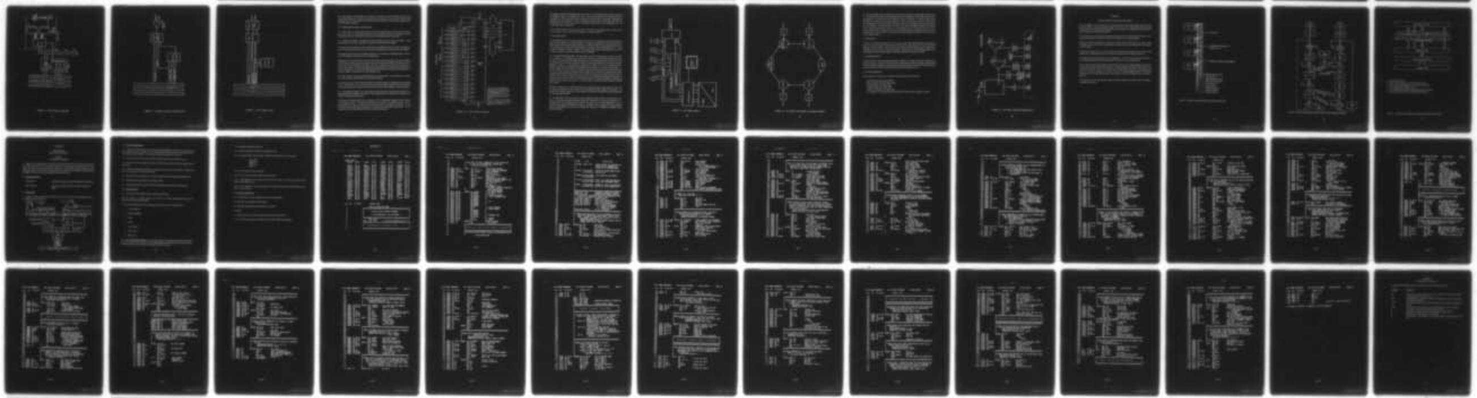
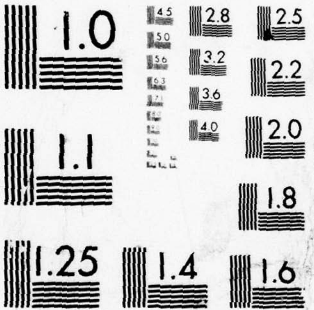
AD-A065 222 ELECTRONICS ENGINEERING GROUP (1842ND) SCOTT AFB IL F/G 17/2
A MICROPROCESSOR-BASED CONTROL EVENT SCANNER.(U)
JAN 79 J R HAMMER, M D VERSTEGEN
UNCLASSIFIED 1842 EEG/EETET-TR-79-6 NL

1 OF 1
AD
A065222



END
DATE
FILMED
4--79
DDC



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

1842 EEG/[redacted]EETET TR 79-6

TECHNICAL REPORT

A MICROPROCESSOR-BASED CONTROL EVENT SCANNER

AFCS SYSTEMS TECHNICAL APPLICATIONS FACILITY
1842 ELECTRONICS ENGINEERING GROUP (AFCS)
SCOTT AIR FORCE BASE, ILLINOIS 62225

409 646

31 JANUARY 1979

1842 ELECTRONICS ENGINEERING GROUP

MISSION

The 1842 Electronics Engineering Group (EEG) has the mission to provide communications-electronics-meteorological (CEM) systems engineering and consultive engineering support for AFCS. In this respect, 1842 EEG responsibilities include: Developing engineering and installation standards for use in planning, programming, procuring, engineering, installing and testing CEM systems, facilities and equipment; performing systems engineering of CEM requirements that must operate as a system or in a system environment; operating a specialized Digital Network System Facility to analyze and evaluate new digital technology for application to the Defense Communications System (DCS) and other special purpose systems; operating a facility to prototype systems and equipment configurations to check out and validate engineering-installation standards and new installation techniques; providing consultive CEM engineering assistance to HQ AFCS, AFCS Areas, MAJCOMS, DOD and other government agencies.

APPROVAL PAGE

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

Z. J. Bara

ZYGMUND J. BARA
Chief, Telecommunications Systems
Engineering Division

William R. Brown

WILLIAM R. BROWN, MAJ, USAF
Chief, Engineering Services Branch

Calvin R. Taylor

CALVIN R. TAYLOR
Technical Area Manager

James R. Hammer

JAMES R. HAMMER, Capt, USAF
Electronics Engineer

Michael D. Verstegen

MICHAEL D. VERSTEGEN, 2LT, USAF
Electronics Engineer

SECURITY CLASSIFICATION OF THIS PAGE (*When Data Entered*)

REPORT DOCUMENTATION PAGE	READ INSTRUCTIONS BEFORE COMPLETING FORM
1. REPORT NUMBER TR 79-6 /	2. GOVT ACCESSION NO. N/A
3. RECIPIENT'S CATALOG NUMBER N/A	
4. TITLE (*and Subtitle*) 6 A Microprocessor-Based Control Event Scanner •	5. TYPE OF REPORT & PERIOD COVERED 9 Final rept.,
	6. PERFORMING ORG. REPORT NUMBER N/A
7. AUTHOR(*s*) 10 ~~Capt~~ James R. Hammer ~~Lt~~ Michael D. Verstegen	8. CONTRACT OR GRANT NUMBER(*s*) N/A
9. PERFORMING ORGANIZATION NAME AND ADDRESS 1842EEG/EETET / Scott AFB, IL 62225	10. PROGRAM ELEMENT, PROJECT, TASK AREA & WORK UNIT NUMBERS N/A
11. CONTROLLING OFFICE NAME AND ADDRESS 1842EEG/EEISG Scott AFB, IL 62225	12. REPORT DATE 11 31 Jan[redacted] 79
	13. NUMBER OF PAGES 12 58 p.
14. MONITORING AGENCY NAME & ADDRESS(*if different from Controlling Office*)	15. SECURITY CLASS. (*of this report*) Unclassified
	15a. DECLASSIFICATION/DOWNGRADING SCHEDULE N/A

16. DISTRIBUTION STATEMENT (*of this Report*)

Approved for public release. Distribution unlimited.

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

14 1842 EEG/EETET-TR-79-6

18. SUPPLEMENTARY NOTES

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

Digital Communications	Performance Monitoring
Microprocessor	IEEE Standard 488-1975
Microcomputer	
Performance Assessment	

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

The Control Event Scanner (CES) is a microprocessor-based device which will control and sense relay contract closures via the IEEE Standard 488-1975 Interface Bus, hereafter known as the General Purpose Interface Bus (GPIB). Through the use of a microcomputer, other functions such as pulse counting, analog to digital conversion, digital timing, etc., are easily implemented. With a fully implemented unit, the capability exists to realize a low-cost, fully programmable automated test fixture suitable for use in a distributed

409646

TDM

- contd

performance assessment network (DPAN). With the use of the GPIB and an internal microcomputer, on-line reconfiguration is possible to implement a variety of dedicated performance assessment instruments. Performance assessment and a fault isolation example are discussed.

ABSTRACT

The Control Event Scanner (CES) is a microprocessor-based device which will control and sense relay contact closures via the IEEE Standard 488-1975 Interface Bus, hereafter known as the General Purpose Interface Bus (GPIB). Through the use of a microcomputer, other functions such as pulse counting, analog to digital conversion, digital timing, etc., are easily implemented. With a fully implemented unit, the capability exists to realize a low-cost, fully programmable automated test fixture suitable for use in a distributed performance assessment network (DPAN). With the use of the GPIB and an internal microcomputer, on-line reconfiguration is possible to implement a variety of dedicated performance assessment instruments. Performance assessment and a fault isolation example are discussed.

-A-

GLOSSARY

ASCII - American Standard Code of Information Interchange

CES - Control Event Scanner

DPAN - Distributed Performance Assessment Network

Defined in 1842 Technical Report TR 77-4 (Automated Data Acquisition Applied to Monitoring of Communications Systems) 1 Oct 76

EPROM - Erasable Programmable Read Only Memory. A semiconductor memory device which is electrically non-volatile.

Event Sensing - Process of sensing relay states indicating equipment and/or parameter status.

Fault Isolation - Process of localizing system malfunctions based on parameter samples at strategic points in the system.

GPIA - General Purpose Interface Adapter. A system of five integrated circuits that implement the protocol of the IEEE Standard 488-1975 General Purpose Interface Bus between a microprocessor and a compatible device.

GPIB - General Purpose Interface Bus. The electrical and logical interface between compatible equipment specified by IEEE Standard 488-1975. See Appendix A.

Histogram - A graphic representation of a frequency or relative frequency distribution consisting of vertical rectangles whose widths correspond to a definite range of frequencies and whose heights correspond to the number of frequencies occuring in the range. (Webster's Second Collegiate Dictionary)

LED - Light Emitting Diode

Nodal Controller - A processor located at a "node" in a distributed performance assessment network (see Figure 3-2). The nodal controller has the mass storage and data analysis capability necessary for a large performance assessment network.

RAM - Random Access Memory. Semiconductor read write memory that is electrically volatile.

Remote Processor - A processor located at a communications site which communicates with the nodal controller, the Control/Event Scanner, and other remote processors in the network (see Figure 3-2). The remote processor may accomplish data analysis in addition to collecting parameter information from the CES.

SPDT - Single Pole, Double Throw.

uP - Microprocessor

TABLE OF CONTENTS

Paragraph No.		Page No.
1.0	BACKGROUND	1
1.1	GENERAL	1
1.2	STATUS	1
2.0	THE CONTROL EVENT SCANNER	1
2.1	DESIGN PHILOSOPHY	1
2.2	DESIGN CRITERIA	1
2.3	THEORY OF OPERATION-HARDWARE	2
2.3.1	Alarm/Event Sensing Subystem	2
2.3.2	Relay Control Subsystem	2
2.3.3	Analog to Digital Converter Card	2
2.3.4	Pulse Counter Card	2
2.3.5	GPIB Interface	7
2.4	SYSTEM SOFTWARE-THEORY OF OPERATION	7
2.4.1	Alarm Scanner	7
2.4.2	Histogram Processing	7
2.4.3	Relay Control	7
2.4.4	GPIA Control	7
2.4.5	Communications Executive	7
3.0	CES APPLICATIONS	8
3.1	STAND-ALONE CES	8
3.2	CES WITH REMOTE PROCESSOR	8
3.3	CES AND DPAN	8
3.4	FAULT ISOLATION	10
4.0	CONCLUSIONS	11
5.0	RECOMMENDATIONS	11
6.0	ACKNOWLEDGEMENTS	11

APPENDICES

A.	IEEE STANDARD 488-1975 (EXCERPT)	A-1 thru A-4
B.	AIR FORCE ACADEMY PROJECT ORDER	B-1 Thru B-3
C.	CES SYSTEM SOFTWARE	C-1 thru C-24
D	CES PROGRAMMING CODES	D-1

LIST OF ILLUSTRATIONS

FIGURE		Page
2-1	Alarm Input Subsystem	3
2-2	Relay Control Subsystem	4
2-3	Analog to Digital Converter Card	5
2-4	Pulse Counter Card	6
2-5	GPIA Block Diagram	8
3-1	CES "Smart Sensor"	10
3-2	Distributed Performance Assessment Network	11
3-3	CES Fault-Isolation Configuration	13

1.0 BACKGROUND.

1.1 GENERAL. The CES was a joint Air Force Academy/1842 EEG project which began during a cadet summer (1976) research program, sponsored by the 1842 EEG. The CES was developed in two stages. Stage I was the interface between CES functions; i.e., relay control/sensing, A/D conversion, and a microcomputer. Stage II was the development of the IEEE Standard Bus 488-1975 interface adapter. The CES is the first joint AFA/1842 EEG project to use the GPIA (see Glossary) to implement the GPIB and will serve as the pilot for any further joint projects using the GPIB. The project served as course material for both students and faculty at the Air Force Academy.

1.2 STATUS. The CES development hardware and software are complete. Some functions, such as analog/digital conversion and pulse counting, were not implemented since it would be easier to do so at a later date. The CES has the capability to scan alarms (relay contact closures) and control selected relays. In addition, it can communicate with an on-site or remote processor to exchange data, commands, and configuration programs via the GPIB. Through the use of a "local" command, the CES may be operated as a stand-alone device with all functions accessed and displayed through local terminal control. The CES easily meets or exceeds all original specifications generated in the project order given to the Air Force Academy (see Appendix 2). Additionally, the CES provides enhanced data acquisition and network communication techniques.

2.0 THE CONTROL EVENT SCANNER.

2.1 DESIGN PHILOSOPHY. One problem area foreseen in a distributed performance assessment network, within a digital communication system, is the overloading of the remote processor with many mundane data acquisition tasks as well as handling data reduction and communication protocol to the nodal controller. To overcome this problem and relieve the remote processor of having to acquire data at a relatively fast acquisition rate, it is desired to have a dedicated monitoring unit to perform specific functions such as monitoring Receive Signal Level, Eye Pattern Degradation, Bit Error Rate, and Frame Alarm State Sensing. Then, on a periodic basis, the transfer of this information from the dedicated monitor to the remote processor may be accomplished. The remote processor can then store the measured parameters for that time period, transmit them upon command from the nodal controller, or further reduce the data for statistical analysis. The use of standard interfaces and communications protocol is assumed in order to reduce development costs and utilize off-the-shelf equipment as much as possible. In addition, with the advent of the microprocessor, the capability to do moderately fast, low-cost data acquisition is readily apparent.

2.2 DESIGN CRITERIA. The project order, sent to the Air Force Academy in the Fall of 1975, outlined all major design concepts and is included in this report as Appendix 2. Air Force Academy project personnel suggested certain hardware and software changes to improve the overall effectiveness of the scanner. For example, instead of activating switches from the front panel to select a particular relay, a terminal device is used under software control for complete control of the scanner in the local mode. Furthermore, by using software control techniques, the front panel controls may be locked-out and control may then be transferred to the remote processor. These types of changes have improved overall performance and increased the capabilities of the scanner without additional expense.

2.3 THEORY OF OPERATION-HARDWARE.

2.3.1 Alarm/Event Sensing Subsystem. The event/alarm scanner is capable of sampling and detecting up to 256 alarm conditions characterized by normally open contacts closing on the alarm. This function is implemented with one decoder/driver card and up to 16 alarm input cards (Figure 2-1).

The decoder/driver card consists of a decoder to select a particular alarm card. Level regeneration and buffering are provided to prevent high system data bus loading when many alarm cards are installed. The decoder board is installed on the microprocessor bus and is connected to the alarm board rack via a 24 line cable. This cable carries the board select code, word select and data bus connections to the alarm rack.

Each alarm card consists of two tri-state octal buffers and 16 LED displays with associated resistors. (Reference Fig. 2-1) One contact of each alarm is tied to the +5 supply and the other contact is connected to an input of the buffer. The inputs are connected to show a logic Ø when the alarm is inactive. When the alarm contact closes, the input is pulled high and also sources the current to the LED to give a visual indication of the alarm. When the microprocessor addresses the alarm card, the actual state of the alarms is available on the system data bus to be read by the processor. Using this input scheme, any alarm input that is not connected shows up as an inactive alarm, and allows the monitoring of TTL level alarm inputs as well as relay contact closures.

2.3.2 Relay Control Subsystem. The relay control system (Figure 2-2) is capable of controlling up to 128 SPDT relay contacts independently. This function is implemented using one decoder/driver card and up to 16 relay cards.

The relay control section is very similar to the design of the alarm section. The decoder/driver card includes a decoder to select one of the relay cards to be accessed. The driver provides a high current capability to drive the inputs of the relay cards and prevent loading on the uP data bus by the relay rack bus. The relay rack is connected to the decoder/driver card by a 24 conductor cable carrying the data bus and the board select.

The relay board consists of an octal D flip-flop, two high voltage inverting buffers and eight relays. When the board is selected by the decoder, the data that is on the data bus is strobed into the latch. The outputs of the flip-flops in the latch are connected to the high voltage drivers which control the coil current in the relays.

2.3.3 Analog to Digital Converter Card. The Analog to Digital (A/D) Converter card is implemented using an off-the-shelf 8-bit A/D module. As shown in Figure 2-3, the A/D card consists of a decoder, and A/D module and an operational amplifier. The analog input is buffered by the operational amplifier to provide isolation and to prevent input loading. Additionally, the operational amplifier may be configured to provide a linear scaling factor, or non-linear (e.g. exponential) transfer function for specialized applications. The A/D conversion begins when a module is selected by the address decoder and ends when the processor "Ready" line goes high. Processor utilization is thus determined by the conversion speed of its A/D converter module since processing cannot continue until the conversion is complete.

2.3.4 Pulse Counter Card. The Pulse Counter Card (Figure 2-4) is based on the Intel 8253 counter. Each of these integrated circuits consists of three independent 16 bit counters that may be configured as one 48 bit counter. The decoder in this application selects only the chip with each counter in the chip being addressable by the system address bus. The signal conditioner shown in Figure 2-4 may include Schmitt triggers, level convertors and/or a prescaler to divide the pulse rate to the 2MHz limit of the 8253.

Through the use of various software configurations, the pulse counter card may be used for a multitude of performance assessment parameters; e.g., percent error free seconds, instantaneous bit error rate, recursive bit error rate, etc.

