A DRIVER TO

INTERFACE THE DGDAC SUBSYSTEM

AND THE GOSEX SYSTEM

A SOFTWARE DRIVER TO

INTERFACE THE DGDAC SUBSYSTEM

AND THE GOSEX SYSTEM

By

DAVID ERNEST SIMMONS, B.A.Sc.

A Project

Submitted to the School of Graduate Studies

in Partial Fulfilment of the Requirements

for the Degree

Master of Science

McMaster University

September 1977

MASTER OF SCIENCE (1977) McMaster University
(Computation) Hamilton, Ontario

TITLE: A Software Driver to Interface the DGDAC Subsystem and the GOSEX System

AUTHOR: David Ernest Simmons, B.A.Sc. (Queen's University)

SUPERVISOR: Dr. P.C. Chakravarti, Dr. J.D. Wright (Chemical Engineering)

NUMBER OF PAGES: vi, 99

ABSTRACT:

A Software Driver has been written to interface the DG/DAC (Data General's Data Acquisition and Control) Subsystem and the GOSEX (Generalized Operating System Executive) System. The interface handles the input-output requirements of GOSEX and was written to allow for dual processing on Data General Nova Computers.

The code was written to appear transparent to the user as well as to add additional features to the GOSEX system.

A discussion of current Industrial Data Acquisition and Control systems is included with recommendations on time and cost saving strategies.

ACKNOWLEDGEMENT

The author wishes to express his gratitude to those who have assisted him in the successful completion of this project. In particular, he wishes to thank:

Dr. J.D. Wright for his helpful guidance throughout this study.

Dr. P.C. Chakravarti for his advice and for the time spent discussing problems associated with this project.

My wife Audrey, for her perseverance.

Mrs. Hilary Karik, for the speedy typing of this project.

McMaster University for assistanceships and scholarships.

TABLE OF CONTENTS

INDEX OF FIGURES

1- INTRODUCTION

In the last five years a dramatic shift has occurred in the cost factors of minicomputer systems. With the increased efficiency of producing hardware and with greater competition among vendors the cost of minicomputers has substantually decreased while the cost of producing software has grown almost inversely proportionally to the fall of hardware prices.

This development of inexpensive minicomputers has lead to the widening interest in Digital Process Control Systems. Digital Control Computers are being used to monitor the operation of plants, log relevant process variable data, indicate alarm conditions, control processes, and to generally optimize the operation of a process.

The principal feature of a process control system, which make it different from an interactive time-sharing system, is the handling of the input and output signals and its process interrupt system.

Within this project, it will be shown how a minicomputer is used to handle the input-output (I/O) operation of a process control system and how the interrupt structure is used in conjunction with the workings of the system. In particular the I/O and interrupt handling structure of the GOSEX[5*] system are examined

* This number refers to references listed in the Bibliography.

with the necessary changes required for the addition of the DG/DAC[3] subsystem.

One of the key developments in Process Control Systems recently has been the writing of executive packages. These packages free the user from the great deal of programming which is necessary before the operator-system communications, in an experiment using digital process control, can be carried out. The user is merely responsible for certain sections of the code to describe the unique parameters of the experiment.

At McMaster University, the Chemical Engineering Department has developed a generalized operating system executive (GOSEX) which is used for on-line data acquisition and control of chemical processes. The package contains a large number of utility programs, executive functions and executive tasks. It allows the user full interactive control over data logging, control software, on-line parameter adjustments, and process variable alarming with minimal programming additions from the user. The GOSEX system has been in use in the Chemical Engineering Control Laboratory since 1974 and is also in industrial use for process research at the Bruce Heavy Water Plant of Ontario Hydro at Tiverton, Ontario.

Recently additional I/O devices have been required and the I/O hardware (which is six years old) has proven inadequate for current modes of operation. The old hardware does not provide enough I/O channels for future use and only allows porting to one of the two computers in the system.

Consequently, the decision was made to order a completely new analog subsystem. The DG/DAC subsystem allows for a dual processing system and flexible I/O applications. It appears to the process system as one interrupting device and has its own programming requirements.

Because of this change to the I/O subsystem in the Chemical Engineering Control Laboratory it was required to change that part of the code in GOSEX which deals with the I/O devices. The DG/DAC programming requirements, the addition of new I/O drivers, and a new interrupt scheme was developed for the change. The change appears transparent to the user except for the requirement of an addition digital to the analog table of values. The system operates in essentially the same way as the original version.

In addition, a discussion of the current status of process control computers is included with some observations from the author on time and cost saving strategies.

2- CURRENT PROCESS CONTROL SOFTWARE SYSTEMS

The large and increasing numbers of firms which are involved in the design and implementation of the hardware and software of Process Control Systems have lead to many distinct methods of handling the problem. The following is a review of some of the techniques which are used and a discussion of problems faced by the users of these products.

2.1 Survey of Process Control Problems

Once a firm has shown justification of a process control system the development of the system can take many paths. A firm without the resources to handle software development or hardware specifications may hire a consulting firm to provide a Turn-Key system. This is usually the most expensive method and can result in the firm being dependent on the consultant for any modifications once the contract requirements are met. The firm must also check the consultants work to see if the code and the resources were used efficiently, if the optimization schemes were properly utilized, if the program is easily expandable, and if all the system specifications were met.

In the author's own experience a Turn-Key system was delivered which failed all of the above criteria. The system, to monitor a gas transmission plant, was designed and written by inexperienced personnel and many years of problems were faced by the gas transmission company. Some other problems were not foreseen which caused many delays and significant expenditures. The CPU which was used was manufactured in Canada, but since the consultant was located in the United States the gas company paid double duty on the unit. Additional hardware ordered from the consultant was charged at significantly higher prices and training for new operating personnel was prohibitively expensive. A simple disc cartridge was quoted at four hundred dollars ($400.00) by the consultant while the same unit was purchased from the original equipment manufacturer for eighty-five dollars ($85.00). The system proved so inflexible that finally a programmer was hired to completely redesign and rewrite the software.

Generally, the best method to develop a Process Control System is to purchase a general package and develop the specific requirements in-house. The firm should have a programmer who is able to develop systems from such a base and be available to maintain these systems. Process Control projects normally change in scope between the design and end-product stages and the firm needs the independence that is gained when having its own employees complete the task.

In the past, when designing process control systems, the system was usually designed from the bottom up. Often the design was started by considering the control loops and the memory cycle speed of the computer as the basic design criterion[6]. The design would work up finally adding the operator interface at the top level.

The principles of software engineering indicate that computer systems should be designed from the top down by successive refinements. We should start by considering what results we want and then design from there. It is essential that the computer be fast enough to do the job but this should not be a base for the design but something checked on after the system's completion. The last decision to be made in the design should be which mini-computer should be included.

2.2 Current Software Systems

The varying applications of process control have led to a wide variety of operating system executives. Some systems maintain a large database and mostly involve I/O scheduling. Others are compute bound and rely on the optimization routines of the computer. Still others exercise little or no control over the process and are merely report producing monitoring systems.

The following paragraphs summarize the packages currently offered by some companies.

1) Fisher Controls, Marshalltown, Iowa.

Fisher has developed its own Process Control language called PC^2 which is very similar to BASIC with some added I/O commands for real-time systems. The system is based on a 32K word processor which is expandable to 64K. The system is good for general use but interfacing to other hardware, except Fisher equipment, is difficult. The PC^2 compiler is hardwired which requires an extra 4K words of memory. The user cannot link his own assembler routines to the package which limits some applications.

2) Foxboro Company, Foxboro, Mass.

Foxboro has developed an integrated control system called INTERSPEC, which allows each Foxboro unit in the system to interact predictably well with every other Foxboro element. Their software incorporates a Fill-In-The-Forms[7] method to allow the user to perform all the functions by merely answering questions about his application.

This approach results in considerable savings in software development costs but the programs tend to be large, and very general, and require substantial memory. It is also possible to require new applications which the "Fill-In-The-Blanks" method does not cover.

3) Taylor Instrument Company.

Taylor Instruments has developed an interactive Process Control language called POL*3 (Process Orientated Language-3)[8]. POL*3 allows the user to avoid the compiling stage and permits the user to execute tasks while the computer is on-line. Thus the user can work with the I/O and change the process requirements while on-line. POL*3 is an interpreter which causes the users programs to access subroutines which handle the requests.

4) The Purdue Fortran

The Fortran Committee of the Purdue Workshop on Standardization of Industrial Programming Languages attempted, in 1972, to unify the different process control extensions to Fortran by developing Purdue Fortran[10]. This has since become an industrial standard. The language takes ANSI Fortran as a base and adds a set of standard procedures for process control. Purdue Fortran has eighteen procedures for starting tasks, process I/O bit string manipulations, and time and date information.

5) Digimatics Ltd.

Digimatics has developed a high level process control language called ABACUS[9] which allows the user to describe his system by means of block schematic control diagrams similar to conventional analogue control and logic signals. The information about the block schematic is entered directly to the computer. The required software, being closely related to block diagrams, is one the user can easily picture and work with.

Other systems which may be of interest to the reader include those supplied by Bailey Meter, Bristol, Honeywell, and Metromation. Each of these has its own special features but all of them are inefficient and inflexible for special purpose applications.

Most firms offer the ability to have a backup computer for important processes. This requires a dual processing system with both computers sharing the same disc. This ensures that when the on-line computer fails no data is lost and that the control process can immediately continue. The back-up computer can be used for other purposes while it awaits for the main computer to fail.

3- DESCRIPTION OF THE SYSTEM

The three basic components of the system which were changed will be described on the following pages. They are the computer system, the GOSEX software package, and the programming requirements of the DG/DAC subsystem.

3.1 The Computer System

Figure 1 shows the layout of the hardware in the Chemical Engineering Control Laboratory. The computers were manufactured by Data General Corporation, have a 16 bit word length, and have a memory size of 32,768 words.

The instruction set of the Nova computers contain instructions that perform fixed point arithmetic and logical operations between accumulators, transfer of operands between accumulators and main memory, transfer of program control, and I/O operations. The instruction set was designed to allow for many operations to be performed by one instruction, allowing complex routines to be written by a small number of instructions.

The computer operation is performed sequentially, executing the instructions of a stored program one by one. In small systems the computer proceeds without interruption, continuing until the solution is complete. In general this mode of operation results in inefficient use of computer resources and does not allow for fast responses to process interrupts which are commonly required in process applications. The following are examples illustrating how interrupts can solve the above two problems.

Consider a case in which the control computer is monitoring the pressure of a gas at the discharge of a compressor station. If the pressure goes below a certain reading the compressor is to be turned on increasing the pressure in the gas transmission line. The compressor is to be turned on within five minutes after the violation of limits occurs if sufficient pressure is to be maintained in the pipeline.

Using Murphy's Law* it is quite probable that the computer would be initiating a lengthy optimization calculation, that requires ten minutes of computing, at the instant the violation occurs. Clearly, without the interrupt capability the computer could not respond to the process event in the required time.

The efficient use of computer resources may be illustrated by examining the operating speeds of the processor and the I/O units. The processor can execute over one million instructions per second compared with thirty characters per second for a typical output terminal. Thus if sequential execution were followed the processor would always be waiting for work while the terminal slowly operated. The interrupt feature allows many thousands of instructions to be executed while the I/O units are operating.

In the basic interrupt scheme there are six functions which must be carried out:

1) Detection of an interrupt
2) Indication of the source of the interrupt
3) Saving of information so the execution of the interrupted program can later be resumed
4) Transfer of control to the interrupt handling program
5) Execution of the interrupt handling program

* If anything can go wrong it will.

6) Return of control to the interrupted program

These functions could all be performed by the program if the programmer added sufficient code. This could be done by having the program periodically compare all input values to pre-set limits and have the program branch to code to service any limit violating conditions. This would require the programmer to time the execution of segments of code so that the program can transfer to the limit checking code at periodic time intervals. The routine would have to store the state of the program at every transfer thus increasing the number of instructions executed, and increasing execution time and storage requirements. Obviously, the hardware interrupt process is preferable for the majority of process control applications.

Advancing the concept of interrupts we find that it is advantageous to assign each possible interrupting device a level of priority. For example, an interrupt from an input device, indicating that digital data is to be read, should be ignored if the computer is presently handling an interrupt to an emergency condition in the process.

Three types of interrupts exist on the system:

1) System Interrupts.

These are generated by the computer system itself and is key to the functioning of the system. When the teletype has finished outputting a character and is ready for another a system interrupt is generated.

2) Timer Interrupts.

In our system the operator must initialize a user clock which indicates what action is to be done periodically by the program. An example is if an optimization calculation is to be done every ten minutes the timer interrupt would be set to ten minutes.

The user clock is essentially a subroutine which contains a counter. The system clock runs at 10 Hz and interrupts the program every one-tenth of a second. The interrupt forces the counter in the clock subroutine to increment and to be compared with a set value which was initialized when the user clock was set by the user. When the counter is equal to the set value a timer interrupt occurs.

3) Process Interrupts.

These interrupts originate from the process and either signal that some routine is to be executed by the computer or signal that a message indicating an alarm condition is to be generated. In the Chemical Engineering Control Laboratory system the user can execute a GOSEX function by typing CTRL E on the teletype, thus issuing a process interrupt.

In Data General systems[11] two levels of priorities are present. High priority interrupt devices are contained in the HINT dispatch table. Examples of these are the real time clock and the power fail/auto restart device. The lower priority devices are wired to a particular bit of the 16 bit priority mask and have entrees in the device interrupt vector table (ITBL). More than one device can be wired to a particular bit. By changing the mask word upon interruptions the servicing program can inhibit specified levels of interrupts and thus establish a priority scheme. By use of the mask bits the user can guarantee that a servicing program will not be interrupted. The masking function only delays the recognition of an interrupt and thus the interrupt is not lost.

Figure 2 shows the system configuration in the Chemical Engineering Control Laboratory, including the two minicomputers, the DG/DAC unit, and the I/O devices. The dual processors share control of the DG/DAC subsystem, each having its own crate controller.

Details of each I/O module is given below:

1) 4290 General Purpose Input Module (to be used as a Contact Sense Device).

The Contact Sense device consists of a 16 bit data register which stores the state of sixteen independent lines. When a transition is detected in any of the lines the module initiates a program interrupt request. The program must then analyze the status of the data register to determine which of the lines changed state and caused the interrupt. The module can also be used to store the state of the sixteen independent lines only upon user request and without causing an interrupt.

2) 4291 TTL Input Module (Two modules in system).

The TTL input modules each contain a 16-bit data register which stores the state of sixteen independent TTL compatable input lines. The data register receives the digital data from the input lines on command from the program or can be programmed to initiate an interrupt upon reception of data from one of the input lines.

3) 4297 Form C Relay Output Module

The relay module controls 17 independent form C reed relays. The module contains a 16-bit data register which controls the value of sixteen of the relays with the seventeenth relay (start pulse relay) being controlled by a module flag command (the S-function).

Physically the Form C relay is a mercury-wetted relay which is activated by a magnetic field. The mercury covers all contact surfaces and thus eliminates bounce when the contacts close.

4) 4299 TTL Output Module (Two modules in system)

The TTL output modules each control eighteen independent TTL compatable output lines. Sixteen of these lines are controlled by bits in a 16-bit data register while the other two are strobe lines which are controlled by module flag commands (the S, C, and IORST functions).

5) 4288 Series D/A Converters (Two converters in system)

The D/A converters provide analog voltage output on four program selectable output lines. The output voltage range is 0-10 volts and the resolution of the digital value to be converted is 12 bits. The digital to analog converter is used to control analog devices such as valves or setpoints.

6) A/D Cluster

The A/D cluster consists of an analog to digital input device plus two voltage multiplexors and one current multiplexor. The cluster is used to provide a digital representation of an analog signal. The three components of the cluster are:

a) 4280 A/D Converter.

The A/D converter operates under Data Channel Operation, that is, the address in core, where the converted digital data is to be stored, is given and all the converted data is stored there. This is opposed to individual digital values being stored by the program as in Programmed I/O Operation. The digital data has 12-bit resolution. The three multiplexors are differential multiplexors which requires two channels per analog signal. The major advantages of this are, first, the ability to handle low-level as well as high level signals, and second, the additional performance requirement of common mode rejection.

b) 4281 Voltage Multiplexors (two in cluster).

The input voltage range is 0 - 10 volts. The Voltage Multiplexors have programmable select gains with values of 1,2,4 and 8.

c) 4281C Current Multiplexors.

The current multiplexor inputs a signal of 10 - 50 ma to the A/D converter.

3.2 The GOSEX Executive System

The GOSEX package is a group of utilities and executive functions used as the base for real time process control applications. The user has to develop a minimal number of subroutines to make use of the executive package. The package is flexible and can be used for a variety of applications on any NOVA-line minicomputer operating under Data General's real time disk operating system (RDOS). The GOSEX package occupies approximately 4K words of central processor memory, leaving approximately 14K words for user programming in a 32K word machine.

The package consists of three distinct types of routines:

1) Utilities.

The utilities are routines which support the package functions. They are composed of twenty-eight core resident programs which are re-entrant to allow for their use in different interrupt levels. Under RDOS, the re-entrant subroutines are essentially read-only code and each reference to the routine creates a separate area for the data and intermediate results. A description of each utility is available.[5]

2) Executive Tasks.

The executive tasks are routines which initialize the system at run-time, check for alarm conditions, and control the transfer of data between the system and the plant. The main program, INTRP, allows initial operator-system communication, and handles operator interrupts. The routine named MNITR allows the user to output messages to the keyboard during alarm conditions. The interface handler is called MTPLX. MTPLX was rewritten as part of this project but its purpose remains unchanged. It initiates the input devices, handles the outputs when requested by the users clock, and handles the inputs when the input interrupts request service. Routines are also available to generate logged output, plots, profile output, and copy output.

3) Functions

The functions supply an individual task to the system and are executed by user request, upon the operator interrupt, given by CTRL E. The user can also supply special purpose functions which can be linked to the system. Examples of functions are STATS, which will print out the status of the run, and LOG, which will initiate a display of data on any of the output devices.

The requirements of the user are to supply a number of programs, tables, and constants which are used by the executive programs to uniquely describe the system. The constants are to be set in a routine called PTAPE which will supply GOSEX with the limiting values and the number of devices in the system. The user must supply routines to shut down the system, and to handle the calculations necessary in his application. Message files must be provided for the alarm conditions and tables must be set-up to hold the data for the I/O devices.

The executive function PREP is called at run-time to study the inputs given by the user and will generate diagnostics upon the location of errors. The PREP routine was changed in this project to reflect the changes to the tables necessary for the enlarged I/O subsystem. A sample run of GOSEX is available.[5]

3.3 The DG/DAC Subsystem

The following is a review of the programming requirements of the DG/DAC subsystem. Each I/O device has its own programming requirements and handles the data in a unique way.

1) 4290 General Purpose Input Device (Contact Sense Device).

