GE-625/635 GECOS-III
I/O Supervision

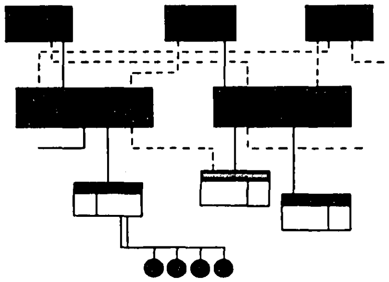
Information
Systems

Information Systems
Equipment

Betty J. Colaluca

TIBS: 600-206

SOFTWARE MAINTENANCE DOCUMENT



GENERAL ELECTRIC

CPB-1494

GENERAL ELECTRIC **INFORMATION SYSTEMS DIVISION** **COMPUTER EQUIPMENT DEPARTMENT**	**GE-600 SERIES** TECHNICAL INFORMATION BULLETIN	DATE Sept. 1968 NO. 600-206
SUBJECT: GECOS-III Remote I/O Supervision		REF. CPB-1494

This TIB includes features implemented in GECOS-III System Development Letter-1.

Replace old pages in GE-625/635 GECOS-III I/O Supervision Software Maintenance Document, CPB-1494, with attached new pages as follows:

Old	New
iii, iv	iii, iv
1, 2	1, 2

Insert attached new pages 186.1 - 186.26 following page 186.

Vertical bars in the margins of these new pages indicate changes or additions to the existing text. This new information will be included in the next edition of the manual. Index entries for this information will also be added at that time.

Place this sheet in the front of your manual to show that the contents of this TIB have been incorporated.

This is at present the only TIB applying to CPB-1494.

CPB-1494

GE-625/635 GECOS-III I/O Supervision

SOFTWARE MAINTENANCE DOCUMENT

May 1968

INFORMATION SYSTEMS

GENERAL ELECTRIC

PREFACE

This manual describes the implementation of input/output supervision for the GE-625/635 General Comprehensive Operating Supervisor (GECOS).

Additional software maintenance documents are as follows:

GE-625/635 GECOS-III Introduction and System Tables, CPB-1488

GE-625/635 GECOS-III Startup, CPB-1489

GE-625/635 GECOS-III System Input, CPB-1490

GE-625/635 GECOS-III Dispatcher and Peripheral Allocation, CPB-1491

GE-625/635 GECOS-III Rollcall, Core Allocation, Operator Interface, CPB-1492

GE-625/635 GECOS-III Fault Processing and Service MME's, CPB-1493

GE-625/635 GECOS-III Error Processing, CPB-1495

GE-625/635 GECOS-III Termination and System Output, CPB-1496

GE-625/635 GECOS-III File System Allocation and Maintenance, CPB-1497

GE-625/635 GECOS-III Utility Routines, CPB-1498

GE-625/635 GECOS-III Comprehensive Index and Glossary, CPB-1499

GE-625/635 GECOS-III Flowcharts, CPB-1500

GE-625/635 GECOS-III Time-Sharing System, CPB-1501

This manual was produced using the General Electric Remote Access Editing System (RAES). RAES is a time-shared disc-resident storage and retrieval system with text-editing and manuscript formatting capabilities. The contents of the manual were entered into RAES from a remote terminal keyboard edited using the system editing language, and formatted by RAES on reproduction masters.

The index was produced using a computer-assisted remote access indexing system. This system produces an index using source strings delimited at manuscript input time.

Suggestions and criticisms relative to form, content, purpose, or use of this manual are invited. Comments may be sent on the Document Review Sheet in the back of this manual or may be addressed directly to Documentation Standards and Publications, B-90, Computer Equipment Department, General Electric Company, 13430 North Black Canyon Highway, Phoenix, Arizona 85029.