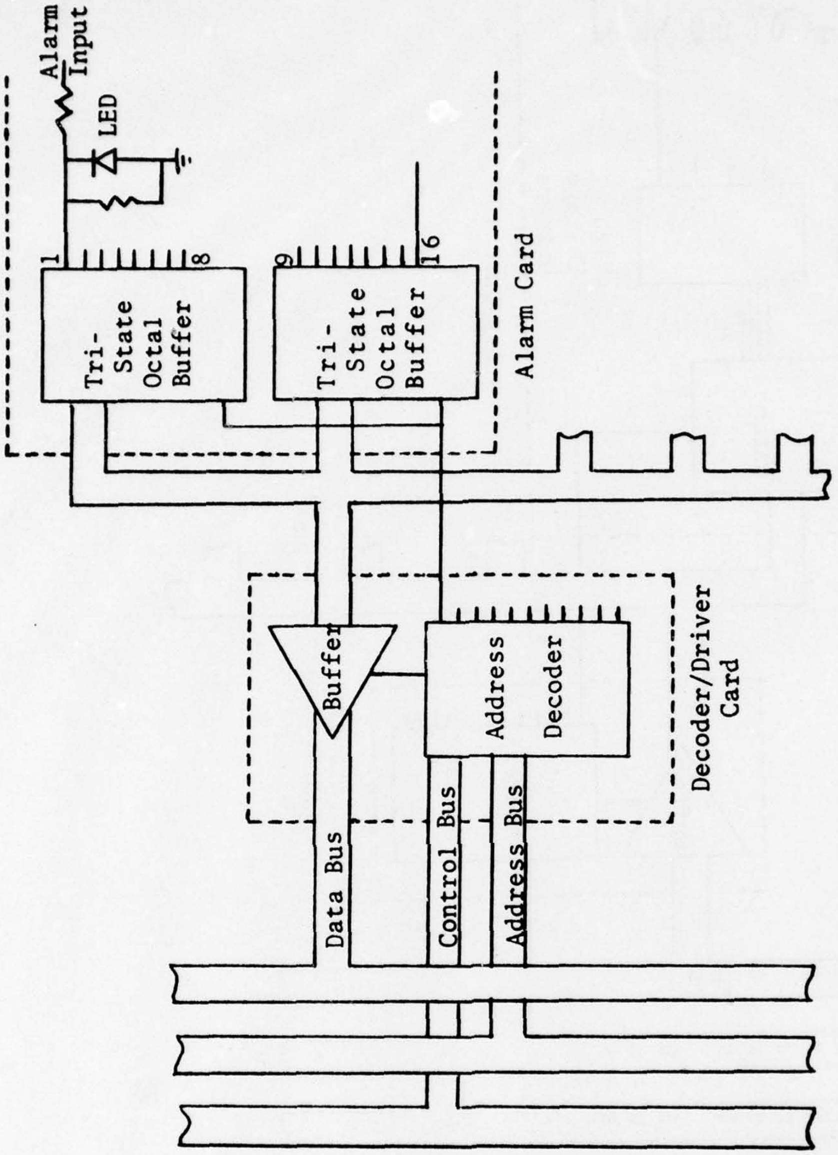


FIGURE 2-1 Alarm Input Subsystem

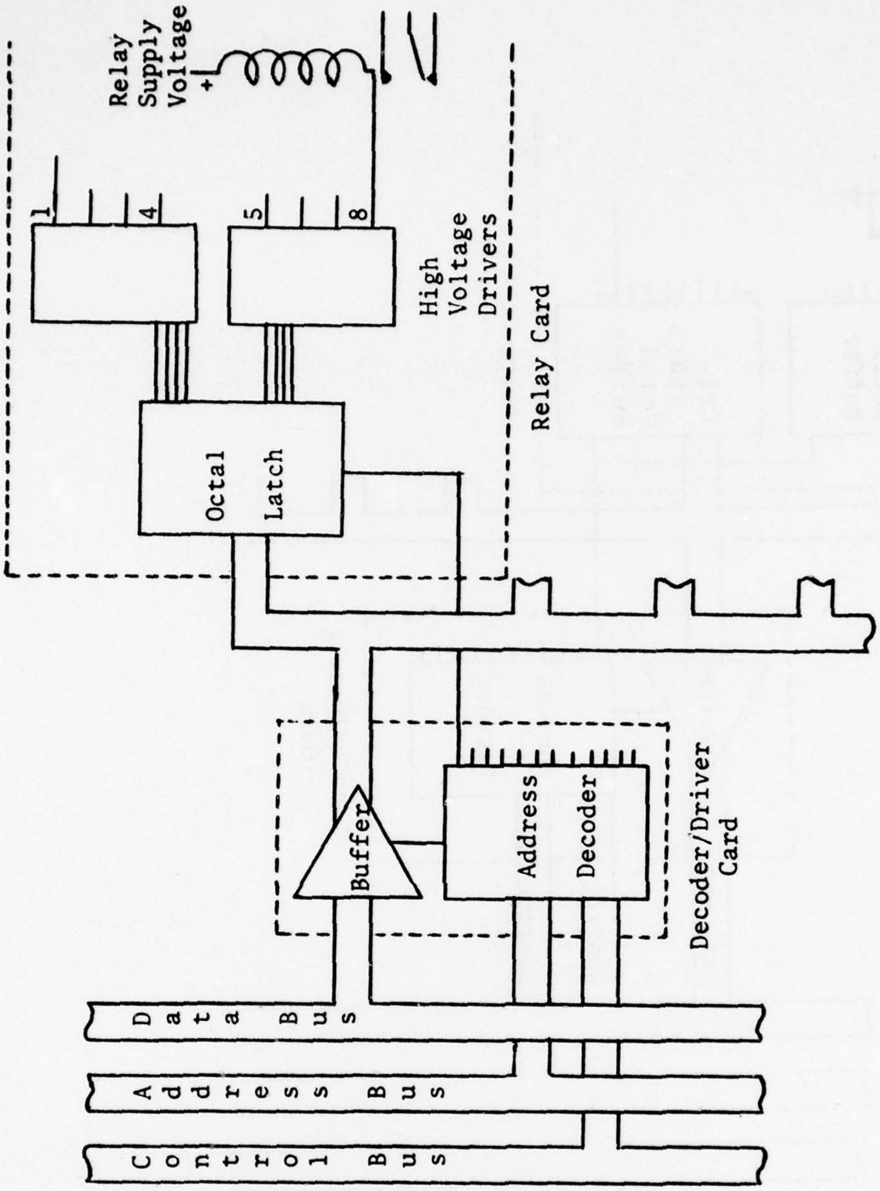


FIGURE 2-2 Relay Control Subsystem

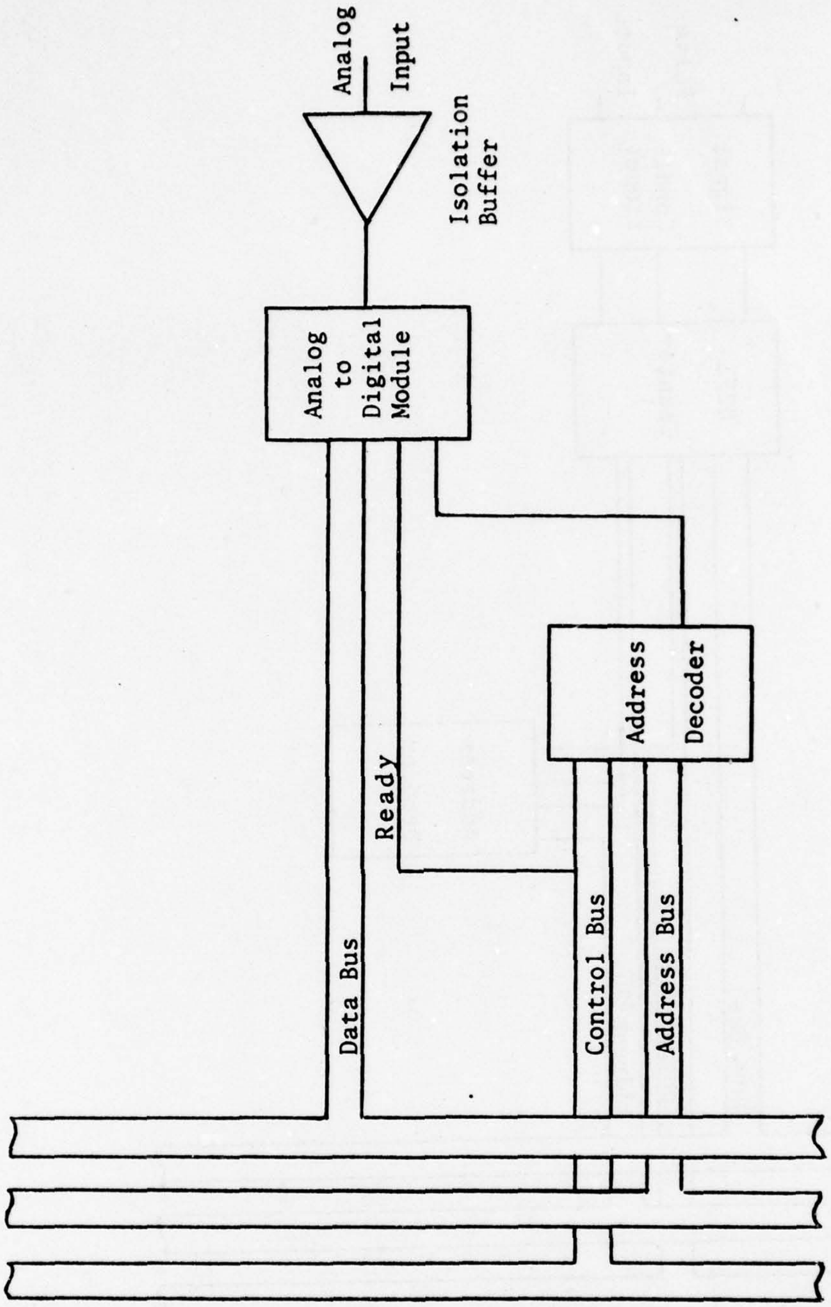


FIGURE 2-3 Analog to Digital Converter Card

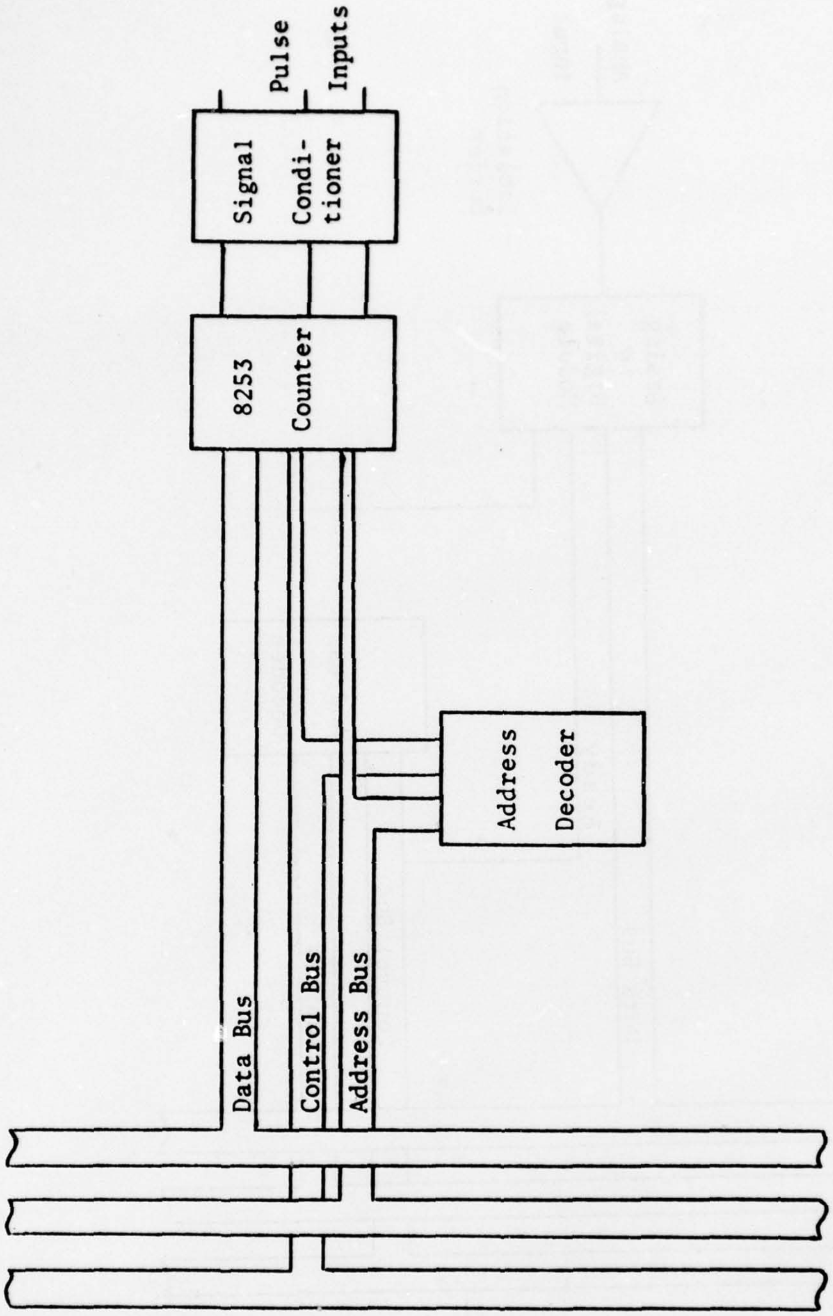


FIGURE 2-4 Pulse Counter Card

2.3.5 GPIB Interface. The GPIB (Figure 2-5) controls all the transfers to and from the GPIB and the uP. The protocol and handshaking are transparent to the uP thus only a minimal amount of overhead is incurred for GPIB communications. Four special bus drivers provide the high drive capability and tri-state or open-collector outputs required for GPIB compatibility.

2.4 SYSTEM SOFTWARE-THEORY OF OPERATION.

2.4.1 Alarm Scanner. The alarm scanner software uses the alarm input subsystem to read the current state of the alarm inputs. Alarms are scanned sequentially and the current condition of the inputs is compared to the status saved during the last scan. Any difference in status causes the histogram subroutine to be invoked.

2.4.2 Histogram Processing. The histogram processing section is broken up into four subroutines. These subroutines allocate and clear the histograms in addition to processing alarm status changes.

The allocation routine computes the size of storage required for histograms and tables. The user must supply the number of alarm cards installed; the processor then sets up the appropriate memory partitions for the histograms.

The histogram clear function sets the histograms and tables to zero values before processing is begun, or after the system has dumped the histogram data to the controlling device.

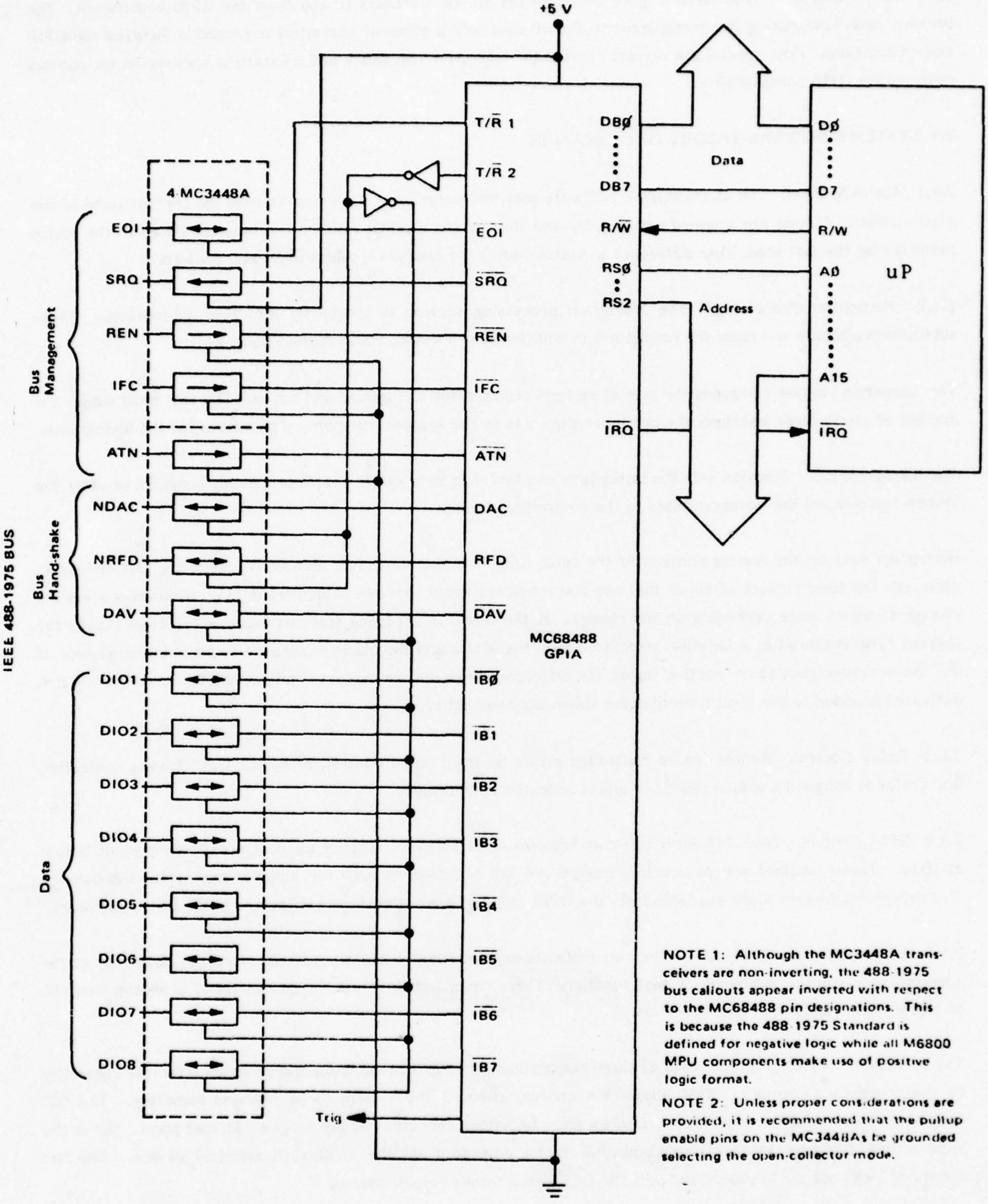
Histogram data on the alarms consists of the total time that the alarm has been active since the last histogram clear and the total number of times that the alarm has activated over the same period. Two routines process the change of alarm state depending on the change. If the alarm is set (goes from an inactive to active state), the current time is stored in a location associated with the alarm and the alarm occurance count is incremented. If the alarm resets (goes to an inactive state), the difference between the current time and the time saved when it is activated is added to the total time that the alarm has been active.

2.4.3 Relay Control. Relays can be controlled either by front panel control, or by the GPIB from a controller. Each relay is assigned a unique identifier and is accessed by that name.

2.4.4 GPIA Control. The GPIA interface chip requires some software support for configuration and input/output of data. These routines are of a utility nature and are accessed through the monitor read/write subroutines. Various special functions are available from the GPIA chip but are not currently supported in the driver software.

2.4.5 Communications Executive. The communications executive routes all requests from the controller to the appropriate subroutine designed for that function. These commands fall into the categories of processor control, histogram commands and relay operation.

The processor control group consists of three commands. The "P" command loads a program from the controller to the processor's memory. This allows the configuration of the system to be changed remotely. The "G" command tells the uP to go to a specified location and start execution of the program at that point. "L" is the local enable function which allows commands to be entered from the processor's terminal device. The "R" command locks out the keyboard and puts the CES into a totally remote status.



NOTE 1: Although the MC3448A transceivers are non-inverting, the 488-1975 bus callouts appear inverted with respect to the MC68488 pin designations. This is because the 488-1975 Standard is defined for negative logic while all M6800 MPU components make use of positive logic format.