The Contact sense device is programmed in two separate steps, first, starting the device, and second, servicing it upon the change of one of its bits. To start the device the module first must be selected by indicating its slot address in the Specify Address and Context (DOA) statement. Then a start flag is given (S) which will start the device and enable the module interrupts.

To service the device, upon an interrupt, the individual bits of the data register must be compared with their previous values to locate which bit changed status. Once the bit is located a message is transmitted to the user disclosing the change and the contact sense device is again started.

2) 4291 TTL Input Module.

The TTL input device has to be initialized, serviced when ready, and started. The device is selected and started with a Specify Address and Context (DOAS) statement with the slot address in the appropriate bits. This enables the module interrupt and allows the device to interrupt the system when the input is present in the data register. When this occurs the data is read and placed in the table TTLI1 or TTLI2 in memory, depending on which TTL input device handled the data.

3) 4297 Form C Relay Output Module.

Part of the requirements of the user is that he must set the frequency at which the system will output data, read the analog to digital and TTL input data. This is given by the constant SCAN in the users own program. From this sampling rate a user clock[11] is defined which will force an interrupt every sampling unit of time. When this timer interrupt occurs a branch to the output handler is made and the Form C relay is selected by a Specify Address and Context (DOA) statement with the slot address in the appropriate bits. The relay settings are then loaded into the data register by a Write Data (DOB) statement. The device is then started by a start flag (S) command which closes the Start Pulse relay for 50 ms and the data is subsequently output.

4) 4299 TTL Output Modules.

After the Form C relay data is output the TTL output modules are selected sequentially. The TTL output is loaded into the data register by a Write Data (DOB) statement and the device is then started by a start flag (S) command. The flag asserts the two external strobe lines and starts a 7 ms interval timer at which time the data is output.

5) 4288 Series D/A Converter

After the TTL data is output the D/A converters are selected sequentially. The eight D/A channels are selected in sequence and the data and channel numbers are loaded into the data register by the Specify Conversion (DOB) instruction. The data is then converted to an analog signal and output. The procedure is then carried out for the other channels.

6) 4280 A/D Converter.

The A/D converter, current multiplexor, and the two voltage multiplexors are grouped together into a cluster. The gain values of the voltage multiplexors must be set, the ADC must be started, and the cluster serviced when ready. The gain values must be set prior to selecting the A/D converter. They are set by first selecting each voltage MPX by the Specify MUX Address (DOC) statement and then using the Specify Gain (DOB) instruction to output the gain value.

The cluster is then started by issuing the following instructions:

a) Select the cluster with a Specify Address and Context (DOB) statement with the slot address in the appropriate bits.

b) Indicate the number of conversions to be done by the Specify Conversion (DOB) statement. The following bits are to be set - initiate program interrupt on a data overrun error, command to propagate to the specified multiplexor when appropriate, trigger is set to DCHI (Data Channel Operation), automatic scanning of multiplexors, data channel mode, and the number of conversions is set to 3.

c) Indicate the first multiplexor to be selected by the Specify

MUX address (DOC) with the number of multiplexors set to 3, and the first channel given in the Channel Select bits.

d) Indicate where in memory the converted data is to be stored by a Load Memory Address (DOC) instruction.

e) A start flag (S) is then given which will start the conversion.

When the conversion is finished and the digital data is stored in the table (DATUM) in memory, the program is interrupted to indicate that the conversion is complete. The validity of the data is then checked with a Read Module Status (DIA). Bit 3 will be one if there was a data overrun error. The request is then cleared with a clear flag (C) command.

4- DESCRIPTION OF THE INTERFACE SOFTWARE

The following is a description of the software changes required for the new DG/DAC subsystem. A general discussion of the Code changes is first given followed by details of each required change. A written description, flow charts, and the actual code is included.

4.1 The DG/DAC - GOSEX Interface

Below is a discussion of the problems, decisions, and analysis which had to be done in implementing the new I/O sub-system within the existing operating system executive.

Initially it had to be determined which code of GOSEX was to be affected by the change. The entire MTPLX routine required change, PTAPE had to be expanded to allow for the additional I/O units, and PREP had to be changed to allow for the expansion.

Problems were encountered in the following areas:

1) Lack of documentation on the flow of the GOSEX routines. As is typical with assembler programs, individual sections of code were well documented but a systems flowchart, showing how the sections fit together, was lacking. This caused serious problems when designing the interface.

2) Lack of proper documentation on the programming of the DG/DAC subsystem. The programmer's reference manual was poorly written, lacked details, and had no examples. The programmers at Data General (Toronto) were most helpful in solving the "Fill-in-the Blanks" documentation.

3) The documentation on the Dual Processor arrangement for the DG/DAC unit was poor and no one at Data General could explain the extra requirements for this configuration. Therefore, no attempt was made in the code to take advantage of the dual processing set-up in the Chemical Engineering Control Laboratory.

4) Data General was not able to deliver the DG/DAC unit by

the promised deadline. Therefore, the code written in this project was not able to be tested.

The following were decisions which had to be made on how flexible the interface should be:

1) It was decided not to plan the code for the dual processing case because of the lack of documentation dealing with this arrangement.

2) It was decided to allow each device to be independent of the slot number in the DG/DAC unit. The user has to specify not only how many slots he is using for his application but which particular slot he is using for each device. The multiplexor still must follow the ADC device in the ADC cluster. The slot selection is done in PTAPE.

3) Presently, for the digital to analog converter, the user always starts with channel 0 and proceeds sequentially to channel N for the (N + 1) channels in his application. To allow for greater flexibility, and to prepare for the future dual-processing situation, it was decided to allow the user to start his application at any channel and proceed sequentially from there. The two DAC devices are treated separately and the user is required to indicate, in PTAPE, the following for each DAC unit:

i) the slot number of each DAC unit

ii) the number of channels each DAC unit is to use

iii) the starting channel of each DAC unit

4) It was decided to allow each of the voltage multiplexors

to have separate gain values. The voltage multiplexors have programmable gains which are set in the PTAPE routine. The user must also set in PTAPE, the slots in which the ADC and each multiplexor are located.

5) It was decided to treat the two TTL input and the two TTL output devices separately to allow for greater flexibility in their use.

6) It was decided to mask out the following devices when the DG/DAC unit is being serviced:

- Teletype Output (TTO)
- Teletype Input (TTI)
- Real Time Clock (RTC)
- Paper Tape Punch (PTP)
- Line Printer (LPT)

This is a change from the previous design where the Line Printer was not mask out during the servicing of the A/O converter.

7) It was decided to design the code so that future expansion could occur with minimal programming effort. The structured programming technique used in GOSEX was again used in the new code. The individual sections of code were written as conditional assemblies to allow for greater efficiency in execution.

8) In the Analog-to-Digital converter cluster the ADC module is in the first slot used by the cluster. It was decided to make the two voltage multiplexors follow in the two slots after the ADC and have the current multiplexor occupy the last slot in the cluster.

This convention was necessary for the coding of the programmable gains.

4.2 Description of the Code

The following is a discussion of the analysis done on each section of the program which had to be added or changed.* Figure 3 shows the flow of GOSEX dealing with the interface to the I/O subsystem.

The code changes are divided into five sections:

1) Changes and additions to PREP.

PREP is the routine which checks the inputs given by the user. Since new tables have been added and others expanded, the code relating to these tables has been changed. Tables for the TTL input (TTLI1 and TTLI2) and TTL output (TTLO1 and TTLO2) have been added. The table for the digital-to-analog converter (ANALG) has been retained but the first four values of ANALG are now reserved for the first DAC device, while positions 5 - 8 in the ANALG table are for the second DAC device. The changed program PREP is given in figure 4.

2) Changes and additions to PTAPE.

PTAPE contains the parameters which the user changes to fit his application. The number of channels per device have been changed and new devices have been added. The slot containing each device is given to increase the flexibility of the system. For the DAC devices the starting channel number for each device is given. Also the gain values for the two voltage multiplexors is included. The changed program PTAPE is given in figure 5.

3) Changes to DCLAR

*The new code is indicated by a bar on the left side of the page for PREP, PTAPE, and OPCOM. MTPLX contains entirely new code.

DCLAR is used to initialize and start the DG/DAC unit, start the contact sense device, and set the gain values of the two voltage multiplexors. DCLAR is called by PREP in the initialization stage of GOSEX. A flowchart for DCLAR is given in figure 6 and the code is given in figure 9.

4) Changes to MTPLX

MTPLX is called when the user clock expires and the ADC device is to be started. The routine sets the relays, outputs the data to the first and second DAC devices and the two TTL output devices. The outputting of the DAC values has to be handled in the following method. The resolution of the digital data is 12-bits which occupies the left most 12 bits of the data register. The two right most bits are to contain the channel number of the DAC device in question. The digital output values are loaded into AC1, then shifted four bits left to place them in the correct bits. The channel number is then added to AC1 to give the required contents of the data register. The Specify Conversion (DOB) statement is then issued to convert the digital value to an analog signal.

After the output devices are handled the routine then starts the two TTL devices and the ADC in Data Channel Operation mode. A flowchart for MTPLX is given in figure 7 and the code is given in figure 9.

5) POLLR-Polling routine and input handler.

POLLR is called upon an interrupt from the DG/DAC unit.

The DG/DAC unit is examined to see which input device caused the interrupt and control is passed to the code which handles the interrupting device. The device can be the contact sense, ADC, or one of the two TTL input devices.

Since the interrupts are to be handled quickly all other modules of the DG/DAC unit are masked out before the inputs are handled. After the contact sense device inputs are handled the device is again started. The flow of POLLR is given in figure 8 and the code is given in figure 9.

5- CONCLUSIONS

The prime objectives of the project have been achieved. These were:

a) To examine GOSEX and determine how it handles the I/O.

b) To examine the DG/DAC unit and determine how to program it.

c) To write code to interface the new DG/DAC subsystem with the existing code of GOSEX.

d) To examine current Process Control Systems and comment on how each system uniquely solves the Process Control problem.

The code written is not completely transparent to the user because the DAC output devices had to be treated as separate devices compared to the previous single treatment. The DG/DAC unit was not delivered in time for the code written in this project to be tested. The documentation of the interface should allow any errors, which may occur in the testing, to be easily corrected.

FIGURES

Figure 1

Layout of the Hardware in the Chemical Engineering Control Laboratory.

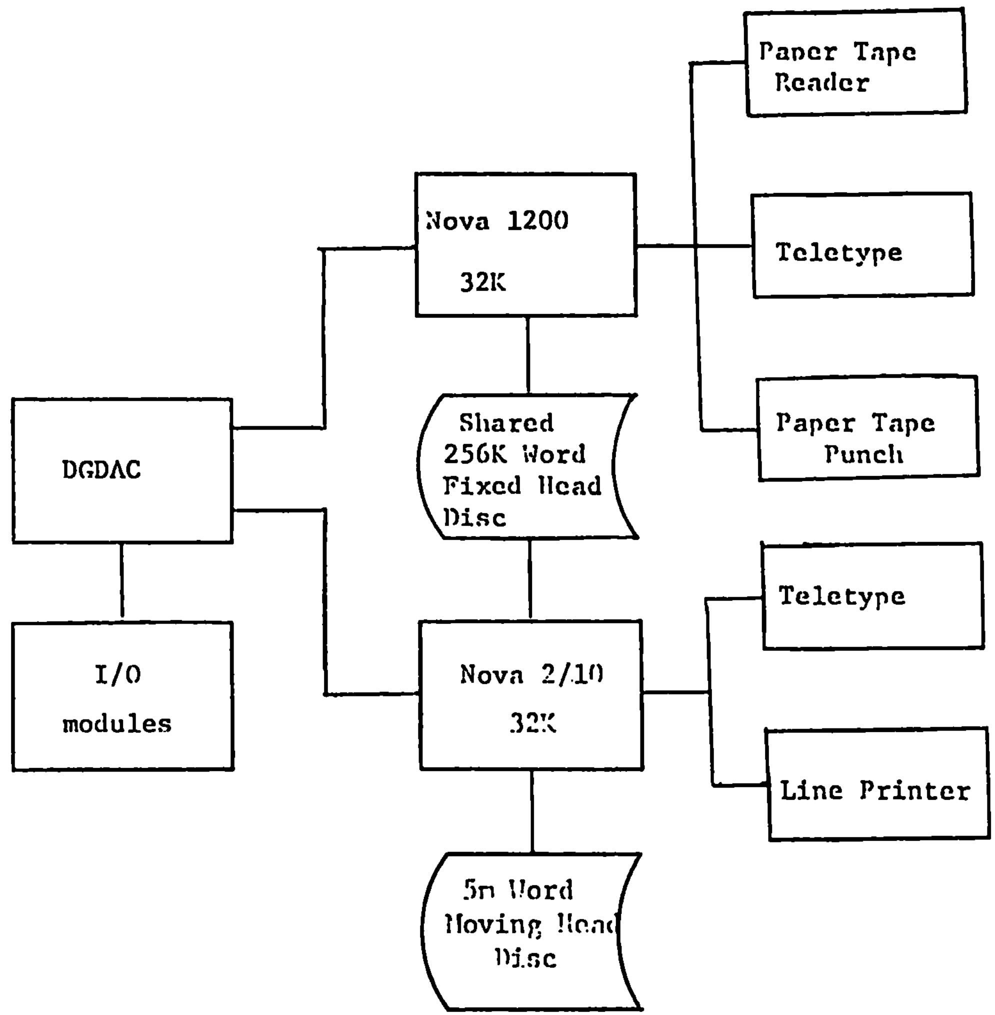
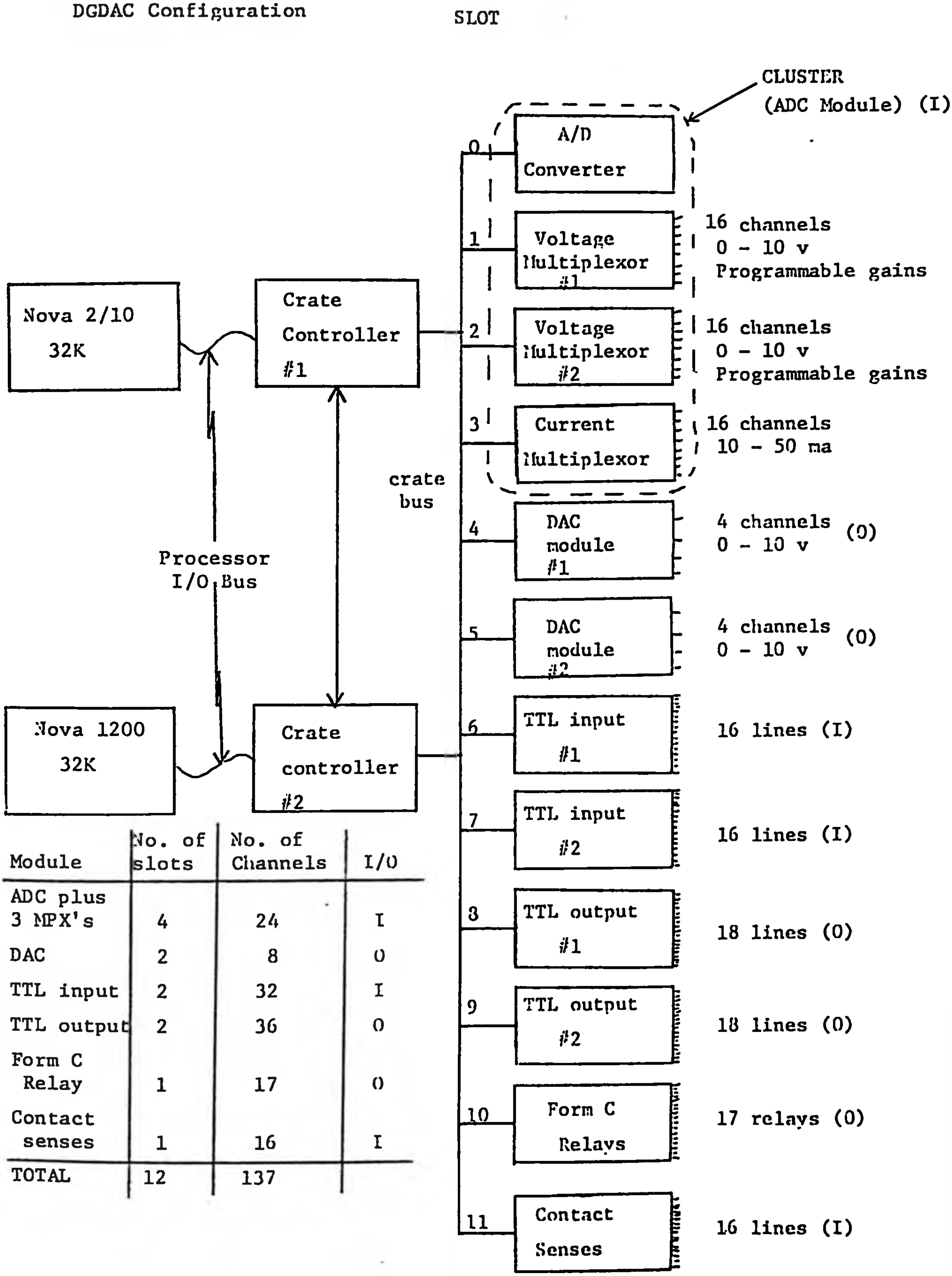


Figure 2

DGDAC Configuration



Module	No. of slots	No. of Channels	I/O
ADC plus 3 MPX's	4	24	I
DAC	2	8	O
TTL input	2	32	I
TTL output	2	36	O
Form C Relay	1	17	O
Contact senses	1	16	I
TOTAL	12	137	

