDP-11/45 or the 16-bit external digital input port of the MIDAS System. The selection of data source is established and stored in the system conditioner and the multiplexing and unpacking of the digital data is accomplished in the digital-data selector of the control section.

For digital input data, the hybrid cards operate in the following manner. After the data source is selected and the 16-bit data word is unpacked into 8-bit samples, the 8-bit samples are stored sequentially in latch "A" on the assigned channel hybrid card. After the data for all the channels has been loaded into the corresponding A latches, it is transferred simultaneously into latch "B." The data from latch B are fed to the word selector on the hybrid card; then, when the system becomes conditioned to accept digital input data, the word stored in latch B is fed to the output three-state gates through the word selector. As indicated on the block diagram of the hybrid card (Fig. 29), the contents of latch B are always presented to the D/A converter. In this way the conversion of the digital data stored on the card can always be observed at the output of the D/A converter whether the system is in a digital classify mode or a system D/A mode.

The output three-state nand gates on the hybrid cards form both 8-bit and 16-bit output data lines; these lines are formed by joining together the corresponding outputs of the three-state gates on each card. The data from a particular card or cards can then be presented to the appropriate destination by enabling the three-state gates of the desired card. The 8-bit data line is used to present data to the classifier and the 16-bit line is used to transfer data to the PDP-11/45. By using a 16-bit data line from the hybrid cards to the PDP-11/45, two 8-bit data samples can be loaded onto the input unibus for each transfer cycle in order to achieve the maximum throughput efficiency.

Each hybrid card contains a diagnostic relay which is used to calibrate the cards and to diagnose faults utilizing the PDP-11/45.

6.3 CLASSIFIER SECTION OF THE SYSTEM

The classifier section of the MIDAS System consists of four bays, each containing 13 wire-wrap cards. These bays are nearly identical in that each has the circuitry for one quadratic pipe computation as described in Subsection 4.2.1. This computation requires twelve cards: (a) a mean card, (b) a variance card, (c) eight matrix multiplier cards, (d) a squaring card, and (e) a square accumulate card. These card types are described in following subsections. In addition, each bay contains one of the following one-of-a-kind wire wrap cards: k^2 card, recognition card, diagnostic/output card, and clock card; these wire-wrap cards are also described in succeeding text.

The wire-wrap hardware employs Augat 8136-URG1TG universal circuit boards and 8170-RG1 card racks and back planes. The cards can hold up to fifty 16-pin integrated circuits (ICs) or eighteen 24-pin ICs.

6.3.1 MEAN CARD

The prime purpose of this card is to sequentially add $-\mu$ to the data vector using two's complement arithmetic. An overflow test which is defined by $A_S \cdot B_S \cdot \overline{C}_S + \overline{A}_S \cdot \overline{B}_S \cdot C_S$ = overflow, where A + B = C and the subscript S indicates sign bit. The $-\mu$ is stored in an 8-bit RAM. The X inputs are multiplexed from a bus located in the A/D cards. The output of the adder is loaded into latches 74174 as processor outputs and into 8T10s for diagnostic outputs. A block diagram of this card is shown in Fig. 30.

There are two selection devices on the mean card, one to select a RAM in that bay for writing into that RAM and the other to select a diagnostic bus output to the diagnostic card. In the RAM selection, a RAM write select comes from clock card which does a bay select decode to enable the decoder on the mean card; this also sends 4 bits to the decoder on mean card. A read pulse is then applied which can now propagate through only one decoder input on one mean card to a RAM on one other card in that bay. In the diagnostic bus output selection an identical procedure is followed except that the origin of the signals is on the diagnostic output card.

6.3.2 VARIANCE CARD

The prime purpose of this card is to multiply an 8-bit number by a 12-bit number and put out an 8-bit result with overflow. There are no latches in the processing stream on this card—but the diagnostic bus has a 9-bit 8T10 latch. The input $X-\mu$ is represented by

XXXXXXXX. (2's complement number)

The coefficient $1/\sigma$ is represented by

X.XXXXXXXXXXX (2's complement number)

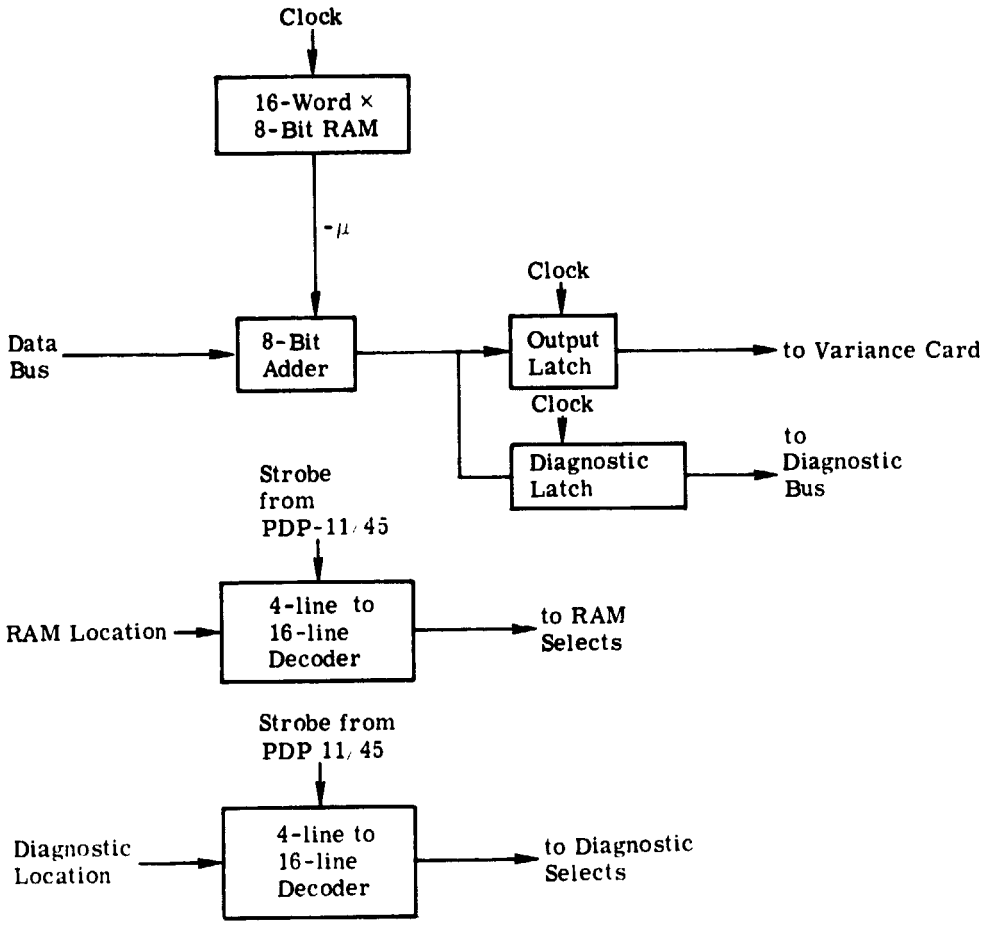


FIGURE 30. BLOCK DIAGRAM OF THE MEAN CARD

Since $1/\sigma$ is never negative, the sign bit is always zero. The eight output bits of the 20-bit product are selected from

```
X-----XXX.XXXX-------
 11  11        76      0
 98  43
```

The overflow test simply consists of testing bits 14 through 19 to see whether they are all at the same level. If so, there is no overflow and the overflow output is high. A block diagram of this card is shown in Fig. 31.

6.3.3 MATRIX MULTIPLIER

This card has two main functions: (1) multiply $[(X - \mu)/\sigma]_i$ by a coefficient, and (2) to accumulate several products of this multiplication. The $[(X - \mu)/\sigma]_i$ input from the variance card is stored in a latch on this card. The accumulator is cleared at the start of each signature computation. Each 16-bit product of the multiplication is truncated to the high-order 10 bits and latched before being applied as one input to the accumulator. The accumulator consists of a 12-bit adder with a feedback latch providing the second input. The two expansion bits in this accumulator are adequate for eight adds since the largest product is 2^8 (the result of $(-2^7) \times (-2^7)$ after truncation). Added eight times, this number yields 2^{11}, which is the largest negative 12-bit number. Since the two operands were negative, the accumulated products should be positive. Hence, for this special case the overflow has caused an error in sign only. However, this number is to be squared in the next step, thus correcting the only possible error. There are processor data-stream output-latches (8T10s with 12 bits) and diagnostic bus latches (8T10s, also with 12 bits). These latches are enabled at the proper time by the MTXMPX signal from the square-accumulator card and loaded by CØ, the fundamental clock frequency. A block diagram of this card is shown in Fig. 32.

6.3.4 SQUARE CARD

The prime purpose of this card is to do a 9×9 square on the 12-bit input from the matrix accumulator. The first operation is to check for overflow in the matrix-accumulator input signal before reduction to nine bits. The four high-order bits (8 through 11) are tested to see whether they are all at the same level. If so, there is no overflow and the overflow output is high. Bits 8, 9, and 10 are then dropped to reduce the square input to nine bits. A block diagram of this card is shown in Fig. 33.

6.3.5 SQUARE ACCUMULATOR

This card's main function is to accumulate the squares from the square card. It has both input and output latches as well as feedback latches. It has a 12-bit input (the sign bit

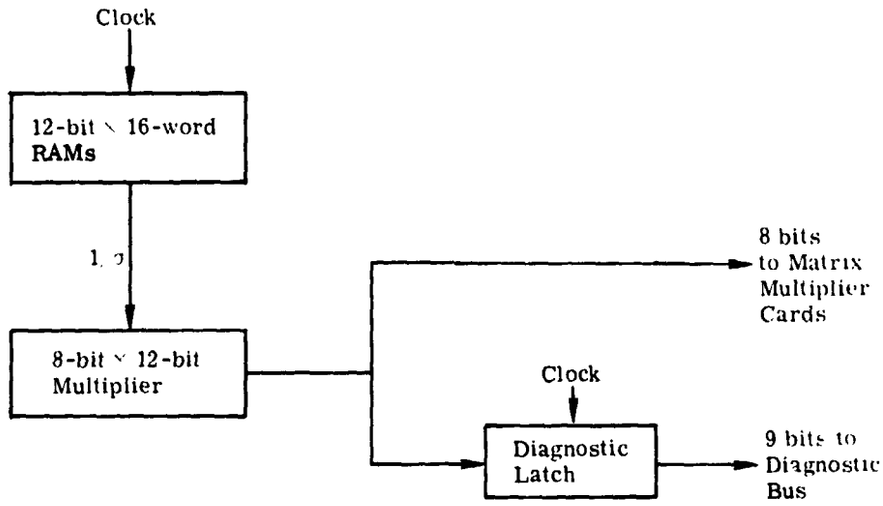


FIGURE 31. BLOCK DIAGRAM OF VARIANCE CARDS

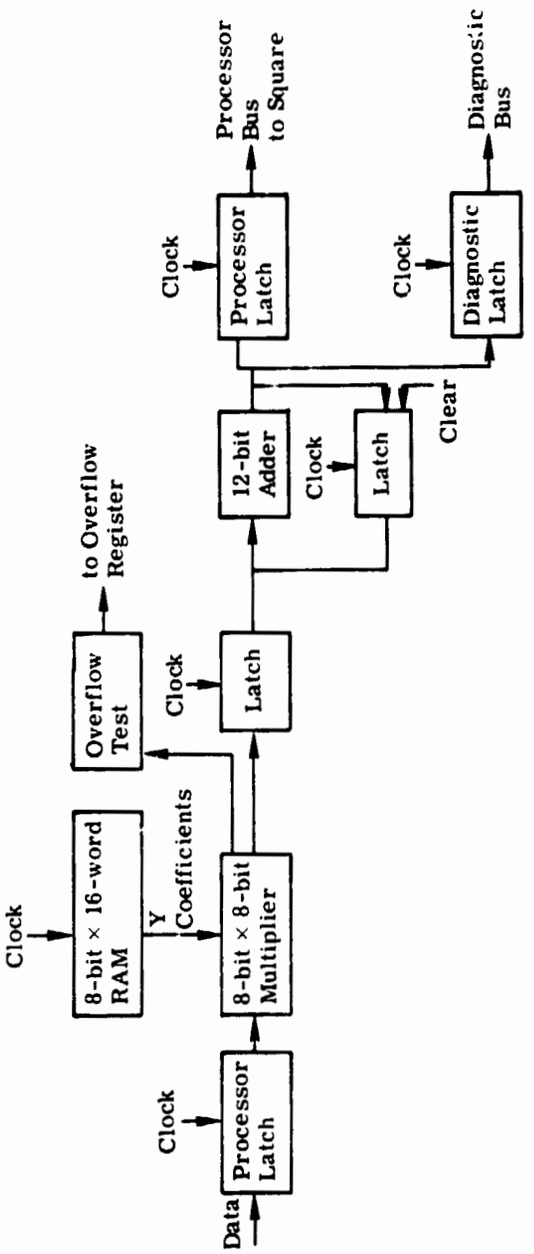


FIGURE 32. BLOCK DIAGRAM OF THE MATRIX MULTIPLIER CARD

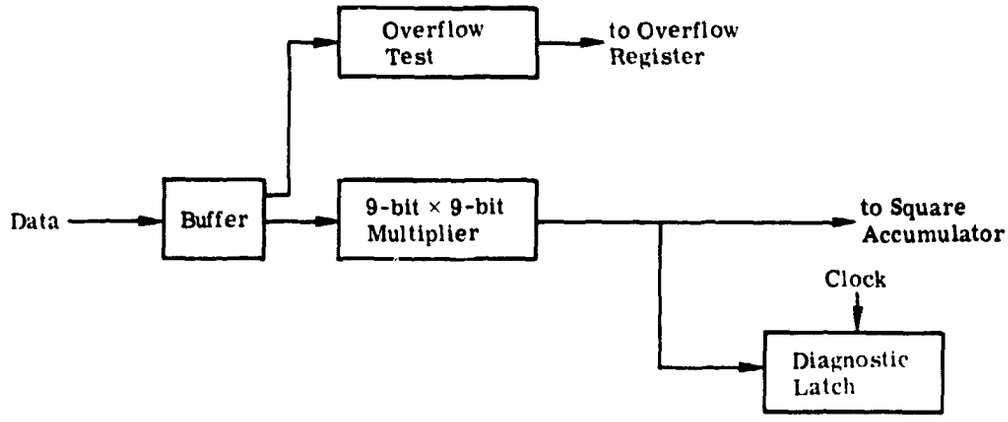


FIGURE 33. BLOCK DIAGRAM OF THE SQUARE CARD

and the low-order seven bits from the 9 × 9 square are not brought in). The adder is 14 bits but bit Ø is not used, so that three expansion bits are provided. The twelve most significant bits are brought out via 8T10s on both the processor data path and diagnostic path. The diagnostic bus is multiplexed from the diagnostic/output card, whereas the processor bus is multiplexed from the k^2 card. The square-accumulator card also has a number of control functions:

(1) Acts as a buffer for the clocks (bay buffer).