© 1968 by General Electric Company

(2M 3-68)

CONTENTS

Page

1. INTRODUCTION TO IOS ... 1

2. MAIN IOS MODULE (.MIOS) ... 3

I/O Request Handler ... 7
- Link I/O to End of Queue, LINK (EP1) ... 8
- Link I/O to Front of Queue, LINKF (EP2) ... 10
- Link Reissued I/O to Front of Queue, LINKR (EP3) ... 11
- Assign an I/O Entry, QUEUE (EP4) ... 12
- MME GEINOS Processor, INOS (EP5) ... 15
- MME GESPEC Processor, SPEC (EP6) ... 20
- Master Message Processor, ITYM (EP7) ... 23
- Pointer Validation, PTRVL ... 27
- DCW Pointer Validation, DCWCK ... 29
- Find File Code, FNDFC ... 31
- Format Error Accounting Record, FMTAR ... 33

Interrupt Handler ... 35
- Interrupt Handler, IOTRM ... 35

Service Subroutines Used by Both Handlers ... 40
- Unlink I/O Entry, UNLNK ... 41
- Calculate Logical Primary Channel Index, ILPCX ... 43
- Status Return, STRET ... 45
- Start I/O, STIO ... 47
- Connect Multirecord Simulation DCW, STIOM ... 50
- Connect Reissue of Second Typewriter Command, TYPER ... 51
- Connect Selected GESPECed Entry, STGPC ... 52

Service Routines Used External to IOS ... 53
- Resume I/O for Program, RSMIO (EP8) ... 54
- Abort I/O for Program, ABTIO (EP9) ... 56
- Format I/O Status Words and Return Status, GSTRT (EP10) .. 58
- Resume I/O on Channel, RSMCH (EP12) ... 60
- Accounting File Request, ACTFL (EP13) ... 62
- Main IOS Module Initialization, .IIOS ... 65

3. CHANNEL MODULES ... 67

(Drivers)

Card Punch Channel Module (.MCPIO) ... 69
- Card Punch Interrupt Handler, CPIT (EP1) ... 70
- Card Punch Request, CPIO (EP2) ... 73
- Card Punch Select, CPSL (EP3) ... 75
- Card Punch Error, CPGP (EP4) ... 77
- Card Punch Initialization, .ICPIO ... 79

Magnetic Drum Subsystem Channel Module (.MDR20) ... 81
- MDS200 Interrupt Handler, DRIT (EP1) ... 82
- MDS200 Request, DRIO (EP2) ... 85
- MDS200 Select, DRSL (EP3) ... 88
- MDS200 Error and EOF Recovery, DRGP (EP4) ... 90
- MDS200 Negative Entry Points ... 92
- MDS200 Initialization, .IDR20 ... 97

- Disc Storage Subsystem Channel Module (.MDS20) 99
  - DSU200 Interrupt Handler, DSIT (EP1) 100
  - DSU200 Request, DSIO (EP2) 103
  - DSU200 Select, DSSL (EP3) 106
  - DSU200 Error and EOF Recovery, DSGP (EP4) 108
  - DSU200 Negative Entry Points 110
  - DSU200 Initialization, .IDS20 118
- Card Reader Channel Module (.MGPIO) 120
  - Card Reader Interrupt Handler, CRIT (EP1) 121
  - Card Reader Request, CRIO (EP2) 124
  - Card Reader Select, CRSL (EP3) 126
  - Card Reader Error, CRGP (EP4) 128
  - Card Reader Initialization, .IGPIO 130
- Magnetic Tape Channel Module (.MMTAP) 132
  - Magnetic Tape Interrupt Handler, MTIT (EP1) 133
  - Magnetic Tape Request, MTIO (EP2) 136
  - Magnetic Tape Select, MTSL (EP3) 138
  - Magnetic Tape Error, MTGP (EP4) 140
  - Magnetic Tape Initialization, .IMTAP 142
- Printer Channel Module (.MPRIO) 144
  - Printer Interrupt Handler, PRIT (EP1) 145
  - Printer Request, PRIO (EP2) 148
  - Printer Select, PRSL (EP3) 150
  - Printer Error, PRGP (EP4) 152
  - Printer Initialization, .IPRIO 154
- Paper Tape Channel Module (.MPTAP).......................... 156
  - Paper Tape Interrupt Handler, PTIT (EP1) 157
  - Paper Tape Request, PTIO (EP2) 159
  - Paper Tape Select, PTSL (EP3) 161
  - Paper Tape Error, PTGP (EP4) 163
  - Paper Tape Initialization, .IPTAP...................... 165
- Typewriter Channel Module (.MTYPE).......................... 167
  - Typewriter Interrupt Handler, TYIT (EP1) 168
  - Typewriter Request, TYIO (EP2) 171
  - Typewriter Select, TYSL (EP3) 174
  - Typewriter Error, TYGP (EP4) 176
  - Typewriter Initialization, .ITYPE...................... 178