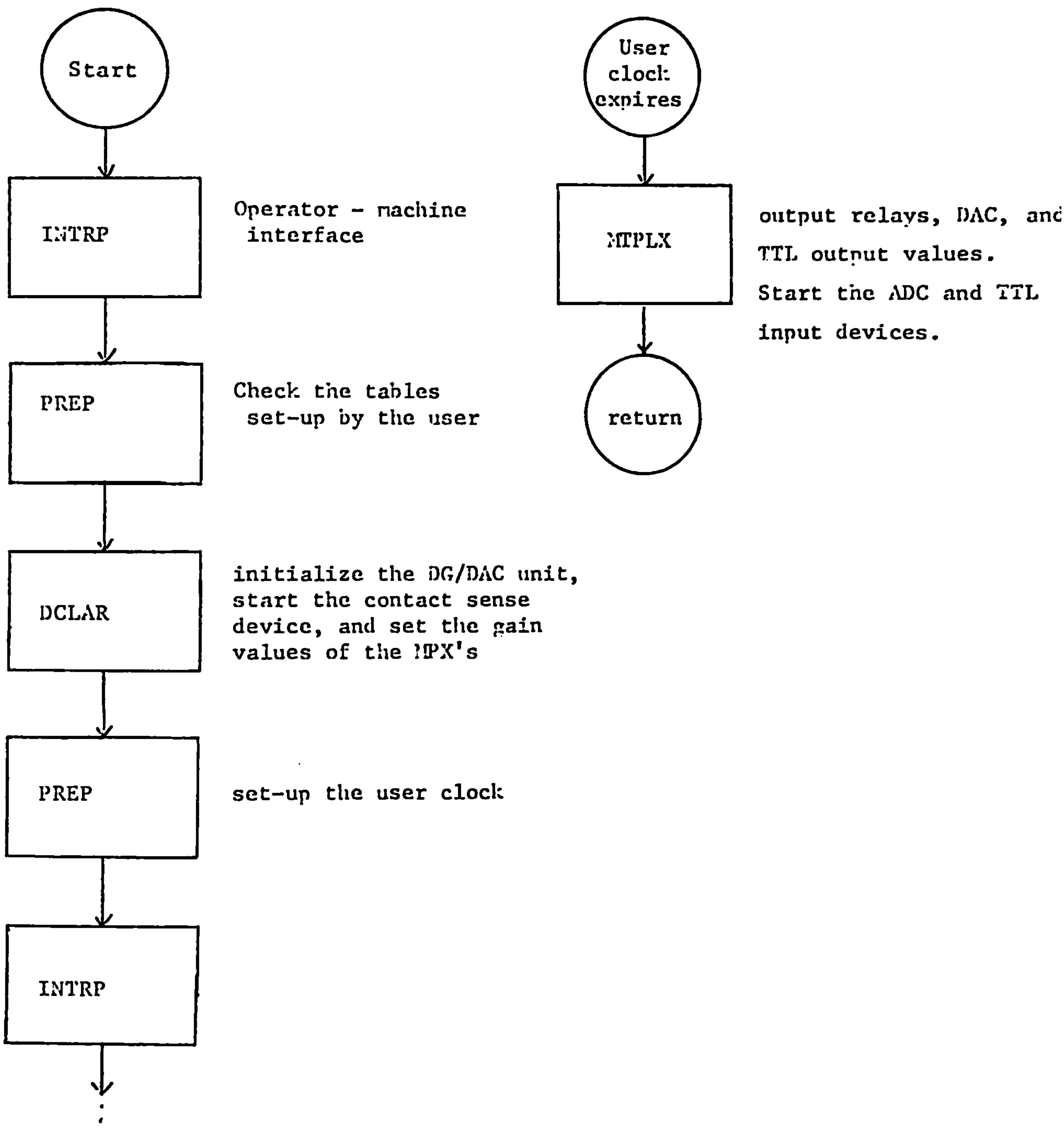


Figure 3

Flow of COSEX pertaining to the interface with the I/O subsystem

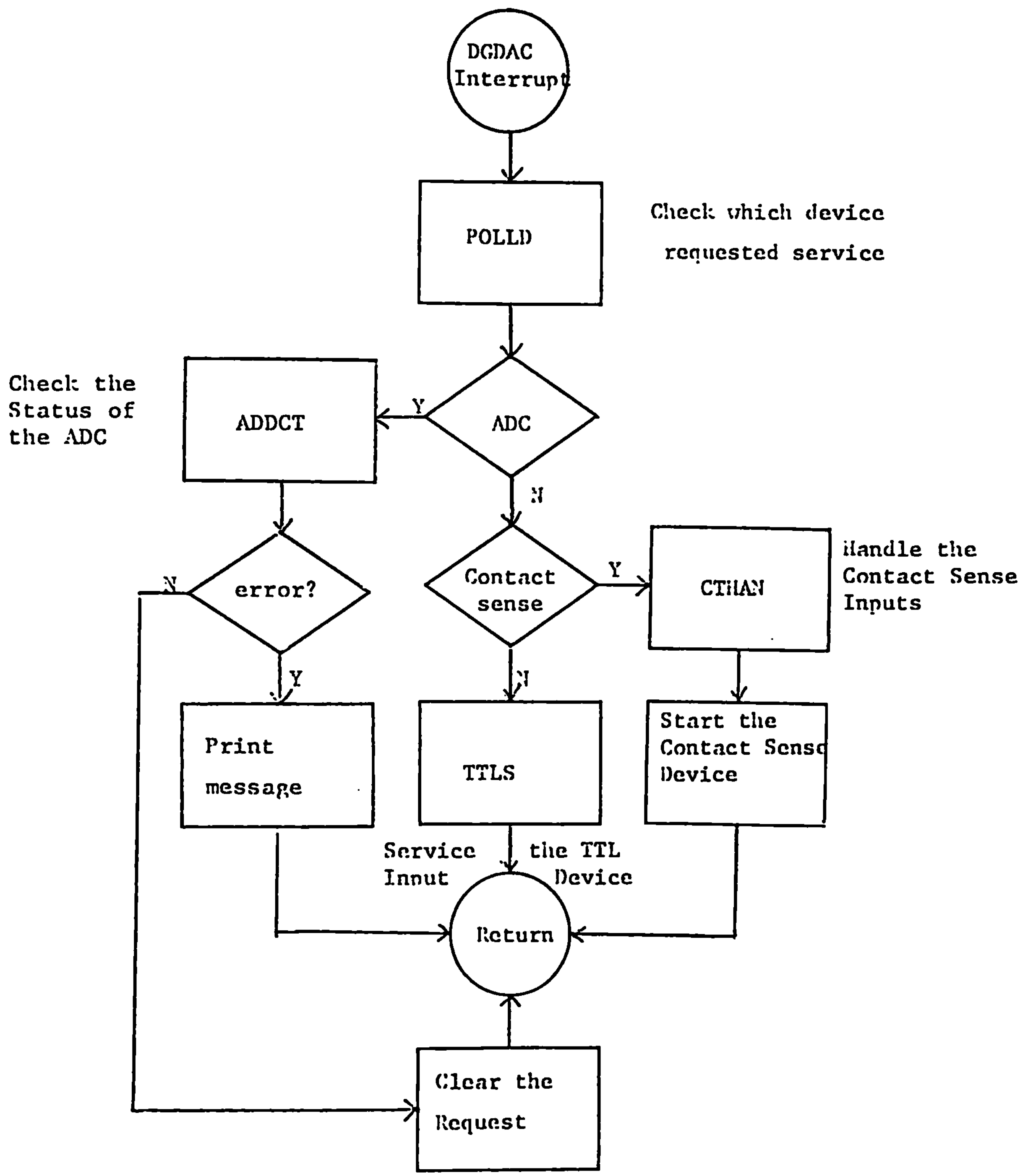

DGDAC
Interrupt
POLLD
Check which device
requested service
ADC
Y
N
ADDCT
Check the
Status of
the ADC
error?
N
Y
Print
message
Contact
sense
Y
N
CTHAN
Handle the
Contact Sense
Inputs
TTLS
Service
Input
the TTL
Device
Start the
Contact Sense
Device
Return
Clear the
Request