(2) Multiplexes the matrix multipliers (enables 8T10s on the two output busses which are then loaded at the next CØ clock).

(3) Controls the matrix multipliers' output bus to the 9 × 9 square card using a 3-line to 8-line decoder.

(4) Clears the matrix multipliers' accumulators.

(5) Clears its own accumulator.

(6) Multiplexes its own output 8T10s to the processor and diagnostic busses.

(7) Contains overflow shift register. Overflows from mean, variance, and 9 × 9 square are loaded into the shift register and then dropped into the overflow output register (8T10s) at proper time.

A block diagram of this card is shown in Fig. 34.

6.3.6 k^2 CARD

The main function of this card is to perform 1Ø-bit by 12-bit multiply. The 12-bit data comes from the square accumulators and the 1Ø-bit data is the k^2 normalizing constant. Its input is buffered and there are no input latches since the outputs of the square-accumulator cards are latched and are bussed into it. There is a 74174 output latch register.

Another function is to control the square accumulators' bus, which it feeds using a 2-line to 4-line decoder. The signal that multiplexes the output bus also multiplexes the overflow bus to the recognition card. A block diagram of this card is shown in Fig. 35.

6.4 FINAL DECISION

The main function of this card is to determine which calculated material's exponent is the minimum. It also performs the Chi^2 test on the sum of the squares. At clock times 5 and 13 a new set of sum of squares from the k^2 card becomes available. A 14-bit add is performed with the $\{n \mid D\}$ input truncated to bits 3 through 14, while the sum of squares is a 14-bit number in bits 0 through 13.

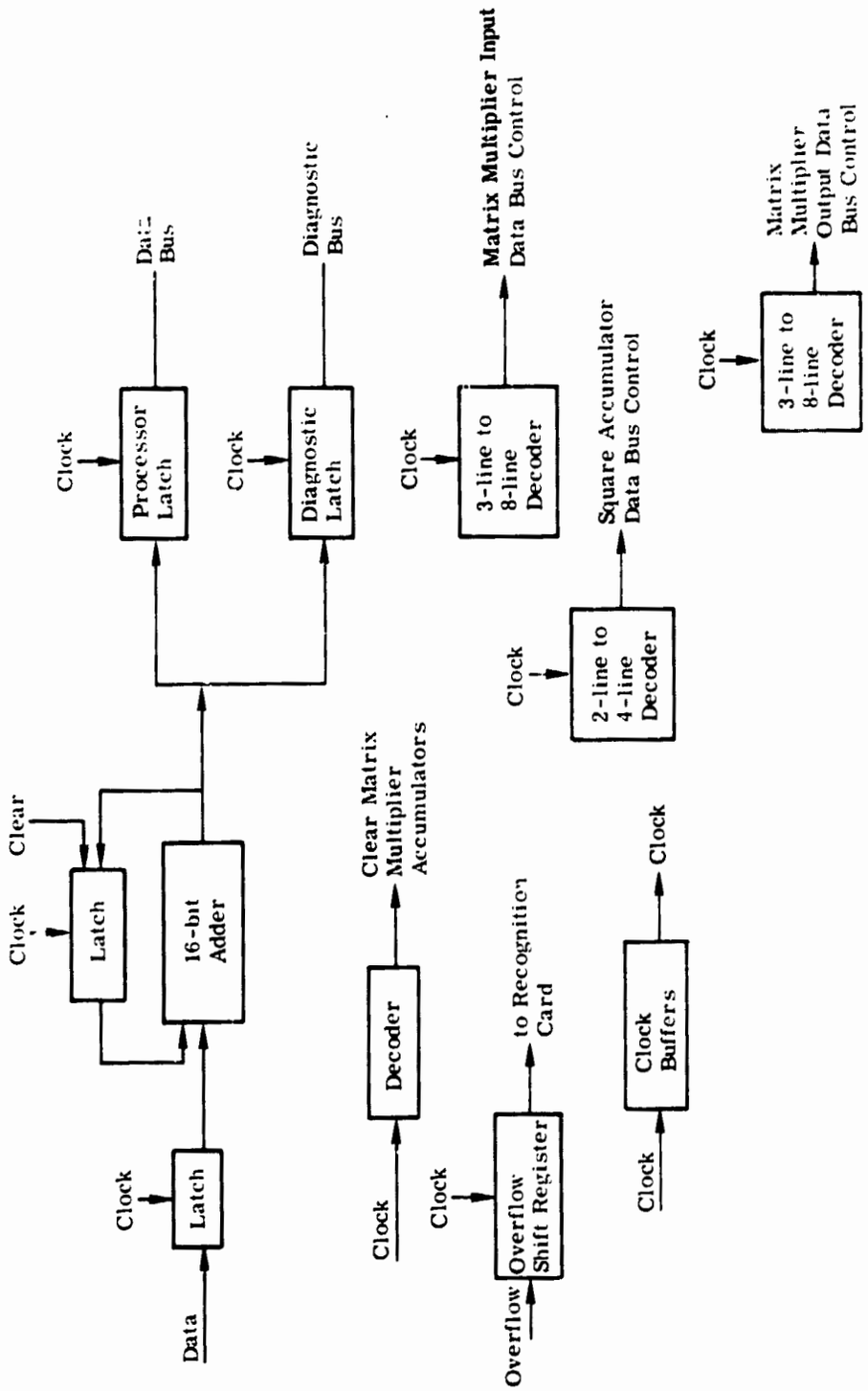


FIGURE 34. BLOCK DIAGRAM OF THE SQUARE-ACCUMULATOR CARD

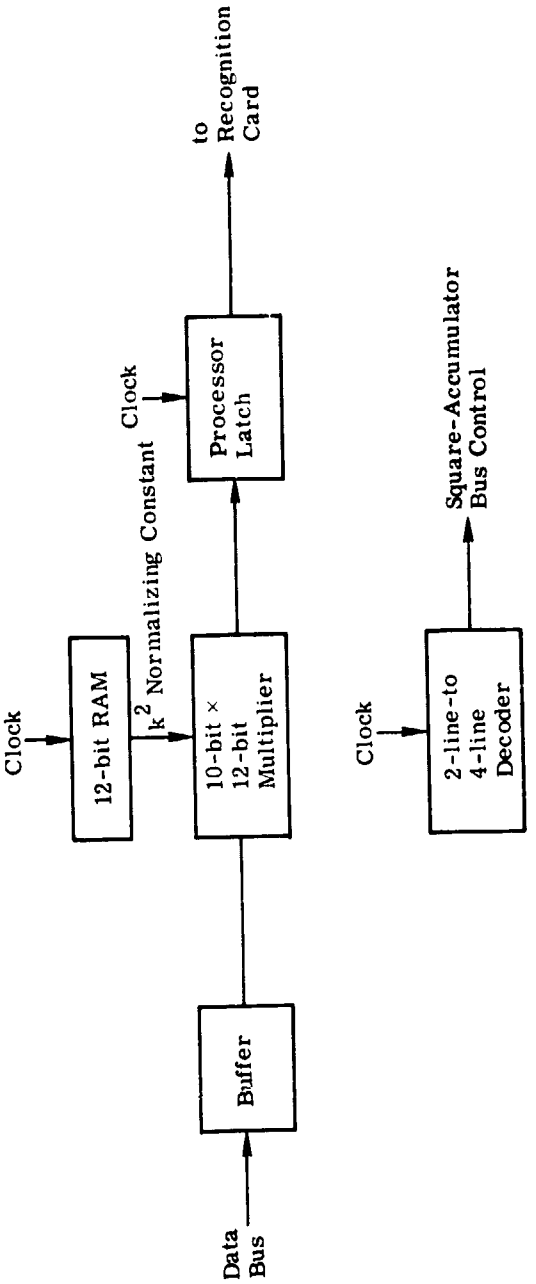


FIGURE 35. BLOCK DIAGRAM OF THE k^2 CARD

The selection time of the overflow into the recognition logic is the same as the selection time of the square accumulation output; therefore the overflow shift register output must have two stages of latches to make up for the delay in the k^2 multiply and in the $\ln |D|$ add on the recognition card. Similarly, the output of the Chi^2 test must be latched to account for the $\ln |D|$ add delay.

A new sequence of testing begins at clock time 8 for the 8-channel set-up or at clock time 14 for the 4-channel set-up. At these times the previous recognition result is transferred to the output latches. There are two internal holding latches, one for exponent and one for material number. After loading these two quantities into the output latches, the internal ones are cleared. There is an artificial seventeenth bit set at this time which makes the stored exponent look like it is larger than any 16-bit computed exponent. Thus the first test will always succeed and the exponent be stored if no overflow has occurred. When this condition is met the artificial bit is reset. The material code is a 5-bit number. It is MSB ØØØXXXX where the X's give the material number (Ø through 15), or code 16 indicates no material selected. A block diagram of this card is shown in Fig. 36.

6.5 CLASSIFIER TIMING

The classifier consists of the four "pipeline" computers in parallel whose outputs finally converge on the circuits to scale the exponents of the density function and intercompare these exponents for a decision. The sequence of operations can be visualized as shown in Fig. 37 where data is shown entering the A-D converters at t = -16 in the upper left corner of the diagram. A sample, consisting of a vector of eight elements of eight bits each is passed through the computational circuits indicated and emerges at the bottom right of the diagram as a classification code of 5 bits. The general appearance and function of the arithmetic operations so diagrammed is that of a "cascade" in which the breadth of the cascade in time is proportional to the computational load of a particular circuit.

Each "pipe" or "cascade" processes the computation of the quadratic form for the exponent of the Gaussian distribution for two distributions. There are, then, a sequence of alternating computations of the first exponent in cascade 1, the second exponent in cascade 1, the third exponent in cascade 2, etc. In the present machine, there are four such cascades operating in parallel, allowing the computation of eight exponents at once. In phase II the machine will be expanded to eight cascades to allow computation of 16 distribution exponents at a time.

The timing diagram, Fig. 37, shows the flow of two exponent computations and the resulting decision, neglecting the fact, for the sake of simplicity, that the cascade would normally contain portions of other computations for the preceding and subsequent samples. Data is entered into the A-D converters and is available for computation at the end of 16 machine