4. ACCOUNTING FILE TAPE SWITCHING.............................. 181

- Accounting Tape Switching Module (MACTS)..................... 181
  - Normal Close of Accounting File, ACTS1 (EP1)............ 182
  - Error Close of Accounting File, ACTS2 (EP2)............. 184
  - External Request for Accounting File Close, ACTS3 (EP3). 185

5. REMOTE I/O SUPERVISION...................................... 186.1

- Remote Interrupt Processor Module (.MDNET).................. 186.2
  - Remote Interrupt Handler, RHNDLR (EP1).................. 186.7
  - I/O Request, DNIO (EP2)................................. 186.10
  - Channel Select, D30SEL (EP3)............................ 186.12
  - Process DATANET-30* Error Status Return, GEEPR (EP4).... 186.14
  - Check for Waiting Connect to Slave, SWAP (EP10)......... 186.16
  - Remote Interrupt Processor Module Initialization, .IDNET 186.18
- Remote Access MME Processor Module (.MROUT)................. 186.20
  - .MROUT MME GEROUT Processor, GROUT (EP1) 186.21

GLOSSARY 187

---

*DATANET, Reg. Trademark of the General Electric Company.

ILLUSTRATIONS

1. Routines And Subroutines Used to Handle I/O Requests And Interrupts 3

2. Schematic of The Control Flow of The I/O Request Handler 4

3. Schematic of The Control Flow of The Interrupt Handler (IOTRM) 5

4. IOS Service Routines And Initialization Subroutine 6

1. INTRODUCTION TO IOS

Input/output supervision is accomplished by a group of GECOS-III modules generically referred to as the I/O Supervisor (IOS). The main functions of IOS are to initiate I/O and to respond to I/O termination. In addition, IOS provides I/O interrupt supervision, queueing I/O requests, translating file codes to physical units, and file protection.

The basic strategy of I/O processing in GECOS is done by the use of I/O queues. A threaded list, one per I/O channel, has its origin in the channel primary System Configuration Table (SCT) and winds its way through Slave Service Areas (SSA), terminating back in the SCT. An I/O queue entry contains images of the primary and secondary mailboxes as required by the hardware. This arrangement helps minimize the elapsed time between a terminate and a connect on an I/O Controller (IOC) channel. The presence of an I/O queue at connect time allows optimum selection of the queue to be connected. It permits the implementation of techniques such as latency reduction and priority scheduling for error recovery.

Each queued I/O request points to a logical device SCT. Only at connect time is the physical IOS device and channel number chosen. This technique permits dynamic switching of the physical device address. For example, assume a tape handler becomes inoperative. Since the I/O queue entry still points to a logical device and channel and since the tape position is recorded in the device secondary SCT, GEPR can switch, upon operator concurrence, all requests for this particular logical device secondary SCT to another unit after it has repositioned the tape. The primary and secondary SCT are described in CPB-1488.

Each primary SCT contains the absolute address of a device-dependent channel module. Entry points, location of constants, seek commands, words per hardware block, etc., for such modules are described in the .CRCTn and in the to the device-dependent channel modules (see Chapter 3) for such functions as building I/O queues, selecting a queue for I/O, processing an interrupt, calculating a seek address, etc. This technique permits adding new devices by simply writing device-dependent modules for the device.