Figure 4

Changes to PREP

0001 PREF

```
                TITL PREF

        ;PREF TAPE # 1 OF 2

                ENTO OV14
                ENT EXC14
                EXTD TIMER
                EXTN TTYPE,TASKS,ALARM,DIVID,OPCOM,MESSG
                EXTN ERROR,DSKRD,MNITR,MESS,MODE,RANGE,DCLAR
                EXTN SUPRS,WEST,UEST,SCAN,MTPLX,CLOOP,UDEF

                IFE RDATA
                EXTN  TASK,  XMIT,  PRI
                ENDC

                IFN RDATA
                EXTN RCVE,.ITASK,XMIT,PRI
                EXTD RETUN
                ENDC

                IFN ALARM
                EXTN ALAR
                ENDC

                IFN DIDAS
                EXTN ANALG
                ENDC

                IFN ATODS
                EXTN DATUM
                ENDC

                IFN CNTCT
                EXTN COUNT
                ENDC

                END
```

```
.DEFINITION OF EXTERNAL SYMBOLS
.[illegible]  ADDR OF WORD DEFINING THE # RTC PULSES IN ONE
.             REAL TIME SECOND
.[illegible]  ADDR OF BYTE POINTER TO THE AUXILIARY TELETYPE FILENAME
.[illegible]  ADDR OF THE USER TASK DEFN TABLE
.ALARM        ADDR OF THE USER ALARM DEFN TABLE
.[illegible]  MNEMONIC SUBROUTINE CALL TO THE SINGLE PRECISION UNSIGNED
.             DIVIDE ROUTINE
.[illegible]  POINTER TO ORIGN.MG FILENAME
.[illegible]  POINTER TO MESSG.MG FILENAME
.ERROR        MNEMONIC SUBROUTINE CALL TO THE RDOS SYSTEM ERROR ROUTINE
.[illegible]  MNEMONIC SUBROUTINE CALL TO THE DISK MESSAGE
              RETRIEVAL ROUTINE
.[illegible]  S.A. OF THE USER ALARM MONITOR
.[illegible]  ADDR OF INTER-TASK COMMUNICATION WORD
.[illegible]  ADDR OF WORD INDICATING THE PROGRAM MODE(ZERO=BACKGROUND,
.             NON-ZERO=FOREGROUND)
.[illegible]  MNEMONIC SUBROUTINE CALL TO DETERMINE THE LOCATION OF
.             AN ITEM IN PROGRAM SPACE
.[illegible]  S.A. OF THE DECLARATION OF USER INTERRUPTING DEVICES
.[illegible]  S.A. OF USER SHUTDOWN ROUTINE
```

```
COMM  PAGE
         .[illegible]   S.A. OF USER BUILT SYMBOL TABLE
         .UEST          END ADDR OF USER BUILT SYMBOL TABLE
          [illegible]   WORD SPECIFYING THE FREQUENCY OF THE USER CLOCK
          [illegible]   S.A. OF THE USER CLOCK INTERRUPT ROUTINE
         .CLOOP         S.A. OF THE CONTROL LOOP DEFN TABLE
         .[illegible]   S.A. OF THE USER DEFINED EXECUTIVE FUNCTION
                        LIST [illegible].
          TASK          TASK CALL TO CREATE A PARALLEL USER TASK
         .[illegible]   TASK CALL TO TRANSMIT A MESSAGE TO ANOTHER ROUTINE
         .              AND AWAIT ITS RECEPTION
         .[illegible]   TASK CALL TO CHANGE THE PRIORITY OF THE CALLER
         .FCALL         [illegible] SUBROUTINE CALL TO FORTRAN IV
         .ITASK         S.A. OF THE FORTRAN IV ROUTINE TO CREATE A
         .              PARALLEL USER TASK
          [illegible]   S.A. OF THE FORTRAN IV ROUTINE TO TRANSMIT A
                        MESSAGE AND WAIT FOR RECEPTION
         .[illegible]   S.A. OF THE FORTRAN IV ROUTINE TO CHANGE
         .              THE PRIORITY OF THE CALLER
          [illegible]   EXECUTIVE'S FORTRAN IV RUN TIME STACK POINTER
         .[illegible]   ADDR OF THE RELAY BIT STATUS TABLE, STARTING AT BIT 0
         .[illegible]   ADDR OF D/A OUTPUT DATA BLOCK, STARTING AT CHANNEL 0
         .[illegible]   ADDR OF THE A/D READ DATA STORAGE, STARTING AT CHANNEL 0
         .[illegible]   ADDR OF THE CONTACT SENSE DISPATCH TABLE, STARTING AT BIT 0
          [illegible]   ADDR OF THE FIRST TTL INPUT STATUS TABLE, STARTING AT BIT 0
         .[illegible]   ADDR OF THE SECOND TTL INPUT STATUS TABLE, STARTING AT BIT 0
          [illegible]   ADDR OF THE FIRST TTL OUTPUT STATUS TABLE, STARTING AT BIT 0
         .[illegible]   ADDR OF THE SECOND TTL OUTPUT TABLE, STARTING AT BIT 0
```

```
;WRITTEN OCTOBER, 1974 BY
;         JEAN-PIERRE TREMBLAY
;         DEPARTMENT OF CHEMICAL ENGINEERING
;         MCMASTER UNIVERSITY
;         HAMILTON, ONTARIO, CANADA
;         CHANGED AUGUST,1977 BY DAVID SIMMONS

;WRITTEN IN DGC 3 NOVA LINE ASSEMBLER LANGUAGE FOR
;RDOS COMPATIBILITY

;         PROGRAM PREP

;CALLING SEQUENCE
;         JSR @XCI4
          RETURN

;ROUTINE TO CHECK AND PREPARE USER AND
;EXECUTIVE CORE SPACE FOR OPERATION IN USER MODE

 ENTRY
;POINTER TO A WORK BLOCK MUST BE PROVIDED IN CALLER'S USP

 EXIT
;RETURN TO THE RDOS CLI OCCURS IF AN ERROR
;IN THE SUPPLIED INFORMATION IS DETECTED.  OTHERWISE, THE
;USER PACKAGE BECOMES ACTIVATED.  AC3 RETURNS WITH C(USP)

 OPERATION
;(1) THE ROUTINE FIRST DETERMINES THE REAL TIME CLOCK
;       PARAMETER:  IF ZERO, THE ERROR RETURN TO
;       THE RDOS CLI IS TAKEN.  IF NON-ZERO, THE # RTC PULSES PER
```

```
        .     SECOND IS STORED IN "TIMER"   THE 'OPCOM.MG" FILE IS CHECKED
        .     IF IT DOES NOT EXIT, THE ERROR RETURN TO THE
        .     RDOS CLI IS TAKEN
        .(2) THE RDOS INTERRUPTS TO THE PROGRAM ARE DISABLED, THE CONSOLE
        .     FILENAME IS WRITTEN TO THE "CONSL" BUFFER.  IF AN AUXILIARY
        .     TLELTYPE HAS BEEN CONFIGURED, ITS FILENAME IS WRITTEN TO
        .     THE "TTYPE" BUFFER
        .(3) THE PROGRAM MODE IS DISPLAYED TO THE OPERATOR.  THE 'MODE'
        .     ADDR IS SET TO ZERO FOR BACKGROUND AND NON-ZERO FOR
        .     FOREGROUND
        .(4) THE EXISTENCE OF A SYMBOL TABLE IS VERIFIED   IF
        .     NONE EXISTS, A MESSAGE IS OUTPUT AND RETURN TO THE
        .     RDOS CLI OCCURS
        .(5) IN TURN  THE INFORMATION FOR THE RELAYS, A/D CONVERTERS
        .     AND CONTACT SENSES IS STUDIED   IF ANY ERRORS ARE DETECTED,
        .     A MESSAGE IS OUTPUT AND THE ERROR FLAG IS SET
        .     IF THE DEVICE INFORMATION IS OK, THEN THE
        .     THE A/D AND CONTACT SENSE DEVICES ARE
        .     INTRODUCED TO RDOS AND THE USER CLOCK IS DEFINED
        .(6) NEXT, THE USER TASK DEFINITION TABLE IS STUDIED   IF IT
        .     HAS ZERO LENGTH, OR IF THE S.A., PRIORITY, ID OR TASK
        .     KILL DATA IS IN ERROR, A MESSAGE IS OUTPUT AND THE ERROR
        .     FLAG IS SET
        .(7) THE ALARM DEFINITION TABLE IS EXAMINED.  IF IN ERROR OR IF
        .     THE FILE "ALMSG.MG" IS NOT FOUND, A MESSAGE IS OUTPUT AND
        .     THE ERROR FLAG IS SET.
        .(8) IF THE ERROR FLAG IS SET, THEN A RETURN TO THE
        .     RDOS CLI OCCURS
        .(9) THE EXECUTIVE IS ASSIGNED A NON-ZERO ID NUMBER
        .     THE ALARM MONITOR TASK IS CREATED
        .(10) AT THIS POINT, ALL IS READY TO RUN   THE OPERATOR IS
        .     PROMPTED TO INITIATE ACTIVITY BY STRIKING A KEY
        .     THE EXECUTIVE IS NOW IN CONTROL OF THE SYSTEM RESOURCES
```

```
00000 055403 EXCI4    STA 3 @ -3       ;SAVE RETURN ADDR

                 ;REAL TIME CLOCK CHECK
00001 006017             .SYSTM          ;GET RTC FREQ
00002 021000             .GHRZ
00003 000000             FADDR
00004 030437             LDA 2 E1004     ;GET ERROR CODE
00005 101005             MOV 0 0 SNR     ;GET LOOP COUNTER
00006 077777             ERROR           ;NO RTC
00007 030412             LDA 2 PULSE     ;GET ADDR OF FREQUENCY TABLE
00010 113000             ADD 0 2         ;FIND ENTRY
00011 025000             LDA 1 0 2       ;GET IT
00012 [illegible]        STA 1 TIMER     ;INSTALL THE #

                 ;OPCOM MG FILENAME CHECK
00013 [illegible]        LDA 0 @ -10     ;GET POINTER TO OPCOM MG FILENAME
00014 [illegible]        LDA 1 SCRAT     ;GET ADDR OF WRITE SPACE
00015 006017             .SYSTM          ;GET FILE'S INFO
00016 [illegible]        .STAT
00017 [illegible]        JMP .+2
00020 [illegible]        JMP DSBLE       ;FILENAME OK
00021 [illegible]        LDA 2 E1005     ;GET ERROR CODE
00022 [illegible]        ERROR
00023 077777             OPCOM
```

```
  0004  PAEP
01 00024 000000 SCRAT    WORK
02 00025 000025 PULSE    PULSE
03 00026 000012          10
04 00027 000144          100
05 00030 001750          1000
06 00031 000074          60
07 00032 000036          30
08 00033 002004 EI004    2004
09 00034 002005 EI005    2005
10 00035 000400 BYTST    400
11
12                 .DISABLE RDOS INTERRUPTS
13 [illegible] [illegible] ENABLE     SYSTM             ,DISABLE RDOS INTERRUPTS TO THE PROGRAM
14 [illegible] [illegible]            ODIS
15 [illegible] [illegible]           MODE
16
17                 .SET UP CONSOLE AND AUXILIARY TELETYPE
18 [illegible] [illegible]           LDA 0 [illegible] 3  ,GET BYTE POINTER TO CONSOLE FILENAME
19 [illegible] [illegible]            SYSTM             ,WRITE FILENAME TO BUFFER
20 [illegible] [illegible]            .GCOUT
21 [illegible] [illegible]           TTYPE
22 [illegible] [illegible]           LDA 1 [illegible]        ,GET CURRENT S.A. OF UST
23 [illegible] [illegible]           LDA 2 [illegible]        ,GET S.A. OF BACKGROUND UST
24 [illegible] [illegible]           SUB 1 2
25 [illegible] [illegible]           STA 2 [illegible]        ,SAVE MODE FLAG
```

```
                      .IFN TTYS-1
00051 022773          LDA 0 @ -5       ;GET POINTER TO AUXILIARY TELETYPE FILENAME
00052 006017           .SYSTM          ;WRITE FILENAME TO BUFFER
00053 021017           .GCOUT
00054 030400          030400           ;ASCII  <1><NULL>
00055 024777          LDA 1 .-1
00056 151004          MOV 2 2 SZR      ;BACKGROUND?
00057 122400          SUB 1 1          ;NO-$TTO IS AUXILIARY
00060 113210          MOVZR 0 3        ;YES-GET ADDR OF BUFFER
00061 040402          STA 1 2 3        ;INSTALL NECESSARY EXTENSION
                       .ENDC

00062 151005          MOV 2 2 SZR      ;FOREGROUND?
00063 000404          JMP .+4          ;YES
00064 077777          DSFMG            ;INDICATE BACKGROUND
00065 000000          MSG00
00066 000405          JMP .+5
00067 [illegible]     DSFMG            ;INDICATE FOREGROUND
00070 000000          MSG01
00071 102401          SUB 0 0 SKP
00072 000000          GOFLG
00073 042777          STA 0 @ .-1      ;ZERO EXECUTE FLAG

               ;USER BUILT SYMBOL TABLE CHECK
00074 [illegible]     LDA 1 POINT      ;GET S.A. OF SYMBOL TABLE
00075 [illegible]     LDA 0 POINT+1    ;GET END ADDR OF SYMBOL TABLE
00076 101113          MOVL# 0 0 SNC    ;IS UST DEFINED?
00077 [illegible]     MOVL# 1 1 SNC    ;IS UST DEFINED?
00100 [illegible]     JMP .+3          ;NO
00101 [illegible]     SUB 0 1 SZR      ;ANY LENGTH?
00102 [illegible]     JMP .+3          ;YES-OK
00103 [illegible]     DSFMG            ;INDICATE ERROR
00104 [illegible]     MSG02
```

```
  0003  PAGE
  00105 002101          JMP @ .+1        ,GO ABORT
  00106 000000          KILIT
  00107 102400          SUB 0 0
  00110 030000          LDA 2 K3
  00111 077777          DIVID            ,DETERMINE # OF SYMBOLS
  00112 101004          MOV 0 0 SZR      ,ANY REMAINDER?
  00113 [illegible]     JMP .+10         ,YES-ERROR

                 ;RELAY CHECK
                        .IFN RLAY1
  00114 [illegible]     LDA 1 .+10       ,GET ADDR OF RELAY BIT STATUS TABLE
  00115 077777          RANGE            ,TEST ADDR
  00116 000407          JMP .+7          ,BAD ADDR
  00117 000406          JMP .+6          ,BAD ADDR
  00120 [illegible]     LDA 1 INPUT+1    ,GET # RELAYS
  00121 [illegible]     LDA 2 .+3        ,GET S.A. OF RELAY TABLE
  00122 004000          JSR CLEAR        ,CLEAR TABLE
  00123 000403          JMP .+3          ,OK
  00124 077777          RELAY
  00125 [illegible]     DIAMG            ,INDICATE ERROR
  00126 [illegible]     MSG63
  00127 [illegible]     JSR @ [illegible] ,SET NO GO FLAG
                         ENDC
```

```
                                      ;CHECK FIRST DAC TABLE
                    IFN DTAS1
00130 024410        LDA 1  +10        ;GET ADDR OF FIRST D/A READ DATA STORAGE
00131 000110        RANGE             ;TEST ADDR
00132 000407        JMP   +7          ;BAD ADDR
00133 000406        JMP   +6          ;BAD ADDR
00134 024374        LDA 1 INFO+2      ;GET # OF FIRST D/A DEVICE CHANNELS
00135 030403        LDA 2   +3        ;GET S.A. OF FIRST D/A DEVICE TABLE
00136 004000        JSR CLEAR         ;CLEAR TABLE
00137 000405        JMP   +5          ;OK
00140 077777        ANALG             ;ADDR OF FIRST D/A ADDRESS TABLE
00141 000123        DSPMG             ;INDICATE ERROR
00142 000000        MSG76
00143 006374        JSR @ INFO+11     ;SET NO GO FLAG
                     ENDC

                    IFN DTAS2         ;CHECK SECOND DAC TABLE
00144 020414        LDA 0   +14       ;LOAD 4 INTO AC0
00145 024412        LDA 1   +12       ;LOAD ADDR OF FIRST D/A DEVICE TABLE
00146 107000        ADD 0 1           ;ADDR OF 2END D/A TABLE IS 4+ADDR OF FIRST TA
00147 000121        RANGE             ;TEST ADDR
00150 000411        JMP   +11         ;BAD ADDR
00151 000410        JMP   +10         ;BAD ADDR
00152 [illegible]   LDA 1 INFO+3      ;GET NO. OF SECOND D/A DEVICE CHANNELS
00153 [illegible]   LDA 2   +4        ;GET S.A. OF FIRST D/A DEVICE TABLE
00154 113000        ADD 0 2           ;S.A. OF 2END D/A TABLE IS 4+S.A. OF FIRST TA
00155 [illegible]   JSR CLEAR         ;CLEAR TABLE
00156 [illegible]   JMP   +4          ;OK
00157 [illegible]   ANALG             ;ADDR OF FIRST D/A ADDR TABLE
00160 [illegible]   4                 ;TO INCREMENT TO ADDR OF SECOND D/A ADDRESS T
00161 [illegible]   DSPMG             ;INDICATE ERROR
00162 [illegible]   MSG101
00163 [illegible]   JSR @ INFO+11     ;SET NO GO FLAG
                     ENDC
```

```
0003 FACF
                                   ;CHECK TTLI1 TABLE
                 IFN TL11
00164 024410     LDA 1  +10        ;GET ADDR OF TTLI1 DATA STORAGE AREA
00165 000147     RANGE             ;TEST ADDR
00166 000407     JMP   +7          ;BAD ADDR
00167 000406     JMP   +6          ;BAD ADDR
00170 024345     LDA 1 IMAG+5      ;GET NUMBER OF TTLI1 ENTRIES
00171 030405     LDA 2  +5         ;GET S A  OF TTLI1 TABLE
00172 004057     JSR CLEAR         ;CLEAR TABLE
00173 000403     JMP   +3
00174 000000     TTLI1             ;ADDRESS OF TABLE
00175 000151     DSPMG             ;INDICATE ERROR
00176 000000     MSG102
00177 006510     JSR @ IMAG+11     ;SET NO GO FLAG
                 ENDC

                                   ;CHECK TTLI2 TABLE
                 IFN TL21
00200 024410     LDA 1  +10        ;GET ADDR OF TTLI2 DATA STORAGE AREA
00201 000150     RANGE             ;TEST ADDR
00202 000407     JMP   +7          ;BAD ADDR
00203 000406     JMP   +6          ;BAD ADDR
00204 024350     LDA 1 IMAG+6      ;GET NUMBER O ENTRIES IN TTLI2 TABLE
00205 030405     LDA 2  +5         ;GET S A  OF TTLI2 TABLE
00206 [illegible] JSR CLEAR        ;CLEAR TABLE
00207 000403     JMP   +3
00210 000000     TTLI2             ;ADDRESS OF TABLE
00211 000175     DSPMG             ;INDICATE ERROR
00212 000000     MSG103
00213 [illegible] JSR @ IMAG+11    ;SET NO GO FLAG
```

```
                        ENDC

                                            ;CHECK TTL01 TABLE
                         IFN TL10
00214 024410            LDA 1  +10          ;GET ADDR OF TTL01 DATA STORAGE AREA
00215 000201            RANGE               ;TEST ADDR
00216 000407            JMP   +7            ;BAD ADDR
00217 000406            JMP   +6            ;BAD ADDR
00220 034510            LDA 1 1WRO+7        ;GET NUMBER OF TTL01 ENTRIES
00221 030403            LDA 2  +3           ;GET S.A. OF TTL01 TABLE
00222 004577            JSR CLEAR           ;CLEAR TABLE
00223 000405            JMP   +5
00224 000000            TTL01               ;ADDRESS OF TABLE
00225 000211            DSKMG               INDICATE ERROR
00226 000000            MSG104
00227 006510            JSR @ 1WRO+11       ;SET NO GO FLAG
                         ENDC

00230 000000                                CHECK TTL02 TABLE
                         IFN TL20
00231 024410            LDA 1  +10          ;GET ADDR OF TTL02 DATA STORAGE AREA
00232 000201            RANGE               ;TEST ADDR
00233 000407            JMP   +7            ;BAD ADDR
00234 000406            JMP   +6            ;BAD ADDR
00235 034501            LDA 1 1WRO+10       ;GET NUMBER OF TTL02 ENTRIES
00236 030403            LDA 2  +3           ;GET S.A. OF TTL02 TABLE
00237 004512            JSR CLEAR           ;CLEAR TABLE
00240 000405            JMP  +5
00241 000000            TTL02               ;ADDRESS OF TABLE
```

```
   0007  FREP
   00242 000111            DSNMG                  ;INDICATE ERROR
   00243 000000            MSG105
   00244 006473            JSR @ INFO+11          ;SET NO GO FLAG
                            ENDC

                  ;A/D CONVERTER CHECK
                            IFN ATODS
   00245 024410            LDA 1  +10             ;GET ADDR OF A/D READ DATA STORAGE AREA
   00246 000232            RANGE                  ;TEST ADDR
   00247 000407            JMP  +7                ;BAD ADDR
   00250 000406            JMP  +6                ;BAD ADDR
   00251 034475            LDA 1 INFO             ;GET # A/D'S
   00252 030403            LDA 2  +3              ;GET S.A. OF A/D TABLE
   00253 004476            JSR CLEAR              ;CLEAR TABLE
   00254 000403            JMP  +3
   00255 077777            DATUM
   00256 000242            DSNMG                  ;INDICATE ERROR
   00257 000000            MSG64
   00260 006457            JSR @ INFO+11          ;SET NO GO FLAG
                            ENDC

                  ;CONTACT SENSE CHECK
                            IFN CNTCT
   00261 024420            LDA 1  +22             ;GET ADDR OF CONTACT SENSE DISPATCH TABLE
   00262 044475            STA 1 POINT            ;INSTALL THE POINTER
   00263 000232            RANGE                  ;TEST ADDR
   00264 000420            JMP  +20               ;BAD ADDR
   00265 000417            JMP  +17               ;BAD ADDR
```

```
00266 024443        LDA 1 INFO+3       ;GET # CONTACT SENSE IN SYSTEM
00267 044452        STA 1 POINT+1      ;INSTALL COUNTER
00270 026450        LDA 1 @ POINT      ;GET A DISPATCH ADDR
00271 000233        RANGE              ;TEST IT
00272 000415        JMP  +15           ;BAD ADDR
00273 000414        JMP  +14           ;BAD ADDR
00274 131000        MOV 1 2
00275 102400        SUB 0 0
00276 041000        STA 0 0 2          ;ZERO RESOLUTION ADDR
00277 010441        ISZ POINT          ;INCR POINTER
00300 014441        DSZ POINT+1        ;DONE?
00301 000767        JMP  -11           ;NO
00302 000410        JMP  +10
00303 077777        SENSE
00304 000236        DSPMG              ;INDICATE ERROR
00305 000000        MSG1
00306 000403        JMP  +3
00307 000304        DSPMG              ;INDICATE ENTRY ERROR
00310 000000        MSG27
00311 006426        JSR @ INFO+11      ;SET NO GO FLAG
                     ENDC

              ;DECLARE USER INTERRUPTING DEVICES
00312 024416        LDA 1 POINT+10     ;GET S A  OF DECLARATION OF USER DEVICES
00313 125113        MOVL# 1 1 SNC      ;IS ADDR OK?
00314 000404        JMP  +4            ;OK
00315 000307        DSPMG              ;INDICATE ERROR
00316 000000        MSG29
00317 006426        JSR @ INFO+11
00320 020000        LDA 0 GOFLG
00321 101004        MOV 0 0 SZR        ;IS NO GO FLAG SET?
```

```
  0006  FACP
U00322 '000002           JMP NOCLK+2     ,YES
 00323 '006425           JSR @ POINT+10  ,NO
U00324 '000000           JMP NEXT

 00325 '000003 K3:       3

U00326 '000000 INFO      ATODS           ,NUMBER OF ENTRIES IN ADC TABLE
U00327 '000000           RLAYS           ,# RELAYS
U00330 '000000           DTAS1           ,# FIRST DAC
U00331 '000000           DTAS2           ,# SECOND DAC
U00332 '000000           CNTCT           ,# CONTACT SENSES
U00333 '000000           TL1I            ;# FIRST TTL INPUTS
U00334 '000000           TL2I            ,# SECOND TTL INPUTS
U00335 '000000           TL1O            ,# FIRST TTL OUTPUTS
U00336 '000000           TL2O            ,# SECOND TTL OUTPTS
U00337 '000000           NOGO

 00340 077777 POINT      USST
 00341 077777            UEST
 00342 077777            SCAN
 00343 077777            MTPLX
 00344 077777            SUPRS
 00345 077777            TASKS
 00346 077777            ALARM
 00347 077777            MESSG
 00350 077777            DCLAR
```

```
00351'102400 CLEAR   SUB 0 0
00352'124405         NEG 1 1 SNR
00353'001400         JMP 0 3
00354'041000         STA 0 0 2
00355'151400         INC 2 2
00356'125404         INC 1 1 SZR
00357'000775         JMP  -3
00360'001400         JMP 0 3

00361'000000 UPDTB   WORK

                      EOT              ;END OF PREP TAPE # 1 OF 2
```

```
  GOOF   FREF

ALARM    000346 X    8/24
ANALG    000157 X    5/35   5/53
ATODS    000000U     1/29   7/07   8/07
BLUST    000035      4/10   4/23
CHECK    000000U     8/49
CLEAR    000351      5/17   5/33   5/51   6/09   6/25   6/41   6/57   7/14
                     8/28
CLOOF    077777 X
CNTCT    000000U     1/33   7/23   8/11
DATUM    000235 X    7/18
DCLAR    000350 X    8/26
DEMUX    000000U     4/18
DIVID    000111 X    3/05
DTLL2    000036      3/55   4/13
DSPMS    000313 X    4/41   4/44   4/56   5/20   5/36   5/55   6/12   6/28
                     6/44   7/01   7/17   7/43   7/46   7/55
DTAS1    000000U     5/23   8/09
DTAS2    000000U     5/41   8/10
DTOAS    000000U     1/29
E1004    000073      3/41   4/05
E1005    000054      3/54   4/07
ERROR    000022 X    3/43   3/57   6/44
EXC14    000000      3/30
FCALL    077777 X
FORT4    000000U     1/12   1/16
FETCH    000002 X
GOFLG    000000U     4/47   7/56
INDIC    000000U     2/41
INPUT    000326      5/15   5/22   5/31   5/38   5/48   5/57   6/07   6/14
                     6/23   6/30   6/37   6/46   6/55   7/03   7/12   7/19
                     7/29   7/48   7/57   8/07
```

```
ITASK   077777 X
KS      000325      5/04    6/05
KILIT   000000U     5/02
MESS    077777 X
MESSG   000347 X    8/25
MWITR   077777 X
MODE    000040 X    4/15
MSG10   000000U     5/56    6/15    6/29    6/45    7/02
MSG60   000000U     4/42
MSG61   000000U     4/45
MSG62   000000U     4/59
MSG63   000000U     5/21
MSG64   000000U     7/18
MSG67   000000U     7/44
MSG65   000000U     7/47
MSG46   000000U     5/37
MSG99   000000U     7/56
NTFLX   000343 X    6/21
NEXT    000000U     6/03
NOCLK   000000U     8/01
NOGO    000000U     8/16
OPCOM   000023 X    3/58
OVL14   000000
POINT   000340      4/51    4/52    7/25    7/30    7/31    7/36    7/39    7/52
                    8/02    8/18
PRI     077777 X
PULSE   000023      3/44    4/02    7/02
PADDR   000000U     3/40
```

0010 FREP

RANGE	000271 X	5/12	5/28	5/45	6/04	6/20	6/36	6/52	7/09
		7/25	7/32						
RELAY	000124 X	5/19							
RLAYS	000000U	1/21	5/10	8/08					
SCAN	000342 X	8/20							
SCRAT	000024	3/31	4/01						
SENSR	000305 X	7/42							
SUPRS	000344 X	8/22							
TABLE	000000U	6/49							
TASKS	000343 X	8/28							
TIMER	000012 X	3/47							
TL11	000000U	6/02	8/12						
TL10	000000U	6/34	8/14						
TL21	000000U	6/18	8/13						
TL20	000000U	6/30	8/15						
TTL11	000000U	6/11							
TTL12	000000U	6/27							
TTL01	000000U	6/43							
TTL02	000000U	6/49	6/59						
TTYPE	000034 X	4/21							
TTYS	000000U	4/27							
UDEF	077777 X								
UEST	000341 X	5/19							
UFLAG	000361	8/37							
USST	000340 X	8/18							
USTR	000000U	4/22							
UDAL	000000U	4/01	8/37						
[illegible]	077777 X								
[illegible]	[illegible]								
[illegible]	[illegible]								
[illegible]	[illegible]								