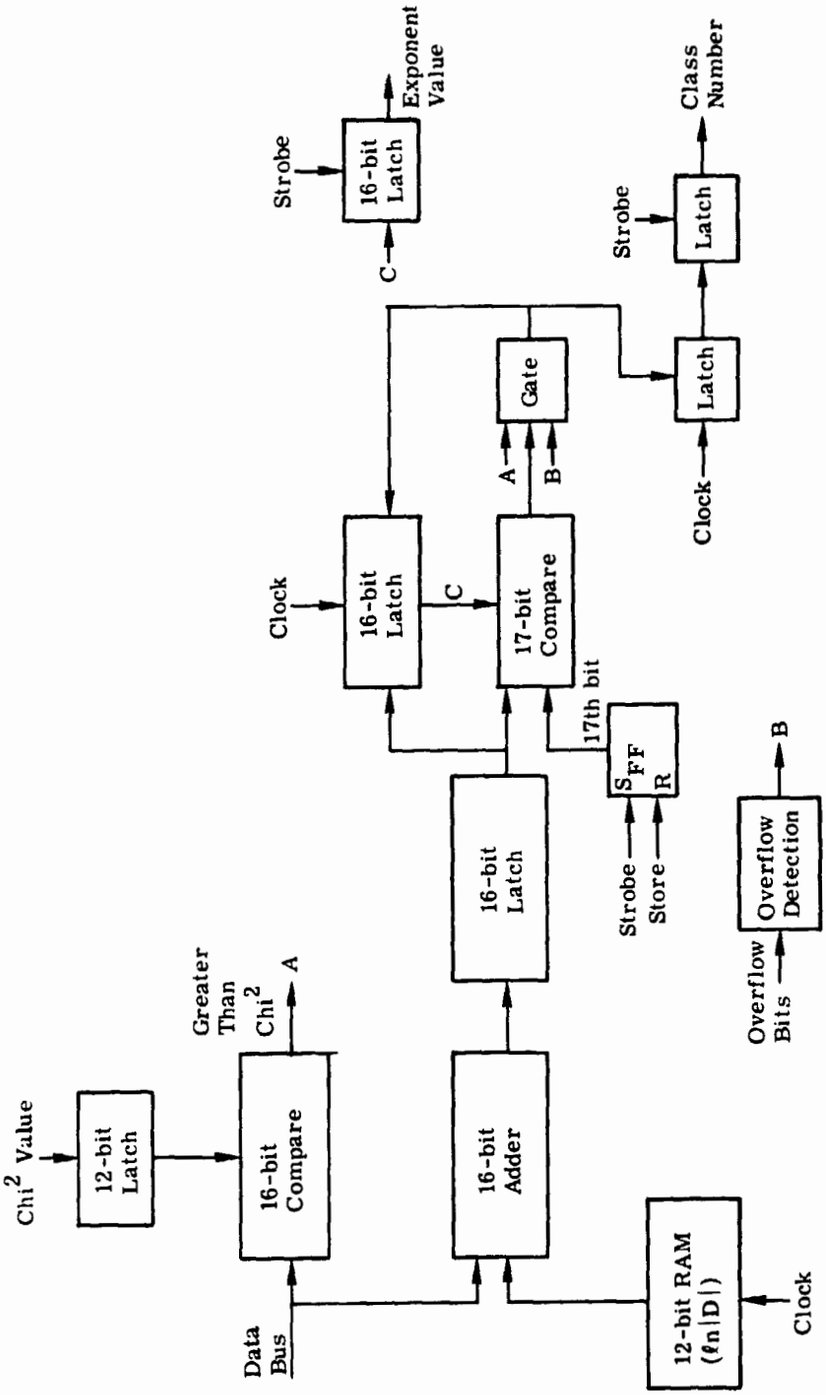


FIGURE 36. BLOCK DIAGRAM OF RECOGNITION CARD

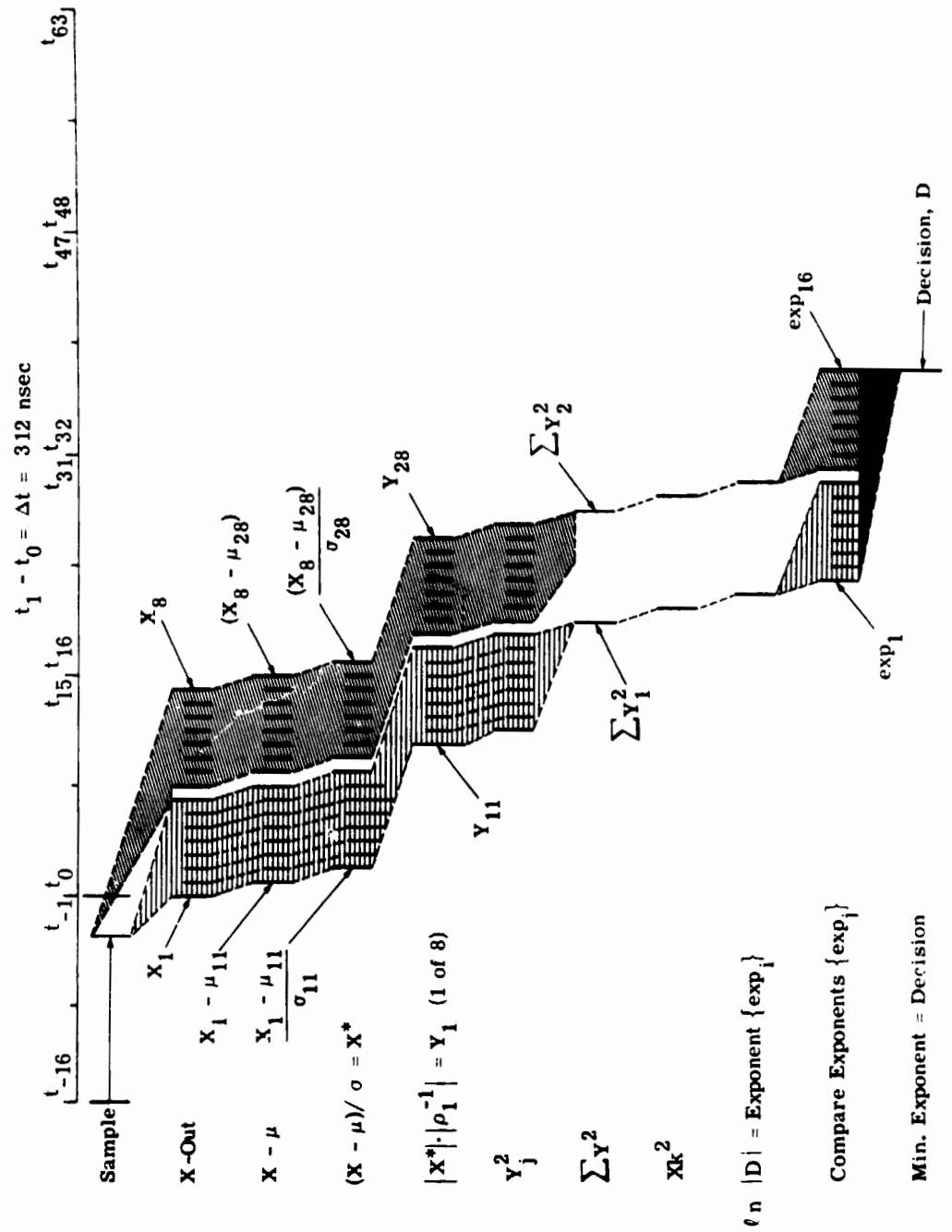


FIGURE 37. DIAGRAM OF CLASSIFIER TIMING

cycles, or approximately 5 microseconds. Data latched in the converter outputs is then supplied sequentially (X_1, X_2, . . . X_8) via a multiplexer and subtracter to a latch, and one clock cycle per element is allowed for this operation. At t = 0, μ_1 is subtracted from X_1; at t = 1, μ_2 is subtracted from X_2, etc. Also, at t = 1, $(X_1 - \mu_1)$ is multiplied by $(1/\sigma_1)$ yielding X_1^*. At t = 2, X_1^* is supplied to each of eight multiplier-summers which compute the products $(X_1^* \cdot P_{11})$, $(X_1^* \cdot P_{21})$, . . . $(X_1^* \cdot P_{81})$ and enter these into the summers. At t = 3, the multiplier-summers compute $(X_2^* \cdot P_{12})$, $(X_2^* \cdot P_{22})$ . . . $(X_2^* \cdot P_{82})$ and add these products to the previous results. Thus at t = 1Ø the summers contain the complete sums of products for all matrix operations. Each of the eight multiplier-summers may, as a result, be considered as a row operator since it accomplishes the sequence of multiplications and summations for a particular row.

This vector has the property as the result of the above transform that all its elements are uncorrelated, and therefore need only have its elements squared and summed to obtain the normalized quadratic form for the exponent. This is accomplished during cycles t = 1Ø to t = 18, allowing one cycle for each squaring operation and a final cycle for storage of the summation.

At this point, the exponent must be re-scaled and the natural logarithm of the determinant of the covariance matrix added to obtain the final exponent of the density function. This requires two cycles. The comparison of these exponents, now becoming available from the normalization circuitry, begins as each exponent appears. The procedure is to examine all exponents sequentially to choose the smallest, assuming that one is less than a threshold test value which is entered first, and to retain at all times the lesser value of two sequentially examined exponents. The number of the exponent retained specifies the class of the input vector. This is available to be displayed, printed on film, or supplied to the computer for logging or subsequent processing.

The cascades may also be used to process an increased number of distributions for a lesser number of channels. Thus for a 4-channel source, such as ERTS, the operations of the various arithmetic units may be time-shared to provide 16 class decisions instead of 8 class decisions for 8 channels.

6.6 CLOCK

This card has two main purposes—generating of timing pulses to control the 16 steps in the processing of a data vector, and controlling the loading of the RAMs during system setup. A block diagram is shown in Fig. 38.

The timing pulses, or clock pulses, are derived from a master oscillator. A countdown is program-selectable in six steps providing clock pulse CØ at rates from 3.2 MHz to 100 kHz. Other clock signals are labeled C1 to C4 and $\overline{C\emptyset}$ to $\overline{C4}$.

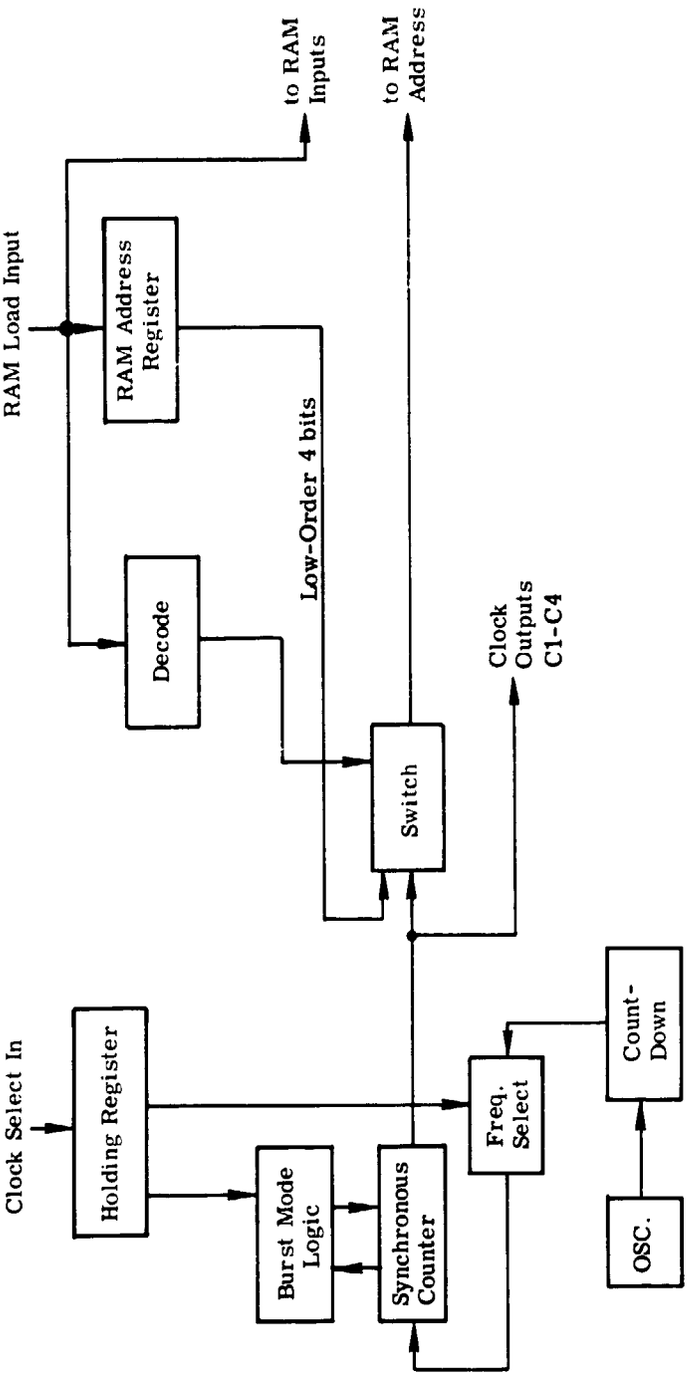


FIGURE 38. BLOCK DIAGRAM OF CLOCK CARD

The signals C1 to C4 are used for classifier control, whereas CØ is used for loading latches at the end of a clock period. The circuitry can operate in a burst mode with or without reset to zero. The burst mode is used when operating the system using data coming from the computer. The clock can be set to count up from zero to 127. For a data vector to be processed, sixteen steps are required. These are provided by setting the clock counter at 15, thereby allowing the sixteen steps 0 through 15. The clock is started by the computer when the computer has loaded the data vector into the second rank of the data register (latch "B" on the hybrid cards). The classifier processes this vector and stops until the computer has loaded a new data vector. If the computer loads a new vector first, however, the logic waits for the clock to finish before a new start signal is accepted by the clock.

Control of RAM loading during system setup is accomplished by means of logic on the clock card. Timing pulses and routing control are provided (see Fig. 38) which transmit the contents to any selected RAM by replacing the clock pulses C1 through C4 with the low-order four bits of the address word. The addressing and loading of RAM constants is accomplished by the following sequence:

(1) SEND WORD 1, DATA READY AND END CYCLE
(2) UPON RECEIPT OF CYCLE REQ. SEND WORD 2
(3) UPON RECEIPT OF CYCLE REQ. SEND NEW WORD 1
(4) REPEAT 2

The process stops when the word count register in the DR-11B increments to zero

The two-word transfers of RAM coefficients are given below:

	CODE	BAY	RAM	ADD
WORD 1	ØØØØ Ø	DDD	DDDD	DDDD
		MSB	MSB	MSB

	CODE	RAM CONSTANT
WORD 2	ØØØ1	DDDDDDDDDDDD
		MSB

NOTES: D is DATA - 0 or 1, as the case may be
BAY is binary-coded 0 to 7 (only 0-3 are used)
RAM is binary-coded:

0 = MEANS
1 = VARIANCE
2 = MTX 0
3 = MTX 1
4 = MTX 2
5 = MTX 3
6 = MTX 4
7 = MTX 5
8 = MTX 6
9 = MTX 7
} ALL BAYS

10 = k2 or CHI { k2 in Bay 2 / CHI in Bay 3
11 = LN (Bay 3 only)

ADD is binary-coded 0-15 for internal address of a RAM.

6.7 DIAGNOSTIC/OUTPUT SECTION

The diagnostic/output (D/O) card is the prime output port for classifier-generated signals. Inputs to the D/O may originate on the recognition card (5-bit material code and/or 16-bit exponent value) or at any access to the classifier diagnostic bus. Outputs from the D/O card are presented as a 16-bit word to the computer interface and as an 8-bit word to a display interface.

The D/O card has two input registers for holding the values of material and exponent from the recognition card and one register for holding the diagnostic value (see Fig. 39). Contents of these three registers are combined and switched to the two output ports under control of the classifier clock and a computer-selected command word.

The command word establishes the desired output formats and selects a time and place for sampling the diagnostic line. The command word is stored in a register and will command the same outputs to repeat twice each classifier cycle until a new command is sent.

Available output formats are as follows.