NOTE 2: Unless proper considerations are provided, it is recommended that the pullup enable pins on the MC3448As be grounded selecting the open-collector mode.

FIGURE 2-5 GPIA Block Diagram

The histogram command section implements three commands. The "A" command allocates a block of memory to the histogram storage. The controller must tell the CES how many alarms are to be monitored when this command is sent. The "D" command dumps the histogram data from the CES to the controller. Depending on the parameters with the command, any or all of the histograms may be dumped. The last function is to clear the histograms with the "C" command. This sets all histogram locations to zero as an initialization step. This command also resets the system clock.

The "K" command specifies a relay to be activitated or deactivated. The parameters specify the relay and the action to be taken. A summary of Executive commands are included in Appendix 4.

3.0 CES APPLICATIONS.

3.1 STAND-ALONE CES. The internal architecture of the CES, in a stand-alone mode, is determined by the software monitor program which resides in EPROM. The monitor allows the operator to do the same functions as under computer control but from local terminal control. In addition, the operator may examine RAM locations to determine what value has been stored in the histogram or if any input card is not responding and so on. A listing of terminal commands is contained in Appendix 3. The stand-alone or "local" mode is intended for use when no controller is available or when side maintenance is being accomplished.

3.2 CES WITH REMOTE PROCESSOR. The full potential of the CES is realized when it is used in conjunction with a calculator or minicomputer-based controller that conforms to the IEEE 488-1975 Standard for GPIB controllers. Commands, data, and configuration programs may be exchanged between the controller and the CES over the GPIB. The controller may initiate any function or read any data contained in the CES by transmitting the appropriate command. It is also possible to reconfigure the software of the CES to change performance assessment algorithms by downloading the appropriate software modules to the CES from the controller. For example, suppose it is desired to have the CES measure percent error free seconds instead of instantaneous bit error rate. Simply download the appropriate software module from the controller and, using the same hardware configuration, the CES will transfer control to the new program in RAM, and will now process percent error free seconds. Naturally the CES will probably already contain most of the performance assessment type algorithms in EPROM, however, should any additions or modifications be desired, the CES has this capability. Should the new algorithm prove more useful than the old one, it is a simple matter to permanently install the program in EPROM within the CES.

When used in conjuntion with a controller, the CES may be thought of as a "smart" sensor (Figure 3-1) since it has the ability to pre-process the data collected into system parameters before passing the information to its controller. This frees the controller to do more important data analysis and prevents the time consuming process of transmitting unprocessed data.

3.3 CES AND DPAN. The CES becomes an even more important concept, in terms of a "smart" sensor, when viewed in a distributed performance assessment network (ref. 1842 EEG TR-77-4) as shown in Figure 3-2. It is conceivable that with the appropriate input cards, the CES might be the only sensor needed at each site to accomplish data acquisition and parameter processing. Through the use of communication executive program control, the nodal controller, as well as the remote processor, would have the capability to reconfigure the CES software architecture.

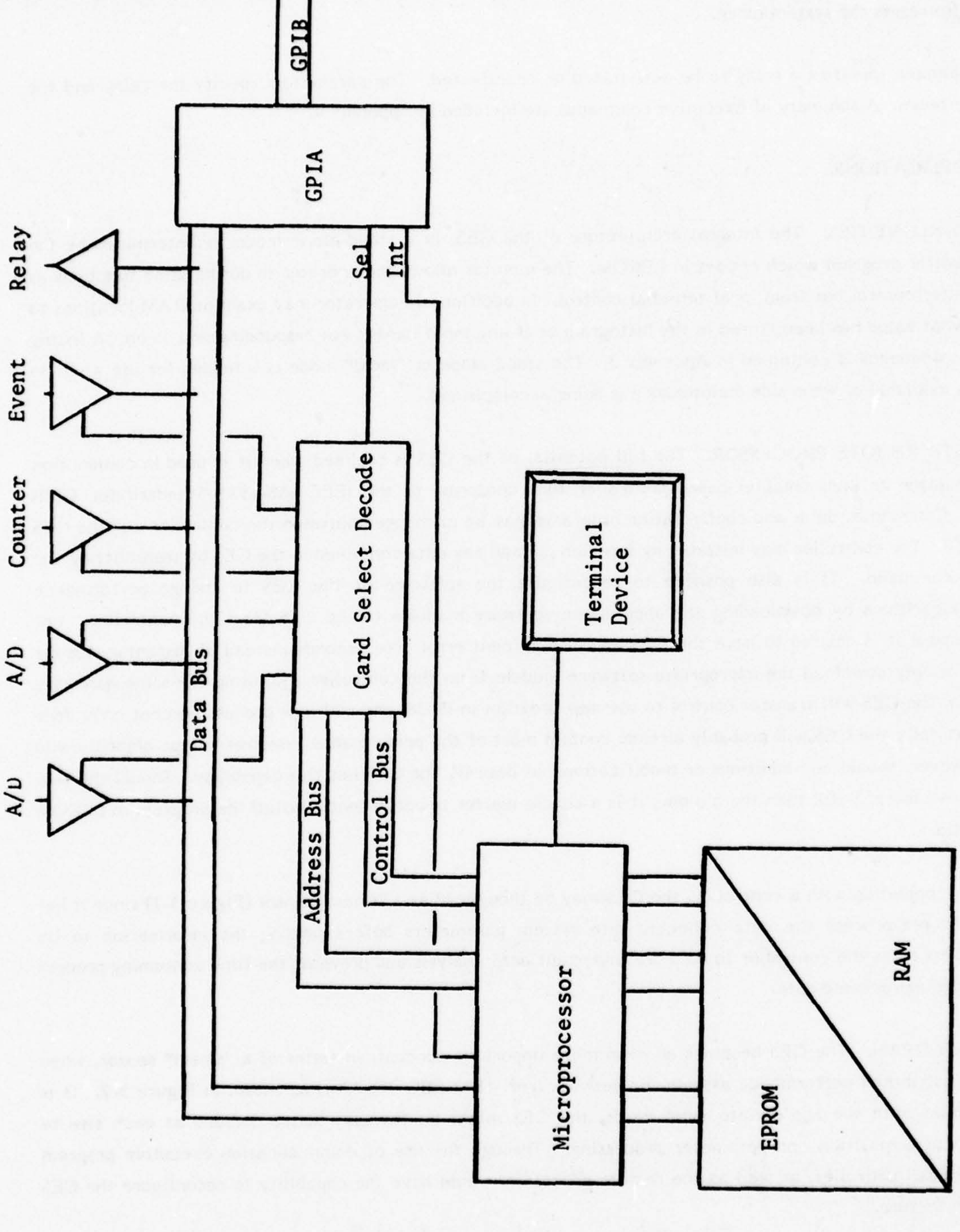


FIGURE 3-1 CES "Smart Sensor"

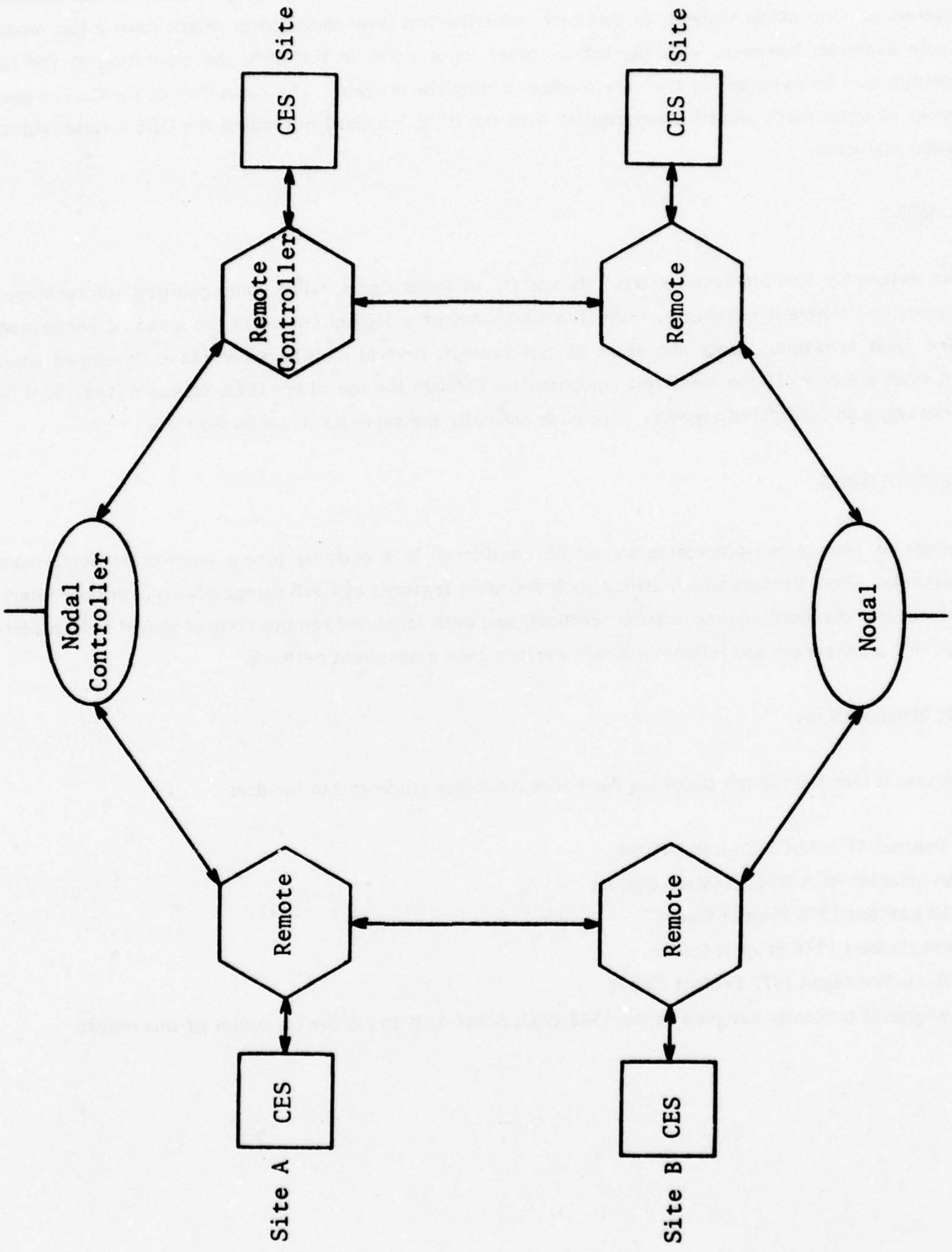


FIGURE 3-2 Distributed Performance Assessment Network

3.4 FAULT ISOLATION. Figure 3-3 shows how the CES might be configured in a typical digital communications site. By connecting various input cards to strategic points, at the site equipment, a level of fault isolation may be accomplished. In this case, the fault may be isolated to a particular equipment. For example, suppose the CES indicates that it is receiving error counts on pulse counter card 1. A quick check of A/D cards 1 and 2 indicate AGC voltage (and thus radio received signal level) and the noise level on the digital eye pattern (radio baseband) are within tolerance. One would suspect, in this case, that the 2nd level multiplexer might have a bad module. This is a simple example, however, with the use of other input cards in the CES, the resolution of the fault isolation algorithm may be extended to the very module causing the problem. The capability of the CES to use an adequate number of input cards and its compatibility with the IEEE Standard Bus makes the CES a viable solution to fault isolation problems.

4.0 CONCLUSIONS.

The CES is an extremely flexible test fixture. Its variety of input cards, GPIB compatability, microcomputer control, low cost, and inherent reliability, make the CES concept a logical choice in the areas of performance assessment and fault isolation. Since the start of this project, several manufacturers have developed similar capabilities in what are now off-the-shelf test equipments. Through the use of the IEEE Standard (488-1975) Bus, these equipments may be configured together to provide basically the same functions as the CES.

5.0 RECOMMENDATIONS.

The CES concept of remote pre-processing should be considered in specifying future large-scale performance assessment networks. Test fixtures which utilize such desirable features as GPIB compatability, variable internal software architecture, standard communication protocol, and both local and remote control should be considered essential to provide an accurate and reliable military performance assessment network.

6.0 ACKNOWLEDGEMENTS.

Special recognition is merited for the following Air Force Academy students and faculty:

Maj. Ed Pearsol AFA AFCS Liaison Officer
Maj. John Schmidt AFA AFCS Liaison Officer
Cadet Zak LaPlant 1976 Project Cadet
Cadet Bert Halbert 1976 Project Cadet
*Cadet M. D. Verstegen 1977 Project Cadet
*2Lt Verstegen is currently assigned to the 1842 EEG, Scott AFB and is the co-author of this report.

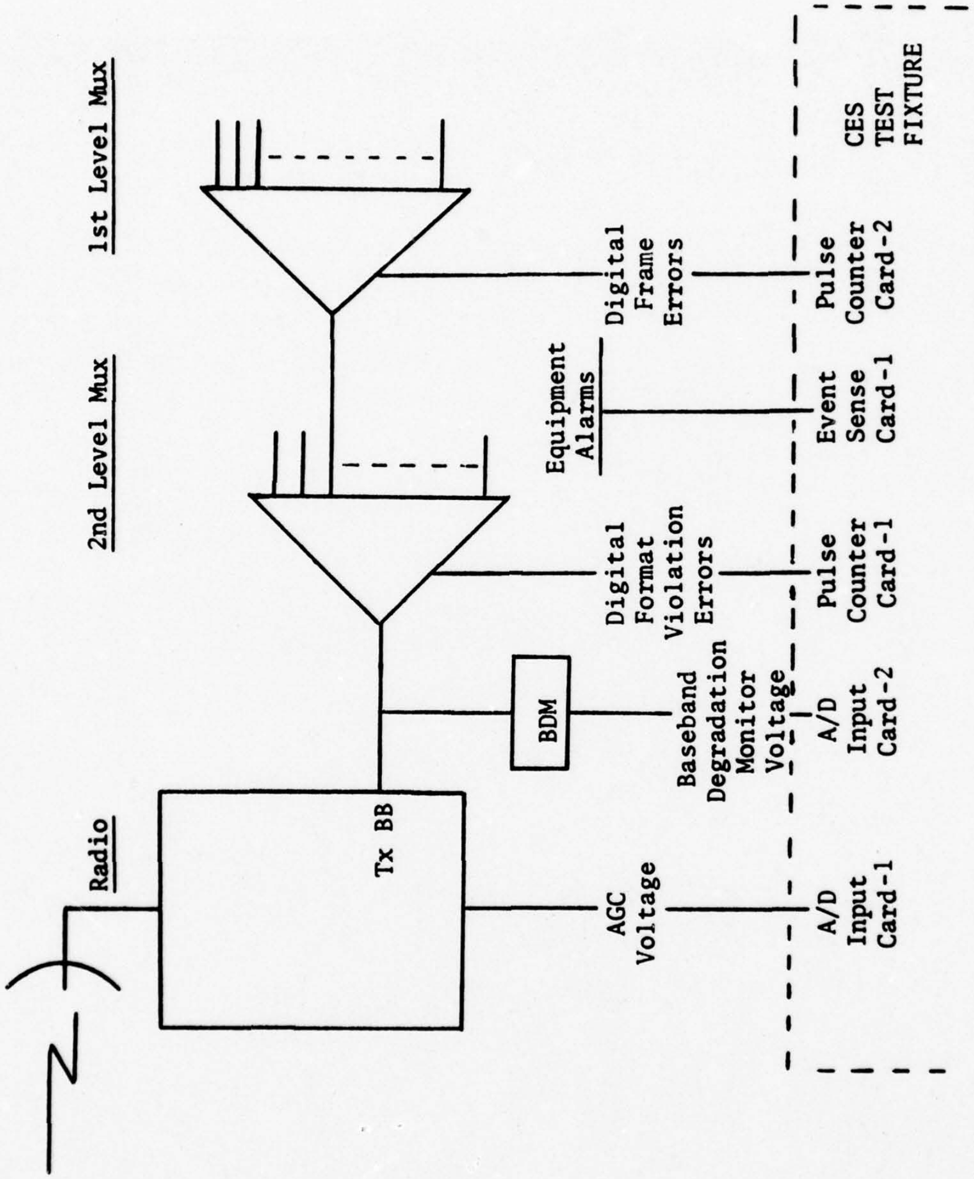


FIGURE 3-3 CES Fault-Isolation Configuration

APPENDIX A

GENERAL PURPOSE INTERFACE BUS DESCRIPTION

The overall purpose of an interface system is to provide an effective communication media so that two or more devices or systems can transmit meaningful messages between each other. The IEEE STD 488-1975 Standard Digital Interface for Programmable Instrumentation is such an interface. This interface will be referred to as a General Purpose Interface Bus (GPIB) hereinafter.

The bus is made up of 16 lines to carry information between instruments connected to the GPIB. Figure A-1 gives a pictorial representation of the GPIB. The 16 lines can be divided into three functional groups.

Eight lines are used to transfer data between devices in ASCII coded bit parallel, byte serial format. Other information codes could be used for device dependent messages, however, and the ASCII code is not required by the IEEE standard.

Three lines are dedicated to the control of data byte transfers between devices and are commonly called handshake lines, thus allowing the data to be transferred asynchronously over the GPIB. See Figure A-2.

Five additional lines are used for general interface management. One line is used to determine the data mode or the command (addressing) mode for the eight data lines. See Figure A-1 for the purpose of each.

Since it takes more than just the physical interconnection of black boxes to have an effective communication media, specific interface functions must be defined. Ten different possible functions are shown in Figure A-3 as defined by the standard. The list gives a general idea of the purpose of each function. A description of each function as it relates to hardware can be found in more detail in the IEEE Standard 488-1975. The handshake functions, however, are what make the GPIB a viable interface and are best described by the flow diagram in Figure A-2.

For any given device, only those functions required to satisfy the operation of the device need to be included in the design of an instrument.