Figure 5

Changes to PTAPE

```
                ;GOSEX PARAMETER TAPE

                ;USERS OF THE GOSEX EXECUTIVE MUST CONFIGURE ALL OF
                ;THE EXECUTIVE PROGRAMS WITH THIS PARAMETERS TAPE
                ;THE USER MAY SPECIFY THE VALUE OF A CERTAIN NUMBER
                ;OF THE PARAMETERS HEREIN.  THESE PARAMETERS ARE.
                ;        LPTS     -MIN=0,MAX=2
                ;        TTYS     -MIN=1,MAX=2
                ;        PTPS     -MIN=0,MAX=2
                ;        DKPC     -MIN=1,MAX=8
                ;        CASS     -MIN=0,MAX=8
                         MAGS     -MIN=0,MAX=8
                ;        PORTA
                ;        NV[illegible]    -MIN=0,MAX=[illegible]

                ;ASIDE FROM THE ABOVE PARAMETERS, THE ONLY OTHER ONES THAT MAY
                ;BE REASSIGNED NEW VALUES ARE THOSE WHICH APPLY TO THE
                ;PERIPHERAL DEVICES INTERFACING THE CPU TO THE PILOT PLANT
                ;THESE ARE LISTED AT THE END OF THIS PARAMETER TAPE

                ;DISPLACEMENTS IN THE GENERAL WORK SPACE

        000012  DUSR    GENAL=  10      ;SIZE OF GENERAL WORK SPACE
        000000  DUSR    RETUR=  0       ;RETURN ADDRESS STORAGE
        000001  DUSR    RETAD=  1       ;RETURN ADDR STORAGE
        000002  DUSR    BYTEP=  2       ;BYTE POINTER STORAGE
        000003  DUSR    COUNT=  3       ;COUNTER
        000004  DUSR    KOUNT=  4       ;COUNTER
        000005  DUSR    WORD1=  5       ;TEMPORARY STORAGE
        000006  DUSR    WORD2=  6       ;TEMPORARY STORAGE
        000007  DUSR    WORD3=  7       ;TEMPORARY STORAGE
        000010  DUSR    WORD4=  8       ;TEMPORARY STORAGE
        000011  DUSR    WORD5=  9       ;TEMPORARY STORAGE
```

```
                ;DISPLACEMENTS USED FOR THE LOG AND PLOT ROUTINES

        000012  DUSR    LOOP1=  GENRL   ;LOOP COUNTER
        000013  DUSR    LOOP2=  GENRL+1 ;LOOP COUNTER
        000014  DUSR    SPOT1=  GENRL+2 ;POINTER
        000015  DUSR    SPOT2=  GENRL+3 ;POINTER
        000016  DUSR    NDATA=  GENRL+4 ;# DATA TO LIST
        000017  DUSR    INTVL=  GENRL+5 ;# SECS BETEEN RECORDS
        000020  DUSR    CHANL=  GENRL+6 ;CHANNEL # USED FOR OUTPUT
        000021  DUSR    LNCNT=  GENRL+7 ;PAGE LINE COUNT
        000022  DUSR    HOUR=   GENRL+8         ;STARTING HOUR OF RECORD
        000023  DUSR    MIN=    GENRL+9         ;STARTING MINUTE OF RECORD
        000024  DUSR    SEC=    GENRL+10        ;STARTING SECOND OF RECORD
        000025  DUSR    TMSHO=  GENRL+11        ;COUNTER FOR TIME OF DAY DISPLAY
        000026  DUSR    MNSIZ=  GENRL+12        ;LOWER RANGE VALUE OF PLOT
        000027  DUSR    MXSIZ=  GENRL+13.       ;UPPER RANGE VALUE OF PLOT
        000030  DUSR    DNAME=  GENRL+14        ;BYTE POINTER OF DEVICE FILENAME
        000031  DUSR    UNIT=   GENRL+15.       ;DEVICE UNIT #
        000032  DUSR    ADSYM=  GENRL+16        ;ADDR OF POINTER TO SYMBOLS LIST
        000033  DUSR    ADVAL=  GENRL+17        ;ADDR OF POINTER TO VALUES TABLE
        000034  DUSR    ADBUF=  GENRL+18.       ;BYTE POINTER TO WRITTING BUFFER
        000035  DUSR    MXVAR=  GENRL+19        ;MAX # VARIABLES FOR OUTPUT BUFFER
        000036  DUSR    RCHAN=  GENRL+20        ;CHANNEL # FOR READ BY COPY TASK
```

```
                ;DISPLACEMENTS IN THE DEVICE STATUS TABLE

        000000  .DUSR   TASSA=  0           ;DEVICE STORAGE BLOCK S.A
        000001  .DUSR   IDENT=  1           ;TASK IDENTIFICATION NUMBER
        000002  .DUSR   PRIOR=  2           ;TASK PRIORITY

                ;CONSTANTS WHICH ARE USED TO DEFINE DISPLACEMENTS
                ;FOR THE DISK MESSAGE ROUTINE

        000012  .DUSR   TMESS=  GENAL       ;TOTAL NUMBER OF MESSAGES IN DISK FILE
        000013  .DUSR   MXLGT=  GENAL+1     ;MAXIMUM MESSAGE LENGTH PERMITTED
        000014  .DUSR   DSKCL=  GENAL+2     ;CHANNEL # TO DISK FILE
        000015  .DUSR   DSKNM=  GENAL+3     ;BYTE POINTER TO DISK FILE NAME
        000016  .DUSR   DEVCL=  GENAL+4     ;CHANNEL # TO OUTPUT DEVICE
        000017  .DUSR   DEVNM=  GENAL+5     ;BYTE POINTER TO OUTPUT DEVICE NAME
        000020  .DUSR   BUFFR=  GENAL+6     ;BYTE POINTER TO BUFFER AREA OF LENGTH MXLGT
        000000  .DUSR   HIBYP=  0           ;HIGH ORDER BYTE POINTER TO MESSAGE
        000001  .DUSR   LOBYP=  1           ;LOW ORDER BYTE POINTER TO MESSAGE
        000002  .DUSR   LNGTH=  2           ;MESSAGE LENGTH IN BYTES

                ;DISPLACEMENTS IN THE USER TASKS DEFINITION BLOCKS.
                ;EACH BLOCK IS FOUR WORDS LONG AND EACH USER TASK HAS
                ;ONE BLOCK DEFINING ITS NECESSARY ATTRIBUTES.

        000000  .DUSR   TSKSA=  0           ;S.A. OF THE TASK
        000001  .DUSR   TSKPR=  1           ;PRIORITY OF THE TASK(1 TO 377-NEVER 0)
        000002  .DUSR   TSKID=  2           ;TASK IDENTIFICATION CODE-1 TO 300,
                                            ;NEVER 0 OR ABOVE 300
        000003  .DUSR   TSKIL=  3           ;TASK'S KILL FLAG
                ;       THE TASK DEFINITION BLOCK IS EXTENDED BY ONE WORD
                ;       IF THE FORT4 PARAMETER, DEFINED BELOW IS NON-ZERO
        000004  .DUSR   TSKMD=  4           ;TASK MODE-IF FORTRAN IV TASK, ZERO
                                            ;           -IF ASSEMBLER ONLY, NON-ZERO
```

```
                ;THE FOLLOWING CONSTANTS DEFINE THE SYSTEM CONFIGURATION
                ;ZERO            DEVICE NOT IN THE SYSTEM
                ;NON-ZERO        NUMBER OF DEVICES CONFIGURED IN SYSTEM

        000001  .DUSR   LPTS=   1           ;LINE PRINTER
        000001  .DUSR   TTYS=   1           ;# TELETYPES ON CPU
        000000  .DUSR   PTPS=   0           ;PAPER TAPE PUNCH
        000002  .DUSR   DKFL=   2           ;DISK FILES
        000000  .DUSR   CASS=   0           ;CASSETTE TAPE FILE
        000000  .DUSR   MAGS=   0           ;MAGNETIC TAPE FILE

                ;RDOS DISPLACEMENTS USED BY THE GOEX PACKAGE

        000012  .DUSR   USTP=   12          ;POINTER TO CURRENT PROGRAM'S UST
        000013  .DUSR   TID=    13          ;TASK ID IN TASK'S TCB
        000014  .DUSR   USTCT=  14          ;CURRENT TCB POINTER TO PROGRAM'S UST
        000017  .DUSR   USTIN=  17          ;START OF USER NREL CODE IN UST


                ;FORTRAN IV PARAMETER
                ;WHEN ZERO, ALL TASKS MUST BE IN ASSEMBLER
                ;WHEN NON-ZERO, THIS PARAMETER SPECIFIES THE MAXIMUM NUMBER OF
                ;PARALLEL FORTRAN IV TASKS THAT MAY RUN AT ONE TIME.  THE
                ;EXECUTIVE MUST BE INCLUDED IN THIS COUNT.  THE TOTAL NUMBER
                ;OF TCB'S  IS STILL SPECIFIED BY THE RLDR LOCAL SWITCH "K"
```

```
        000002  DUSR PORT#=     1

                ;CONTROLLER BLOCK DISPLACEMENTS

        000012  DUSR    WRKSP=   GENRL    ,WORK SPACE-2 WORDS
        000014  DUSR    SIGN=    GENRL+2 ,SIGN STORAGE
        000015  DUSR    EROR0=   GENRL+3 ,ERROR AT T
        000016  DUSR    EROR1=   GENRL+4 ,ERROR AT T-1
        000017  DUSR    EROR2=   GENRL+5 ,ERROR AT T-2
        000020  DUSR    EROR3=   GENRL+6 ,ERROR AT T-3
        000021  DUSR    ITSUM=   GENRL+7 ;INTEGRAL SUM STORAGE
        000022  DUSR    OUTPT=   GENRL+8 ,CURRENT OUTPUT ACTION
        000023  DUSR    SMOTH=   GENRL+9 ,;FILTERED DATA STORAGE
        000024  DUSR    CNTHI=   GENRL+10 ,HIGH LIMIT ALARM COUNTER
        000025  DUSR    CNTLO=   GENRL+11 ,LOW LIMIT ALARM COUNTER
        000026  DUSR    FILTR=   GENRL+12 ,FILTER CONSTANT
        000027  DUSR    VABLE=   GENRL+13 ,ADDR OF RAW DATA
        000030  DUSR    INHI=    GENRL+14 ,INPUT HIGH LIMIT
        000031  DUSR    INLO=    GENRL+15 ,INPUT LOW LIMIT
        000032  DUSR    MESHI=   GENRL+16 ,MESSAGE # FOR HIGH LIMIT ALARM IN MESSG MG
        000033  DUSR    MESLO=   GENRL+17 ,MESSAGE # FOR LOW LIMIT ALARM IN MESSG MG
        000034  DUSR    AWAIT=   GENRL+18 ,INTERVAL BETWEEN SUCCESSIVE ALARMS
        000035  DUSR    SETPT=   GENRL+19 ,SET POINT
        000036  DUSR    KP=      GENRL+20 ,PROPORTIONAL GAIN
        000037  DUSR    KPD=     GENRL+21 ,PROPORTIONAL DERIVATIVE GAIN
        000040  DUSR    KPI=     GENRL+22 ,PROPORTIONAL INTEGRAL GAIN
        000041  DUSR    SCALE=   GENRL+23 ,SCALE FACTOR WITH ACTION DIRECTION
        000042  DUSR    DELAY=   GENRL+24 ,TIME INTERVAL BETWEEN CONTROL ACTIONS
        000043  DUSR    CASCD=   GENRL+25 ,CASCADE FLAG
        000044  DUSR    DESTN=   GENRL+26 ,ADDR OF DESTINATION DATA BLOCK OR
                                          ,D/A CHANNEL #
```

```
        000045  DUSR    OUTHI=  GENRL+27 ,HIGH LIMIT ON OUTPUT
        000046  DUSR    OUTLO=  GENRL+28 ,LOW LIMIT ON OUTPUT
        000047  DUSR    MEAN=   GENRL+29 ,MEAN OUTPUT VALUE
        000050  DUSR    SPCON=  GENRL+30 ,ADDR OF CONSTANT BLOCK FOR SETPT CONVERSION
        000051  DUSR    SPAZE=  GENRL+31 ,ADDR OF ROUTINE TO CONVERT FROM
                                         ,A/D TO ENGINEERING UNITS
        000052  DUSR    SPEZA=  GENRL+32 ;ADDR OF ROUTINE TO CONVERT FROM
                                         ,ENGINEERING TO A/D UNITS
        000053  DUSR    SPCAP=  GENRL+33 ,MESSAGE # FOR SETPT CAPTION IN OPCOM MG
        000054  DUSR    MNCON=  GENRL+34 ,ADDR OF CONSTANT BLOCK FOR MEAN CONVERSION
        000055  DUSR    MNAZE=  GENRL+35 ;ADDR OF ROUTINE TO CONVERT FROM
                                         ;D/A (OR A/D) TO ENGINEERING UNITS
        000056  DUSR    MNEZA=  GENRL+36 ,ADDR OF ROUTINE TO CONVERT FROM
                                         ;ENGINEERING TO D/A (OR A/D) UNITS
        000057  DUSR    MNCAP=  GENRL+37 ,MESSAGE # FOR MEAN CAPTION IN OPCOM MG
        000060  DUSR    LPMES=  GENRL+38 ,LOOP'S DESCRIPTIVE MESSAGE # IN OPCOM MG


                ,DEFINITION OF PERIPHERAL DEVICES INTERFACING CPU TO THE PILOT PLANT
                ,THE ANALOG DEVICES ARE ASSUMED TO COVER THE RANGE 0 TO 10 VDC.

        000030  DUSR    ATODI=  24       ,A/D CONVERTER-ANALOG INPUT
        000777  DUSR    RNIAG=  777      ,RANGE OF THE ANALOG INPUT DEVICE-POSITIVE
        000004  DUSR    DTAS1=  4        ,FIRST D/A CONVERTER-ANALOG OUTPUT
        000004  DUSR    DTAS2=  4        ;SECOND D/A CONVERTER-ANALOG OUTPUT
        000777  DUSR    ANORG=  777      ;RANGE OF THE ANALOG OUTPUT DEVICE-POSITIVE
        000020  DUSR    RLAYS=  16       ,RELAYS
```

```
000020  DUSR   CNTCT=   16       ,CONTACT SENSE
000020  DUSR   TL1I=    16       ,FIRST TTL INPUT DEVICE
000020  DUSR   TL2I=    16       ,SECOND TTL INPUT DEVICE
000020  DUSR   TL1O=    16       ,FIRST TTL OUTPUT DEVICE
000020  DUSR   TL2O=    16       ,SECOND TTL OUTPUT DEVICE

                                 ,STARTING CHANNELS OF THE DAC DEVICES
000000  DUSR   SCDA1=   0        ,STARTING CHANNEL OF THE FIRST DAC DEVICE
000000  DUSR   SCDA2=   0        ,STARTING CHANNEL OF THE SECOND DAC DEVICE

                                 ,SLOT IN WHICH EACH DEVICE IS LOCATED
000000  DUSR   SADC=    0        ,ADC SLOT
000001  DUSR   SVM1=    1        ,FIRST VOLTAGE MPX SLOT
000002  DUSR   SVM2=    2        ,SECOND VOLTAGE MPX SLOT
000003  DUSR   SCMP=    3        ,CURRENT MPX SLOT
000004  DUSR   SDAC1=   4        ,FIRST DAC SLOT
000005  DUSR   SDAC2=   5        ,SECOND DAC SLOT
000006  DUSR   STI1=    6        ,FIRST TTL INPUT SLOT
000007  DUSR   STI2=    7        ,SECOND TTL INPUT SLOT
000010  DUSR   STO1=    8        ,FIRST TTL OUTPUT SLOT
000011  DUSR   STO2=    9        ,SECOND TTL OUTPUT SLOT
000012  DUSR   SRLY=    10       ,RELAY SLOT
000013  DUSR   SCNT=    11       ,CONTACT SENSE SLOT
```

```
                                              ;INDICATE THE GAIN VALUES OF THE VOLTAGE MPX
000001  DUSR   GAIN1=  1                      ;FIRST VOLTAGE MULTIPLEXOR GAIN VALUE
000002  DUSR   GAIN2=  2                      ;SECOND VOLTAGE MULTIPLEXOR GAIN VALUE

        ;OUTPUT BUFFER SIZE DEFINITIONS
        ;WARNING:        MINIMUM BUFFER SIZE MUST BE 11. IF A
        ;                DEVICE IS CONFIGURED FOR USE

000017  DUSR   NVCO=   15                     ;CONSOLE
000000  DUSR   NVTT=   0                      ;AUXILLIARY TTY
000050  DUSR   NVLP=   40                     ;LINE PRINTER
000000  DUSR   NVPN=   0                      ;PAPER TAPE PUNCH
000050  DUSR   NVDK=   40                     ;DISK
000000  DUSR   NVCS=   0                      ;CASSETTE
000000  DUSR   NVMG=   0                      ;MAGNETIC TAPE
000000  DUSR   NVANO=  0                      ;ANALOG OUTPUTS.  THE CHART OUTPUT TASK CLAI
                                              ;THE FIRST NVANO ANALOG OUTPUT CHANNELS
                                              ;SPECIFIED BY DTOAS ABOVE.

               EOT                            ;END OF GOSEX PARAMETER TAPE
```

Figure 6

Flow of DCLAR

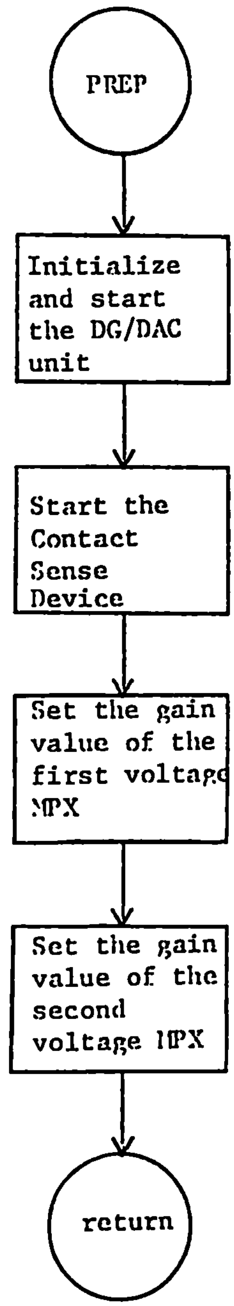
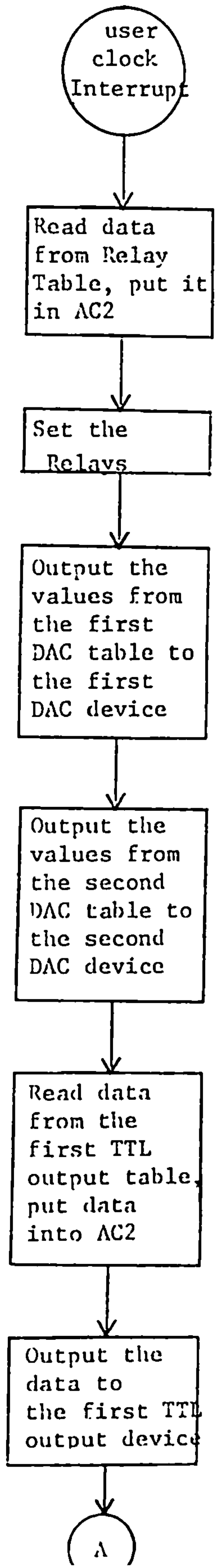
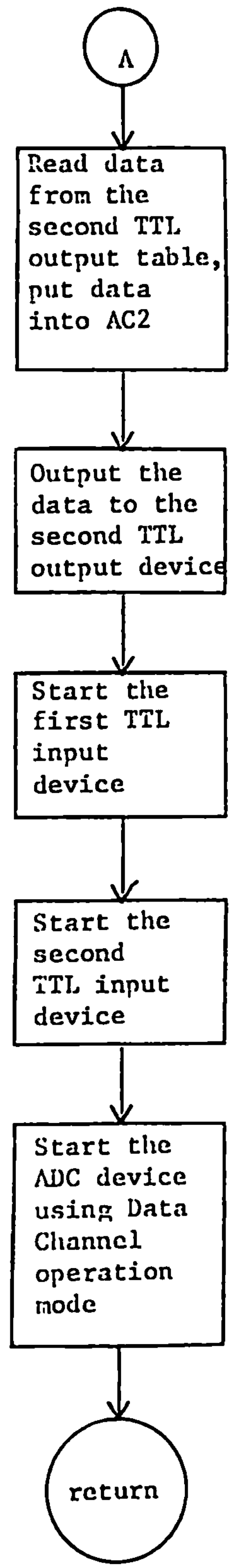
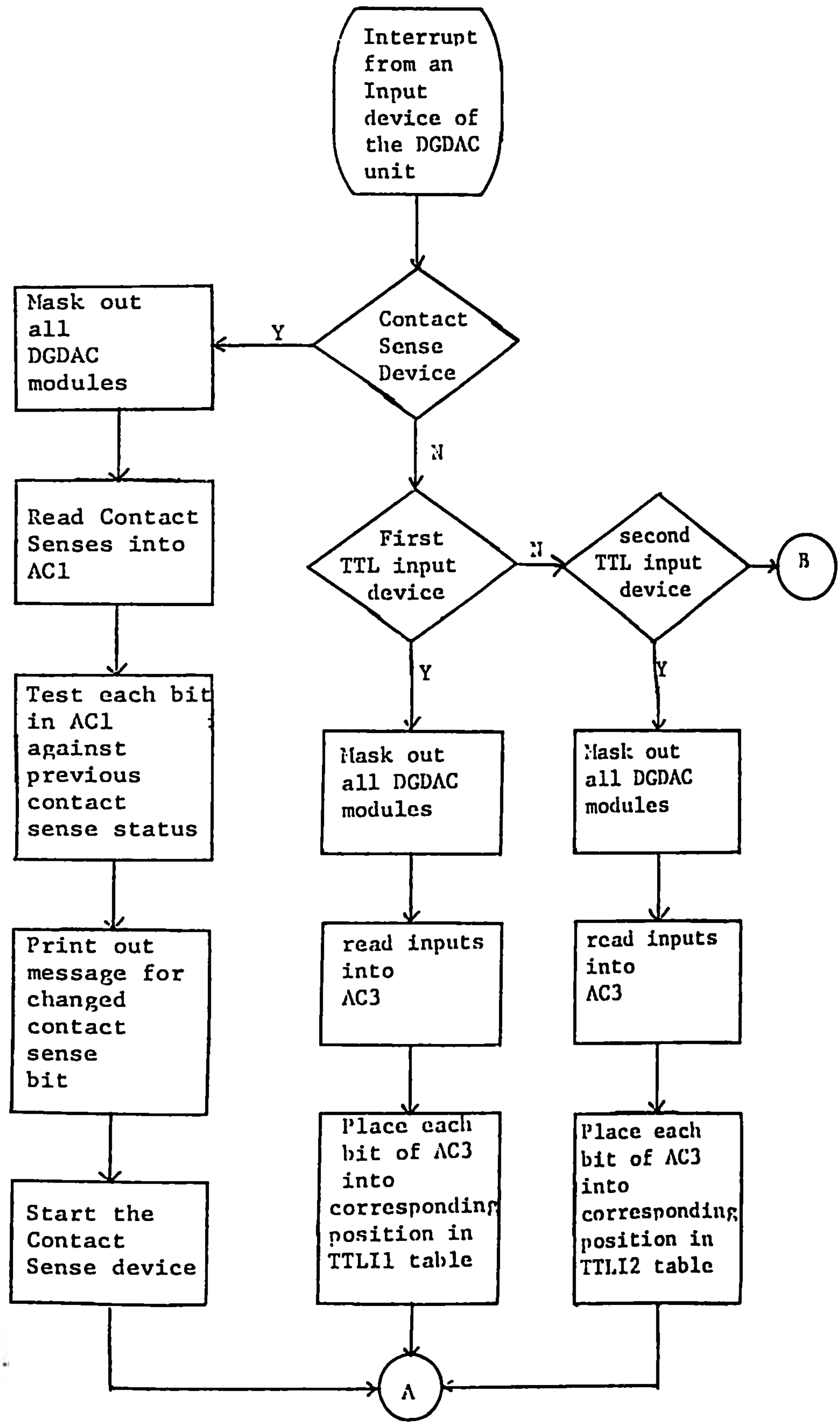