(1) At the computer interface:
  (a) the full 16-bit exponent value
or (b) eleven most significant exponent bits plus a 5-bit material code
or (c) 12-bit diagnostic plus 4 zeros
or (d) a composite word, presented once each cycle, consisting of two 5-bit material codes

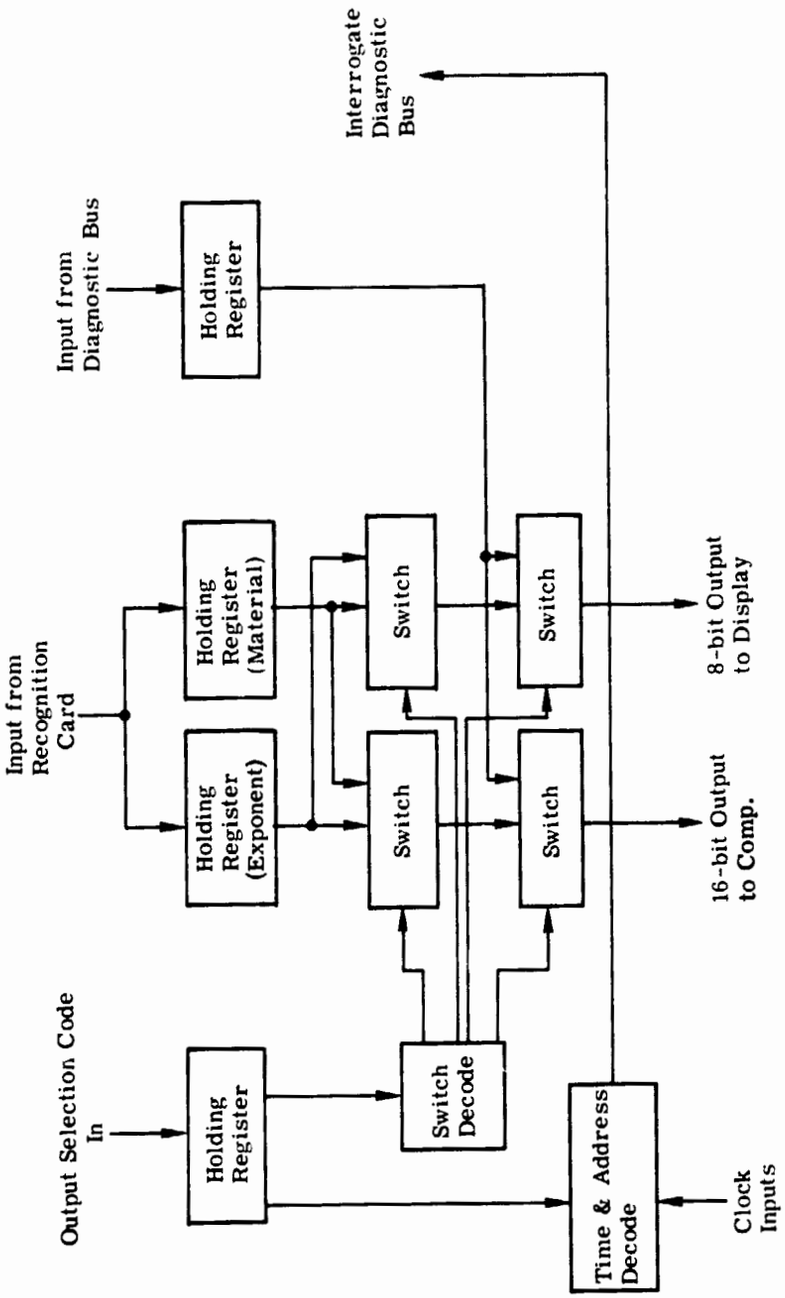


FIGURE 39. BLOCK DIAGRAM OF DIAGNOSTIC/OUTPUT CARD

(2) At the display interface:

(e) the eight MSBs of the exponent

or (f) the eight MSBs of the diagnostic

or (g) 5-bit material code

Any combination of the above computer and display outputs may be selected by the appropriate command except that (d) and (e) are not available simultaneously. The outputs are specified by the codes in Table 11.

6.8 INTER-CARD-FILE WIRING

The wiring between card files is accomplished in two main ways. The control file and the hybrid file use one technique, the classifier files another. A combination of the two is used in wiring the classifier files to the computer and other files.

The first method uses four standard card slots in the hybrid and control files as connectors. Standard cards (Douglas or DEC boards) are not wired with ICs but rather with ribbon cable wired to the printed circuit connector lands. The ribbon cables from the computer's DR-11B were also wired to these cards. This scheme is outlined in the block diagram of Fig. 40. It allows each of these two bays to be unplugged from the system. On the control file one connector cables to the control panel, a second connector cables to the DR-11C interface, a third to the hybrid card file, and a fourth to the classifier for clock control. On the hybrid file two cards cable to a back panel containing BNC connectors for analog input and output, a third connector cables to the computer DR-11B and a fourth one to the control file.

The classifier section of the system employs wire-wrap connectors, and some hand wire-wrapping was needed to get the data and control signals to each of the card files. The detailed wiring between the various files is shown in Fig. 41. In addition, the cabling between the entire classifier and the control file and second DR-11B in the computer was accomplished by using two DEC wiring blocks of the type used in the PDP-11/45 computer. The classifier was connected to these blocks with wire-wrap connections and standard DEC boards were plugged into the blocks. Classifier connections to the second DR-11B in the computer are detailed in Figs. 42 and 43.

6.9 INTERIM DISPLAY AND RECORD FACILITY

To provide a pictorial output showing system capability near the end of the first phase of the MIDAS development program, a subsystem consisting of several available instruments and parts was assembled.

The display equipment consisted of a large-screen, long-persistence X-4 cathode ray oscilloscope, a dual-sweep oscilloscope to provide deflection voltages for the display, and

TABLE 11. CODE SELECTION FOR THE DIAGNOSTIC/ OUTPUT CARD

CODE	TO COMP		TO DISPLAY
1ØØØ1	ØØØ MAT ④	ØØØ MAT	ØØØ MAT
Ø1ØØ1	ØØØ MAT ④	ØØØ MAT	DIAG ①
1Ø1Ø1	11-BIT EXP ②	MAT	ØØØ MAT
Ø11Ø1	11-BIT EXP ②	MAT	DIAG ①
111Ø1	11-BIT EXP ②	MAT	EXP ③
1ØØ11	DIAG	ØØØØ	ØØØ MAT
Ø1Ø11	DIAG	ØØØØ	DIAC ①
11Ø11	DIAG	ØØØØ	EXP ③
1Ø111	EXP		ØØØ MAT
Ø1111	EXP		DIAG ①
11111	EXP		EXP ③

NOTES:
① 8 most significant bits of 12
② 11 most significant bits of 16
③ 8 most significant bits of 16
④ This output appears every second process cycle; all other outputs appear every cycle.

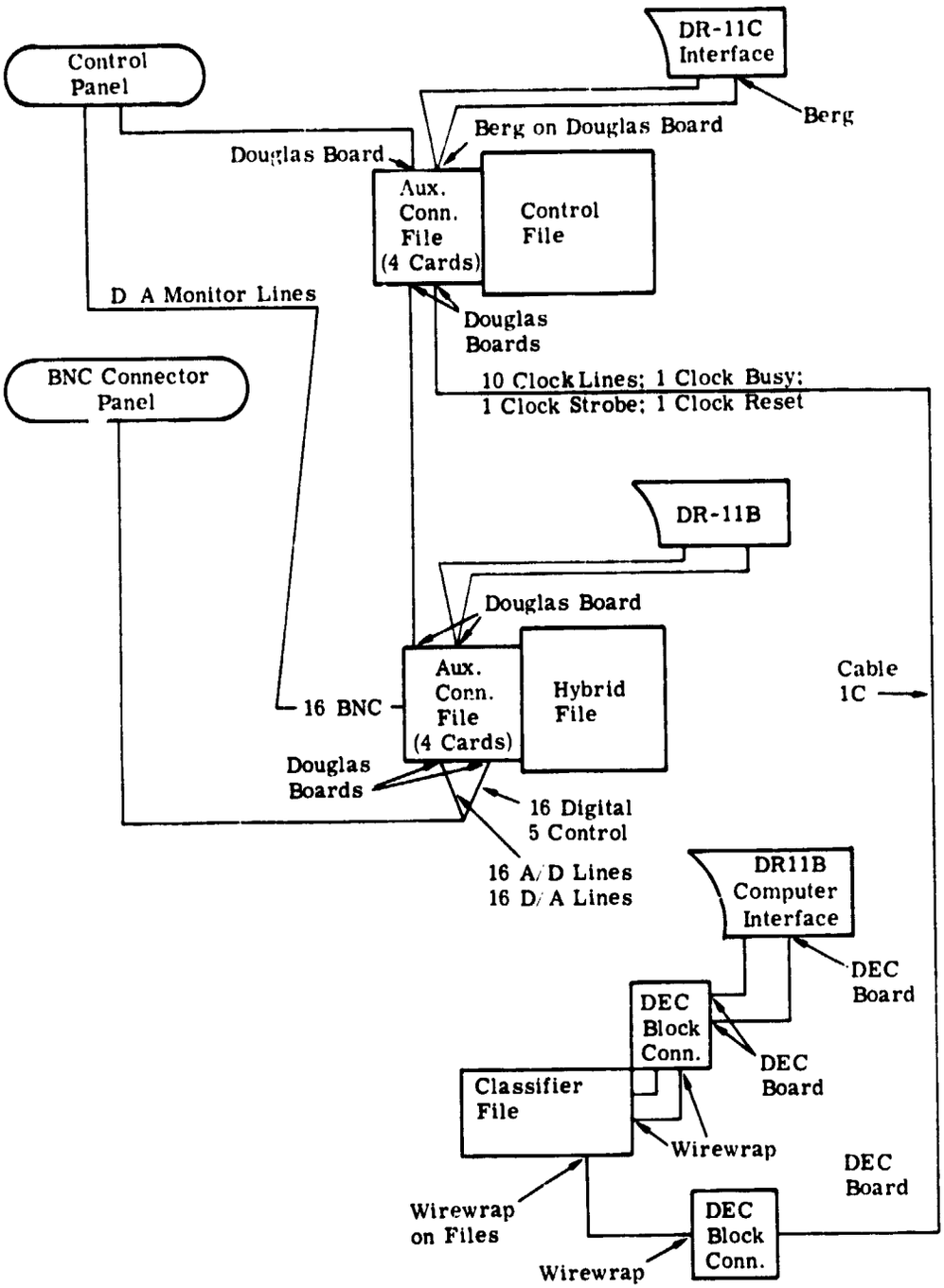
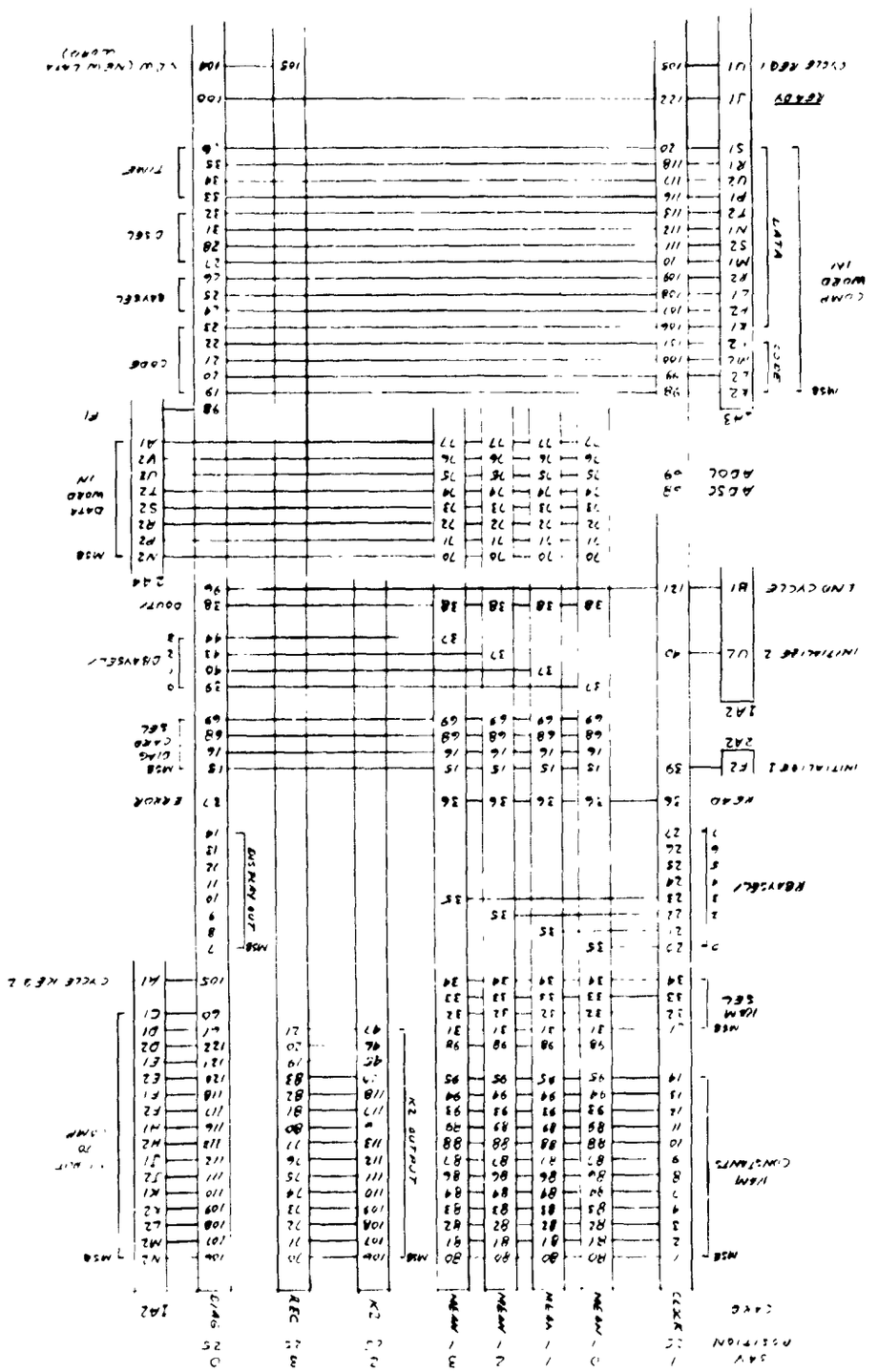
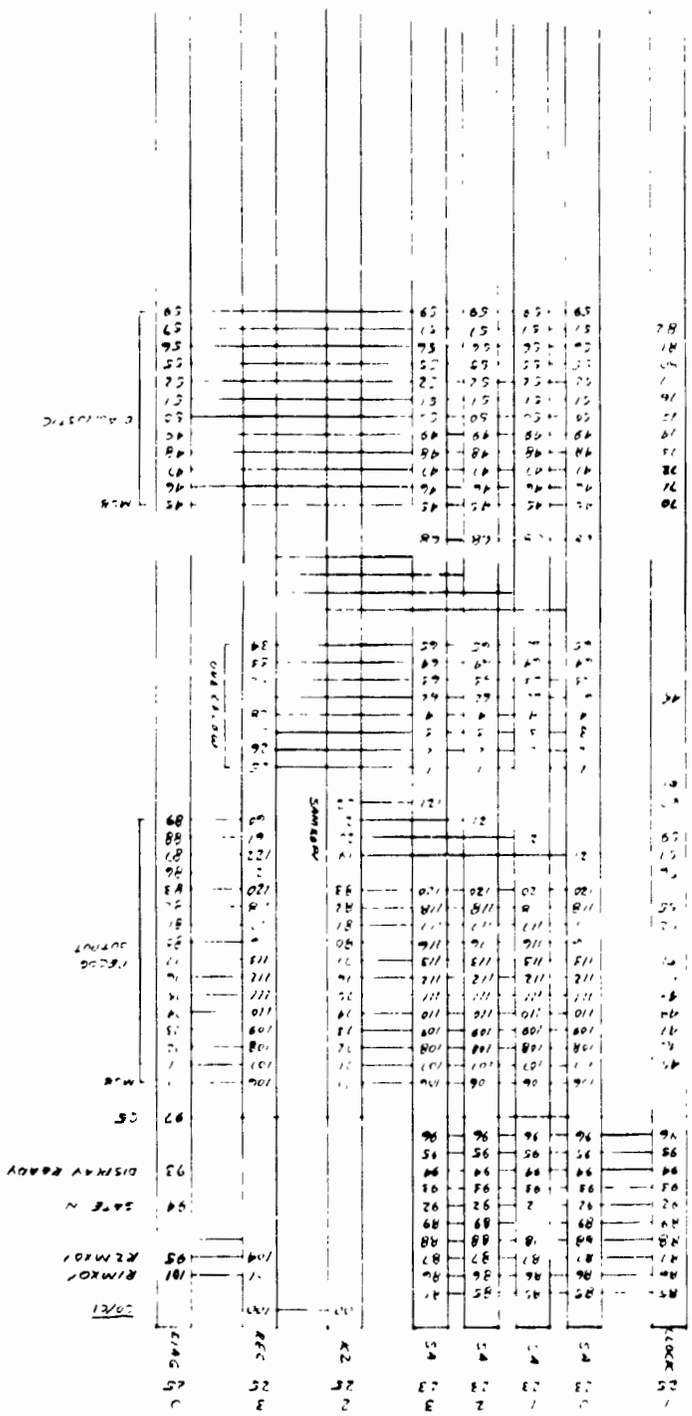