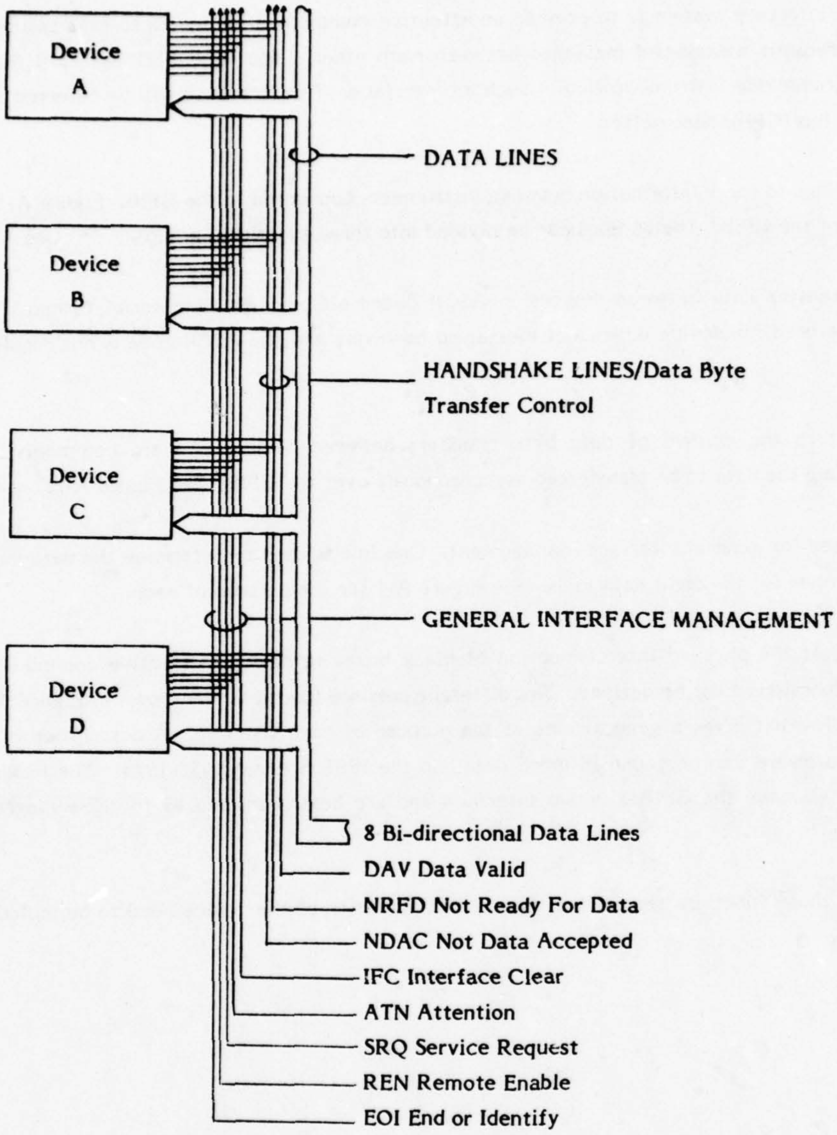


FIGURE A-1 GENERAL PURPOSE INTERFACE BUS (GPIB) STRUCTURE

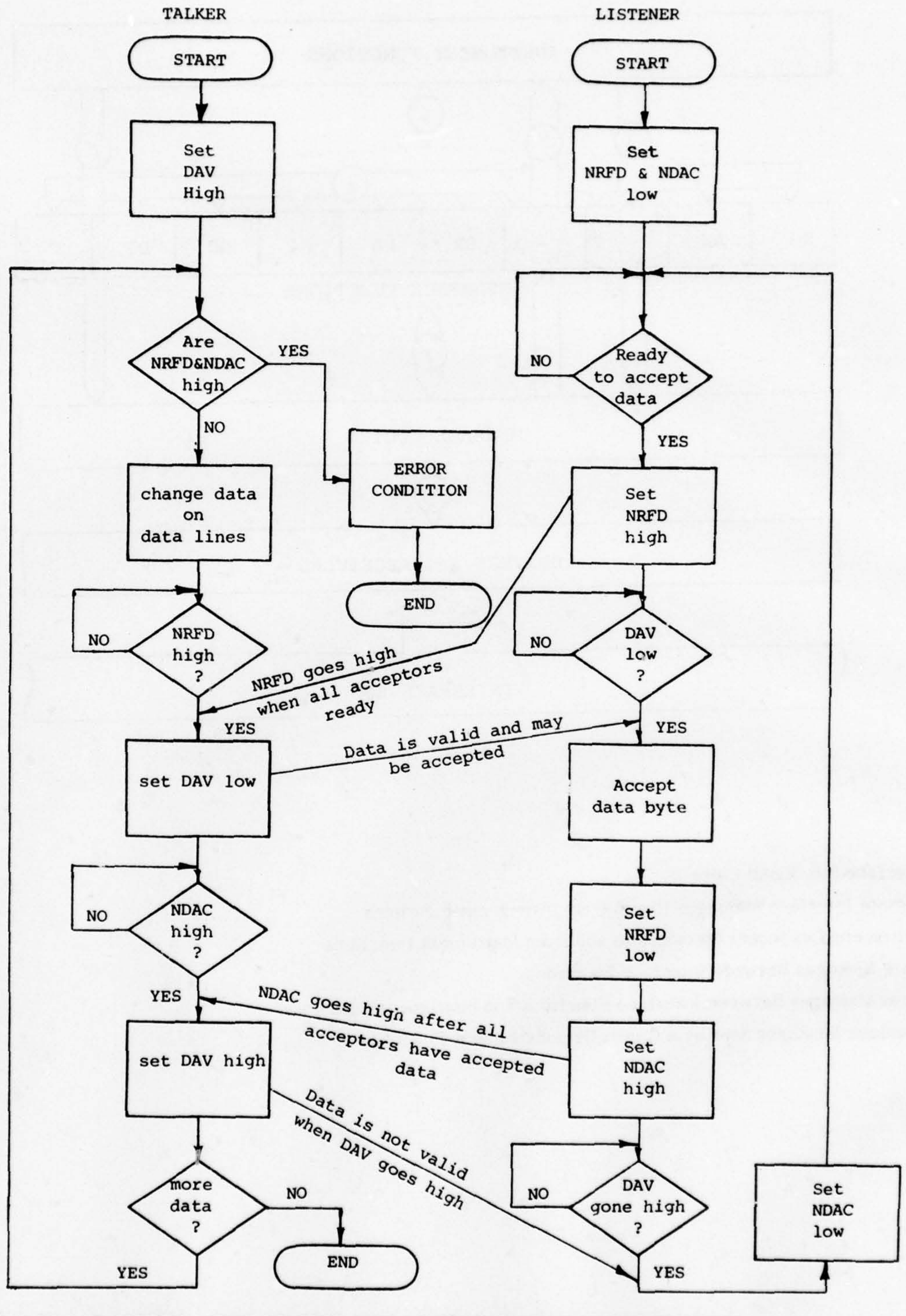
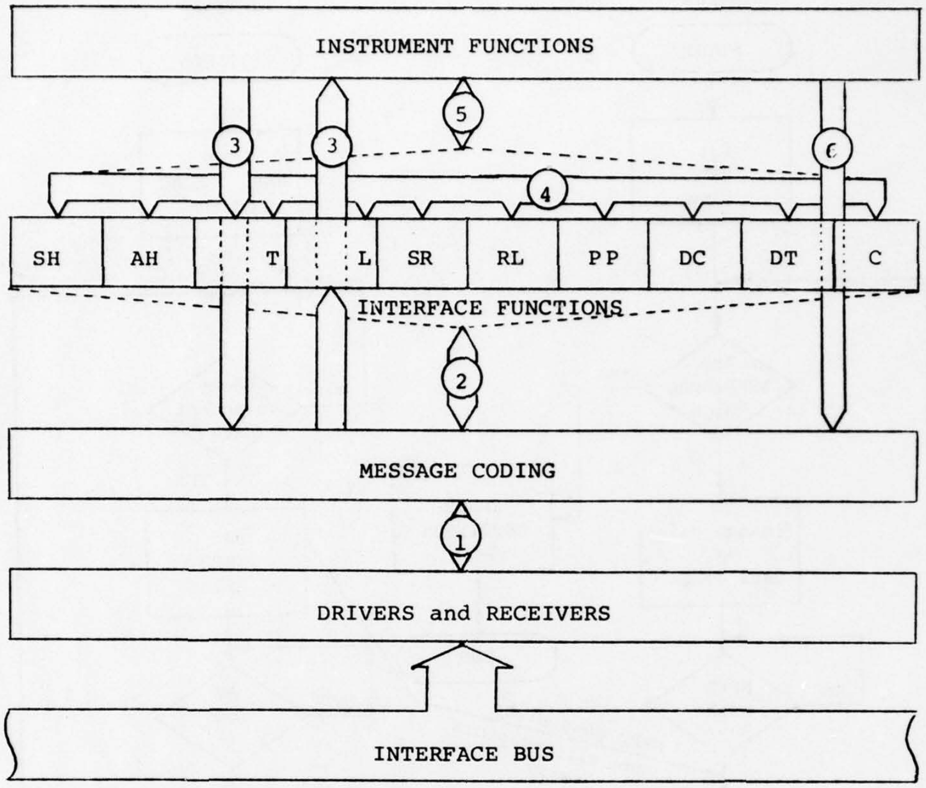


FIGURE A-2 DATA TRANSFER USING SOURCE AND ACCEPTOR HANDSHAKE PROCESS



1. Interface Bus Signal Lines
2. Remote Interface Messages to and from Instrument Functions
3. Instrument Dependent Messages to and from Instrument Functions
4. State Linkages Between Interface Functions
5. Local Messages Between Interface Functions and Instrument Functions
6. Interface Messages Sent by a Controllers Instrument Functions

FIGURE A-3 INTERFACE FUNCTIONS AND INSTRUMENT MESSAGE FLOW DIAGRAM

APPENDIX B

Proposed Hardware
Concept of Dedicated Monitors

SECTION I
CONTROL/EVENT SCANNER

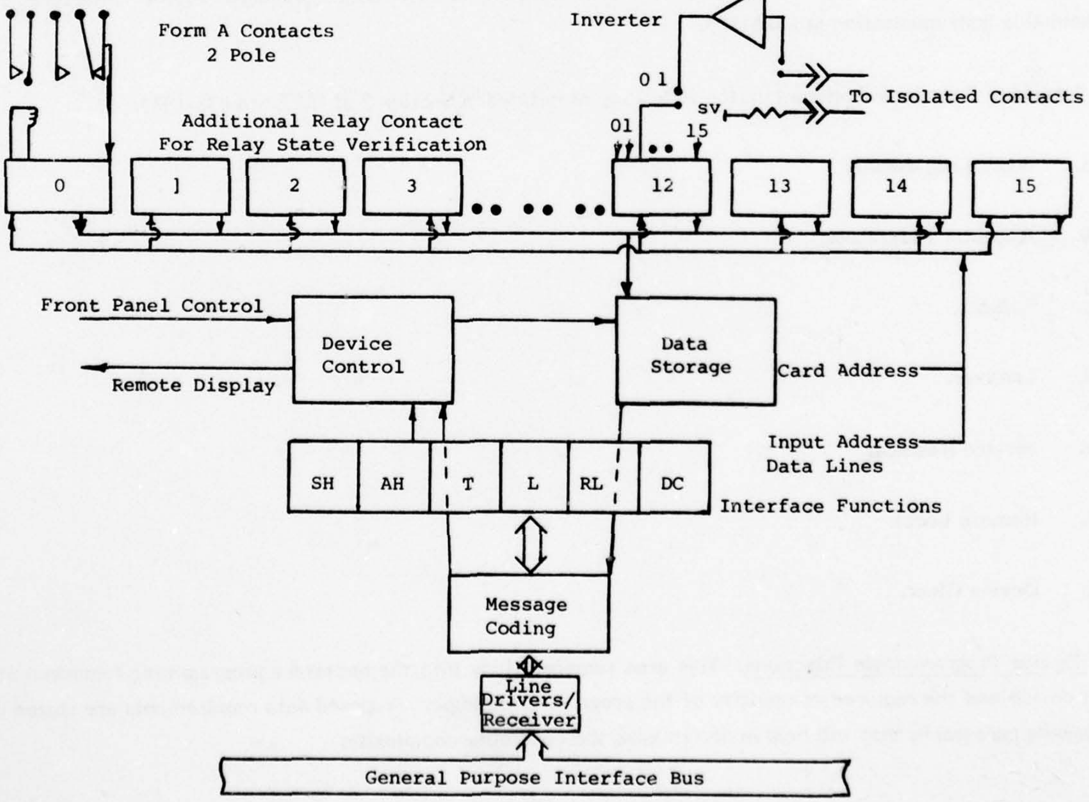
1.0 Scope: This attachment applies to a stand alone monitoring instrument that can (1) detect and process the state of up to 256 alarms (Form C contact closure) in groups of 16 inputs per card, or (2) provide 8 Form A relay contact closures per card for a total of 128 relay control points. Any combination of alarm and relay cards will be allowed per mainframe. The instrument will be automated with interface to system controller conforming to IEEE STD 488-1975. Local control will be accomplished with front panel controls.

2.0 Applicable Documents:

IEEE 488-1975	IEEE Standard Digital Interface for Programmable Instrumentation.
DNSF TR-75-100	Radio Service Channel Techniques in a Digital Communication System

3.0 Requirements:

Design and construct two prototype Control/Event Scanner that will perform the following functions. A block diagram is shown in Figure B-1.



3.1.0 Stand Alone Requirement.

3.1.1 Sense on-off (True-False) indications from communication equipment that uses Form C relay contacts for alarm indications. Necessary voltage or current sources should be contained within the monitoring equipment. Current sensing shall be limited to .5 amp. Voltage sensing shall be possible from 4 to 50 volts.

3.1.2 Sample rate for each alarm should be at least one sample/second or 256 alarms/second scan rate.

3.1.3 Each unit shall have the capability to sense up to 256 alarm states in groups of 16 alarms per plug in printed circuit card.

3.1.4 Control communication equipment by the activation of two pole Form C relay contacts. Contacts shall have a current rating of at least 1.0 amps at 48 volts.

3.1.5 Each unit shall have the capability of controlling 128 separate relays in groups of eight relays per card.

3.1.6 Display alarms on front panel of each card using red LED's.

3.1.7 Activate selected control relays from front panel and display activated relays with LED indicators.

3.2.0 Interface Functions.

3.2.1 All functions of the Control/Event Scanner should be programmable.

3.2.2 The interface to a system controller shall conform to the IEEE Standard Digital Interface for Programmable Instrumentation Std 488-1975.

3.2.3 Interface Functions shall contain the following, as defined in Section 2 of IEEE Std 488-1975:

a. Source Handshake.

b. Acceptor Handshake.

c. Talker.

d. Listener.

e. Service Request.

f. Remote Local.

g. Device Clear.

3.3.0 Device Programmable Functions. This area requires study into the necessary programming requirements for this device and the required complexity of the programmable logic. Proposed data requirements are stated in the following paragraphs that will help in determining the controller complexity.

3.3.1 Scan inputs to determine alarm states.

3.3.2 Transmit information over interface when addressed to talk.

3.3.2.1 Each alarm should be characterized by a multiple ASCII coded character. As an example:

```
Unit Number
┌Transmitter
│┌RF Carrier
││
3TC ┘
```

3.3.2.2 Only active alarms will be transmitted.

3.3.2.3 Active control relays will be included in the data transfer.

3.3.2.4 When addressed to listen, accept control information to activate or deactivate relays specified by a multiple ASCII coded character.

3.3.2.5 Store histogram of each alarm giving number of times each alarm was activated and total time each alarm is in the active state.