Input/Output Supervision is accomplished by a main IOS module:

- .MIOS Main IOS Module

and various channel modules which service peripherals:

.MCPIO	Card Punch
.MDR20	MDS200 Magnetic Drum Subsystem
.MDS20	DSU200 Disc Storage Subsystem
.MGPIO	Card Reader
.MMTAP	Magnetic Tape
.MPRIO	Printer
.MPTAP	Paper Tape
.MTYPE	Typewriter

and an accounting tape switching module:

- .MACTS

With the exception of the .MACTS module, all IOS modules are part of the hard core monitor (HCM). The .MACTS module is called into memory only when required.

Remote Input/Output Supervision is accomplished by the following two modules:

.MDNET	Remote Interrupt Processor
.MROUT	Remote Access MME Processor

The various routines comprising the .MIOS module are described in Chapter 2. Chapter 3 contains descriptions of the channel modules. The .MACTS module is described in Chapter 4. Remote I/O Supervision is described in Chapter 5.

A glossary and an index are included for user convenience.

2. MAIN IOS MODULE (.MIOS)

The .MIOS module is the major portion of IOS. It is composed of a number of routines used to accomplish two functions: handle I/O requests and handle interrupts. These routines, the majority of which are labeled and numbered by entry points (EP), are shown in Figure 1. Also shown in the figure are various subroutines used jointly by the two handlers. The main IOS module, designated .MIOS, then, is divided by function into two major parts:

- I/O Request Handler
- Interrupt Handler

The flow of the I/O Request Handler is shown in Figure 2; the flow of the Interrupt Handler is shown in Figure 3.

I/O REQUEST HANDLER ROUTINES AND SUBROUTINES

Routines

LINK	(EP1)	Link I/O to End of Queue
LINKF	(EP2)	Link I/O to Front of Queue
LINKR	(EP3)	Link Reissued I/O to Front of Queue
QUEUE	(EP4)	Assign an I/O Entry
INOS	(EP5)	MME GEINOS Processor
SPEC	(EP6)	MME GESPEC Processor
ITYM	(EP7)	Master Message Processor

Subroutines

PTRVL	Pointer Validation
DCWCK	DCW Pointer Validation
FNDFC	Find File Code
FMTAR	Format Error Accounting Record

INTERRUPT HANDLER ROUTINE

IOTRM

SERVICE SUBROUTINES USED BY BOTH HANDLERS

UNLINK	Unlink I/O Entry
ILPCX	Calculate Logical Primary Channel Index
STRET	Status Return
STIO	Start I/O
STIOM	Connect Multirecord Simulation DCW
TYPER	Connect Reissue of Second Typewriter Command
STGPC	Connect Selected GESPECed Entry