FIGURE 40. INTER-CARDFILE WIRING



(a) Sheet 1

FIGURE 41. INTER-BAY WIRING (Continued)



(b) Sheet 2

FIGURE 41. INTER-BAY WIRING (Concluded)

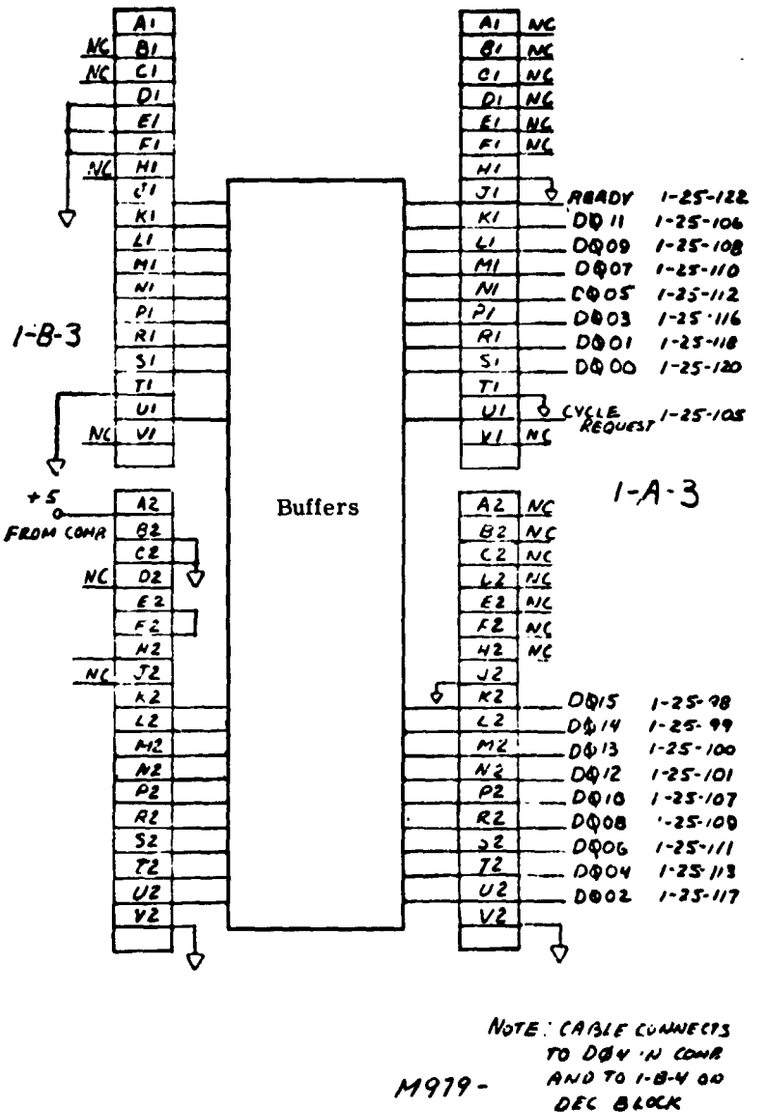


FIGURE 42. CABLE CONNECTIONS FOR DATA TRANSFER FROM COMPUTER TO CLASSIFIER

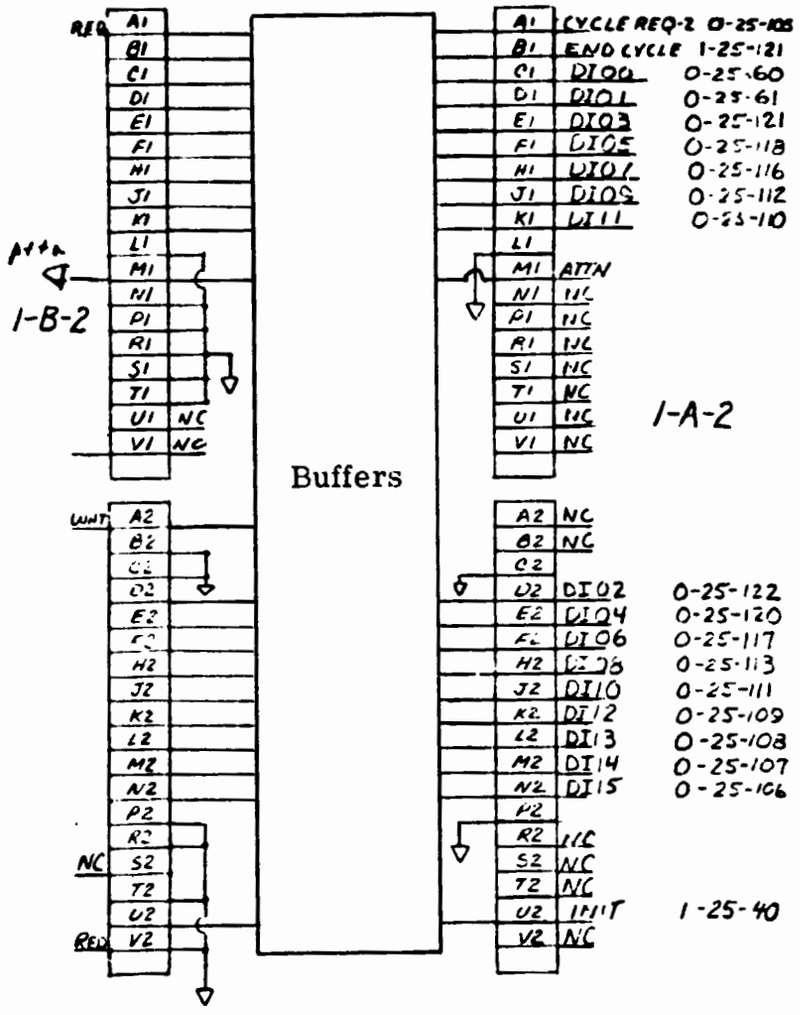


FIGURE 43. CABLE CONNECTIONS FOR DATA TRANSFER TO COMPUTER FROM CLASSIFIER

various modular amplifiers to control display quality. The incoming signals for the display were the decoded and selected classification codes from the MIDAS classifier.

Briefly, the display sweeps slowly down the large-screen CRO, drawing one horizontal line across the CRO for each line of the data source or record. During this sweep the CRO beam is modulated with the decoded classification code; this generates a picture in which those areas which are bright are those points (or pixels) classified as the selected code. The picture is held by the phosphor (P-7) for a time long enough to allow the image to be seen in a darkened room.

At the same time the display is generated, the class code is decoded and the output lines (13) are adjusted in level (0, +1V) to allow recording on a multi-track recorder for subsequent printing on an available black and white film-strip printer. These signals and a sync signal are recorded on an instrumentation tape recorder employing the FM intermediate band mode so that the recording is compatible with the printing system.

6.10 PHYSICAL DESCRIPTION

The Phase-I MIDAS Classifier is housed in one 6-ft rack assembly: a photograph of it is shown in Fig. 44. The power is single-phase, 120-volt AC with 20-ampere service. The system contains two power supplies—a 5-volt, 80-ampere supply and ±15-volt, 3-ampere supply. A matrix multiplier card has been measured as drawing about 1.4 amperes; therefore it would appear that all 52 wire-wrap cards draw about 70 amperes. A photograph of the four card files housing the wire-wrap cards is shown in Fig. 45. The physical location of the major components is diagrammed in Fig. 46.