4.0 Mechanical Considerations.

4.1 It is desired that all caros be accessible from the front of the equipment.

4.2 Figure A2-2 shows a proposed front panel layout.

4.3 Unit shall be capable of being rack mounted in a 19-inch rack.

5.0 Power.

5.1 Unit shall operate with a power line frequency anywhere between 47 and 63 Hz.

5.2 Unit shall operate with a power line voltage of 110V or 220VC (±20%) switch selectable.

```
AFCS 8080 ASSEMBLER      CES SYSTEM SOFTWARE       00:40 322/78        PAGE   1

 SYMBOL TABLE
    SYMBOL    VALUE       SYMBOL    VALUE       SYMBOL    VALUE       SYMBOL    VALUE

    LWAM      0F FF       EXEC      09 00       JMPTBL    09 14       NOGOOD    09 3A
    ALARMS    7F E0       NCARDS    0F FF       STATUS    0F FD       TIMEON    0F FB
    OCCUR     0F F9       TOTTIM    0F F7       TYPCHG    0F F5       HSTTOP    0F F4
    HSTMEM    09 3C       DBLSUB    09 71       HSTCLR    09 78       CLRH1     09 80
    SCAN      09 8D       CHECK     09 98       UPDATE    09 AC       UPLOOP    09 B4
    UPSTOR    09 BF       UPDSET    09 C7       UPDRST    09 D2       HSTGRM    09 DA
    HSTSET    09 F0       HSTRST    0A 0C       HSDMDR    0A 35       HSDMAL    0A 49
    HSTDMP    0A 54       EXCUTE    0A A6       PLOAD     0A AA       PLLOOP    0A C2
    GPIA      BF C8       GPTALK    0A D5       GTTRAN    0A EA       GTLAST    0A F8
    GPLIST    0B 03       GLREC     0B 19       GLSTAT    0B 21       GLSKIP    0B 30
    CTLPRT    00 0C       BDRTPT    00 0B       CHSLPT    00 09       USTAT     00 08
    DATAIN    00 09       DATOUT    00 0A       ASNIO     0B 38       TTY       0B 44
    CASETT    0B 47       READS     0B 4A       RNOERR    0B 56       WRITES    0B 60
    DMPASC    0B 6C       BLKDMP    0B 77       BKDP1     0B 8A       SEQFIL    0B 97
    SQFIL1    0B A0       MEMACC    0B B2       MEMAC1    0B DB       ERROR     0B DE
    ERRMSG    0B F8       READ      0B 19       WRITE     0A EA       CNTR      BF C0
    TIME      09 FE       CCTLWD    BF C3       IN8253    0C 0A       RS8253    0C 1E
    RD8253    0C 26       HEX1      0C 37       HEX1A     0C 45       HEX2      0C 49
    ASCII     0C 59       CMPHND    0C 63       RD1BHX    0C 68       RD2BHX    0C 73
    RD4BHX    0C 7C       RD1BI3    0C 85       WR1BHX    0C A4       WR2BHX    0C B7
    WR1BI3    0C C0       WR1BI2    0C CF       CR        00 0D       LF        00 0A
    CRLF      0C E3       MLTPLY    0C EE       MULT0     0C F2       DIVIDE    0D 01
    DIV0      0D 04       DIV1      0D 15       DIV2      0D 1D

LINE  LOC   B1 B2 B3          SOURCE LINE

   1                                 NAM CES SYSTEM SOFTWARE
   2                          ******************************************************
   3                          *                                                    *
   4                          *         CONTROL EVENT SCANNER SOFTWARE             *
   5                          *                                                    *
   6                          *        LT MIKE VERSTEGEN, 1842 EEG/EETET           *
   7                          *                                                    *
   8                          ******************************************************
   9                                 ORG '0900
  10                          LWAM   EQU '0FFF        LAST WORD AVAILABLE R/W MEMORY
  12                          ******************************************************
  13                          *                                                    *
  14                          *     C O M M A N D   E X E C U T I V E              *
  15                          *                                                    *
  16                          ******************************************************
```

```
AFCS 8080 ASSEMBLER      CES SYSTEM SOFTWARE       00:40 322/78         PAGE  2

LINE  LOC   B1 B2 B3            SOURCE LINE

  18                     *
  19                     * EXEC USES THE ASCII CHARACTER IN THE ACCUMULATOR
  20                     *    TO JUMP TO THE APPROPRIATE SUBROUTINE.
  21                     *       REGISTERS AFFECTED: ALL
  22                     *
  23  0900  D6 41        EXEC   SUI "A"                 OFFSET FOR COMMANDS
  24  0902  FE 13               CPI "S"-"A"+1           LIMIT NUMBER OF COMMANDS
  25  0904  D2 3A 09            JNC NOGOOD              INVALID COMMAND
  26  0907  07                  RLC                     MULTIPLY BY 2
  27  0908  5F                  MOV E,A                 MOVE TO LSB OF DE
  28  0909  16 00               MVI D,0                 CLEAR MSB
  29  090B  21 14 09            LXI H, JMPTBL           LOAD BASE OF JUMP TABLE
  30  090E  19                  DAD D                   COMPUTE ABSOLUTE INDEX TO TBL
  31  090F  56                  MOV D,M                 GET MSB OF ADDRESS
  32  0910  23                  INX H                   POINT TO LSB
  33  0911  5E                  MOV E,M                 GET LSB
  34  0912  EB                  XCHG                    PUT ADDRESS IN HL
  35  0913  E9                  PCHL                    JUMP THERE - RETURN TO SECTION
  36                     *                                 THAT CALLED EXEC
  37                     *                               VERIFIED 22 AUG 78
  38  0914  09 3C        JMPTBL DW HSTMEM               A - ALLOCATE MEMORY
  39  0916  09 3A               DW NOGOOD               B NOT USED
  40  0918  09 78               DW HSTCLR               C - CLEAR HISTOGRAM
  41  091A  0A 35               DW HSDMDR               D - DUMP HISTOGRAM DATA
  42  091C  09 3A               DW NOGOOD               E
  43  091E  09 3A               DW NOGOOD               F
  44  0920  0A A6               DW EXCUTE               G - GO
  45  0922  09 3A               DW NOGOOD               H
  46  0924  09 3A               DW NOGOOD               I
  47  0926  09 3A               DW NOGOOD               J
  48  0928  00 00               DW 0                    K - RELAY CONTROL
  49  092A  00 00               DW 0                    L - LOCAL ENABLE
  50  092C  09 3A               DW NOGOOD               M
  51  092E  09 3A               DW NOGOOD               N
  52  0930  09 3A               DW NOGOOD               O
  53  0932  0A AA               DW PLOAD                P - PROGRAM LOAD
  54  0934  09 3A               DW NOGOOD               Q
  55  0936  00 00               DW 0                    R - REMOTE
  56  0938  09 8D               DW SCAN                 S - SCAN
  57  093A  FB           NOGOOD EI                      INVALID COMMAND
  58  093B  C9                  RET                        NO ACTION
  60                     ******************************************************
  61                     *                                                    *
  62                     *E X E C U T I V E   R O U T I N E S   L I B R A R Y*
  63                     *                                                    *
  64                     ******************************************************
  65                     ******************************************************
  56                     *
  67                     *          SYSTEM MEMORY MAP
```

```
LINE  LOC   B1 B2 B3            SOURCE LINE

  68                  *
  69                  * LABEL     SIZE                DESCRIPTION
  70                  *
  71                  * ALARMS '7FE0-'7FFF     MEMORY MAPPED I/O LOCATION OF
  72                  *                        ALARM INPUTS. ADDRESS IS
  73                  *                        HARDWIRED ON DECODER BOARD.
  74                  *
  75                  * STATUS 1 BIT/ALARM     STATUS OF ALARMS DURING LAST
  76                  *        2 BYTES/CARD    SCAN CYCLEE
  77                  *
  78                  * TIMEON 2 BYTES/ALARM   TIME ALARM WAS ACTIVATED.
  79                  *        32 BYTES/CARD
  80                  *
  81                  * TOTTIM 2 BYTES/ALARM   TOTAL TIME ALARM HAS BEEN ON
  82                  *        32 BYTES/CARD   SINCE LAST HISTOGRAM CLEAR.
  83                  *
  84                  * OCCUR  1 BYTE/ALARM    NUMBER OF TIMES THE ALARM HAS
  85                  *        16 BYTES/CARD   HAD AN OFF TO ON TRANSITION.
  86                  *
  87                  *
  88                  ************************************************************
  89                  ALARMS EQU '7FE0         HARDWIRED ADDRESS OF ALARMS
  90                  NCARDS EQU LWAM          NUMBER ALARM CARDS INSTALLED
  91                  * THE FOLLOWING FOUR WORDS CONTAIN THE ADDRESS OF THE
  92                  *    BASE OF THE NAMED HISTOGRAM TALBES
  93                  STATUS EQU NCARDS-2      ALARM STATUS
  94                  TIMEON EQU STATUS-2      TIME ALARM TURNED ON
  95                  OCCUR  EQU TIMEON-2      COUNT OF ALARM ACTIVATIONS
  96                  TOTTIM EQU OCCUR-2       TOTAL TIME ALARM ON
  97                  TYPCHG EQU TOTTIM-2      TYPE OF UPDATE PROCESSED
  98                  *                           0 - TURNED OFF (RESET)
  99                  *                           1 - TURNED ON (SET)
 100                  HSTTOP EQU TYPCHG-1      TOP OF HISTOGRAMS
 102                  ************************************************************
 103                  *
 104                  * HSTMEM ALLOCATES MEMORY TO THE HISTOGRAMS IN BLOCKS
 105                  *    FROM THE HIGHEST AVAILABLE MEMORY (PASSED IN HL)
 106                  *    DOWN.  NUMBER OF ALARM CARDS INSTALLED IN REG. C.
 107                  *       REGISTERS AFFECTED: A, B, C, H, L
 108                  *       MEMORY AFFECTED: STATUS, OCCUR, TIMEON, TOTTIM
 109                  *       SUBROUTINES CALLED: DBLSUB
 110                  *
 111  093C  06 00     HSTMEM MVI B,0           CLEAR MSB
 112  093E  79               MOV A,C           GET CARDCOUNT
 113  093F  E6 0F            ANI '0F           LIMIT TO 15 CARDS
 114  0941  32 FF 0F         STA NCARDS        SAVE NUMBER OF CARDS CONFIGURE
 115  0944  07               RLC               MULTIPLY BY 2
 116  0945  4F               MOV C,A           SAVE BLOCKSIZE (2 BYTES/CARD)
 117  0946  21 F4 0F         LXI H,HSTTOP     LOAD BOUNDARY VALUE
 118  0949  CD 71 09         CALL DBLSUB       GENERATE BASE OF BLOCK
```

```
LINE  LOC   B1 B2 B3            SOURCE LINE
 119  094C  22 FD 0F          SHLD STATUS        SAVE ADDRESS
 120  094F  AF                XRA A              CLEAR CARRY
 121  0950  07                RLC                MULTIPLY BY 8
 122  0951  07                RLC                   NOW IS NUMBER CARDS X 16
 123  0952  07                RLC                      FOR 16 BYTES/CARD
 124  0953  4F                MOV C,A            SAVE BLOCKSIZE
 125  0954  CD 71 09          CALL DBLSUB        NEXT TABLE BASE
 126  0957  22 F9 0F          SHLD OCCUR         SAVE ADDRESS
 127  095A  79                MOV A,C            GET BLOCKSIZE
 128  095B  07                RLC                TIMES 2 FOR CARDS X 32
 129  095C  D2 60 09          JNC $+4            BUMP MSB ONLY IF ACC
 130  095F  04                INR B                 OVERFLOWED (IE CARDS > 8)
 131  0960  4F                MOV C,A            SAVE BLOCKSIZE (LSB)
 132  0961  CD 71 09          CALL DBLSUB        FIND BASE OF BLOCK
 133  0964  22 FB 0F          SHLD TIMEON        SAVE ADDRESS
 134  0967  CD 71 09          CALL DBLSUB        LAST BLOCK
 135  096A  22 F7 0F          SHLD TOTTIM        SAVE IT
 136  096D  CD 78 09          CALL HSTCLR+3      ZERO ALL NEW HISTOGRAMS
 137  0970  C9                RET                WITH MEMORY CONFIGURED PER MAP
 139                   ***********************************************************
 140                   *
 141                   * DBLSUB   HL = HL - BC
 142                   *     REGISTERS AFFECTED: A, H, L
 143                   *
 144  0971  7D         DBLSUB MOV A,L            GET LSB
 145  0972  91                SUB C              SUBTRACT LSB
 146  0973  6F                MOV L,A            SAVE IT
 147  0974  7C                MOV A,H            GET MSB
 148  0975  98                SBB B              INCLUDE BORROW FROM LSB
 149  0976  67                MOV H,A            SAVE IT
 150  0977  C9                RET
 152                   ***********************************************************
 153                   *
 154                   * HSTCLR ZEROES ALL HISTOGRAM ENTRIES. CALLED BY
 155                   *     HSTMEM (INITIALIZATION) OR ON COMMAND.
 156                   *        REGISTERS AFFECTED: A, B, C, H, L
 157                   *        MEMORY AFFECTED: ALL HISTOGRAM STORAGE
 158                   *
 159  0978  21 F7 0F   HSTCLR LXI H,TOTTIM       LOWEST ADDRESS IN HISTOGRAM
 160  097B  3A FF 0F          LDA NCARDS         GET NUMBER OF CARDS INSTALLED
 161  097E  47                MOV B,A            MAKE IT A COUNTER
 162  097F  AF                XRA A              CLEAR ACC
 163  0980  0E 52      CLRH1  MVI C,82           NUMBER OF BYTES OF HIST/CARD
 164  0982  77                MOV M,A            ZERO MEMORY LOCATION
 165  0983  23                INX H              BUMP POINTER UP
 166  0984  0D                DCR C                 AND BYTE COUNTER DOWN
 167  0985  C2 82 09          JNZ CLRH1+2        SMALL LOOP
 168  0988  05                DCR B              BUMP CARD COUNTER
 169  0989  C2 80 09          JNZ CLRH1          DO ALL CARDS
 170  098C  C9                RET
```

```
LINE  LOC   B1 B2 B3         SOURCE LINE
 172                  ******************************************************
 173                  *
 174                  * SCAN IS A SHORT LOOP THAT COMPARES THE PRESENT STATE
 175                  *    OF AN ALARM TO THE STATUS LAST TIME IS WAS
 176                  *    CHECKED. ANY CHANGES DETECTED REQUIRE UPDATING
 177                  *    THE STATUS TABLE AND HISTOGRAM DATA.
 178                  *
 179  098D  FB        SCAN   EI               WINDOW FOR GPIA
 180  098E  F3               DI                  TO INTERRUPT
 181  098F  21 E0 7F         LXI H,ALARMS     ALARM POINTER
 182  0992  11 FD 0F         LXI D,STATUS     STATUS TABLE POINTER
 183  0995  01 00 00         LXI B,0000       CLEAR WORD & BIT CTRS
 184                  *                       START SCAN LOOP
 185  0998  1A        CHECK  LDAX D           GET STATUS WORD
 186  0999  AE               XRA M            CHECK FOR STATUS CHANGE
 187  099A  C2 AC 09         JNZ UPDATE       PROCESS IF CHANGE OCCURRED
 188  099D  23               INX H            UPDATE
 189  099E  13               INX D               POINTERS
 190  099F  0C               INR C            INCREMENT WORD COUNTER
 191  09A0  79               MOV A,C          GET WORD COUNTER
 192  09A1  3A FF 0F         LDA NCARDS       NUMBER OF CARDS INSTALLED
 193  09A4  07               RLC              ADJUST FOR TWO WORDS/CARD
 194  09A5  B9               CMP C            CHECK IF DONE
 195  09A6  CA 8D 09         JZ SCAN          REINTIALIZE PTRS IF DONE
 196  09A9  C3 98 09         JMP CHECK        LOOP BACK AND SCAN
 198                  ******************************************************
 199                  *
 200                  * UPDATE HANDLES THE THE LOWEST ORDER BIT CHANGED
 201                  *    OF THE STATUS WORD. MULTIPLE CHANGES IN A STATUS
 202                  *    WORD ARE HANDLED BY THE ROUTINE JUMPING BACK TO
 203                  *    "CHECK" SO THE POINTERS HAVE NOT BEEN INCREMENTED
 204                  *    THIS SECTION EXPECTS:
 205                  *       ALARMS EXCLUSIVE-OR STATUS IN ACCUMULATOR
 206                  *       REG DE POINTING TO STATUS WORD
 207                  *
 208  09AC  E5        UPDATE PUSH H           SAVE SCAN
 209  09AD  D5               PUSH D              STATUS BEFORE UPDATE
 210  09AE  67               MOV H,A          COPY XRA'D STATUS WORD
 211  09AF  AF               XRA A            CLEAR ACC
 212  09B0  47               MOV B,A          ZERO COUNTER
 213  09B1  37               STC              SET CARRY BIT
 214  09B2  17               RAL                 AND GET IT INTO ACC
 215  09B3  6F               MOV L,A          SAVE COPY OF ACC
 216  09B4  A4        UPLOOP ANA H            AND IT WITH STATUS WORD
 217  09B5  C2 BF 09         JNZ UPSTOR       EXIT LOOP WHEN SET BIT FOUND
 218  09B8  7D               MOV A,L          COPY BACK ACC
 219  09B9  07               RLC              ROTATE ACC AND
 220  09BA  04               INR B               UPDATE COUNTER
 221  09BB  6F               MOV L,A          SAVE COPY OF MASK
 222  09BC  C3 B4 09         JMP UPLOOP       LOOP UNTIL MATCH FOUND
```

```
LINE  LOC   B1 B2 B3            SOURCE LINE

 223                   *                     MODIFY AND STORE NEW STATUS
 224  09BF  C5         UPSTOR PUSH B         SAVE COUNTERS
 225  09C0  1A                LDAX D         GET STATUS WORD
 226  09C1  4F                MOV C,A        SAVE STATUS WORD
 227  09C2  A4                ANA H          AND IN XRA'D BYTE
 228  09C3  A5                ANA L          AND IN MASK
 229  09C4  C2 D2 09          JNZ UPDRST     BIT CHANGE WAS A RESET
 230                   *                        OTHERWISE IT WAS SET
 231  09C7  79         UPDSET MOV A,C        COPY OF STATUS
 232  09C8  B5                ORA L          OR IN MASK BIT
 233  09C9  12                STAX D         RETURN UPDATED BYTE TO TABLE
 234  09CA  3E 01             MVI A,1        SET CHANGE CODE TO 1
 235  09CC  32 F5 0F          STA TYPCHG     SAVE CODE
 236  09CF  C3 DA 09          JMP HSTGRM     UPDATE HISTOGRAM DATA
 237  09D2  7D         UPDRST MOV A,L        GET MASK
 238  09D3  2F                CMA            COMPLEMENT IT
 239  09D4  A1                ANA C          AND IN RESET BIT
 240  09D5  12                STAX D         RETURN UPDATED BYTE TO TABLE
 241  09D6  AF                XRA A          CLEAR ACC
 242  09D7  32 F5 0F          STA TYPCHG     SAVE TYPE OF CHANGE
 244                   ***************************************************************
 245                   *
 246                   * HSTGRM HANDLES UPDATES NEEDED TO THE HISTOGRAM
 247                   *      WHEN AN ALARM CHANGES STATE BY USING HSTSET
 248                   *      OR HSTRST DEPENDING ON WHETHER THE ALARM TURNED
 249                   *      ON OR OFF.
 250                   *
 251  09DA  C1         HSTGRM POP B          RECOVER A COPY
 252  09DB  C5                PUSH B            OF COUNTERS
 253  09DC  79                MOV A,C        ROTATE WORD CNT
 254  09DD  07                RLC            ROTATE
 255  09DE  07                RLC              LEFT
 256  09DF  07                RLC                THREE BITS
 257  09E0  B0                ORA B          OR IN BIT CNTR
 258  09E1  07                RLC            ACC NOW HAS RELATIVE ADDRESS
 259                   *                        INTO TWO BYTE (PER ENTRY)
 260                   *                        TABLE
 261  09E2  5F                MOV E,A        MOVE ACC TO
 262  09E3  16 00             MVI D,00          TO REG PAIR
 263  09E5  21 FB 0F          LXI H,TIMEON   LOAD TABLE BASE
 264  09E8  19                DAD D          HL NOW HAS ABSOLUTE
 265                   *                        ADDRESS INTO TABLE
 266  09E9  3A F5 0F          LDA TYPCHG     GET TYPE OF CHANGE
 267  09EC  A7                ANA A          SET FLAGS
 268  09ED  CA 0C 0A          JZ  HSTRST     GO TO APPROPRIATE SECTION
 269                   *                     PROGRAM FALLS THROUGH TO HERE
 270                   *                     IF TYPCHG = 1
```

```
LINE  LOC   B1 B2 B3        SOURCE LINE

 272                  *******************************************************
 273                  *
 274                  * HSTSET PROCESSES CHANGES TO THE HISTOGRAM WHEN THE
 275                  *      ALARM IS SET (TURNED ON).
 276                  *      THE ACTION REQUIRED IS:
 277                  *        1) THE TIMEON WORD FOR THAT ALARM IS SET
 278                  *           TO THE CURRENT TIME
 279                  *        2) THE OCCURANCE COUNT FOR THAT ALARM IS
 280                  *           INCREMENTED.
 281                  *
 282  09F0  D5        HSTSET PUSH D             SAVE BYTES
 283  09F1  11 FE 09         LXI D,TIME         POITER TO TIME STORED
 284                  *                            BY RD8253 ROUTINE
 285  09F4  CD 26 0C         CALL RD8253        GET CURRENT TIME
 286  09F7  1A               LDAX D             GET LSB OF TIME
 287  09F8  77               MOV M,A            STORE IN TIME-ON TABLE
 288  09F9  13               INX D              GET AND
 289  09FA  23               INX H                STORE
 290  09FB  1A               LDAX D                 MSB OF
 291  09FC  77               MOV M,A                  TIME
 292  09FD  D1               POP D              RECOVER BYTES
 293  09FE  7B               MOV A,E            RELATIVE ADDRESS IN
 294  09FF  0F               RRC                   2 BYTE TABLE SHIFTED TO
 295  0A00  5F               MOV E,A               USE IN ONE BYTE TABLE
 296  0A01  21 F9 0F         LXI H,OCCUR        BASE OF OCCURANCE TABLE
 297  0A04  19               DAD D              GENERATE ABSOLUTE ADDRESS
 298  0A05  34               INR M              INCREMENT ENTRY IN TABLE
 299  0A06  C1               POP B              RECOVER
 300  0A07  D1               POP D                 SCANNING
 301  0A08  E1               POP H                    STATUS
 302  0A09  C3 98 09         JMP CHECK          END OF PROCESSING FOR
 303                  *                            THIS CHANGE OF ALARM STATE
 305                  *******************************************************
 306                  *
 307                  * HSTRST PROCESSES THE CHANGES TO THE HISTOGRAM WHEN
 308                  *     THE ALARM RESETS (TURNS OFF).
 309                  *     WHEN THE ALARM RESETS,
 310                  *       1) THE CURRENT TIME IS SUBTRACTED FROM TIMEON
 311                  *          (SINCE IT IS A DOWN COUNTER) AND THE
 312                  *          DIFFERENCE IS ADDED TO TOTTIM.
 313                  *       2) TIMEON IS CLEARED.
 314                  *
 315  0A0C  CD 26 0C  HSTRST CALL RD8253        LOAD CURRENT TIME
 316  0A0F  D5               PUSH D             SAVE RELATIVE POINTER
 317  0A10  11 FE 09         LXI D,TIME         SET POINTER TO TIME
 318  0A13  1A               LDAX D             GET LSB OF TIME
 319  0A14  4F               MOV C,A            SAVE IT
 320  0A15  7E               MOV A,M            GET LSB OF TIMEON
 321  0A16  91               SUB C              FIND DIFFERENCE
```

```
LINE  LOC   B1 B2 B3           SOURCE LINE

 322  0A17  4F                 MOV C,A      SAVE DIFFERENCE
 323  0A18  13                 INX D        POINT TO MSB OF TIME
 324  0A19  23                 INX H        POINT TO MSB OF TIMEON
 325  0A1A  1A                 LDAX D       GET MSB OF TIME
 326  0A1B  47                 MOV B,A      SAVE IT
 327  0A1C  7E                 MOV A,M      GET MSB OF TIME ON
 328  0A1D  98                 SBB B        FIND DIFFERENCE INCLUDING
 329                    *                      BORROW FROM FIRST SUBTRACT