Figure 1. Routines And Subroutines Used to Handle I/O Requests And Interrupts

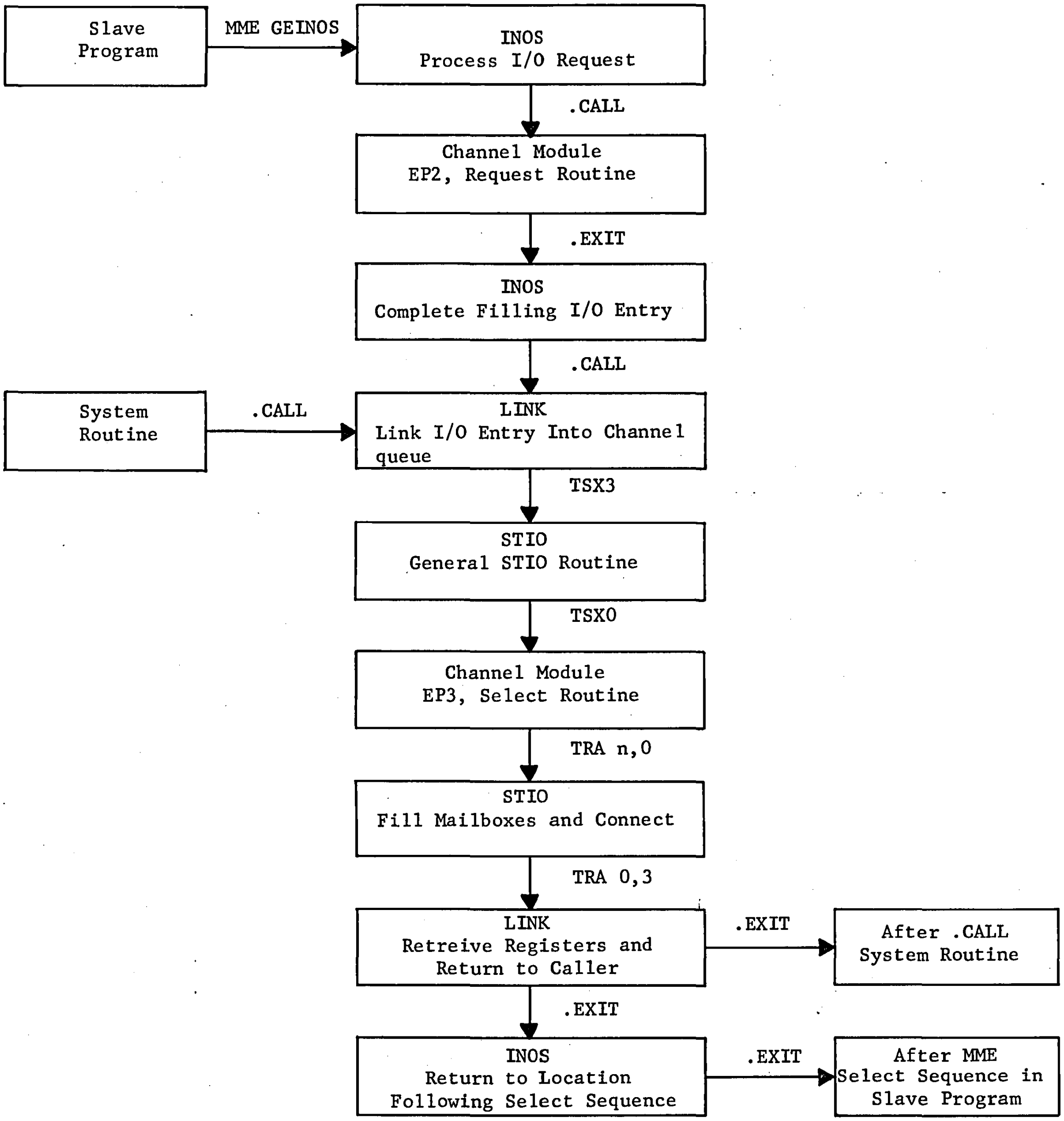

Slave Program
MME GEINOS
INOS
Process I/O Request
.CALL
Channel Module
EP2, Request Routine
.EXIT
INOS
Complete Filling I/O Entry
.CALL
System Routine
.CALL
LINK
Link I/O Entry Into Channel queue
TSX3
STIO
General STIO Routine
TSX0
Channel Module
EP3, Select Routine
TRA n,0
STIO
Fill Mailboxes and Connect
TRA 0,3
LINK
Retreive Registers and Return to Caller
.EXIT
After .CALL System Routine
.EXIT
INOS
Return to Location Following Select Sequence
.EXIT
After MME Select Sequence in Slave Program