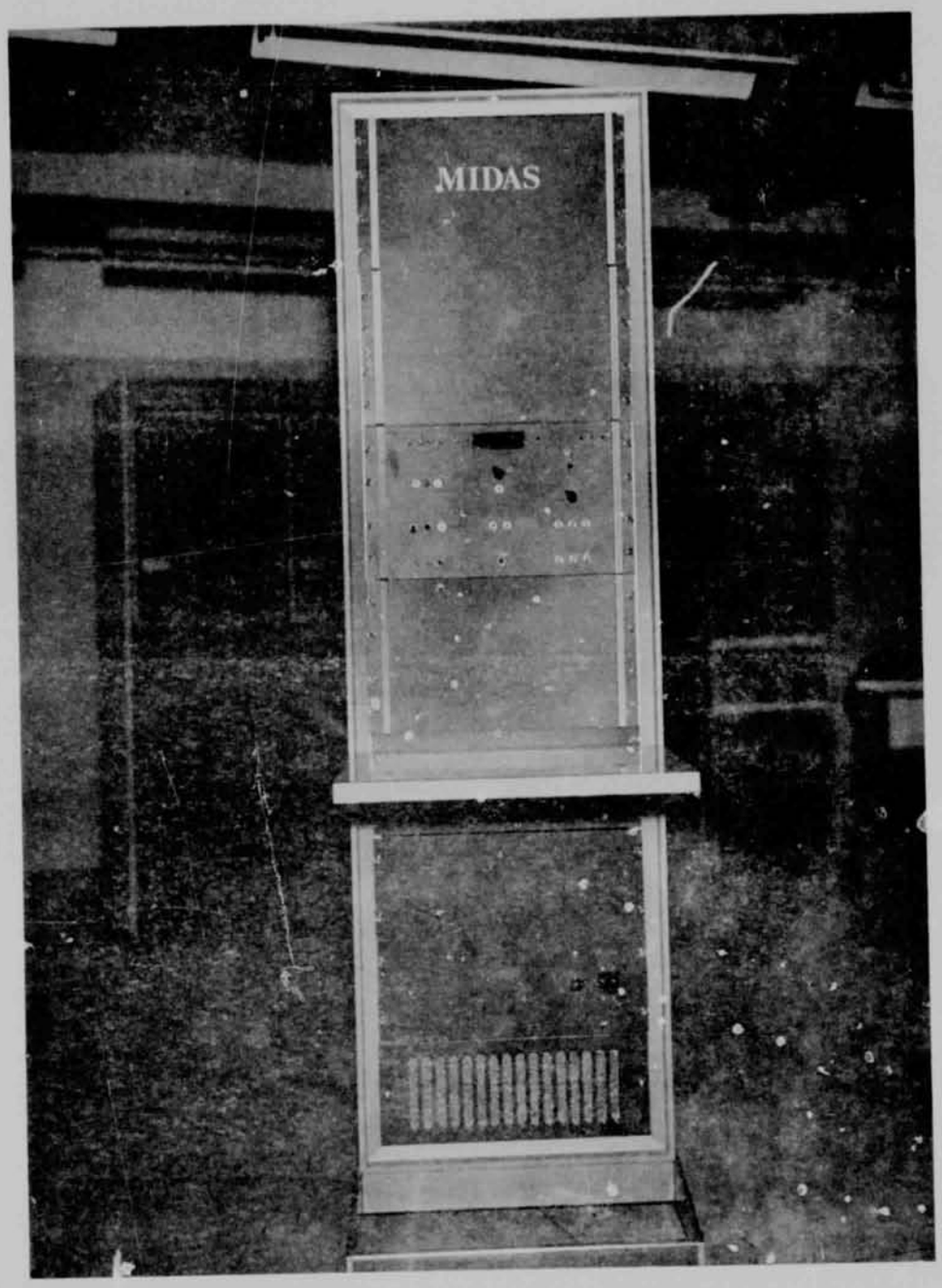


FIGURE 44. MIDAS CLASSIFIER

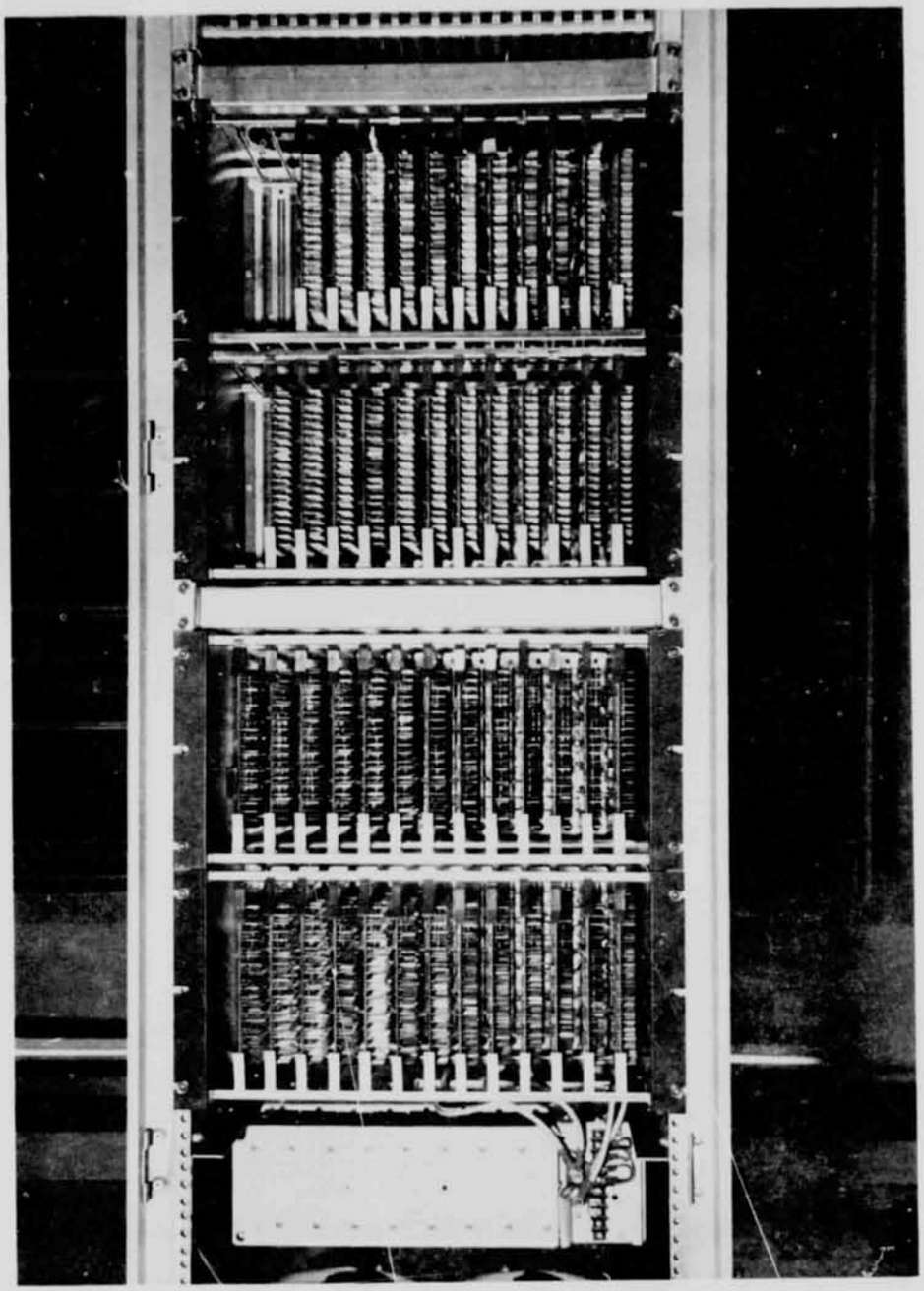


FIGURE 45. MIDAS CLASSIFIER WIREWRAP CARD FILES

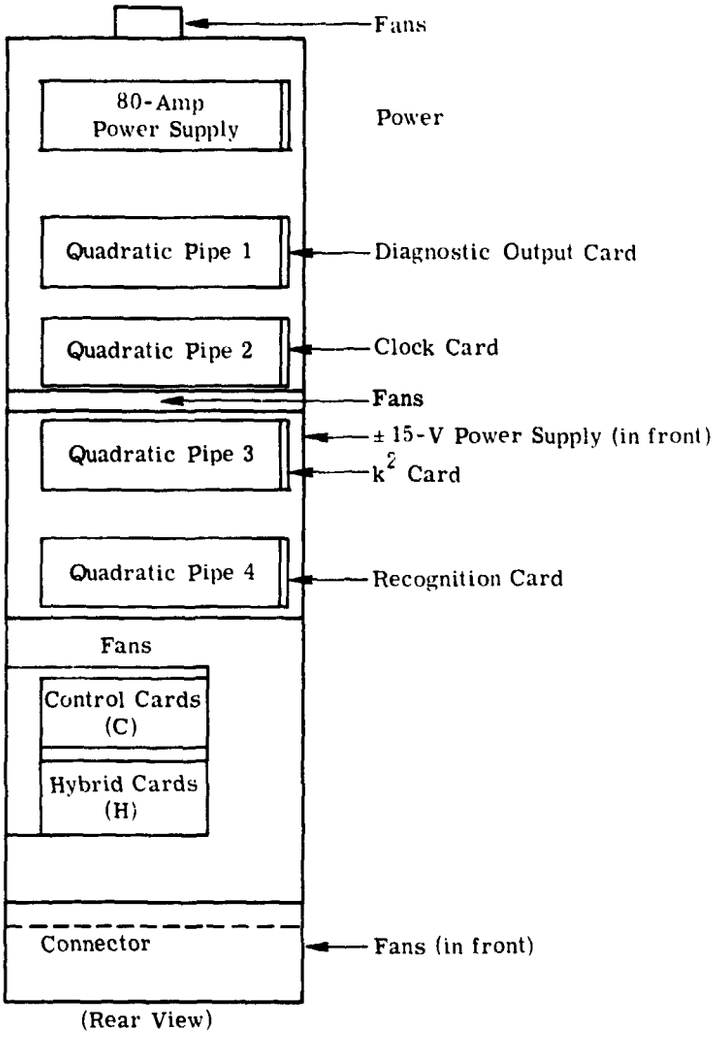


FIGURE 46. PHYSICAL LOCATION OF MAJOR MIDAS COMPONENTS

Appendix A
COMPARATIVE TEST OF MIDAS (PHASE I) PERFORMANCE

Our test of MIDAS on an ERTS data-set was performed and compared with results obtained in separate processing of the same data-set using the IBM 7094. At the time of the comparison, the 7094 processing task conducted under the ERTS program was not yet complete, but its objectives and interim results nonetheless serve as background for the processed imagery examples submitted with the present report. The current status of the ERTS Yellowstone project as recently reported by ERIM appears below in Section A.1. Following this, Section A.2 describes the MIDAS Phase I procedures employed in processing a quarter-section of the same ERTS Yellowstone data to obtain an imagery sample for comparison purposes.

A.1 YELLOWSTONE PARK DATA

A fundamental task of the National Park Service (N.P.S.) under Yellowstone's new Master Plan for park management is that of assigning geographic areas within the park their optimum role in meeting important multiple-use needs in accordance with established criteria for protection of the environment. Although specific management measures for accomplishing this goal are to be identified through "mission oriented" research, their implementation is certain to require an organized body of data on the type and distribution of natural resources throughout the park.

An objective of this project is to obtain, from ERTS-A MSS data, detailed vegetation and terrain feature maps for Yellowstone suitable for these many purposes. Completion of these maps will mark the first time a park-wide environmental inventory of Yellowstone has been available as a basis for management planning. They will provide not only an up-to-date record of conditions within the park, but will, in addition, serve as quantitative baseline data for monitoring natural ecosystem development and the ecological consequences of management decisions over extended periods of time.

The project is a cooperative one between ERIM, Dr. Harry Smedes of U.S.G.S. and Mr. Don DeSpain of N.P.S.

A.1.1 PRELIMINARY 5-CATEGORY COVER-TYPE RECOGNITION MAP

Computer-compatible tapes (CCTs) of bulk-processed MSS data for the 6 August 1972 (frame 1015-17404) overflight of the test site were received at ERIM in early December. The decision to process this data reflected both the fact that this orbit marked the first cloud-free observation of the park, and, from an ecological standpoint, occurred during a period of distinctive seasonal differences in the appearance of various types of vegetation cover.

ERIM's initial task consisted of converting the data on the ERTS tapes to a format compatible with in-house software programs, and subsequently generating a digital graymap of MSS 5 (red band) for the entire park. A copy of this map was sent to Dr. Smedes in mid-December, 1972, for his use in transcribing cover-type training set locations onto a common base. A list of these training sets, along with their respective environmental parameters and graymap coordinates, was transmitted to ERIM by Dr. Smedes in January, 1973. ERIM then began an extensive program centered around the extraction and analysis of multispectral statistical signatures for the training sets, which comprised our effort for February. Based on the environmental complexity of cover types observed during this period, a decision was made to initially produce a basic, 5-category map representing the endpoints of important spectral and environmental cover type continua. We reasoned that it was important initially to determine the level of recognition accuracy we could attain using spectral categories which were easily distinguishable.

The 5-category map was produced in early March and reported in a paper, "Terrain Classification Maps of Yellowstone National Park," given at the GSFC during the March 5-9 ERTS Symposium [6]. An analysis of recognition accuracy for the five categories resulted in the following hierarchical ranking:

(1) water
(2) coniferous forest
(3) light rock
(4) grasslands
(5) wet meadows

Estimates of recognition success based on percent area correctly classified indicate an overall map accuracy greater than 82 percent.

A.1.2. FINAL COVER-TYPE RECOGNITION MAP

In an attempt to resolve the causes underlying poorer recognition associated with the grassland and lowland meadow types of the 5-category map, to refine these and the other categories, and to enlarge upon the total number of training sets, Mr. Ralph Root, a Colorado State graduate working for Dr. Smedes, spent a week at ERIM in April. During his visit, high-altitude RB-57 color and color infrared photography of the park was examined on a VARISCAN. The optical enlarging capability of the VARISCAN permitted us to closely match scales between a scene viewed on the film and the graymaps. Not only was identifying an appropriate training set on the film much easier, but transcription of this location onto the graymaps was far more accurate.

As a result of this analysis, several of the original training sets, especially those in the grasslands and lowland meadows categories,were found to be less homogeneous than previously believed. This appears to be the major explanation for the poorer recognition of these types in the preliminary map.

On this premise, we proceeded with another program of signature extraction and analysis for refined versions of original training sets and some representing several new categories. This work occupied our efforts for May. The results of our findings at the end of this period, based solely on training set signature separation, indicated that we now had 18 potential categories for inclusion in the final map. These categories were:

1. Lodgepole pine (young-dense type)
2. Lodgepole pine (mature-more open type)
3. Douglas fir
4. Spruce fir
5. 30-45% conifer cover over rock rubble
6. 30-45% conifer cover over grass
7. 15-30% conifer cover over rock rubble
8. 15-30% conifer cover over grass
9. Wetland shrubs
10. Upland grass
11. Lowland grass
12. Grass-brush mixture
13. Brush
14. Dark rock
15. Light rock
16. Thermal deposits
17. Alpine meadows
18. Water

Recognition maps of these 18 cover types were generated for restricted test areas within the park and sent to Dr. Smedes for his evaluation of their accuracy.

Based on Dr. Smedes' and Mr. Root's findings, several steps were taken preparatory to more extensive mapping. First, the category alpine meadow was dropped because of excessive recognition commission errors as light rock. Similarly, a disappointing number of recognition commission errors between categories of morphologically similar vegetation types prompted us to combine several similar categories. Included in this situation were the merging of the two Lodgepole pine types, Douglas fir and Spruce fir categories back into a single coniferous forest type, and merging of the upland and lowland grass types into a single grasslands category.

In the case of the forest species, it is possible that many of the recognition commission errors resulted from varying signal contributions of path radiance in MSS 4 and 5. Comparison of signal means in these channels for the various forest species (Table A.1) reveals they are rarely more than 1 to 2 data values apart. By our estimations, path radiance at critical altitudes within the park was at least this great. This was determined by comparison of MSS 4 signal levels for deep-pure water from two lakes at elevations of 6000-7000 ft. A signal increase of 1.11 data values attributable to path radiance was observed for the water signature at the lower elevation.

Recognition processing of the entire park was completed in June, using the following 13 categories:

1. Coniferous forest
2. 30-45% conifer cover over rock rubble
3. 30-45% conifer cover over grass
4. 15-30% conifer cover over rock rubble
5. 15-30% conifer cover over grass
6. Wetland shrubs
7. Grasslands
8. Grass-brush mixture
9. Brush
10. Dark rock
11. Light rock
12. Thermal deposits
13. Water

The classification output was then mapped and sent to Dr. Smedes and Mr. DeSpain.

Dr. Smedes, Mr. Root, and Mr. DeSpain are currently concentrating on evaluating the recognition accuracy of this latest product. Additionally, Mr. DeSpain is beginning an assessment of its impact on National Park Service management-decision processes.

A.1.3 CONCLUSIONS

This project is expected to contribute much new ecological knowledge about the Yellowstone landscape. In particular, the revegetation patterns of old wildfire burns are being analyzed to determine successional trends occurring in the park. This information should have a significant impact on the Park Service's fire protection and wildlife management programs.

On a more general level, we are working closely with Mr. DeSpain to determine if there are more desirable, or additional formats in which to present data abstracted from the final map in terms of making it a more effective part of or basis for the "mission oriented" research involving formulation of Yellowstone's future management policies.

One format we are currently experimenting with is the production of thematic maps. Small examples of these have been made illustrating important ecological cover type continua, e.g., forest to bare rock, and forest to grassland. This type of information on vegetation type boundaries and transitions has considerable value in the management of "edge" species, i.e., plants or animals that depend entirely, or in part, on the ecological character of a transition zone between major cover types for their existence.

We are also interested in investigating a method of automatically computing from the map data the acreage of the various cover types and amount of edge within specified corridors where a land use shift has been proposed. This capability would provide the Park Service with a means of obtaining a timely, specific, and detailed data base for assessing the environmental impact of proposed management alternatives.

A.1.4 RECOMMENDATIONS

Many of the most important cover types have signature means in MSS 4 and 5 separated by only 2 percent of the sensor's dynamic range, i.e., 2-3 data values in 127 (Tables A.1 and A.2). Excepting thermal deposits, which appear essentially white, the brightest natural objects in the scene rarely exceed the data value of 50. This represents only 40 percent of the dynamic range and indicates that over one-half the scanner's sensitivity is not being used.

Since the danger of clipping important data appears minimal, we believe a pass over the park exercising the high gain option is highly desirable. We anticipate that this data will be extremely useful in assessing the potential separability of important forest species. Using this option the data will be expanded 3-fold as compared to MSS 4 and 5. In effect, the data takes on more significance because now there will be a greater number of quanta within the relative separation of the signatures.

It would also appear that preprocessing the data to remove the effects of path radiance as experienced in MSS 4 and 5 is essential to consistent recognition of forest types that occur in interspersion over a wide range of base altitudes.

A.2 MIDAS PHASE I PROCESSING PROCEDURES

The following software procedures were implemented in carrying out the Phase I demonstration of MIDAS software capability on the ERTS Yellowstone National Park data-set recorded on 6 August 1972.

(1) Program DISPLAY was employed to select from this data set one quarter-frame which would be most representative for demonstration purposes. Quarter-frame 3 was determined to be the most suitable area in view of distinct surface features which would be clearly identifiable on the limited display facilities of the Phase I MIDAS.

(2) Since the Phase I signature-extraction facility was not operational as of 1 November 1973, suitable signatures for use with the Yellowstone data-set were obtained on the IBM 7094 data processing facility, employing the identical algorithms specified in this report for the PDP-11/45 software system. These signatures were subsequently used to obtain a multiple-category classification of the ERTS Yellowstone data-set comparable to that described in the preceding section. In all, 15 signatures were formulated by combining signatures of homogeneous areas over several regions of the entire data-set. Program SIGCVT performed the transformations necessary and created a disk file of the proper format which provided an input file for subsequent processing.