Figure 7 MTPLX - Output handler




A
Read data from the second TTL output table, put data into AC2
Output the data to the second TTL output device
Start the first TTL input device
Start the second TTL input device
Start the ADC device using Data Channel operation mode
return


Interrupt from an Input device of the DGDAC unit
Contact Sense Device
Y
N
Mask out all DGDAC modules
Read Contact Senses into AC1
Test each bit in AC1 against previous contact sense status
Print out message for changed contact sense bit
Start the Contact Sense device
First TTL input device
N
second TTL input device
B
Y
Y
Mask out all DGDAC modules
read inputs into AC3
Place each bit of AC3 into corresponding position in TTLI1 table
Mask out all DGDAC modules
read inputs into AC3
Place each bit of AC3 into corresponding position in TTLI2 table
A

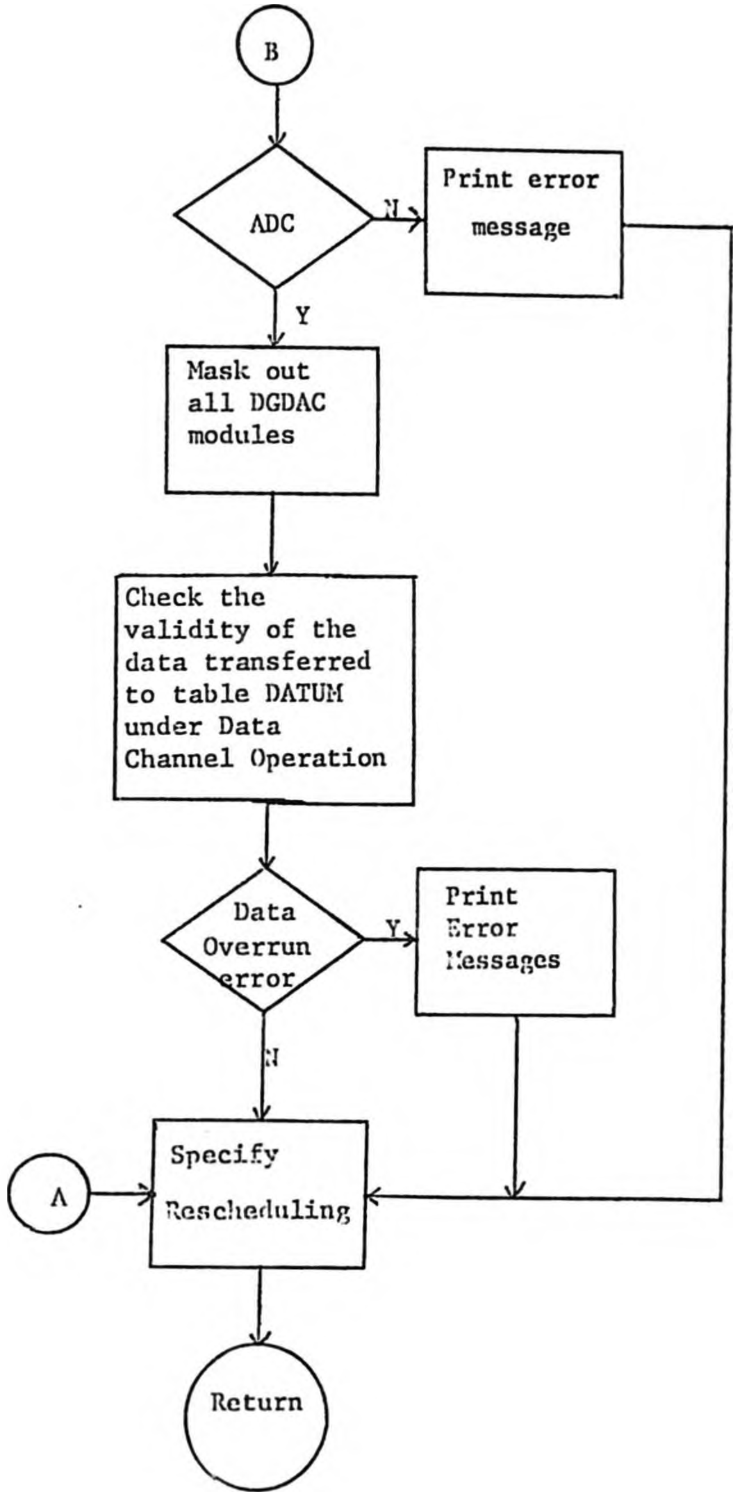

B
ADC
N
Print error message
Y
Mask out all DGDAC modules
Check the validity of the data transferred to table DATUM under Data Channel Operation
Data Overrun error
Y
Print Error Messages
N
A
Specify Rescheduling
Return