 330  0A1E  47                 MOV B,A      SAVE DIFF
 331                    *                   B,C NOW HAS TWO BYTE DIFFERENC
 332                    *                       INDICATING TIME ALARM WAS O
 333  0A1F  D1                 POP D        RECOVER RELATIVE ADDRESS
 334  0A20  AF                 XRA A        CLEAR ACC
 335  0A21  77                 MOV M,A      CLEAR MSB OF TIMEON
 336  0A22  2B                 DCX H        CLEAR
 337  0A23  77                 MOV M,A         LSB
 338  0A24  21 F7 0F           LXI H,TOTTIM POINTR TO TOTAL-TIME-ON
 339                    *                      HISTOGRAM
 340  0A27  19                 DAD D        ADD RELATIVE ADDRESS
 341  0A28  7E                 MOV A,M      GET LSB FROM HISTOGRAM
 342  0A29  81                 ADD C        ADD NEW DATA IN
 343  0A2A  77                 MOV M,A      SAVE UPDATED TIME
 344  0A2B  23                 INX H        POINT TO MSB OF HISTOGRAM
 345  0A2C  7E                 MOV A,M      GET MSB FROM HISTOGRAM
 346  0A2D  88                 ADC B        ADD NEW DATA
 347  0A2E  77                 MOV M,A      SAVE MSB
 348  0A2F  C1                 POP B        RECOVER
 349  0A30  D1                 POP D           SCANNING
 350  0A31  E1                 POP H              STATUS
 351  0A32  C3 98 09           JMP CHECK    DONE PROCESSING TIMING
 352                    *                      ALARM
 354                    *************************************************************
 355                    *
 356                    * HSDMDR IS THE HISTOGRAM DUMP DRIVER. A NUMBER PULLED
 357                    *    OFF THE INPUT DEVICE SPECIFIES THE HISTOGRAM TO
 358                    *    BE DUMPED. A NUMBER > NUMBER OF ALARMS INSTALLED
 359                    *    DUMPS ALL HISTOGRAMS.
 360                    *       REGISTERS AFFFECTED: ALL
 361                    *       SUBROUTINES CALLED: HSTDMP
 362                    *
 363  0A35  CD 68 0C    HSDMDR CALL RD1BHX  GET ALARM NUMBER
 364  0A38  4F                 MOV C,A      SAVE IT
 365  0A39  3A FF 0F           LDA NCARDS   GET NUMBER OF CARDS
 366  0A3C  07                 RLC          MULTIPLY BY 16
 367  0A3D  07                 RLC             TO GET NUMBER
 368  0A3E  07                 RLC                OF ALARMS
 369  0A3F  07                 RLC                   INSTALLED
 370  0A40  3D                 DCR A        ADJUST TO START COUNT AT ZERO
 371  0A41  B9                 CMP C        CHECK IF INPUT IN VALID RANGE
 372  0A42  DA 49 0A           JC HSDMAL    OUT OF RANGE, DUMP ALL
```

```
LINE  LOC   B1 B2 B3            SOURCE LINE

 373  0A45  CD 54 0A          CALL HSTDMP       OTHERWISE, DUMP ONE
 374  0A48  C9                RET
 375  0A49  4F        HSDMAL  MOV C,A           SAVE NUMBER OF ALARMS
 376  0A4A  C5                PUSH B            SAVE REG
 377  0A4B  CD 54 0A          CALL HSTDMP       DUMP THAT ONE
 378  0A4E  C1                POP B             RECOVER ALARM NUMBER
 379  0A4F  0D                DCR C             BUMP COUNTER
 380  0A50  F2 4A 0A          JP HSDMAL+1       LOOP UNTIL ALL ALARMS DONE
 381  0A53  C9                RET
 382                  ***************************************************
 383                  *
 384                  * HSTDMP DUMPS HISTOGRAM DATA FOR THE ALARM DESIGNATED
 385                  *     BY THE C REGISTER.
 386                  *        REGISTER AFFECTED: ALL
 387                  *        SUBROUTINES CALLED: WR1BI3, WR1BI2, DIVIDE
 388                  *
 389  0A54  06 00     HSTDMP  MVI B,0           CLEAR B BEFORE PUSH
 390  0A56  C5                PUSH B            SAVE ALARM NUMBER IN C
 391  0A57  CD C0 0C          CALL WR1BI3        OUTPUT ALARM NUMBER
 392  0A5A  C1                POP B             RECOVER
 393  0A5B  C5                PUSH B               ALARM NUMBER
 394  0A5C  79                MOV A,C           GET ALARM NUMBER
 395  0A5D  E6 07             ANI #00000111     MASK OFF 3 LSB
 396  0A5F  47                MOV B,A           SAVE BIT NUMBER
 397  0A60  79                MOV A,C           GET ALARM NUMBER
 398  0A61  E6 78             ANI #01111000     MASK OFF WORD NUMBER
 399  0A63  0F                RRC               RIGHT
 400  0A64  0F                RRC                  JUSTIFY
 401  0A65  0F                RRC
 402  0A66  5F                MOV E,A           WORD NUMBER IN LSB
 403  0A67  16 00             MVI D,0           CLEAR MSB
 404  0A69  2A FD 0F          LHLD STATUS       BASE OF STATUS TABLE
 405  0A6C  19                DAD D             GENERATE POINTER INTO TABLE
 406  0A6D  7E                MOV A,M           GET STATUS WORD
 407  0A6E  05                DCR B             BUMP BIT COUNTER
 408  0A6F  0F                RRC               ROTATE STATUS BIT INTO CARRY
 409  0A70  F2 6E 0A          JP $-2            LOOP UNTIL COUNTER UNDERFLOWS
 410  0A73  3E 30             MVI A,"0"         0 = ALARM INACTIVE
 411  0A75  D2 79 0A          JNC $+4           IF CARRY NOT SET, I.E. ALARM
 412  0A78  3C                INR A                INACTIVE,LEAVE 0 OTHERWISE
 413                  *                            CHANGE TO 1
 414  0A79  CD EA 0A          CALL WRITE        OUTPUT CHARACTER
 415  0A7C  C1                POP B             ANOTHER COPY
 416  0A7D  C5                PUSH B              OF ALARM NUMBER
 417  0A7E  2A F9 0F          LHLD OCCUR        BASE OF OCCURANCE TABLE
 418  0A81  09                DAD B             PTR INTO TABLE
 419  0A82  7E                MOV A,M           GET TABLE ENTRY
 420  0A83  C5                PUSH B            SAVE BC
 421  0A84  CD C0 0C          CALL WR1BI3        OUTPUT IN I3 FORMAT
 422  0A87  C1                POP B             RECOVER
```

```
LINE  LOC   B1 B2 B3        SOURCE LINE

 423  0A88  AF              XRA A         CLEAR CARRY
 424  0A89  79              MOV A,C       MULTIPLY ALARM # BY 2
 425  0A8A  17              RAL              INSURING LSB RESET
 426  0A8B  4F              MOV C,A       RESTORE LSB OF OFFSET
 427  0A8C  2A F7 0F        LHLD TOTTIM   BASE OF TOTAL TIME TABLE
 428  0A8F  09              DAD B         ADD OFSET
 429  0A90  46              MOV B,M       GET MSB FROM TABLE
 430  0A91  23              INX H         BUMP POINTER
 431  0A92  4E              MOV C,M       GET LSB FROM TABLE
 432  0A93  16 64           MVI D,100     DIVISOR
 433  0A95  CD 01 0D        CALL DIVIDE   SEPERATE TIME
 434  0A98  4F              MOV C,A       GET QUOTIENT
 435  0A99  C5              PUSH B        SAVE DATA
 436  0A9A  CD C0 0C        CALL WR1BI3    OUTPUT TIME DIV 100 IN
 437                *                          IN I3 FORMAT
 438  0A9D  C1              POP B         RECOVER
 439  0A9E  48              MOV C,B       GET REMAINDER
 440  0A9F  CD CF 0C        CALL WR1BI2    OUTPUT REMIANDER IN I2 FORMAT
 441  0AA2  CD E3 0C        CALL CRLF     SPECIFY END OF RECORD
 442  0AA5  C9              RET
 444                *************************************************************
 445                *
 446                * EXCUTE READS FOUR CHARACTERS FROM THE BUS AS A START
 447                *    ADDRESS FOR EXECUTION.
 448                *       REGISTERS AFFECTED: A, B, C, H, L
 449                *       SUBROUTINES CALLED: RD1BHX
 450                *
 451  0AA6  CD 73 0C  EXCUTE CALL RD2BHX   GET TRANSFER ADDRESS
 452  0AA9  E9              PCHL          TRANSFER TO THAT ADDRESS
 454                *************************************************************
 455                *
 456                * PLOAD LOADS OBJECT CODE FROM THE ASSEMBLER
 457                *    IN INTEL FORMAT VIA THE INPUT DEVICE.
 458                *       REGISTERS AFFECTED: ALL
 459                *       MEMORY AFFECTED: DEPENDENT ON ASSEMBLER ORG
 460                *       SUBROUTINES CALLED: RD1BHX, RD2BHX
 461                *
 462  0AAA  CD 19 0B  PLOAD  CALL READ     GET CHARACTER OFF THE BUS
 463  0AAD  FE 3A           CPI ":"       CHECK FOR COLON INDICATING
 464                *                         BEGINNING OF RECORD
 465  0AAF  C2 AA 0A        JNZ PLOAD     LOOP UNTIL BEGIN RECORD
 466  0AB2  CD 68 0C        CALL RD1BHX   GET RECORD LENGTH
 467  0AB5  5F              MOV E,A       SAVE BYTE COUNTER
 468  0AB6  57              MOV D,A       START ON CHECKSUM
 469  0AB7  CD 73 0C        CALL RD2BHX   GET LOAD ADDRESS
 470  0ABA  82              ADD D         ADD CHECKSUM TO LO BYTE
 471  0ABB  84              ADD H         ADD HI BYTE
 472  0ABC  57              MOV D,A       SAVE IT
 473  0ABD  CD 68 0C        CALL RD1BHX   GET RECORD TYPE
 474  0AC0  82              ADD D         ONLY USE FOR CHECKSUM
```

```
LINE  LOC   B1 B2 B3         SOURCE LINE

 475  0AC1  57                MOV D,A           OTHERWISE IGNORED
 476  0AC2  CD 68 0C  PLLOOP CALL RD1BHX      GET PROGRAM WORD
 477  0AC5  77                MOV M,A         STORE IT
 478  0AC6  82                ADD D           UPDATE
 479  0AC7  57                MOV D,A           CHECKSUM
 480  0AC8  23                INX H           BUMP POINTER UP
 481  0AC9  1D                DCR E             AND BYTE CNTR DOWN
 482  0ACA  C2 C2 0A          JNZ PLLOOP      LOOP UNTIL RECORD READ
 483  0ACD  CD 68 0C          CALL RD1BHX     READ CHECKSUM FROM RECORD
 484  0AD0  82                ADD D           FINAL ADD - SHOULD BE ZERO
 485  0AD1  C8                RZ              DONE IF CHECKSUM OK
 486  0AD2  3E 01             MVI A,01        SET ERROR CODE
 487                  *       CALL ERROR      OUTPUT MESSAGE
 488  0AD4  C9                RET             VERIFIED 22 AUG 78
 489                  GPIA    EQU 'BFC8       HARDWIRED ADDRESS
 490                  *                         OF GPIA REGISTER 0
 492                  ****************************************************
 493                  *                                                  *
 494                  *     G P I A   T A L K E R   R O U T I N E S      *
 495                  *                                                  *
 496                  ****************************************************
 498                  *
 499                  *GPTALK WILL CONFIGURE THE GPIA TO BE A TLAKER
 500                  *       REGISTERS AFFECTED: A
 501                  *
 502  0AD5  21 CB BF  GPTALK LXI H,GPIA+3     AUX COMMAND REGISTER
 503  0AD8  36 80             MVI M,'80       STROBE RESET BIT
 504  0ADA  36 00             MVI M,'00         TO POWER ON CONDITION
 505  0ADC  23                INX H           ADDRESS SWITCH REGISTER
 506  0ADD  7E                MOV A,M         READ SWITCHES
 507  0ADE  77                MOV M,A         WRITE TO ADDRESS SWITCH REGIST
 508  0ADF  3E 00             MVI A,0         CLEAR INTERRUPT MASK
 509  0AE1  32 C8 BF          STA GPIA          IN MASK REGISTER
 510  0AE4  3E 40             MVI A,'40       TALKER ONLY ADDRESS MODE
 511  0AE6  32 CA BF          STA GPIA+2        IN ADDRESS MODE REGISTER
 512  0AE9  C9                RET
 514                  ****************************************************
 515                  *
 516                  *GTTRAN TRANSMITS THE CHARACTER IN THE ACC VIA THE
 517                  *   GPIB.  NOT USED FOR LAST CHAR OF STRING.
 518                  *       REGISTERS AFFECTED: A
 519                  *
 520  0AEA  F5        GTTRAN PUSH PSW         SAVE DATA TO BE OUTPUT
 521  0AEB  3A C8 BF          LDA GPIA        READ INTERRUPT STATUS REGISTER
 522  0AEE  E6 40             ANI '40         MASK OFF BIT 0 (B0)
 523  0AF0  CA EB 0A          JZ  GTTRAN+1    CONITNUE LOOP UNTIL B0 GOES HI
 524  0AF3  F1                POP PSW         GET DATA
 525  0AF4  32 CF BF          STA GPIA+7        AND WRITE TO DATA OUT REG
 526  0AF7  C9                RET
```

```
LINE  LOC   B1 B2 B3           SOURCE LINE

 528                  ******************************************************
 529                  *
 530                  *GTLAST TRANSMITS A CHARACTER FROM THE A REG. USED
 531                  *    ONLY FOR THE LAST CHARACTER OF A STRING.
 532                  *      REGISTERS AFFECTED: A
 533                  *
 534  0AF8  F5        GTLAST PUSH PSW        SAVE DATA
 535  0AF9  3E 20            MVI A,'20       SET EOI BIT
 536  0AFB  32 CB BF         STA GPIA+3      STORE IN AUX COMMAND
 537                  *                         REG PRIOR TO LAST CHAR
 538  0AFE  F1               POP PSW         GET CHARACTER
 539  0AFF  32 CF BF         STA GPIA+7      STORE IN DATA OUT REG
 540  0B02  C9               RET
 542                  ******************************************************
 543                  *                                                    *
 544                  *      G P I A   L I S T E N E R   R O U T I N E S   *
 545                  *                                                    *
 546                  ******************************************************
 547                  *
 548                  *GPLIST CONFIGURES THE GPIA TO BE A LISTENER
 549                  *      REGISTERS AFFECTED: A
 550                  *
 551  0B03  21 CB BF  GPLIST LXI H,GPIA+3    AUX COMMAND REGISTER
 552  0B06  36 80            MVI M,'80       STROBE RESET BIT
 553  0B08  36 00            MVI M,'00
 554  0B0A  23               INX H           ADDRESS SWITCH REGISTER
 555  0B0B  7E               MOV A,M         READ ADDRESS SWITCHES
 556  0B0C  77               MOV M,A         WRITE TO ADDRESS REGISTER
 557  0B0D  3E 81            MVI A,'81       INTERRUPT ON BYTE RECEIVED
 558  0B0F  32 C8 BF         STA GPIA           IN MASK REGISTER
 559  0B12  3E 20            MVI A,'20       LISTENER ONLY ADDRESS MODE
 560  0B14  32 CA BF         STA GPIA+2         IN ADDRESS MODE REG
 561  0B17  FB               EI              GPIA WILL INTERRUPT
 562                  *                         WHEN READY
 563  0B18  C9               RET
 565                  ******************************************************
 566                  *
 567                  *GLREC RECEIVES ON CHARACTER FROM THE GPIB AND
 568                  *     STORES IT IN THE ACCUMULATOR.  RETURNING
 569                  *     WITH THE CARRY SET INDICATES EOI WAS RECIEVED
 570                  *     AND THUS IS THE END OF THE INPUT STRING.
 571                  *        REGISTERS AFFECTED: A
 572                  *
 573  0B19  C5        GLREC  PUSH B          MAKE REGISTER AVAILABLE
 574  0B1A  06 40            MVI B,'40       SET A DELAY TIME
 575  0B1C  05               DCR B           BUMP COUNTER
 576  0B1D  C2 1C 0B         JNZ $-1         LOOP ON DELAY
 577  0B20  C1               POP B           DONE WITH DELAY
 578  0B21  3A C8 BF  GLSTAT LDA GPIA        READ INTERRUPT STATUS REG
```

```
LINE  LOC   B1 B2 B3          SOURCE LINE

 579  0824  E6 01              ANI 01             MASK OFF BIT 0 (BI)
 580  0826  CA 19 0B           JZ GLREC           CONTINUE CHECKING UNTIL
 581                   *                            BI GOES HIGH
 582  0829  E6 02              ANI '02            MASK OFF BIT 1 (EOI)
 583  082B  37                 STC                SET CARRY FLAG
 584  082C  CA 30 0B           JZ GLSKIP          EOI NOT SENT,LEAVE CARRY SET
 585  082F  3F                 CMC                RESET CARRY WHEN EOI IS SENT
 586  0830  3A CF BF   GLSKIP LDA GPIA+7          GET DATA IN
 587  0833  FE 0A              CPI '0A            LINE FEED ALSO SIGNALS
 588  0835  C0                 RNZ                   END OF RECORD
 589  0836  37                 STC                      SO SET CARRY
 590  0837  C9                 RET
 591                   ********************************************************
 592                   *
 593                   *  TELETYPE TERMINAL DRIVERS.  USED AS THE TERMINAL
 594                   *  DEVICE IN THE LOCAL MODE.
 595                   *
 596                   ********************************************************
 597                   CTLPRT EQU 'C              UART CONTROL PORT
 598                   BDRTPT EQU 'B              BAUD RATE GENERATOR PORT
 599                   CHSLPT EQU '9              CHANNEL SELECT PORT
 600                   USTAT  EQU '8              UART STATUS PORT
 601                   DATAIN EQU '9              DATA INPUT BUFFER
 602                   DATOUT EQU 'A              DATA OUTPUT BUFFER
 604                   ********************************************************
 605                   *
 606                   * ASNIO ASIGNS THE OUTPUT DEVICE ON THE SERIAL IO
 607                   *    CARD BY SELECTING THE PROPER DATA FORMAT, BAUD
 608                   *    RATE AND CHANNEL. ON ENTRY, HL MUST POINT TO THE
 609                   *    APPROPRIATE TABLE ENTRY.
 610                   *       REGISTERS AFFECTED: A, H, L
 611                   *
 612                   *
 613  0838  7E         ASNIO  MOV A,M           GET UART CCMMAND
 614  0839  D3 0C              OUT CTLPRT
 615  083B  23                 INX H
 616  083C  7E                 MOV A,M           GET BAUD RATE
 617  083D  D3 0B              OUT BDRTPT
 618  083F  23                 INX H
 619  0840  7E                 MOV A,M           GET CHANNEL NUMBER
 620  0841  D3 09              OUT CHSLPT
 621  0843  C9                 RET
 622  0844  18         TTY     DB #11000           UART COMMAND
 623  0845  03                 DB '03              BAUD RATE
 624  0846  02                 DB 2                CHANNEL SELECT
 625  0847  00         CASETT DB 0
 626  0848  00                 DB 0
 627  0849  00                 DB 0
```

```
LINE LOC   B1 B2 B3       SOURCE LINE

 629                  ******************************************************
 630                  *
 631                  * READS INPUTS DATA FROM THE SERIAL IO CARD RETURNING
 632                  *    WITH THE INPUT CHARACTER IN THE ACC.
 633                  *       REGISTERS AFFECTED: A
 634                  *
 635 0B4A DB 08      READS  IN USTAT
 636 0B4C E6 0E             ANI #1110       ERROR MASK
 637 0B4E CA 56 0B          JZ RNOERR
 638 0B51 3E 02             MVI A,2         SET ERROR CODE
 639 0B53 CD DE 0B          CALL ERROR
 640 0B56 DB 08      RNOERR IN USTAT        GET STATUS
 641 0B58 E6 10             ANI '10         MASK DATA AVAILABLE
 642 0B5A CA 4A 0B          JZ READS        LOOP UNTIL UART RECIEVES
 643 0B5D DB 09             IN DATAIN       GET DATA
 644 0B5F C9                RET
 646                  ******************************************************
 647                  *
 648                  * WRITES OUPTUS A CHARACTER TO THE SERIAL IO CARD
 649                  *    FROM THE ACC.
 650                  *       REGISTERS AFFECTED: NONE
 651                  *
 652 0B60 F5         WRITES PUSH PSW        SAVE OUTPUT DATA
 653 0B61 DB 08             IN USTAT
 654 0B63 E6 01             ANI 1           MASK TBMT
 655 0B65 CA 61 0B          JZ WRITES+1     LOOP UNTIL BUFFER EMPTY
 656 0B68 F1                POP PSW         RECOVER DATA
 657 0B69 D3 0A             OUT DATOUT      FILL BUFFER
 658 0B6B C9                RET
 660                  ******************************************************
 661                  *
 662                  * DMPASC DUMPS AN ASCII STRING OF LENGTH (HL) STARTING
 663                  *    AT HL+1
 664                  *       REGISTERS AFFECTED: A,.....E, H, L
 665                  *       SUBROUTINES CALLED: WRITE
 666                  *
 667 0B6C 5E         DMPASC MOV E,M          GET LENGTH
 668 0B6D 23                INX H            POINT TO CHARACTER
 669 0B6E 1D                DCR E            BUMP CHARACATER COUNTER
 670 0B6F F8                RM               DONE WHEN CNTR UNDERFLOWS
 671 0B70 7E                MOV A,M          GET CHARACTER
 672 0B71 CD EA 0A          CALL WRITE       OUTPUT IT
 673 0B74 C3 6D 0B          JMP DMPASC+1
```

```
LINE  LOC   B1 B2 B3          SOURCE LINE

                      *************************************************
                      *
                      * BLKDMP DUMPS THE CONTENTS OF MEMORY STARTING AT
                      *    (HL) AND ENDING AT (DE) TO THE OUTPUT DEVICE.
                      *       REGISTERS AFFECTED: ALL
                      *       SUBROUTINES CALLED: RD4BHX, WR1BHX, WRITE
                      *
      0B77  CD 7C 0C  BLKDMP CALL RD4BHX        GET START AND END ADDRESS
      0B7A  CD 63 0C         CALL CMPHND        CHECK IF DONE
      0B7D  D0               RNC
      0B7E  7D               MOV A,L            GET LSB OF ADDRESS
      0B7F  E6 0F            ANI '0F            CHECK 4 LS BITS OF LSB FOR ZER
      0B81  C2 8A 0B         JNZ BKDP1          SKIP CRLF AND WR2BHX IF NOT
      0B84  CD E3 0C         CALL CRLF          SPACE LISTING
      0B87  CD B7 0C         CALL WR2BHX        SHOW CURRENT LOCATION
      0B8A  7E        BKDP1  MOV A,M            GET DATA
      0B8B  CD A4 0C         CALL WR1BHX        OUTPUT DATA
      0B8E  3E 20            MVI A," "          SPACE BETWEEN
      0B90  CD EA 0A         CALL WRITE            DATA ITEMS
      0B93  23               INX H              BUMP POINTER
      0B94  C3 7A 0B         JMP BLKDMP+3
                      *************************************************
                      *
                      * SEQFIL IS USED TO FILL OR EXAMINE A BLOCK OF MEMORY.
                      *       REGISTERS AFFECTED: A, B, C, H, L
                      *       SUBROUTINES CALLED: RD2BHX, WR2BHX,
                      *                               CRLF, MEMACC
                      *
      0B97  CD 73 0C  SEQFIL CALL RD2BHX        GET BEGIN ADDRESS
      0B9A  CD E3 0C         CALL CRLF          NEXT LIST LINE
      0B9D  CD B7 0C         CALL WR2BHX        OUTPUT BEGIN ADDRESS
      0BA0  3E 20     SQFIL1 MVI A," "          SPACE FOR READABILITY
      0BA2  CD EA 0A         CALL WRITE
      0BA5  CD B2 0B         CALL MEMACC        LOOK AT THAT LOCATION
      0BA8  D8               RC                 DONE IF MEMACC SETS CARRY
      0BA9  7D               MOV A,L            CHECK 4 LSB
      0BAA  E6 0F            ANI '0F               FOR ZERO I.E. XXX0
      0BAC  CA 9A 0B         JZ SEQFIL+3        IF SO NEW LINE
      0BAF  C3 A0 0B         JMP SQFIL1         OTHERWISE LOOP
                      *************************************************
                      *
                      * MEMACC DISPLAYS A MEMORY LOCATION AND MODIFIES IT IF
                      *    DESIRED.  COMMANDS: TWO HEX CHARACTERS, REPLACE
                      *    DATA; SPACE, NO CHANGE; X, CHANGE PREVIOUS DATA;
                      *    Z, END ROUTINE; OTHERS INVALID. CARRY SET = STOP.
                      *       REGISTERS AFFECTED: A, B, C, H, L
                      *       SUBROUTINES CALLED: WR1BHX, WRITE, READ, RD1BH
                      *
      0BB2  7E        MEMACC MOV A,M            GET DATA
```

```
LINE  LOC   B1 B2 B3         SOURCE LINE

 726  0BB3  CD A4 0C         CALL WR1BHX       DISPLAY IT
 727  0BB6  CD 19 0B         CALL READ         GET COMMAND
 728  0BB9  FE 20            CPI " "           SPACE ?
 729  0BBB  CA DB 0B         JZ MEMAC1
 730  0BBE  FE 58            CPI "X"           BACKSPACE COMMAND ?
 731  0BC0  C2 C6 0B         JNZ $+6
 732  0BC3  2B               DCX H             BACKSPACE
 733  0BC4  AF               XRA A             CLEAR CARRY