TABLE A.1. TRAINING-SET SIGNATURES OF CONIFEROUS FOREST COVER TYPES FOR ERTS-A YELLOWSTONE PARK MSS DATA, 6 AUGUST 1972

Training Set	Signature Mean and Standard Deviation			
	MSS 4*	MSS 5*	MSS 6†	MSS 7†
Lodgepole Pine (young)	17.23 ± 1.01	12.79 ± 0.98	22.07 ± 1.51	11.76 ± 0.94
Lodgepole Pine (mature)	16.18 ± 0.88	11.01 ± 0.80	21.71 ± 1.37	11.46 ± 0.72
Douglas Fir	16.67 ± 1.00	10.89 ± 1.05	22.26 ± 1.80	11.82 ± 1.61
Spruce - Fir	15.58 ± 0.92	10.09 ± 1.12	20.01 ± 2.22	10.64 ± 1.42

TABLE A.2. TRAINING-SET SIGNATURES OF MAJOR TERRAIN COVER TYPES FOR ERTS-A YELLOWSTONE PARK MSS DATA, 6 AUGUST 1972

Training Set	Signature Mean and Standard Deviation			
	MSS 4*	MSS 5*	MSS 6†	MSS 7†
Coniferous Forest	17.93 ± 1.96	13.45 ± 2.67	23.54 ± 3.28	12.61 ± 1.98
30-45% Conifer cover over rock rubble	19.15 ± 2.03	15.81 ± 2.74	27.11 ± 3.95	14.87 ± 2.30
30-45% Conifer cover over grass	18.72 ± 1.33	13.50 ± 1.46	31.90 ± 6.26	18.36 ± 4.02
15-30% Conifer cover over rock rubble	25.68 ± 2.97	25.07 ± 4.09	37.07 ± 4.31	19.52 ± 2.15
15-30% Conifer cover over grass	21.24 ± 2.18	16.93 ± 2.99	40.74 ± 6.05	23.74 ± 3.88
Wetland Shrubs	19.33 ± 1.05	13.38 ± 1.31	43.87 ± 3.81	26.35 ± 2.98
Grasslands	22.64 ± 1.55	18.52 ± 1.55	47.62 ± 5.11	29.17 ± 3.83
Grass/Brush	24.17 ± 1.29	21.72 ± 2.30	37.97 ± 3.52	21.52 ± 2.59
Brush	26.32 ± 1.50	25.28 ± 1.88	31.30 ± 2.78	15.64 ± 1.79
Dark Rock	27.72 ± 4.42	26.41 ± 5.32	27.24 ± 3.80	12.51 ± 2.05
Light Rock	37.62 ± 8.29	40.40 ± 11.14	43.57 ± 10.83	19.78 ± 5.54
Thermal Deposits	65.48 ± 9.69	72.52 ± 11.23	71.00 ± 9.31	31.58 ± 3.60
Water	13.65 ± 0.79	5.85 ± 0.87	2.99 ± 0.70	0.27 ± 0.45

*Current potential range of 127 data values.

†Current potential range of 64 data values.

(3) Operating on the data file created by SIGCVT, program SCALER performed the mathematical manipulations necessary to obtain constants in a form suitable for loading into the MIDAS classifier coefficient RAMs. SCALER uses the mean vector and covariance matrix to generate this set of constants —which play a key role in implementing the maximum-likelihood rule. (See Section 3.2.5 on the algorithms used.)

(4) Once these constants were generated via SCALER, program RAMLD allowed the user to select a subset of signatures for actual loading into classifier RAMs. This subset selected, it was necessary to normalize certain constants within the subset, viz., the normalization constant for the decomposed inverse correlation matrix, and the additive constant specifying the natural logarithm of the determinant of the covariance matrix. After these constants had been calculated, all of the coefficients derived from the signatures were loaded in several large DMA transfers via the DR-11B general interface.

Next, the user specified an upper-limit for the chi-squared value derived for each resolution element. This chi-squared test, comprising the last overflow test performed by the hardware, acts as upper bound on the maximum-likelihood criterion.

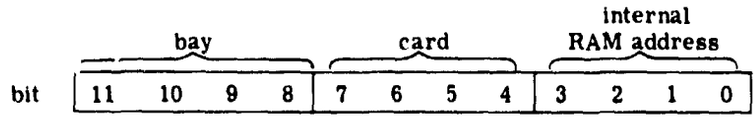
(5) With the signature coefficients loaded into the classifier, program CLAS1 set up the parameters for the classification procedure to be performed next in the logical operational sequence. The user now specified, via console keyboard, the region to be classified; designated the I/O media to be used; and specified the output format from the classifier. This information was stored in a temporary data-file on disk, and a program CLAS2D was loaded into core and executed. To control the classification procedure, program CLAS2D used the temporary disk file created by CLAS1. After supervising the initiation of the classification process and controlling the subsequent flow of data both in and out, CLAS2D returned control to CLAS1 for input of new classification parameters by the user.

(6) Upon completing the classification process, we again called upon program DISPLAY to generate grayscale maps exhibiting the classification results. Here, because the line printer is only intermittently available and slower than the CRT, we employed the latter as the output device.

Appendix B
RAM LOADING REQUIREMENTS

Loading of coefficients into the RAMs is somewhat involved because of the machine's pipeline nature. Also, it appears that two programs may have to be written, one for 4-channel and one for 8-channel operation. At this time, however, it appears that only the 4-channel case need be considered since that is what we use for the Phase I demonstration. Basically, there are three sets of coefficients to be loaded: (1) a $-\mu$ vector, (2) a $1/\sigma$ vector, and (3) a U-Square matrix of eigenvectors.

To load the RAMs requires a 12-bit address located as follows in the computer word:



The bay and card bits are relatively straightforward except that the internal RAM address is more complex. Up to 16 materials, designated by the letters A,B,C,D,E,F,H,J,K,L,M,N,P,R, S,and T, can be stored in the processor. (These letters correspond to standard wiring letters—including DEC's.) To load the means ($-\mu$ vectors) for letters A,B,C, and D, the address codes are

$-\vec{\mu}_A$ = ØØØØ|ØØØØ|ØØØØ first element
ØØØØ|ØØØØ|ØØØ1
ØØØØ|ØØØØ|ØØ1Ø
ØØØØ|ØØØØ|ØØ11 fourth element

$-\vec{\mu}_B$ = ØØØ1|ØØØØ|ØØØØ
ØØØ1|ØØØØ|ØØØ1
ØØØ1|ØØØØ|ØØ1Ø
ØØØ1|ØØØØ|ØØ11

$-\vec{\mu}_C$ = ØØ1Ø|ØØØØ|ØØØØ
ØØ1Ø|ØØØØ|ØØØ1
ØØ1Ø|ØØØØ|ØØ1Ø
ØØ1Ø|ØØØØ|ØØ11

$-\vec{\mu}_D$ = ØØ11|ØØØØ|ØØØØ
ØØ11|ØØØØ|ØØØ1
ØØ11|ØØØØ|ØØ1Ø
ØØ11|ØØØØ|ØØ11

For the next set of four $-\vec{\mu}$ vectors, we have, first

$-\vec{\mu}_E$ = ØØØØ|ØØØØ|Ø1ØØ to Ø111

Then similarly, for the other three materials

$-\vec{\mu}_F$ = ØØØ1|ØØØØ|Ø1ØØ to Ø111

$-\vec{\mu}_H$ = ØØ1Ø|ØØØØ|Ø1ØØ to Ø111

$-\vec{\mu}_J$ = ØØ1Ø|ØØØØ|Ø1ØØ to Ø111

In the next set of four vectors, we use high-address numbers in the internal RAM address

$-\vec{\mu}_K$ = ØØØØ|ØØØØ|1ØØØ to 1Ø11

down to

$-\vec{\mu}_N$ = ØØ11|ØØØØ|1ØØØ to 1Ø11

The last set of materials is

$-\vec{\mu}_P$ = ØØØØ|ØØØØ|11ØØ to 1111

down to

$-\vec{\mu}_T$ = ØØ11|ØØØØ|11ØØ to 1111

The next vector to be loaded is the $1/\sigma$ vector. These are loaded in almost the same way as the mean vector with this exception: We add 1 to the least significant bit in the internal RAM address to compensate for the pipeline nature of the processor. Thus,

$\overrightarrow{1/\sigma}_A$ = ØØØØ|ØØØ1|ØØØ1 first element
ØØØØ|ØØØ1|ØØ1Ø
ØØØØ|ØØØ1|ØØ11
ØØØØ|ØØØ1|Ø1ØØ fourth element

$\overrightarrow{1/\sigma}_B$ = ØØØ1|ØØØ1|ØØØ1
ØØØ1|ØØØ1|ØØ1Ø
ØØØ1|ØØØ1|ØØ11
ØØØ1|ØØØ1|Ø1ØØ

and so on until we come to the last set of four materials P through T. Here

$\overrightarrow{1/\sigma}_P$ = ØØØØ|ØØØ1|11Ø1
ØØØØ|ØØØ1|111Ø
ØØØØ|ØØØ1|1111
ØØØØ|ØØØ1|ØØØØ

and

$\overrightarrow{1/\sigma}_T$ = ØØ11|ØØØ1|11Ø1 first element
ØØ11|ØØØ1|111Ø
ØØ11|ØØØ1|1111
ØØ11|ØØØ1|ØØØØ fourth element

Next to be loaded are the U-square matrix coefficients. For these, there are eight cards, each designed to hold a row of coefficients for an 8×8 matrix.

To store four 4×4 matrices on these cards, only half the amount of coefficient storage is needed. The classifier is organized so that the coefficients of four 4×4 matrices can be stored in the two 8×8 matrices as follows:

Matrix Mult. Card No. in Bay Ø															
Ø 1 2 3	Material Ø				E (zeroes)				Ø				P		
4 5 6 7	A				Ø				K				Ø		
	2	3	4	5	6	7	8	9	10	11	12	13	14	15	0	1

Internal Address Number

Bay Ø is selected by the ØØØØ code in bits (11,8). Matrix multiplier Ø is selected by the ØØ1Ø code in bits (7,4) and each succeeding matrix multiplier is selected in turn by advancing the code by 1 up to number 7, code 1ØØ1. Note the permutation of the internal RAM address by 2. The other bays employ the following code for loading material coefficients:

bay 1	Ø	F	Ø	R
code ØØØ1	B	Ø	L	Ø
bay 2	Ø	H	Ø	S
code ØØ1Ø	C	Ø	M	Ø
bay 3	Ø	J	Ø	T
code ØØ11	D	Ø	N	Ø

In the configuration using eight 8 × 8 matrices, the material coefficients would be stored as

bay Ø	A	E
bay 1	B	F
bay 2	C	H
bay 3	D	J

Two other sets of coefficients are to be stored: (1) a set of normalizing coefficients k^2, and (2) the log of the determinants of the covariance matrices. The k^2 set is addressed by

ØØ1Ø	1Ø1Ø	XXXX
bay	card	material number

The materials are assigned internal RAM addresses as follows:

A 1Ø11
B 11ØØ
C 11Ø1
D 111Ø
E 1111
F ØØØØ
H ØØØ1
J ØØ1Ø

K ØØ11
L Ø1ØØ
M Ø1Ø1
N Ø11Ø
P Ø111
R 1ØØØ
S 1ØØ1
T 1Ø1Ø

The log of the determinants is addressed by

ØØ11	1Ø11	XXXX
bay	card	material number

Here, the material numbers are different from those above in that a 1 has to be added, thus

A 11ØØ

down to

F ØØØ1

and

T 1Ø11

One other constant for the Chi^2 test has to be loaded at address

ØØ11	1Ø1Ø	(material number not needed)
bay	card	

Appendix C
SPECIFICATION OF OUTPUT FILE FROM SIGNATURE PROGRAM

C.1 SIG AND SIGCVT OUTPUT

The following provides a description of the output file generated by programs SIG and SIGCVT.

(1) The file must be a contiguous DOS-file.

(2) Structure:

BLOCK 1

Word 1	:	No. of signatures in the file
Word 2	:	No. of channels (NCHAN)
Word 3	:	No. of words used per signature

BLOCK 2 etc.

Word 1-3	:	6-character ID (ASCII)
Successive Words	:	Mean vector of NCHAN elements (REAL*8, i.e., 4 words/element)
Successive Words	:	Covariance matrix (lower half), A, of (NCHAN+1)* N/2 elements (REAL*8) such that the first element passed is A_{11}, the next A_{21}, then A_{22}, A_{31}, A_{32}, A_{33}, etc.

NOTE: Start each new signature on a new block. For example, if we have a 12-channel signature, then we have the following numbers of words used:

3	Signature ID
48	Mean vector (12 elements × 4 words/element)
312	Cov. matrix (78 elements × 4 words/element)
363	Total number of words (two blocks/signature)

Thus, the first signature begins on block 2, the second on block 4, the third on block 6, etc.

C.2 NOTES ON DATA PRESENTATION TO *WIREWRAP

(1) An overview:

Passing data to *WIREWRAP involves three things—what, where, and connections.

(a) What—What kind of IC or module does the user now want to put on the board?

The user specifies this by calling in a macro description of the IC he has in mind. He does this by simply giving the name of the macro. *WIREWRAP then looks in the user's macro list until it finds the right one, and then is ready to accept more information.

In short, macros are used to pass information statistics to *WIREWRAP and to implement the conversion from a net-name into a pin number.

Here, as examples, are two forms of macro description for a hex inverter module:

```
I:  MACRO
    INVERTER MODULE=SN7404 POWER=44 COST=25 TAG=14
    ASSIGN NETS=&INPUT PINS=(1,3,5,9,11,13) LOAD=1
    ASSIGN NETS=&OUTPUT PINS=(2,4,6,8,10,12) DRIVE=10
    ASSIGN NETS=&POWER PINS=(7,14)
    MEND

II: MACRO
    INV TAG=14
    ASGN NETS=&P PINS=(1,2,3,4,5,6,7,8,9,10,11,12,13,14)
    MEND
```

The words "MACRO" and "MEND" must accompany each macro description.