Figure 9

New MTPLX Routine

```
        TITL MTFLX

                ;DGDAC INITIALIZATION , USER CLOCK , AND
                ;DEVICE INTERRUPT HANDLERS

        ENT DCLAR,MTFLX
        EXTN RELAY,DATUM,SENCE,ANALG,TTLO1,TTLO2,TTLI1,TTLI2
        EXTN ERROR, IXMT, UCEX, UIEX

        NREL

                ;DEFINITION OF ENTRY SYMBOLS
;MTFLX   S.A.  OF THE USER CLOCK ROUTINE
;DCLAR   S.A.  OF THE DECLARATION OF THE DGDAC INTERRUPT

                ;DEFINITION OF EXTERNAL SYMBOLS
;RELAY   ADDR OF RELAY BIT STATUS TABLE, STARTHNG AT BIT 0
;DATUM   ADDR OF A/D READ DATA BLOCK FOR BOTH D/A DEVICES
;SENSE   ADDR OF CONTACT SENSE DISPATCH TABLE
;ANALG   ADDR OF D/A OUTPUT DATA BLOCK, FOR BOTH D/A DEVICES
;TTLO1   ADDR OF FIRST TTL OUTPUT DATA BLOCK
;TTLO2   ADDR OF SECOND TTL OUTPUT DATA BLOCK
;TTLI1   ADDR OF FIRST TTL INPUT DATA BLOCK
;TTLI2   ADDR OF SECOND TTL INPUT DATA BLOCK
;ERROR   MNEMONIC SUBROUTINE CALL TO THE FATAL ERROR HANDLER
; IXMT   SYSTEM CALL TO TRANSMIT A MESSAGE FROM A USER
;        DEVICE HANDLER TO A USER TASK
; UCEX   SYSTEM CALL TO EXIT A USER CLOCK
; UIEX   SYSTEM CALL TO EXIT A USER DEVICE INTERRUPT HANDLER

;WRITTEN AUGUST, 1977 BY
;                DAVID SIMMONS

;WRITTEN IN DGC'S NOVA-LINE ASSEMBLER LANGUAGE FOR
;RDOS COMPATIBILITY
```

```
37A00000'054000 DCLAR   STA 3 RADDR     ,SAVE RETURN ADDRESS
38                                      ,DEFINE THE DGDAC UNIT TO THE SYSTEM
39 00001 020406         LDA 0  +6       ;AC0 CONTAINS THE DEVICE CODE
40 00002'024406         LDA 1 . +6      ,AC1 CONTAINS THE ADDRESS TO BRANCH ON
41                                      ,INTERRUPTS
42 00003 006017          SYSTEM         ,DEFINE THE DGDAC UNIT TO THE SYSTEM
43 00004'021007         . IDEF
44 00005'077777         ERROR
45 00006'000403         JMP . +3
46 00007'000000         DAC             ,MNEMONIC OF DGDAC DEVICE
47 00010'000323'        FOLLD           ,BRANCH TO FOLLD UPON AN INTERRUPT FROM DGDA
48 00011 060100         NIOS DAC        ,START THE DG/DAC DEVICE
49                      . IFN CNTCT     ;INITIALIZE THE CONTACT SENSE DEVICE
50 00012'060400         DIA 0 DAC       ,READ AND SAVE THE STATUS OF THE DG/DAC
51                                      ,CONTROLLER
52 00013'030000         LDA 2 SCNS      ,LOAD THE SLOT ADDRESS OF THE CONTACT
53                                      ,SENSE DEVICE
54A00014'024000         LDA 1 SFADC     ,OF=1,ID=1,CM=0, LOAD 1,000,000,010,000,000
55 00015 133000         ADD 1 2         ,SETUP TO LOCATE THE CONTACT SENSE DEVICE
56 00016'071100         DOAS 2 DAC      ,SELECT AND START THE CONTACT SENSE DEVICE
57 00017'063000         DOC 0 DAC       ,RESTORE THE STATE OF THE DGDAC CONTROLLER
58                       ENDC
59
```

```
  0002  MTPLX
01                                          ;SET THE GAIN VALUE OF THE FIRST VOLTAGE MPX
02U                         IFN SVM1
03A00020'020000            LDA 0 SFMXA      ;MUX NO=1,MUX CHAN=0
04U00021'063000            DOC 0 DAC        ;SELECT THE FIRST MPX
05U00022'020000            LDA 0 GAIN1
06U00023'062000            DOB 0 DAC   .    ;SPECIFY THE GAIN OF THE FIRST VOLTAGE MPX
07K                         ENDC
08
09                                          ;SET THE GAIN VALUE OF THE SECOND VOLTAGE MP:
10U                         IFN SVM2
11A00024'020000            LDA 0 SFMX2      ;MUX NO=2,MUX CHAN=0
12U00025'063000            DOC 0 DAC        ;SELECT THE SECOND MPX
13U00026'020000            LDA 0 GAIN2
14U00027'062000            DOB 0 DAC        ;SPECIFY THE GAIN OF THE SECOND VOLTAGE MPX
15K                         ENDC
16
17 00030'002000            JMP @ RADDR      ;RETURN
18
19A00031'054000 MTPLX      STA 3 RADDR      ;SAVE RETURN ADDRESS
20                                          ;HANDLE THE RELAY OUTPUTS
21U                         IFN RLAYS
22 00032 132400            SUB 1 1          ;SETUP BIT COUNTER
23A00033'034000            LDA 3 RS         ;GET ADDR OF RELAY BIT STATUS TABLE
24A00034'021400            LDA 0 RS+1       ;GET # RELAYS IN SYSTEM
25 00035 124400            NEG 0 0          ;MAKE LOOP COUNTER
26 00036 025400            LDA 1 0 3        ;GET A BIT FLAG,STARTING AT BIT 0
27 00037 125004            MOV 1 1 SZR      ;IS BIT TO BE SET?
```

```
26 00040'151141      MOVOL 2 2 SKP      ;YES-SET IT IN COMPOSITE
27 00041'151120      MOVZL 2 2          ;NO-RESET IT IN COMPOSITE
30 00042'175400      INC 3 3            ;PUSH POINTER
31 00043'101404      INC 0 0 SZR        ;SCAN DONE?
32 00044'000772      JMP -6             ;NO-CONTINUE
33A00045'020000      LDA 0 RS+1         ;GET # RELAY BITS SCANNED
34A00046'024000      LDA 1 N16          ;GET NEG OF # OF RELAY BITS IN SYSTEM
35 00047'107003      ADD 0 1 SNR        ;ANY BITS NOT SCANNED
36 00050'000404      JMP +4
37 00051'151120      MOVZL 2 2          ;MOV COMPOSITE ALONG
38 00052'125404      INC 1 1 SZR
39 00053'000776      JMP -2
40U00054'060400      DIA 0 DAC          ;SAVE THE STATUS OF THE DGDAC CONTROLLER
41U00055'034000      LDA 3 SRLY         ;LOAD THE SLOT ADDRESS OF THE RELAY DEVICE
42A00056'024000      LDA 1 SFADC        ;CP=1,ID=1,CM=0 LOAD 1,000,000,010,000,000
43 00057'133000      ADD 1 3            ;SETUP TO LOCATE THE RELAY DEVICE
44U00060'073000      DOA 3 DAC          ;SELECT THE RELAY DEVICE
45U00061'072100      DOBS 2 DAC         ;SETS BIT PATTERN AND STARTS THE RELAY DEVIC
46U00062'065000      DOC 0 DAC          ;RESTORE THE STATE OF THE DGDAC CONTROLLER
47                    ENDC
48                                      ;HANDLE THE FIRST DAC DEVICE
49                    IFN DTAS1
[illegible]          DIA 2 DAC          ;READ THE STATUS OF THE DGDAC CONTROLLER
[illegible]          STA 2 DGCON        ;SAVE THE STATUS OF THE DGDAC CONTROLLER
[illegible]          LDA 2 SDAC1        ;LOAD THE FIRST SLOT ADDRESS OF THE DAC DEVI
[illegible]          LDA 1 SFADC        ;CP=1,ID=1,CM=0 LOAD 1,000,000,010,000,000
[illegible]          ADD 1 2            ;SETUP TO LOCATE THE FIRST DAC DEVICE
[illegible]          LDA 1 SCDA1        ;LOAD THE STARTING CHANNEL OF THE FIRST DAC
[illegible]          STA 1 CHDAC        ;STORE THE CHANNEL NUMBER OF THE DAD
[illegible]          LDA 3 DIA          ;GET THE ADDRESS OF THE D/A OUTPUT TABLE
[illegible]          LDA 0 DIA+1        ;GET THE NUMBER OF DAC CHANNELS FOR FIRST DE
[illegible]          NEG 0 0            ;START LOOP COUNTER
```

```
 0003  MTFLX
  00075 025400   LDA 1 0 3      ;GET AN OUTPUT VALUE
 U00076 071000   DOA 2 DAC      ;SELECT THE FIRST DAC DEVICE
 A00077 040000   STA 0 LOOPC    ;STORE LOOP COUNTER
 A00100 020000   LDA 0 CHDAC    ;LOAD NEXT CHANNEL NUMBER
  00101 125120   MOVZL 1 1      ;SHIFT
  00102 125120   MOVZL 1 1      ;DIGITAL OUTPUT
  00103 125120   MOVZL 1 1      ;FOUR BITS
  00104 125120   MOVZL 1 1      ;LEFT
  00105 107000   ADD 0 1        ;MERGE VALUES FOR DATA TRANSFER
 U00106 062000   DOB 1 DAC      ;SPECIFY CONVERSION
  00107 101400   INC 0 0        ;INCREMENT CHANNEL NUMBER
  00110 040571   STA 0 CHDAC    ;STORE CHANNEL NUMBER
  00111 175100   INC 3 3        ;INCREMENT TABLE POSITION
  00112 020566   LDA 0 LOOPC    ;RESTORE LOOP VALUE
  00113 101[illegible]   INC 0 0 SZR    ;DONE?
  00114 000715   JMP -15        ;NO
  00115 030345   LDA 2 DGCON    ;LOAD THE STATE OF THE DGDAC CONTROLLER
 U00116 073000   DOC 2 DAC      ;RESTORE THE STATE OF THE DGDAC CONTROLLER
                  ENDC
                                ;HANDLE THE SECOND DAC DEVICE
                  IFN DTAS2
 U00117 072400   DIA 2 DAC      ;READ THE STATUS OF THE DGDAC CONTROLLER
  00120 [illegible]   STA 2 [illegible]   ;SAVE THE STATUS OF THE DGDAC CONTROLLER
 [illegible]00121 [illegible]   LDA 2 SLOT2    ;LOAD THE FIRST SLOT ADDRESS OF THE DAC DEVIC
```

```
 00122'024554     LDA 1 SFADL      ,CP=1,ID=1,CM=0 LOAD 1,000,000,010,000,000
 00123'133000     ADD 1 2          ,SETUP TO LOCATE THE SECOND DAC DEVICE
U00124'024000     LDA 1 SCDA2      ,LOAD THE STARTING CHANNEL OF THE SECOND DAC
 00125 044554     STA 1 CNDAC      ,STORE THE CHANNEL NUMBER OF THE DAC
 00126 034566     LDA 3 D2A        ,GET THE ADDRESS OF THE D/A OUTPUT TABLE
 00127'024404     LDA 1  +4        ,ADD 4 TO ADDRESS OF THE D/A OUTPUT TABLE
 00130'137000     ADD 1 3          ;TO FIND STARTING LOCATION FOR SECOND DAC DEV
 00131'020565     LDA 0 D2A+2      ;GET THE NUMBER OF DAC CHANNELS FOR SECOND DE
 00132 100400     NEG 0 0          ,START LOOP COUNTER
 00133'025400     LDA 1 0 3        ,GET AN OUTPUT VALUE
U00134'071000     DOA 2 DAC        ,SELECT THE SECOND DAC DEVICE
 00135'040543     STA 0 LOOPC      ,STORE LOOP COUNTER
 00136'020543     LDA 0 CNDAC      ,LOAD NEXT CHANNEL NUMBER
 00137'125120     MOVZL 1 1        ,SHIFT
 00140 125120     MOVZL 1 1        ,DIGITAL OUTPUT
 00141'125120     MOVZL 1 1        ,FOUR BITS
 00142'125120     MOVZL 1 1        ,LEFT
 00143'107000     ADD 0 1          ,MERGE VALUES FOR DATA TRANSFER
U00144 066000     DOB 1 DAC        ,SPECIFY CONVERSION
 00145'101400     INC 0 0          ,INCREMENT CHANNEL NUMBER
 00146'040533     STA 0 CNDAC      ,STORE CHANNEL NUMBER
 00147 175400     INC 3 3          ,INCREMENT TABLE POSITION
 00150'020530     LDA 0 LOOPC      ,RESTORE LOOP VALUE
 00151 101404     INC 0 0 SZR      ,DONE?
 00152 000765     JMP  -13         ;NO
 00153 030527     LDA 2 DGCON      ,LOAD THE STATE OF THE DGDAC CONTROLLER
U00154 073000     DOC 2 DAC        ,RESTORE THE STATE OF THE DGDAC CONTROLLER
                   ENDC
                                   ,HANDLE THE FIRST TTL OUTPUT DEVICE
                   IFN TLIO
 00155'153400     SUB 2 2          ,SETUP BIT COUNTER
 00156'034541     LDA 3 FTTLO      ,GET ADDR OF RELAY BIT STATUS TABLE
 00157 020541     LDA 0 FTTLO+1    ,GET # RELAYS IN SYSTEM
 00160'100400     NEG 0 0          ,MAKE LOOP COUNTER
 00161 025400     LDA 1 0 3        ,GET A BIT FLAG,STARTING AT BIT 0
```

```
0004  MTPLX
00162 125004     MOV 1 1 SZR        ;IS BIT TO BE SET?
00163 151141     MOVOL 2 2 SKP      ;YES-SET IT IN COMPOSITE
00164 151120     MOVZL 2 2          ;NO-RESET IT IN COMPOSITE
00165 175400     INC 3 3            ;PUSH POINTER
00166 101404     INC 0 0 SZR        ;SCAN DONE?
00167 000772     JMP  -6            ;NO-CONTINUE
00170 020530     LDA 0 PTTLO+1      ;GET # RELAY BITS SCANNED
00171 024503     LDA 1 N16          ;GET NEG OF # OF RELAY BITS IN SYSTEM
00172 107005     ADD 0 1 SNR        ;ANY BITS NOT SCANNED
00173 000404     JMP  +4
00174 151120     MOVZL 2 2          ;MOV COMPOSITE ALONG
00175 125404     INC 1 1 SZR
00176 000776     JMP  -2
00177 060400     DIA 0 DAC          ;SAVE THE STATUS OF THE DGDAC CONTROLLER
00200 034000     LDA 3 ST01         ;LOAD THE SLOT ADDRESS OF THE FIRST TTL OUTPI
00201 024475     LDA 1 SPADC        ;CP=1,ID=1,CM=0 LOAD 1,000,000,010,000,000
00202 137000     ADD 1 3            ;SETUP TO LOCATE THE RELAY DEVICE
00203 075000     DOA 3 DAC          ;SELECT THE FIRST TTL OUTPUT DEVICE
00204 072100     DOBS 2 DAC         ;SETS THE FIRST TTL OUTPUT DEVICE
00205 063000     DOC 0 DAC          ;RESTORE THE STATE OF THE DGDAC CONTROLLER
                  ENDC
                                    ;HANDLE THE SECOND TTL OUTPUT DEVICE
                  IFN TL20
00206 132400     SUB 1 1
00207 034512     LDA 3 STTLO        ;GET ADDR OF RELAY BIT STATUS TABLE
```

```
00210 020312    LDA 0 STTLO+1     ;GET # RELAYS IN SYSTEM
00211 100400    NEG 0 0           ;MAKE LOOP COUNTER
00212 025400    LDA 1 0 3         ;GET A BIT FLAG,STARTING AT BIT 0
00213 125004    MOV 1 1 SZR       ;IS BIT TO BE SET?
00214 151141    MOVOL 2 2 SKP     ;YES-SET IT IN COMPOSITE
00215 151120    MOVZL 2 2         ;NO-RESET IT IN COMPOSITE
00216 175400    INC 3 3           ;PUSH POINTER
00217 101404    INC 0 0 SZR       ;SCAN DONE?
00220 000772    JMP  -6           ;NO-CONTINUE
00221 020501    LDA 0 STTLO+1     ;GET # RELAY BITS SCANNED
00222 034452    LDA 1 N16         ;GET NEG OF # OF RELAY BITS IN SYSTEM
00223 107005    ADD 0 1 SNR       ;ANY BITS NOT SCANNED
00224 000404    JMP .+4
00225 151120    MOVZL 2 2         ;MOV COMPOSITE ALONG
00226 125404    INC 1 1 SZR
00227 000776    JMP  -2
00230 060100    DIA 0 DAC         ;SAVE THE STATUS OF THE DGDAC CONTROLLER
00231 034000    LDA 3 STO2        ;LOAD THE SLOT ADDRESS OF THE SECOND TTL OUTP
00232 024444    LDA 1 SPADC       ;CP=1,ID=1,CM=0 LOAD 1,000,000,010,000,000
00233 137000    ADD 1 3           ;SETUP TO LOCATE THE RELAY DEVICE
00234 07[illegible]    DOA 3 DAC         ;SELECT THE SECOND TTL OUTPUT DEVICE
00235 072100    DOBS 2 DAC        ;SETS THE SECOND TTL OUTPUT DEVICE
00236 063000    DOC 0 DAC         ;RESTORE THE STATE OF THE DGDAC CONTROLLER
                 ENDC
                                  ;START THE FIRST TTL INPUT DEVICE
                 IFN TL11
00237 0[illegible]    DIA 0 DAC         ;READ AND SAVE THE STATUS OF THE DGDAC CONTR
0024[illegible] [illegible]    LDA 2 STI1        ;LOAD THE SLOT ADDRESS OF THE FIRST TTL INPU
00241 [illegible]    LDA 1 SPADC       ;CP=1,ID=1,CM=0
00242 133000    ADD 1 2           ;SETUP TO LOCATE THE FIRST TTL INPUT DEVICE
00243 071100    DOAS 2 DAC        ;SELECT AND START THE FIRST TTL INPUT DEVICE
00244 [illegible]    DOC 0 DAC         ;RESTORE THE STATE OF THE DGDAC CONTROLLER
                 ENDC
                                  ;START THE SECOND TTL INPUT DEVICE
```

```
                    IFN TL2I
00245 060400        DIA 0 DAC        ;READ AND SAVE THE STATUS OF THE DGDAC CONTRO
00246 030000        LDA 2 STI2       ;LOAD THE SLOT ADDRESS OF THE SECOND TTL INPU
00247 024427        LDA 1 SPADC      ;CP=1,ID=1,CHA0
00250 133000        ADD 1 2          ;SETUP TO LOCATE THE SECOND TTL INPUT DEVICE
00251 071100        DOAS 2 DAC       ;SELECT AND START THE SECOND TTL INPUT DEVICE
00252 [illegible]   DOC 0 DAC        ;RESTORE THE STATE OF THE DGDAC CONTROLLER
                    .ENDC

                                     ;START THE ADC DEVICE

                    IFN ATODS
[illegible]         DIA 0 DAC        ;READ AND SAVE STATUS OF THE DGDAC CONTROLLER
[illegible]         LDA 3 SADC       ;LOAD THE SLOT ADDRESS OF THE ADC DEVICE
[illegible]         LDA 1 SPADC      ;CP=1,ID=1,CHA0
[illegible]         ADD 1 3          ;SETUP TO LOCATE THE ADC CLUSTER
[illegible]         DOA 3 DAC        ;SELECT THE ADC CLUSTER
[illegible]         LDA 2 ATODS      ;NUMBER OF CONVERSIONS TO BE PERFORMED
[illegible]         LDA 1 SPCON      ;INT OVR=1, MUX BUS=1, TRIG=01, AUTO SCAN=1,
[illegible]         ADD 1 2          ;SET UP TO INDICATE THE NUMBER OF CONVERSIONS
[illegible]         DOB 2 DAC        ;SPECIFY CONVERSIONS
[illegible]         LDA 1 SPMXA      ;MUX NO=3, MUX CHAN=0
[illegible]         DOC 1 DAC        ;SPECIFY MUX ADDRESS FOR DATA CHANNEL OPERATI
[illegible]         LDA 1 DATUM      ;ADDRESS WHERE CONVERTED DATA IS TO BE STORED
[illegible]         DOAS 1 DAC       ;SPECIFY MEMORY ADDRESS LOCATION AND START TH
[illegible]         DOA 3 DAC        ;LOCATE THE ADC CLUSTER
[illegible]         DOC 0 DAC        ;RESTORE THE STATE OF THE DGDAC CONTROLLER
                    .ENDC

                                     ;RETURN
[illegible]         LDA 3 RTURN      ;GET RETURN ADDRESS
                    [illegible]
```

```
                                        ;INPUT DATA
00274 177760  N16     -16
00275 000000  RADDR   0                 ;RETURN ADDR STORAGE
00276 100220  SFADC   100220            ;OF=1,ID=1,CH=0 IN SPECIFY ADDRESS AND CONTE
                                        ;EQUIVALENT TO 1,000,000,010,000,000
00277 000020  SFMX1   000020            ;MUX NO=1,MUX CHAN=0 IN SPECIFY MUX ADDRESS
                                        ;EQUIVALENT TO 0,000,000,000,010,000
00300 000000  LOOPC   0                 ;LOOP COUNTER FOR BIT COUNTER
00301 000000  CNDAC   0                 ;CHANNEL NUMBER OF THE DAC
00302 000000  DSCON   0                 ;STATUS OF THE DSDAC CONTROLLER
00303 000060  SFMXA   000060            ;MUX NO=3,MUX CHAN=0 IN SPECIFY MUX ADDRESS
                                        ;EQUIVALENT TO 0,000,000,000,110,000
00304 154000  SPCON   154000            ;INT OVR=1,MUX BUS=1,TRIG=01,AUTO SCAN=1,DCH
                                        ;NEG=0,TST=0,CONVERSION COUNT=0 IN SPECIFY C
                                        ;EQUIVALENT TO 1,101,110,000,000,000
00305 000040  SFMX2   000040            ;MUX NO=2 MUX CHAN=0 IN SPECIFY MUX ADDRESS
                                        ;EQUIVALENT TO 0,000,000,000,100,000
00306 000017  SLOOR   000017            ;VALUE=0,000,000,000,001,111 USED TO FIND TH
                                        ;SLOT ADDRESS OF THE INTERRUPTING DEVICE IN
                                        ;POLLING ROUTINE
00307 010000  FDOVR   010000            ;FOR DATA OVERRUN,  0,001,000,000,000,000
00310 000020  SFAD1   000020            ;OF=0,ID=0,CH=1 IN SPECIFY ADDRESS AND CONTE
                                        ;EQUIVALENT TO 0,000,000,000,010,000
00311 010000  MSKDA   010000            ;SETUP MODULE MASK TO MASK OUT ALL DSDAC SLO
                                        ;EQUIVALENT TO 0,001,000,000,000,000
                                 .ZREL
00312 000000               RELAY         ;ADDR OF RELAY BIT STATUS TABLE
00313 000000               RLNO          ;NUMBER OF RELAYS IN SYSTEM
                                 .END
```

```
[illegible]
[illegible]
[illegible]                                IFN DTOAS
[illegible]               DIA    ANALG     ;ADDR OF DAC STORAGE TABLE
[illegible]                      DTAS1     ;NUMBER OF CHANNELS FOR FIRST DAC DEVICE
[illegible]                      DTAS2     ;NUMBER OF CHANNELS FOR SECOND DAC DEVICE
[illegible]                       ENDC
07
[illegible]                        IFN TLIO
[illegible]              FTTLO   TTLO1     ;ADDR OF FIRST TTL OUTPUT DEVICE TABLE
[illegible]                      TL1O      ;NUMBER OF OUTPUTS FOR FIRST TTL OUTPUT DEVI
[illegible]                       ENDC
[illegible]
[illegible]                        IFN TL2O
[illegible]              STTLO   TTLO2     ;ADDR OF SECOND TTL OUTPUT DEVICE TABLE
[illegible]                      TL2O      ;NUMBER OF OUTPUTS FOR SECOND TTL OUTPUT DEV
[illegible]                       ENDC
17
[illegible]                                ;INPUT INTERRUPT HANDLER
[illegible]                                ;INTERRUPT STATE SAVE AREA
[illegible]                                ;MASK WORD (IO,TTI,RTC,PTR,LPT MASKED OUT
[illegible]                                ;INTERRUPT ROUTINE ADDRESS
                                           ;POLLING ROUTINE FOR DGDAC INTERRUPTS
[illegible]                                ;IDENTIFY SUBSYSTEM INTERRUPT
[illegible]                                ;DETERMINE SLOT WHICH CAUSED THE INTERRUPT
[illegible]                                ;SAVE STATUS OF DGDAC CONTROLLER
[illegible]                                ;LOAD IN SLOT ADDRESS BIT LOCATION
[illegible]                                ;SLOT ADDRESS IN BITS 12 - 15
```

```
        LDA 2 SCNS      ;CHECK IF CONTACT SENSE DEVICE WAS
        SUB 1 2 SZR     ;RESPONSIBLE FOR THE INTERRUPT
        JMP CTHAN       ;JUMP TO CONTACT SENSE HANDLER
        LDA 2 STT1      ;CHECK IF FIRST TTL INPUT DEVICE
        SUB 1 2 SNR     ;CAUSED THE INTERRUPT
        JMP @TTL1       ;JUMP TO FIRST TTL INPUT HANDLER
        LDA 2 STT2      ;CHECK IF THE SECOND TTL INPUT DEVICE
        SUB 1 2 SNR     ;CAUSED THE INTERRUPT
        JMP @TTL2       ;JUMP TO SECOND TTL INPUT HANDLER
        LDA 2 SADC      ;CHECK IF THE ADC DEVICE
        SUB 1 2 SNR     ;CAUSED THE INTERRUPT
        JMP ADHAN       ;JUMP TO ADC INPUT HANDLER
        ERRMG           ;ERROR IF NONE OF THE ABOVE
        MSG10           ;PRINT ERROR MESSAGE
        ADC 1 1         ;SPECIFY RESCHEDULING
         QUIT
          IFN CNTCT
        SENTB           ;ADDR OF CONTACT SENSE DISPATCH TABLE
        CNTCT           ;NUMBER OF CONTACT SENSES IN SYSTEM
        0               ;LAST SENSE PATTERN READ
        0               ;RETURN ADDR STORAGE
        0               ;COUNTER STORAGE

        [illegible]
                        ;CONTACT SENSE HANDLER
        [illegible]     ;SAVE RETURN ADDR
        [illegible]     ;PLACE SUBSYSTEM IN
        [illegible]     ;CONTACT MODE
        [illegible]     ;MASK OUT ALL MODULE
        [illegible]     ;LEVEL INTERRUPTS
        [illegible]     ;LOCATE
        [illegible]     ;CONTACT
```

```
0007  NTFLX

        ADD 1 2              ,SENSE
        DOA 2 DAC            ,MODULE
        DIB 0 DAC            ,READ CONTACT SENSE INPUTS
        LDA 1 CTACT+1
                             ,
        COM 0 2
        AND 1 2 SZR          ,CHANGE FROM 0 TO 1?
        JMP .+3              ,YES
        COM 1 2              ,NO
        AND 0 2 SZR          ,CHANGE FROM 1 TO 0?
        JMP .+25             ,NO-INTERRUPT NOT ON CONTACT SENSE
        SUBZR 1 1 SKP        ,FORM 100000 FOR 0 TO 1 CHANGE
        SUB 1 1              ,FORM 0 FOR 1 TO 0 CHANGE
        LDA 3 BIT            ,GET IBI
        ADD 3 1              ,FORM LEFT PART OF MESSAGE WORD
        LDA 3 CTACT          ,GET # CONTACT SENSES IN SYSTEM
        STA 3 CTACT+5        ,INSTALL COUNTER
        SUB 3 3
        MOVL 2 2 SZC         ,IS BIT SET, STARTING AT BIT 0
        JMP .+5              ,YES
        INC 3 3              ,NO-INCR POINTER
        DSZ CTACT+5          ,SCAN DONE?
        JMP .-4              ,NO-CONTINUE
        JMP .+10             ,YES-NO BITS CHANGED-INTERRUPT NO GOOD
        STA 2 CTACT+1        ,BIT FOUND-SAVE NEW SENSE PATTERN
        ADD 3 1              ,FORM FINAL MESSAGE
        LDA 0 CTACT+1        ,GET S.A. OF CONTACT SENSE DISPATCH TABLE
```

```
          ADD 0 3           ;GET POINTER TO DISPATCH ADDR
          LDA 0 0 3         ;GET DISPATCH ADDR
           IXMT             ;TRANSMIT MESSAGE TO DISPATCH TABLE
          JMP  .+1          ;RESTART THE CONTACT SENSE DEVICE
          LDA 2 SCNS        ;LOAD THE SLOT ADDRESS OF THE CONTACT SENSE I
          LDA 1 SFDAC       ;CF=1, ID=1, CMA=0
          ADD 1 2           ;SETUP TO LOCATE THE CONTACT SENSE DEVICE
          DOAS 2 DAC        ;SELECT AND START THE CONTACT SENSE DEVICE
          LDA 0 DGDCW       ;LOAD THE STATE OF THE DGDAC CONTROLLER
          DOC 0 DAC         ;RESTORE THE STATE OF THE DGDAC CONTROLLER
          LDA 3 CTNCT+2     ;GET RETURN ADDR
          ADC 1 1           ;SPECIFY RESCHEDULING
           UIEX
          .ENDC

                            ;ADC HANDLER
ADHAN:    LDA 1 SADCW       ;PLACE SUBSYSTEM IN
          DOA 1 DAC         ;CONTROL MODE
          LDA 1 MSKAL       ;MASK OUT ALL MODULE
          DOC 1 DAC         ;LEVEL INTERRUPTS
          LDA 2 SADC        ;LOCATE
          LDA 1 SPADC       ;THE
          ADD 1 2           ;ADC
          DOA 2 DAC         ;MODULE
          DIAC 1 DAC        ;CHECK THE VALIDITY OF THE DATA FROM THE ADC
          LDA 0 D.OVR       ;CHECK THE DATA OVERRUN BIT
          AND 0 1 SZR       ;GETS DATA OVERRUN BIT
          DSPMS             ;INDICATE ERROR
          ADSIDO            ;ERROR - PRINT MESSAGE
          LDA 0 DGDCW       ;LOAD THE STATE OF THE DGDAC CONTROLLER
          DOC 0 DAC         ;RESTORE THE STATE OF THE DGDAC CONTROLLER
          ADC 1 1           ;SPECIFY RESCHEDULING
```

```
0003 KTPLX
00437'000436'           UIEX
                        ENDC

                                        ;FIRST TTL INPUT HANDLER
                        .IFN TLI1
00460 077777   FTTLI    TTLI1           ;ADDRESS OF FIRST TTL INPUT TABLE
00461 000000            TLI1            ;NUMBER OF TTL INPUTS FOR FIRST TTL INPUT DE
00462 000000            0               ;RETURN ADDR STORAGE
00463 000000            0               ;STORE CURRENT TABLE VALUE
00464 [illegible] HTLI1 STA 3 FTTLI+2   ;SAVE RETURN ADDR
00465 [illegible]       LDA 1 SADCN     ;PLACE SUBSYSTEM IN
00466 [illegible]       DOA 1 DAC       ;CONTROL MODE
00467 [illegible]       LDA 1 MASKL     ;MASK OUT ALL MODULE
[illegible]             DOC 1 DAC       ;LEVEL INTERRUPTS
[illegible]             LDA 2 ITI1      ;LOCATE
[illegible]             LDA 1 SFADC     ;FIRST
[illegible]             ADD 1 2         ;TTL INPUT
[illegible]             DOA 2 DAC       ;MODULE
[illegible]             DIB 3 DAC       ;READ FIRST TTL INPUTS INTO AC3
00476 [illegible]       LDA 1 FTTLI     ;GET ADDR OF FIRST TTL INPUT TABLE
00477 [illegible]       LDA 0 FTTLI+1   ;GET NUMBER OF FIRST TTL INPUTS IN SYSTEM
[illegible]             NEG 0 0         ;MAKE LOOP COUNTER
[illegible]             STA 1 FTTLI+3   ;LOAD CURRENT TABLE POSITION INTO AC1
[illegible]             LDA 2 +1        ;ASSUME BIT IS A ONE
[illegible]             MOVL 3 0 3 SNC  ;SHIFT BIT INTO CARRY, EXAMINE CARRY
[illegible]             LDA 2 +0        ;IF BIT IS A ZERO
[illegible]             STA 2 @ FTTLI+3 ;LOAD VALUE INTO FIRST TTL INPUT TABLE
[illegible]             INC 1 1         ;INCREMENT TABLE POSITION
[illegible]             INC 0 0 SZR     ;CHECK IF ALL INPUTS HAVE BEEN LOADED
[illegible]             JMP .-7         ;IF NOT CONTINUE
[illegible]             LDA 0 DACON     ;LOAD THE STATE OF THE DGDAC CONTROLLER
[illegible]             DOA 0 DAC       ;RESTORE THE STATE OF THE DGDAC CONTROLLER
[illegible]             LDA 3,FTTLI+2   ;GET RETURN ADDR
[illegible]             JMP 1 3         ;SPECIFY RESCHEDULING
                        .ENDC
                        ENDC
```

```
                                              ;SECOND TTL INPUT HANDLER
                      IFN TL2I
00516 077777  STTLI   TTLI2                   ;ADDRESS OF SECOND TTL INPUT TABLE
00517 000000          TL2I                    ;NUMBER OF TTL INPUTS FOR SECOND TTL INPUT DE
00520 000000          0                       ;RETURN ADDR STORAGE
00521 000000          0                       ;STORE CURRENT TABLE VALUE
[illegible]   HTLI2   STA 3 STTLI+2           ;SAVE RETURN ADDR
[illegible]           LDA 1 SADCN             ;PLACE SUBSYSTEM IN
[illegible]           DOA 1 DAC               ;CONTROL MODE
[illegible]           LDA 1 MSKAL             ;MASK OUT ALL MODULE
[illegible]           DOC 1 DAC               ;LEVEL INTERRUPTS
[illegible]           LDA 2 STI2              ;LOCATE
[illegible]           LDA 1 SPADC             ;SECOND
[illegible]           ADD 1 2                 ;TTL INPUT
[illegible]           DOA 2 DAC               ;MODULE
[illegible]           DIB 3 DAC               ;READ SECOND TTL INPUTS INTO AC3
[illegible]           LDA 1 STTLI             ;GET ADDR OF SECOND TTL INPUT TABLE
[illegible]           LDA 0 STTLI+1           ;GET NUMBER OF SECOND TTL INPUTS IN SYSTEM
[illegible]           NEG 0 0                 ;MAKE LOOP COUNTER
[illegible]           STA 1 STTLI+3           ;LOAD CURRENT TABLE POSITION INTO AC1
[illegible]           LDA 2 =1                ;ASSUME BIT IS A ONE
[illegible]           MOVL 3 3 SNC            ;SHIFT BIT INTO CARRY, EXAMINE CARRY
```

```
          LDA 2 +0          ,IF BIT IS A ZERO
          STA 2 @ STTLI+5   ,LOAD VALUE INTO SECOND TTL INPUT TABLE
          INC 1 1           ,INCREMENT TABLE POSITION
          INC 0 0 SZR       ,CHECK IF ALL INPUTS HAVE BEEN LOADED
          JMP -7            ,IF NOT CONTINUE
          LDA 0 DGCON       ,LOAD THE STATE OF THE DGDAC CONTROLLER
          DOC 0 DAC         ,RESTORE THE STATE OF THE DGDAC CONTROLLER
          ADC 1 1           ,SPECIFY RESCHEDULING
           UIEX
           ENDC

DGSTA      BLK 6            ,STATE SAVE AREA
           END              ;END OF MTFLX
```