 734  0BC5  C9               RET               WITH CARRY RESET
 735  0BC6  FE 5A            CPI "Z"           EXIT COMMAND
 736  0BC8  C2 CD 0B         JNZ $+5
 737  0BCB  37               STC               SET END FLAG (CARRY)
 738  0BCC  C9               RET               WITH CARRY SET
 739  0BCD  CD 6B 0C         CALL RD1BHX+3     ALTERNATE ENTRY POINT
 740                   *                          SINCE CHARACTER ALREADY REA
 741  0BD0  D2 DA 0B         JNC MEMAC1-1      RETURNS WITH CARRY IF BAD DATA
 742  0BD3  3E 3F            MVI A,"?"         UNRECOGNIZED COMMAND
 743  0BD5  CD EA 0A         CALL WRITE
 744  0BD8  AF               XRA A             CLEAR CARRY
 745  0BD9  C9               RET
 746  0BDA  77               MOV M,A           STORE NEW DATA
 747  0BDB  23         MEMAC1 INX H             NEXT LOCATION
 748  0BDC  AF               XRA A             CLEAR CARRY
 749  0BDD  C9               RET
 751                   ***************************************************************
 752                   *
 753                   * ERROR PRINTS THE ERROR MESSAGE:
 754                   *               "ERROR XX NEAR LOC XXXX"
 755                   *       REGISTERS AFFECTED: ALL
 756                   *       SUBROUTINES CALLED: DMPASC, WR1BI3, WR2BHX
 757                   *
 758  0BDE  21 F8 0B   ERROR  LXI H,ERRMSG      BASE OF ERROR MESSAGE
 759  0BE1  47               MOV B,A           SAVE ERROR CODE
 760  0BE2  CD E3 0C         CALL CRLF         NEXT LINE
 761  0BE5  CD 6C 0B         CALL DMPASC       OUTPUT "ERROR "
 762  0BE8  78               MOV A,B           GET ERROR CODE
 763  0BE9  CD C0 0C         CALL WR1BI3       WRITE CODE
 764  0BEC  CD 6C 0B         CALL DMPASC       OUTPUT " NEAR LOC "
 765  0BEF  E1               POP H             GET A COPY OF RETURN ADDRESS
 766  0BF0  E5               PUSH H
 767  0BF1  CD B7 0C         CALL WR2BHX       OUTPUT LOC THAT CALLED ERROR
 768  09F4  CD E3 0C         CALL CRLF         NEXT LINE
 769  0BF7  C9               RET
 770  0BF8  06         ERRMSG DB 6              MESSAGE LENGTH
 771  0BF9  45 52            DW "ER"           MESSAGE CHARACTERS
 772  0BFB  52 4F            DW "RO"
 773  0BFD  52 20            DW "R "
 774  0BFF  0A               DB 10             LENGTH
 775  0C00  20 4E            DW " N"
 776  0C02  45 41            DW "EA"
```

```
LINE  LOC   B1 B2 B3           SOURCE LINE

 777  0C04  52 20              DW "R "
 778  0C06  4C 4F              DW "LO"
 779  0C08  43 20              DW "C "
 780                    READ   EQU GLREC
 781                    WRITE  EQU GTTRAN
 782                    CNTR   EQU 'BFC0        HARDWIRED ADDRESS OF COUNTER 0
 783                    TIME   EQU '09FE
 784                    CCTLWD EQU CNTR+3       COUNTER CONTROL WORD
 786                    **********************************************
 787                    *                                            *
 788                    *       8 2 5 3   T I M E R   C O N T R O L  *
 789                    *                                            *
 790                    **********************************************
 791                    *
 792                    * THE 8253 INTERVAL TIMER CONTROL WORD IS DEFINED AS
 793                    *    BITS 7,6  SELECT WHICH COUNTER THIS CONTROL
 794                    *              WORD IS FOR 00, 01, 10 ARE #0, 1, 2
 795                    *              RESPECTIVELY.  11 IS NOT ALLOWED
 796                    *    BITS 5,4  OPERATION MODE: 00 LATCH COUNTER TO
 797                    *              INTERNAL REG; 01 READ/LOAD MSB ONLY
 798                    *              10 R/L LSB ONLY; 11 R/L LSB THEN MSB
 799                    *              CONSECUTIVELY
 800                    *    BITS 3,2,1 COUNTER MODE 000  ALLOWS
 801                    *              NORMAL (DOWN) COUNTING.  MODE 010
 802                    *              CONFIGURES THE 8253 TO A DIVIDE BY N
 803                    *              COUNTER. (DISCUSSED IN MANFUFACTURER'S
 804                    *              DOCUMENTS)
 805                    *    BIT 0     0 SELECTS 16 BIT BINARY,
 806                    *              1 SELECTS 4 DECADES OF BCD
 807                    *
 808                    **********************************************
 810                    *
 811                    *IN8253 INITIALIZES THE 8253 AS FOLLOWS:
 812                    *      CNTR 0, DIVIDE BY 960
 813                    *        CLK 0 = 8080 CLK, OUT 0 = 960 HZ
 814                    *      CNTR 1, DIVIDE BY 960
 815                    *         CLK 1 = OUT 0 = 960 HZ, OUT 1 = 1 HZ
 816                    *      CNTR 2, 16 BIT DOWN COUNTER (CLOCK)
 817                    *         CLK 2 = OUT 1 = 1 HZ
 818                    *             REGISTERS AFFECTED: B, C, H, L
 819                    *
 820  0C0A  21 C3 BF    IN8253 LXI H,CCTLWD     PTR TO CONTROL WORD
 821  0C0D  36 34              MVI M,'34        CNTR 0 MODE 2
 822  0C0F  36 74              MVI M,'74        CNTR 1 MODE 2
 823  0C11  36 B0              MVI M,'B0        CNTR 2 MODE 0
 824  0C13  21 C0 BF           LXI H,CNTR       PTR TO CNTR 0
 825  0C16  01 C0 03           LXI B,960        SET DIVIDE QUANTITY
 826  0C19  71                 MOV M,C          WRITE LSB
 827  0C1A  70                 MOV M,B             AND MSB TO CNTR 0
 828  0C1B  23                 INX H            PTR TO CNTR 1
```

```
LINE  LOC    B1 B2 B3          SOURCE LINE

829   0C1C   71                 MOV M,C            WRITE LSB
830   0C1D   70                 MOV M,B                AND MSB TO CNTR 1
831                     *************************************************************
832                     *
833                     * RS8253 RESETS COUNTER 2 TO MAX COUNT ('FFFF)
834                     *       THIS ALTERNATE ENTRY POINT SHOULD BE USED ONLY
835                     *       RESET THE REAL TIME CLOCK.
836                     *               REGISTERS AFFECTED: H, L
837                     *
838   0C1E   21 C2 BF   RS8253  LXI H,CNTR+2       PTR TO CNTR 2
839   0C21   36 FF              MVI M,'FF          SET MAX COUNT IN LSB AND
840   0C23   36 FF              MVI M,'FF              MSB OF CNTR 2
841   0C25   C9                 RET                CNTR 2 IS DOWN COUNTING
842                     *                              FROM 'FFFF
843                     ***********************************************************
844                     *
845                     *RD8253 READS THE CURRENT TIME FROM THE COUNTER
846                     *       WITH THE RESULTS IN TIME (LSB) AND TIME+1 (MSB)
847                     *               REGISTERS AFFECTED: A, H, L
848                     *               MEMEORY AFFECTED: TIME, TIME+1
849                     *
850   0C26   3E 80      RD8253  MVI A,'80          SELECTS COUNTER 2 TO LATCH
851   0C28   32 C3 BF           STA CCTLWD         WRITE TO CONTROL WORD
852   0C2B   21 C2 BF           LXI H,CNTR+2       PNTR TO CNTR 2
853   0C2E   7E                 MOV A,M            READ LSB
854   0C2F   32 FE 09           STA TIME           STORE LSB
855   0C32   7E                 MOV A,M            READ MSB
856   0C33   32 FF 09           STA TIME+1         STORE MSB
857   0C36   C9                 RET                COUNTER IS STILL COUNTING
859                     *************************************************************
860                     *                                                           *
861                     *     M E N I A L    F U N C T I O N    L I B R A R Y     *
862                     *                                                           *
863                     *************************************************************
865                     *************************************************************
866                     *
867                     * HEX1 CONVERTS ONE ASCII CHARACTER IN THE ACCUMULATOR
868                     *      TO 4 BITS RIGHT JUSTIFIED IN THE ACCUMULATOR
869                     *      RETURNING WITH CARRY SET FOR A CHARACTER OUT OF
870                     *      THE RANGE 0-F.
871                     *          REGISTERS AFFECTED: A
872                     *
873   0C37   D6 30      HEX1    SUI "0"
874   0C39   D8                 RC                 FILTER OUT 00-2F
875   0C3A   C6 E9              ADI "0"-"G"
876   0C3C   D8                 RC                 FILTER OUT 47-FF
877   0C3D   C6 06              ADI 6
878   0C3F   F2 45 0C           JP HEX1A           BRANCH FOR A-F
879   0C42   C6 07              ADI 7
880   0C44   D8                 RC                 FILTER OUT 3A-40
```

```
LINE  LOC   B1 B2 B3         SOURCE LINE

881   0C45  C6 0A     HEX1A  ADI 10
882   0C47  B7               ORA A           CLEAR ERROR FLAG
883   0C48  C9               RET             VERIFIED 18 AUG 78
885                   ****************************************************
886                   *
887                   * HEX2 CONVERTS THE ASCII CHARACTERS IN ACC (LSB) AND
888                   *     B (MSB) TO ONE 8 BIT HEX WORD RETURNED IN THE
889                   *     ACCUMULATOR.
890                   *        REGISTERS AFFECTED: A, C
891                   *        SUBROUTINES CALLED: HEX1
892                   *
893   0C49  CD 37 0C  HEX2   CALL HEX1       CONVERT LSB
894   0C4C  D8               RC              RETURN IF BAD DATA
895   0C4D  4F               MOV C,A         SAVE IT
896   0C4E  78               MOV A,B         GET MSB CHARACTER
897   0C4F  CD 37 0C         CALL HEX1       CONVERT MSB
898   0C52  D8               RC
899   0C53  07               RLC             ROTATE
900   0C54  07               RLC                  FOUR
901   0C55  07               RLC                    BIT
902   0C56  07               RLC                        POSITIONS
903   0C57  B1               ORA C           CONCATONATE LSB
904   0C58  C9               RET             WITH RESULTANT WORD IN ACC
905                   *                      VERIFIED 18 AUG 78
907                   ****************************************************
908                   *
909                   * ASCII CONVERTS THE 4 LSB OF THE ACC TO ONE ASCII
910                   *    CHARACTER 0-9, A-F.
911                   *    REGISTERS AFFECTED: A
912                   *
913   0C59  E6 0F     ASCII  ANI '0F         LIMIT TO 4 LSB
914   0C5B  C6 30            ADI "0"         ADD ASCII OFFSET FOR DIGITS
915   0C5D  FE 39            CPI "9"         CHECK IF IT WAS A DIGIT
916   0C5F  D8               RC              DONE IF DIGIT
917   0C60  C6 07            ADI "A"-"9"-1   OTHERWISE ADD ALPHA OFFSET
918   0C62  C9               RET             VERIFIED 21 AUG 78
920                   ****************************************************
921                   *
922                   * CMPHND COMPARES HL 'N' DE (HENCE THE NAME).
923                   *    CARRY = 1 IF DE < HL OTHERWISE 0
924                   *       REIGSTERS AFFECTED: A
925                   *
926   0C63  7B        CMPHND MOV A,E         GET LSB OF DE
927   0C64  95               SUB L           LSB OF HL
928   0C65  7A               MOV A,D         GET MSB
929   0C66  9C               SBB H           MSB OF HL
930   0C67  C9               RET             VERIFIED 21 AUG 78
```

```
LINE  LOC  B1 B2 B3           SOURCE LINE
 932                  ******************************************************
 933                  *                                                    *
 934                  *    F O R M A T T I N G   R E A D    L I B R A R Y  *
 935                  *                                                    *
 936                  ******************************************************
 938                  ******************************************************
 939                  *
 940                  * RD1BHX GETS TWO CHARACTERS FROM THE INPUT DEVICE AND
 941                  *    RETURNS WITH THE HEX WORD IN THE ACCUMULATOR.
 942                  *       REGISTERS AFFECTED: A, B, C
 943                  *       SUBROUTINES CALLED: READ , HEX2
 944                  *
 945  0C68 CD 19 0B   RD1BHX CALL READ        GET ONE CHARACTERS OFF THE BUS
 946  0C6B 47                MOV B,A          SAVE MSB CHARACTER
 947  0C6C CD 19 0B          CALL READ        GET NEXT CHARACTER
 948  0C6F CD 49 0C          CALL HEX2        CONVERTS TO HEX WORD
 949  0C72 C9                RET              VERIFIED 18 AUG 78
 951                  ******************************************************
 952                  *
 953                  * RD2BHX READS FOUR CHARACTERS AND RETURNS THE TWO
 954                  *    BYTES IN HL.
 955                  *    REGISTERS AFFECTED: A, B, C
 956                  *    SUBROUTINES CALLED: RD1BHX
 957                  *
 958  0C73 CD 68 0C   RD2BHX CALL RD1BHX      GET MSB OF DATA
 959  0C76 67                MOV H,A          SAVE IT
 960  0C77 CD 68 0C          CALL RD1BHX      GET LSB
 961  0C7A 6F                MOV L,A
 962  0C7B C9                RET              VERIFIED 21 AUG 78
 963                  ******************************************************
 964                  *
 965                  * RD4BHX USES EIGHT INPUT CHARACTERS TO FILL THE HL
 966                  *    PAIR, THEN THE DE PAIR.
 967                  *       REGISTERS AFFECTED: ALL
 968                  *       SUBROUTINES CALLED: RD2BHX
 969                  *
 970  0C7C CD 73 0C   RD4BHX CALL RD2BHX      GET HL
 971  0C7F EB                XCHG             SAVE THEM
 972  0C80 CD 73 0C          CALL RD2BHX      GET DE
 973  0C83 EB                XCHG             PUT THEM WHERE THEY SHOULD BE
 974  0C84 C9                RET
 975                  ******************************************************
 976                  *
 977                  * RD1BI3 READS 3 CHARACTERS FROM THE INPUT DEVICE AND
 978                  *    INTERPRETS THESE AS DECIMAL DIGITS TO PRODUCE A
 979                  *    NUMBER BETWEEN 0 AND 255 RETURNED IN THE ACC.
 980                  *       REGISTERS AFFECTED: A, B, C, D, E
 981                  *       SUBROUTINES CALLED: READ, MLTPLY, HEX1
 982                  *
```

```
LINE  LOC   B1 B2 B3        SOURCE LINE

 983  0C85  CD 19 0B  RD1BI3 CALL READ          GET 100'S DIGIT
 984  0C88  CD 37 0C         CALL HEX1          ASCII TO BINARY
 985  0C8B  4F               MOV C,A            MOV TO MULTIPLICAND
 986  0C8C  16 0A            MVI D,10           SET MULTIPLIER
 987  0C8E  CD EE 0C         CALL MLTPLY        C = 10 X HUNDREDS DIGIT (HD)
 988  0C91  CD 19 0B         CALL READ          GET TENS DIGIT
 989  0C94  CD 37 0C         CALL HEX1          CONVERT IT
 990  0C97  81               ADD C              ACC = 10 X HD + TENS DIGIT (TD
 991  0C98  4F               MOV C,A            MOVE TO MULTIPLICAND
 992  0C99  CD EE 0C         CALL MLTPLY        X 10
 993  0C9C  CD 19 0B         CALL READ          GET UNITS DIGIT (UD)
 994  0C9F  CD 37 0C         CALL HEX1
 995  0CA2  81               ADD C              ACC = 100 X HD + 10 X TD + UD
 996  0CA3  C9               RET                VERIFIED 21 AUG 78
 998                  ********************************************************
 999                  *                                                      *
1000                  * F O R M A T T I N G   W R I T E   L I B R A R Y    *
1001                  *                                                      *
1002                  ********************************************************
1004                  ********************************************************
1005                  *
1006                  * WR1BHX OUTPUTS ONE BYTE OF HEX DATA FROM THE ACC
1007                  *    AS TWO CONSECUTIVE ASCII CHARACTERS.
1008                  *       REGISTERS AFFECTED: A, B
1009                  *       SUBROUTINES CALLED: ASCII, WRITE
1010                  *
1011  0CA4  47        WR1BHX MOV B,A            SAVE DATA
1012  0CA5  07               RLC                SWAP
1013  0CA6  07               RLC                   HALF
1014  0CA7  07               RLC                      BYTES
1015  0CA8  07               RLC
1016  0CA9  CD 59 0C         CALL ASCII         CONVERT 4 LSB TO ASCII CHAR
1017  0CAC  CD EA 0A         CALL WRITE         OUTPUT MSB OF DATA
1018  0CAF  78               MOV A,B            GET LSB OF DATA
1019  0CB0  CD 59 0C         CALL ASCII         CONVERT IT
1020  0CB3  CD EA 0A         CALL WRITE         OUTPUT LSB OF DATA
1021  0CB6  C9               RET                VERIFIED 22 AUG 78
1023                  ********************************************************
1024                  *
1025                  * WR2BHX OUTPUTS THE CONTENTS OF HL AS FOUR ASCII CHAR
1026                  *    REGISTERS AFFECTED: A, B
1027                  *    SUBROUTINES CALLED: WR1BHX
1028                  *
1029  0CB7  7C        WR2BHX MOV A,H            GET MSB OF HL
1030  0CB8  CD A4 0C         CALL WR1BHX        OUTPUT IT
1031  0CBB  7D               MOV A,L            GET LSB
1032  0CBC  CD A4 0C         CALL WR1BHX        OUTPUT IT
1033  0CBF  C9               RET                VERIFIED 22 AUG 78
```

```
LINE  LOC   B1 B2 B3          SOURCE LINE

1035                    **************************************************
1036                    *
1037                    * WR1BI3 FORMATS THE DATA IN THE ACCUMULATOR INTO AN
1038                    *    I3 FORMAT (3 ASCII DIGITS).  WR1BI2 CONVERTS
1039                    *    THE DATA IN THE C REGISTER INTO AN I2 FORMAT.
1040                    *       REGISTERS AFFECTED: A, B, C, D, E
1041                    *       SUBROUTINES CALLED: DIVIDE, WRITE
1042                    *
1043  0CC0  4F          WR1BI3 MOV C,A          MOV TO LSB
1044  0CC1  06 00              MVI B,0          CLEAR MSB
1045  0CC3  16 64              MVI D,100        SET DIVISOR
1046  0CC5  CD 01 0D           CALL DIVIDE
1047  0CC8  79                 MOV A,C          GET QUOTIENT FOR OUTPUT
1048  0CC9  48                 MOV C,B          MOV REMAINDER TO LSB
1049  0CCA  C6 30              ADI '30          ADJUST TO ASCII
1050  0CCC  CD EA 0A           CALL WRITE       OUTPUT DATA DIV 100
1051                    * ENTER HERE FOR I2 FORMAT - DATA (REG C) MUST
1052                    *    BE LESS THAN 100 !!!
1053  0CCF  06 00       WR1BI2  MVI B,0          CLEAR MSB
1054  0CD1  16 0A              MVI D,10         DIVISOR
1055  0CD3  CD 01 0D           CALL DIVIDE
1056  0CD6  79                 MOV A,C          MOV QUOTIENT TO ACC
1057  0CD7  C6 30              ADI '30          ADJUST TO ASCII
1058  0CD9  CD EA 0A           CALL WRITE       OUTPUT 10'S DIGIT
1059  0CDC  79                 MOV A,C          GET REMAINDER
1060  0CDD  C6 30              ADI '30          ASCII
1061  0CDF  CD EA 0A           CALL WRITE       OUTPUT UNITS DIGIT
1062  0CE2  C9                 RET              VERIFIED 18 AUG 78
1064                    **************************************************
1065                    *
1066                    * CRLF OUTPUTS A CARRIAGE RETURN AND LINE FEED ON THE
1067                    *    OUTPUT DEVICE.
1068                    *       REIGSTERS AFFECTED: A
1069                    *
1070                    CR     EQU '0D          CARRIAGE RETURN IN ASCII
1071                    LF     EQU '0A          LINE FEED
1072  0CE3  3E 0D       CRLF   MVI A,'0D        CARRIAGE RETURN
1073  0CE5  CD EA 0A           CALL WRITE
1074  0CE8  3E 0A              MVI A,'0A        LINE FEED
1075  0CEA  CD EA 0A           CALL WRITE       LAST CHARACTER OF STRING
1076  0CED  C9                 RET              VERIFIED 22 AUG 78
1078                    **************************************************
1079                    *                                                *
1080                    *   T W O   B Y T E   A R I T H M E T I C        *
1081                    *                                                *
1082                    **************************************************
```

```
LINE  LOC   B1 B2 B3         SOURCE LINE

1084                   ************************************************************
1085                   *
1086                   * MLTPLY MULTIPLIES THE UNSIGNED 8 BIT NUMBERS IN THE
1087                   *    C AND D REGISTERS AND RETURNS THE PRODUCT IN THE
1088                   *    B (MSB) AND C (LSB) REGISTERS.
1089                   *       REGISTERS AFFECTED: A, B, C, E
1090                   *
1091  0CEE  06 00      MLTPLY MVI B,0          CLEAR MSB
1092  0CF0  1E 09             MVI E,9          INITIALIZE COUNTER
1093  0CF2  79         MULT0  MOV A,C          ROTATE LS BIT OF MULTIPLIER
1094  0CF3  1F                RAR                 TO CARRY AND SHIFT LOW ORDE
1095  0CF4  4F                MOV C,A                OF PRODUCT
1096  0CF5  1D                DCR E            BUMP COUNTER
1097  0CF6  C8                RZ               EXIT IF COMPLETE
1098  0CF7  78                MOV A,B          GET MSB OF PRODUCT
1099  0CF8  D2 FC 0C          JNC $+4          ADD MULTIPICAND TO MSB OF PROD
1100  0CFB  82                ADD D               ONLY IF RAR ABOVE PRODUCED
1101  0CFC  1F                RAR              SHIFT MSB OF PRODUCT (CARRY =
1102  0CFD  47                MOV B,A          SAVE MSB
1103  0CFE  C3 F2 0C          JMP MULT0        LOOP UNTIL ALL BITS DONE
1104                   *                       VERIFIED 21 AUG 78
1106                   ************************************************************
1107                   *
1108                   * DIVIDE DIVIDES AN UNSIGNED 16 BIT NUMBER IN THE
1109                   *    B (MSB) AND C (LSB) REGISTERS BY THE UNSIGNED
1110                   *    8 BIT NUMBER IN THE D REGISTER.  THE 8 BIT
1111                   *    QUOTIENT IS IN THE C REGISTER AND THE REMAINDER
1112                   *    IS IN THE B REGISTER.
1113                   *       REGISTERS AFFECTED: A, B, C, E
1114                   *
1115  0D01  1E 09      DIVIDE MVI E,9          BIT COUNTER
1116  0D03  78                MOV A,B          GET LSB OF DIVIDEND
1117  0D04  47         DIV0   MOV B,A          SAVE ACC
1118  0D05  79                MOV A,C          GET MSB OF DIVIDEND
1119  0D06  17                RAL
1120  0D07  4F                MOV C,A
1121  0D08  1D                DCR E            BUMP COUNTER
1122  0D09  CA 1D 0D          JZ DIV2
1123  0D0C  78                MOV A,B
1124  0D0D  17                RAL
1125  0D0E  D2 15 0D          JNC DIV1
1126  0D11  92                SUB D
1127  0D12  C3 04 0D          JMP DIV0
1128  0D15  92         DIV1   SUB D
1129  0D16  D2 04 0D          JNC DIV0
1130  0D19  82                ADD D
1131  0D1A  C3 04 0D          JMP DIV0
1132  0D1D  17         DIV2   RAL
1133  0D1E  5F                MOV E,A
```

```
LINE  LOC   B1 B2 B3          SOURCE LINE

1134  0D1F  3E FF            MVI A,'FF
1135  0D21  A9               XRA C
1136  0D22  4F               MOV C,A
1137  0D23  7B               MOV A,E
1138  0D24  1F               RAR
1139  0D25  C9               RET                    VERIFIED 18 AUG 78 FOR WR1B13
1140                         END

** ASSEMBLY COMPLETE, NO ERRORS 1062 BYTES ASSEMBLED
```