1. In the first case above, the name of the module is "INVERTER"; in the second it is "INV."
2. The manufacturer's model number of the first is "SN7404," and the second is unspecified.
3. The first module consumes 44mW; for the second this is unspecified (default 0).
4. The first module costs 25 cents; the second is unspecified (default 0).
5. Both are known to have 14 pins (present routines require the "TAG=" to be present).
6. For the first module it is known that the unit load of the inputs is 1, that its output fanout is 10, and that its power pins (ground, +5 volts) are 7 and 14 respectively; in the second case this information is unspecified.

The order of appearance of COST, POWER, TAG, MODULE is not significant. Both macros I and II are valid, but I is more logically apparent to the user, and he is less likely to make mistakes with it. ("ASGN" is "ASSIGN" in short form.)

In I, the names "INPUT," "OUTPUT," "POWER" are known as logical net names and are distinguished by the "&" which precedes them. They can be 8 or less characters long.

(b) Where —Where does the user want this IC or module placed on the board?

*WIREWRAP automatically starts placing modules on the board in locations Ø,1,2,3,4, . . . unless otherwise instructed.

The "$POSITION nn" command instructs *WIREWRAP to place the next module at location nn, and unless told not to, the next one after that at position (nn+1), then (nn+2), etc.

If the user wished to place hex inverters, for example, in locations 0,1,3,5, 48, 49*, the following would accomplish that:

```
INV
INV
$POSITION 3
INV
$POSITION 5
INV
$POSITION 48
INV
INV
```

If necessary, these positions can be thought of as pin locators, not IC seats. For example, if a 3-pin crystal clock is being used, it could be inserted into several different empty module positions to allow for its widely spaced pins. The output would then think that there had been several strange ICs inserted, instead of one unique device.

In short, unless $POSITION'ed, *WIREWRAP will get its new position by adding 1 to its old position.

(c) Connections —How does the user connect the IC pins?

Since each logical net-name is associated with a group of pin numbers, all that has to be done is to associate the names of the pins with the logical net-names. A pin's name is called its net-name.

Any number of pins which receive the same net-name from the data become wired together. The net-name might be thought of as a signal name —any pins labeled with that name must have that name's signal value. Of course there can be only one name per pin.

For example, take the "INVERTER" again. Since "INPUT" is associated with the input pins in the order 1,3,5,9,11,13, we can assign any or all of those pins a name by using "INPUT." Suppose now that we want the name "XX" to be

*NOTE: For the user to know, for example, exactly where location 48 is, he must have a board layout diagram. In the Generalized Augat URG-1, layout locations 0-49 are for 14- or 16-pin ICs only, 50-67 are for 24-pin ICs only, 68-75 are for I/O, 76-81 are for Ground, and 82-87 for +5 volts (VCC).

on pin 1 and "ZERO" on pin 3. Then, INPUT=XX,ZERO would accomplish this. What about pins 5,9,11,13? Since they are not called or specified, they are not wired.

Suppose further that we only wanted to wire pins 9 and 13 on another SN7404 inverter. Then INPUT= ,,,XX,,ZERO would be the data. Notice that in the first case, trailing commas were not necessary. *WIREWRAP detects the end of the list by seeing a blank at the end. In the second case we need three commas as null-element spacers, then the name "XX" followed by its comma, followed by a null-element comma for pin 11, then the name "ZERO" for pin 13.

There can be no blanks internal to an assignment statement. Therefore the following would be illegal: INPUT=AA, BB, CC, , However, there can be many blanks between different assignment statements. For example, suppose we wanted pin 1 to be AA, pin 2 to be BA, pin 7 to be G114, and pin 14 to be V54, then

POWER=G114,V54 OUTPUT=BA INPUT=AA

The order is insignificant as far as assignment statements go, but the order of the net-names in the assignment statements is one-to-one with the pin numbers in the marco for that given logical net name. Logical net-names may be duplicated in different macros, e.g., we used "&PWR" in all of them, but net-names can be presented in the data only on pins which are to be wired together.

Notice that in the marco description the logical net-name has an ampersand (&) preceding it, but in the data the ampersand disappears. What happens if some data runs over one card? It is then necessary to use a dash (—) in column 80 to indicate continuation to the next card. (Recall that there can be no blanks inside of an assignment statement.) Thus if the data spills over, column 1 on the next card is considered column 80 (the dash disappears), and column 2 is column 81, etc.

Example:

. . ."INPUT=ZERO,". . .becomes:

Column

		73	74	75	76	77	78	79	80	1	2	3	4	
. . .	card 1	I	N	P	U	T	=	Z	-					. . .
	Card 2									E	R	O	,	

And there is nothing wrong with this occurrence:

. . ."INPUT=ZERO" becomes:

	Column									
. . .	71	72	73	74	75	76	77	78	79	80
	I	N	P	U	T	=	Z	E	R	O

if "ZERO" is the last net name in that marco (IC) to be assigned.

(2) A specific example:

Two inverters might be actually wired as in Fig. C.1. Call the first inverter BA, and the second one BB. We want BA in position 17 and BB at 18.

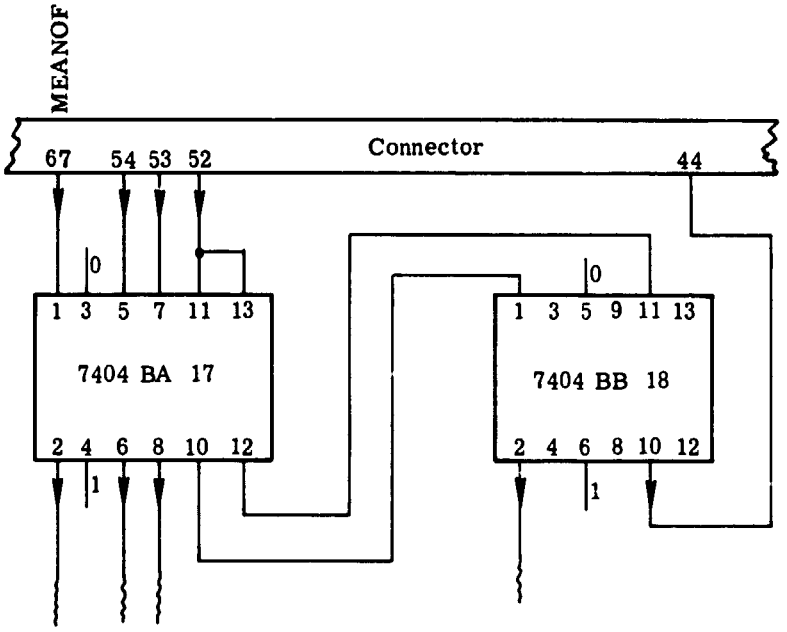


FIGURE C.1. SAMPLE INVERTER WIRING

```
$POSITION 17
INVERTER INPUT=MEANOF,GBA,P54,P53,P52,P52 OUTPUT=BA2,BA4, BA6, BA8,
  BA10,BA12 POWER=GBA,VBA
INVERTER OUTPUT=BB2,,BB6,,BB10 POWER=GBB,VBB INPUT=BA10,,GBB,,BA12
```

Notice that the marco name comes first, then one of the assignment statements. Also, it is not necessary to mention all of the logical net names if they are not all used. If we wished somehow to hand-wire the power pins, we could just not mention "POWER=" and pins 7 and 14 would then be left free.

Also notice in the example above that many names are singular, such as "VBB." If these were never mentioned before or again in the data they would not be wired, and would appear in the output in a "Singles List." Frequently this catches mis-punched cards.

A good way to generate meaningful net-names is to give each chip a letter code for its type, then a letter for its number ("BB" is the 2nd inverter, "MF" is the 6th multiplier, "LA" is the first latch, etc), then name pins by their source output pins. For example, the 10th pin on BA is BA1Ø because it is an output pin, then pin 1 on BB is BA1Ø because it is connected to the 10th pin of BA.

Also, "GBA" and "VBA" are the ground and voltage pins of BA respectively, and pin 54 on the connector becomes "P54." Pin 67 on the connector is an important control signal, so it gets a mnemonic name "MEANOF."

(3) Running *WIREWRAP on MTS at U of M Computing Center

(a) For your $SIGNON card, the TIME=12 and PAGES=25 CARDS=XXXX*

(b) Next you need a $RUN *WIREWRAP+Subs+WØ64:SQGRD
where "Subs" is the name of your compiled object subroutine package, and WØ64:SQGRD is the system assembly program that punches the cards.

(c) Next comes the $SET command where you can choose your options (see the MTS manual writeup). Some of the more useful ones are (default underlined):

PUNCH=ON/<u>OFF</u>	controls production of Gardner-Denver deck
ABORT=ON/<u>OFF</u>	if ON, prevents punching a deck in case of error
DEBUG=ON/<u>OFF</u>	if ON, prevents costly extras, like ordering of the pins, etc. (inhibits punching)
LEADLEN=n/<u>Ø</u>	n=11 (1/10 inch per wrap of wire)
WRAPCOST=n/<u>Ø</u>	n=5 (¢ per wrap)
WIRECOST=n/<u>Ø</u>	n=1 (¢ per foot of wire)
XREF=<u>ON</u>/OFF	cross-reference is not so useful after data is debugged.

$SET uses no commas: <u>ex</u>: $SET BAYS=7 BAYCOST=2665Ø PUNCH=ON

(d) $TITLE, followed by a card containing the title you want to use

(e) $DATA indicating the start of data

(f) Then the Macro description. (*WIREWRAP won't know what "INVERTER" is until it has encountered it in a macro list.)†

*If you intend to produce cards you <u>must</u> specify CARDS=XXXX, where XXXX is just an upper limit on the number of cards you expect, and is typically less than 1000.

†Frequently it is convenient to have your bunch of macros in a file (say, "MACROS") and have "$CONTINUE WITH MACROS RETURN" just after your $DATA card. Curiously, the Macro list is searched last first, so the most used macros should go at the end.

(g) Wiring data, including $POSITION cards

(h) $END signaling the end of data

(i) $ENDFILE

(j) If an asterisk (*) occurs in column 1, the card becomes a comment card.

Suppose that your subroutine package is in a file named "SUBS," already compiled, and that your macro list is in a file named "MACROS."

Also suppose that you plan to hand-wire your power connections and your I/O, and that you want an inverter at position 36, with one line on pins 2 to 3 called "ONLY," and the title of the device is "SMALL TEST." The following would be a complete deck which would punch one Gardner-Denver machine card:

```
$SIGNON CCNO   T=12 P=25 CARDS=1ØØØ  'TEST DECK'
Password
$RUN *WIREWRAP+SUBS+WØ64:SQGRD
$SET BAYS=1 BAYCOST=88ØØ LEADLEN=11 WIRECOST=1
$SET WRAPCOST=5 ABORT=ON PUNCH=ON
$TITLE
      SMALL TEST
$DATA
$CONTINUE WITH MACROS RETURN
$POSITION 36
*   HERE IS YOUR ONLY DATA LINE:
INVERTER INPUT=,ONLY OUTPUT=ONLY
$END
$ENDFILE
```

(4) Output Interpretation

(a) Source Listing—the input deck is listed (including the Macro file if one is $CONTINUE'ed). When a macro is called, that device is assigned an element number (E1, E2, E3, . . .) by *WIREWRAP, then it is assigned a location (either by adding one to its last position or by being $POSITION'ed). The error count given at the end applies only to *WIREWRAP syntactical errors, not mispunches or wiring errors.

(b) Net-Name Cross Reference—each net-name is listed alphabetically along with the input deck line number in which it is used (once per use).

(c) Net-Name Dictionary—each net-name is listed alphabetically followed by

1 an "O" if any two output pins with that net-name are collector OR'ed (hard-wire OR)

2 a "*" if any output pin with that net-name is overloaded

3 the sum of the normalized loads of the input pins with that net-name

4 the normalized drive (fanout) of the lowest-drive output pin with that net-name

5 the list of pins with that net-name. Output pins are underlined. The pin names used are the output of WW4 and are also found on the interpreted Gardner-Denver machine deck. In general, the first 5 characters in the name give its layout position and the last 3 characters give the physical board location.

Since *WIREWRAP wraps pins serially (daisy-chain) it determines in which order to wrap a certain group of pins (those logically related by possessing the same net-name) so that the minimum amount of wire is used. If the DEBUG option is OFF, the Net-Name Dictionary listing of pins will be in the same order as will actually be wrapped on the board.

The load and drive information is obtained from the marco description A pin is considered to be an output pin if in the macro it has the "DRIVE=" parameter. The default for LOAD and DRIVE is zero. In the Net-Name Dictionary a drive of 0 is assumed to be like an output or power pin so it won't call a non-zero load an overload.

(d) Module Dictionary —Totals the number and cost of all modules (macros) used.

(e) Itemized Cost Analysis —Totals the cost of the device.

(f) Documentation —This is the output of WW6, labeled "***SUMMARY OF PIN CONNECTIONS," which documents which net-names were actually assigned where.

(g) Statistics —Lists things like total power consumption, total feet of wire, etc.

(h) Singles —Lists the net-names which were only assigned to one pin. Presence of a Singles List indicates a wire data error —usually a mispunching but sometimes a conflict in naming-conventions. The list is deleted if no singles are discovered.

REFERENCES

1. Erickson, J., Automatic Extraction of Information from Multispectral Scanner Data, 12th International Congress of the International Society for Photogrammetry, Ottawa, August 1972.

2. Disk Operating System Monitor Programmer's Handbook, DEC-11-OMONA-A-D. Digital Equipment Corporation, Maynard, MA. 1972.

3. Lanczos, C., Applied Analysis, Prentice Hall, Inc., Englewood Cliffs, N. J., 1961, pp. 52-81.

4. Marshall, R. E. and F. J. Kriegler, An Operational Multispectral Survey, Proc., Seventh International Symposium on Remote Sensing of Environment, Report No. 10259-1-X, Willow Run Laboratories of the Institute of Science and Technology, The University of Michigan, Ann Arbor, 1971, Vol. III, pp. 2169-91.

5. Crane, R. B., T. Crimmins, and J. F. Reyer, Feature Extraction of Multispectral Data, Proc., Machine Processing of Remotely Sensed Data, Purdue University, West Lafayette, Ind., 16-18 October 1973, pp. 4B-6 thru 4B-15.

6. Thomson, F. J. and N. Roller, Terrain Classification Map of Yellowstone National Park, Proc., Symposium on Significant Results Obtained from ERTS Resources Technology Satellite-1, Goddard Space Flight Center, New Carrollton, March 1973, Vol. I, pp. 1091-96.