0010 HTTLX

ADHON	000377	6/17	7/44						
ANALG	000314	6/03							
ATODF	000000	5/10	5/13						
BIT1	000360	6/51	7/14						
CNDAC	000301	2/55	3/04	3/12	3/28	3/37	3/45	3/59	
CNTCT	000000	1/49	3/19	3/46					
CTACT	000301	6/46	6/55	7/04	7/16	7/17	7/22	7/25	7/27
		7/36							
CTRAN	000311	6/50	6/58						
DIP	000314	2/37	2/55	3/29	3/32	6/03			
DAC	000000	1/42	1/45	1/50	1/56	1/57	2/04	2/06	2/12
		2/14	2/40	2/44	2/45	2/46	2/50	3/02	3/10
		3/16	3/22	3/35	3/43	3/51	4/14	4/16	4/19
		4/20	4/42	4/46	4/47	4/48	4/52	4/56	4/57
		5/02	5/06	5/07	5/11	5/15	5/19	5/21	5/23
		5/24	5/26	[illegible]	6/24	6/55	6/57	7/02	7/03
		7/35	7/37	7/45	7/47	7/51	7/52	7/56	8/12
		8/14	8/15	8/17	8/22	8/46	8/46	8/52	8/53
		9/07							
[illegible]	000007	5/51	[illegible]						
[illegible]	077777	5/22							
[illegible]	000100	1/37							
[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	3/50	5/40	6/23	7/36	7/37
		8/51	[illegible]						

DAFFA	00001F	6/19	7/12						
DFLAG	000000U	6/40	7/35						
DTAB1	000000U	2/44	6/04						
DTAB2	000000U	3/21	6/03						
DTAB3	000000U	6/02							
ERROR	000005 X	1/44							
FTTLI	000106	5/06	6/10	6/20	6/21	6/23	6/27	6/33	
FTTLO	000317	3/36	3/57	4/07	5/09				
GAIN1	000000U	2/05							
GAIN2	000000U	2/13							
HTLI1	[illegible]	6/33	8/10						
HTLI2	000322	6/36	6/44						
LOOPC	000100	3/03	3/14	3/36	5/47	5/58			
NEGIO	000000U	6/41	7/36						
[illegible]	000511	5/34	6/36	7/46	8/16	8/47			
[illegible]	000071	2/19							
[illegible]	000174	2/34	4/05	4/36	5/32				
[illegible]	000321	1/47	6/19						
[illegible]	[illegible]	6/21	6/23						
[illegible]	[illegible]	1/37	2/17	[illegible]	5/26	5/33			
RELAY	000312 X	5/57							
[illegible]	000000U	2/21	[illegible]	[illegible]					
RS	[illegible]	[illegible]	[illegible]	[illegible]	5/37				
[illegible]	[illegible]	[illegible]	4/17	[illegible]					
[illegible]	[illegible]	5/02	6/34	7/44	6/11	6/45			
[illegible]	[illegible]	5/48	[illegible]						
[illegible]	000000U	[illegible]							
[illegible]	000000U	[illegible]							
[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	7/42				
[illegible]	[illegible]	[illegible]							
[illegible]	[illegible]	[illegible]							
[illegible]	[illegible]								
[illegible]	[illegible]	[illegible]							
[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	4/16	[illegible]	4/04	5/04
		[illegible]	[illegible]	[illegible]	7/4?	6/16	6/30		

[illegible] HTFLX

SFCON	000304	5/17	5/45				
SFDAC	000000	7/33					
SFMX1	000277	5/36					
SFMX2	000[illegible]03	2/11	5/46				
SFMXA	000[illegible]03	2/05	5/20	5/41			
SRLY	000000	2/41					
STI1	000000	4/53	6/21	6/15			
STI2	000000	5/06	6/34	6/49			
STO1	000000	4/15					
STO2	000000	4/43					
STTL1	000016	6/40	6/44	6/54	6/55	6/57	7/02
STTL2	000[illegible]21	4/25	4/26	4/33	6/14		
SVM1	000000	2/02					
SVM2	000000	2/10					
TCIT	000000	4/51	6/07	6/07			
TI10	000000	6/08	6/10				
TI11	000000	5/01	6/2[illegible]	6/41			
TI20	000000	4/23	6/16	6/16			
TI[illegible]	000000	3/54					
TTL11	[illegible]	6/06					
TTL12	[illegible]	6/40					
TTL01	[illegible]	6/05					
TTL02	000[illegible]21	6/14					
[illegible]	[illegible]	7/30					
[illegible]	[illegible]	5/24					
[illegible]	[illegible]	6/43	7/10	6/0[illegible]	6/53	7/07	

Figure 10

Additions to OPCOM Messages

```
<13><12>
<12>NOT AN EXECUTIVE FUNCTION
<12>EXECUTIVE FUNCTION NOT AVAILABLE
<12>TASK NAME
<12>NOT A USER TASK<15>
<12>USER TASK CREATED
<12>TASK IS ALREADY ACTIVE
<12>USER TASK KILLED
<12>TASK IS ALREADY INACTIVE
<12>TASK IS INACTIVE
<12>TASK IS ACTIVE
<12>DEVICE NAME
<12>NOT A DEVICE NAME<15>
<12>DEVICE NOT PART OF SYSTEM
<12>SPECIFIED OUTPUT KILLED
<12>SPECIFIED OUTPUT ALREADY INACTIVE
<12>ALL OUTPUTS KILLED
<12>DISK DIRECTORY & FILENAME
<12>DELETION CONFIRMED
<12>FILENAME ERROR<15>
<12>CURRENT DISPLAY INTERVAL(SECONDS)
<12>RESET DISPLAY INTERVAL(SECONDS)
<12>CHANGE CONFIRMED
<12>INTERVAL IMPROPERLY SPECIFIED<15>
<13><12>END OF REPORT
[illegible]
[illegible]
[illegible]
[illegible]
```

```
FILTER CONSTANT(0 TO +1000 [illegible]
INPUT HIGH ALARM LIMIT-
INPUT LOW ALARM LIMIT-
CURRENT SET POINT-
PROPORTIONAL GAIN(+VE)/(40)
PROPORTIONAL-DERIVATIVE GAIN(+VE)/(40)
PROPORTIONAL-INTEGRAL GAIN(+VE)/(40)
SCALE FACTOR & DIRECTION OF ACTION/(40)
CONTROL INTERVAL(SECONDS)/(40)
OUTPUT HIGH LIMIT-
OUTPUT LOW LIMIT-
OUTPUT MEAN-
[illegible] CONVERSION ERROR
[illegible]
[illegible] FILE IN USE
[illegible] DELETION NOT POSSIBLE. LOGGING
[illegible] VARIABLES LIST-MAX.
[illegible] DEVICE IN USE [illegible]
[illegible] NO BUFFERS IN USE
[illegible] FORMAT [illegible]
[illegible] LIST [illegible]
[illegible] END OF LIST [illegible]
[illegible] LOGGING INTERVAL(SECONDS)
[illegible] OUTPUT TASK CREATED
[illegible] UNABLE TO INITIATE OUTPUT TASK
[illegible] DEVICE UNIT #
[illegible] DEVICE NOT CONFIGURED FOR PLOTTING
[illegible] SCALE FACTOR
[illegible] PLOTTING INTERVAL(SECONDS)
[illegible] MIN RANGE OF PLOT
[illegible] MAX RANGE OF PLOT
[illegible] - BACKGROUND [illegible]
[illegible] - BACKGROUND [illegible]
```

```
<15><12>USER FAULT STATUS TABLE [illegible] DEFINED<15><12>
<15><12>RELAY BIT STATUS TABLE NOT DEFINED<15><12>
<15><12>A-D READ DATA STORAGE NOT DEFINED<15><12>
<15><12>USER CLOCK SET TO [illegible]<15><12>
<15><12>USER CLOCK NOT DEFINED<15><12>
<15><12>CONTACT SENSE DISPATCH TABLE NOT DEFINED<15><12>
<15><12>USER SHUTDOWN ROUTINE NOT DEFINED<15><12>
<15><12>USER TASK DEFINITION TABLE NOT DEFINED<15><12>
<15><12>ERROR DETECTED IN USER TASK DEFINITION TABLE<15><12>
<15><12>USER ALARM DEFINITION TABLE NOT DEFINED<15><12>
<15><12>USER MESSAGE FILE NOT DEFINED<15><12>
<15><12>EXECUTION ABORTED<15><12>
<15><12>[illegible]-TO RUN, STRIKE ANY KEY<15><12>
<12>DIRECTORY OR DEVICE NAME
<12>DIRECTORY OR DEVICE INITIALIZED
<12>DIRECTORY OR DEVICE ALREADY INITIALIZED
<12>INITIALIZATION NOT POSSIBLE, CODE<40>
<12> DIRECTORY OR DEVICE RELEASED
<12>DIRECTORY OR DEVICE IN USE
<12>RELEASE NOT POSSIBLE, CODE<40>
<15><12>USER SPACE RELEASED<15><12>
<15><12>CONTROL LOOP DEFINITION TABLE NOT DEFINED<15><12>
<15><12>ERROR DETECTED IN CONTROL LOOP DEFINITION TABLE<15><12>
<15><12>ERROR DETECTED IN CONTACT SENSE DISPATCH TABLE<15><12>
<12>AXIAL POSITION
<15><12>USER EXECUTIVE FUNCTION LIST EXTENSION NOT DEFINED<15><12>
<12>ASCII OR BINARY OUTPUT
[illegible]
[illegible]
<12>[illegible] LIST [illegible]
```

```
<12>INPUT BUFFER ERROR
<12>LIST ERROR, ENTRY #
<12>SOURCE FILENAME
<12>NO COPY TO TTY DEVICE
<12>CHART IN USE, KILL? (Y/N)
<12>NO ANALOG OUTPUTS
<15><12>FIRST D/A OUTPUT DATA STORAGE NOT DEFINED<15><12>
<15><12>OTHER INTERRUPTING DEVICES NOT DECLARED<15><12>
<15><12>DATA OVERRUN ERROR - ADC INPUTS<15><12>
<15><12>SECOND D/A OUTPUT DATA STORAGE NOT DEFINED<15><12>
<15><12>FIRST TTL INPUT DATA STORAGE NOT DEFINED<15><12>
<15><12>SECOND TTL INPUT DATA STORAGE NOT DEFINED<15><12>
<15><12>FIRST TTL OUTPUT DATA STORAGE NOT DEFINED<15><12>
<15><12>SECOND TTL OUTPUT DATA STORAGE NOT DEFINED<15><12>
<15><12>ERROR-POLLING LOOP-INTERRUPT CAUSED BY UNDEFINED DEVICE<15><12>
[illegible] SET HYDROGEN (AO)
[illegible] VALVE OPEN (AO)
<12>HYDROGEN FLOW CONTROL
[illegible] FEED NITROGEN (AO)
<12>NITROGEN FLOW CONTROL
<15><12>INITIALIZATION: PROGRAM SETUP
<15><12>OIL TEMPERATURE SET POINT
<15><12>USE FUNCTION TIMER TO SET REACTANTS FLOWRATES
<15><12>REACTOR [illegible] INITIATED
<15><12>ALL FLOWS OFF, COOLING ON, FEED VENTING ON
[illegible]-DEGREES CELCIUS (AO)
<15><12>TEMPERATURE FEEDBACK CONTROL.
```

6- BIBLIOGRAPHY

1- Harrison, Thomas J. Handbook of Industrial Control Computers. New York: John Wiley and Sons Inc., 1972.

2- Tremblay, Pierre. "Direct Digital Control of a Butane Hydrogenolysis Chemical Reactor". M. ENG. Thesis, McMaster University, 1973.

3- Data General Corporation. DGDAC and Process Controls Programmer's Reference Manual, 1977

4- Smith, Cecil L. Digital Computer Process Control. Scranton, Pennsylvania: Intext Educational Publishers, 1972.

5- Tremblay, Pierre. "Generalized Operating System Executive", Internal Report, SOC Group, McMaster University, (March 1975).

6- Carpenter, B.E. "A Computer Scientist's View of Process Control", Massey University Computer Unit, Report No. 21 (June 1975).

7- Smith, C.L. "Fill-In-The-Forms Computer Languages For Process Control", Chemical Engineering, (March 3, 1975).

8- Shave, John. "Process Control Software", Control and Instrumentation, Vol. VII, No. 10 (November 1975), 30.

9- Smythe, Clare. "Integrated Control of Paper Machine", Control and Instrumentation, Vol. VI, No. 1 (November 1975), 30.

10- Gertler, Janos and Sedlak, Jan. "Software For Process Control - A Survey", Automatica, Vol. XI, Pergamon Press, 1975 (613-625).

11- Data General Corporation. User Device Driver Implementation In The Real Time Operating System, Manual No. 017-000006-00, March, 1974.

12- Data General Corporation. Extended Assembler, Manual No. 093-000040-01, May 1974.

13- Data General Corporation. How To Use The Nova Computers, 1971.