APPENDIX D

CES Programming Code Summary

Commands from the GPIB are interpreted as ASCII characters and are decoded as follows.

Command	Description
Axx	Allocate memory block to histogram. xx is the two digit number of alarm cards installed.
C	Clear histograms and reset system clock.
Dxxx	Dump Histograms.xxx specifies the number of the histogram to dump. An out-of-range number dumps all histograms.
Gxxxx	Go to specific location in memory and start execution xxxx is the four hex digit address to start at.
Kxxxy	Relay control. xxx is the relay code. y = Ø turns the relay off, y > Ø turns the relay on.
L	Local enable. Terminal device is enabled for operator input.
P	Load program. Program in Intel format representation is loaded into RAM.
R	Remote. Locks out the local control of the processor.
S	Start Scanning. If not preceded by an "A" command, defaults to scanning only one alarm card.

DISTRIBUTION LIST

	NO OF CYS
DCA/470 Washington DC 20305	3
DCEC/R220 1860 Wiehle Avenue Reston VA 22070	4
Naval Telecommunications Command ATTN: 2633 Massachusetts Avenue, NW Washington DC 10390	
Naval Electronics System Command ATTN: 41014, 41025 Washington DC 20360	
US Department of Commerce National Telecommunications and Information Administration Institute for Telecommunication Sciences Boulder CO 80303.	4
NSA/S254 Ft Meade MD 20755.	2
USACEEIA/CCC-CED	2
USACEEIA/CCC-TED	2
USACEEIA/CCC-CED-XEM Ft Huachuca AZ 85613	1
AFLC/MMWE Wright-Patterson AFB OH 45433	1
AFCD/MMCY Kelly AFB TX 78241	1
AFSS/SREE Kelly AFB TX 78241	1
ESD/DCDE Hamscom AFB MA 01730	1
NCA/EPE Griffiss AFB NY 13440	2
ECA/EPE APO New York 09633	1
SCA/EPE Oklahoma City AFS OK 73145	1
PCA/EPE APO San Francisco 96515	1
Defense Documentation Center Cameron Station Alexandria VA 22314	15

1839 EIS	
EMC Measurements Division/EPEF	
Keesler AFB MS 39534	1
HQ AFCS	
DOY	1
LGX	1
LGMEE	1
LGMSR	1
OA	1
XPQC	1
1842 EEG	
EEISG	2
EETTC	1
EETTW	1
EETET	5
HQ AFCS	
DAPL	10
DFEE/CSL	
USAF Academy, CO 80840	1