Figure 2. Schematic of The Control Flow of The I/O Request Handler

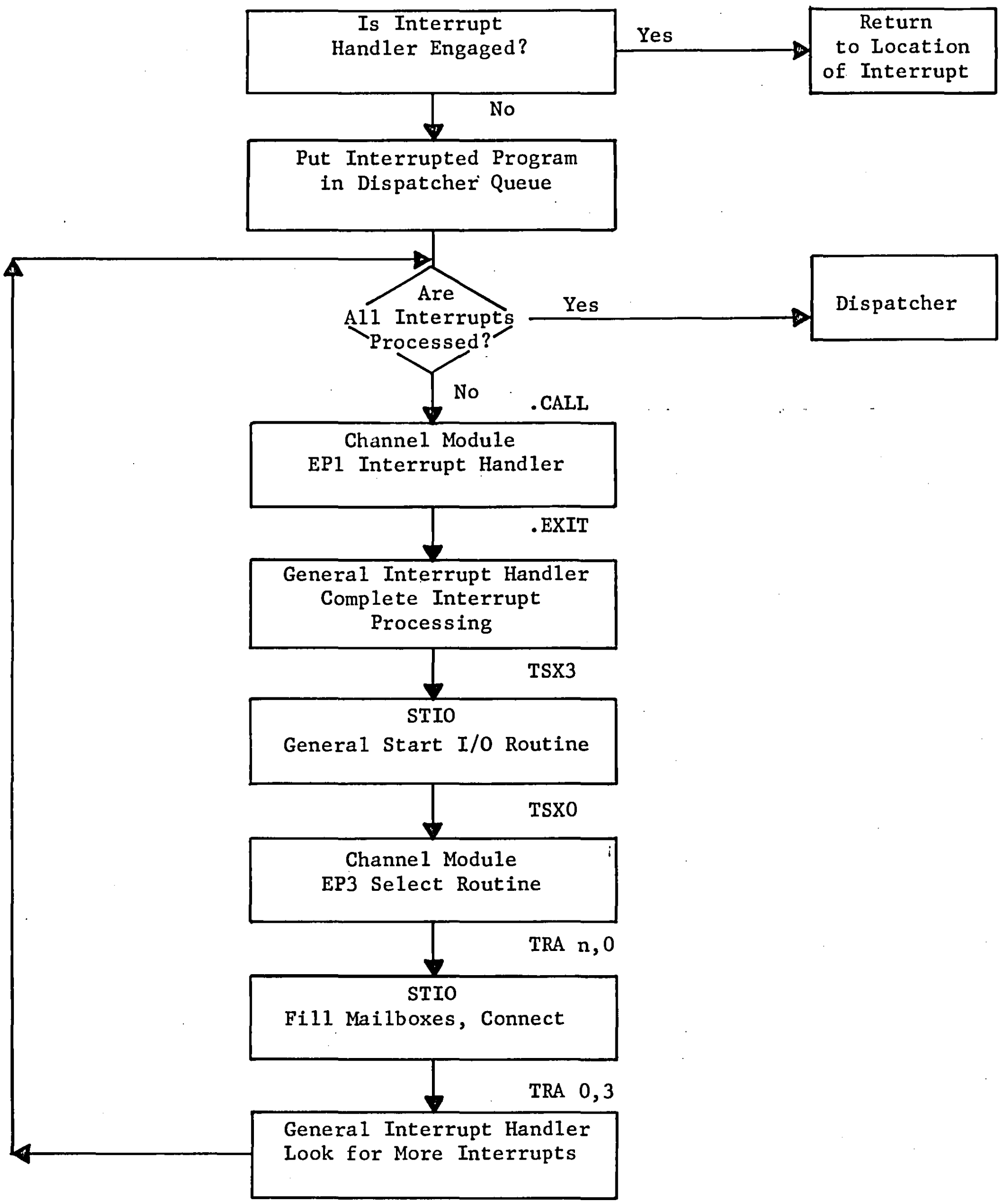

Is Interrupt Handler Engaged?
Yes
Return to Location of Interrupt
No
Put Interrupted Program in Dispatcher Queue
Are All Interrupts Processed?
Yes
Dispatcher
No
.CALL
Channel Module EP1 Interrupt Handler
.EXIT
General Interrupt Handler Complete Interrupt Processing
TSX3
STIO General Start I/O Routine
TSX0
Channel Module EP3 Select Routine
TRA n,0
STIO Fill Mailboxes, Connect
TRA 0,3
General Interrupt Handler Look for More Interrupts

Figure 3. Schematic of The Control Flow of The Interrupt Handler (IOTRM)

In addition to the two major Handlers, IOS contains various service routines that are used by other (external to IOS) GECOS modules. These routines and the services they provide are listed in Figure 4.

SERVICE ROUTINES USED EXTERNAL TO IOS

RSMIO (EP8)	Resume I/O for Program
ABTIO (EP9)	Abort I/O for Program
GSTRT (EP10)	Format I/O Status Words and Return Status
RSMCH (EP12)	Resume I/O on Channel
ACTFL (EP13)	Accounting File Request

INITIALIZATION SUBROUTINE

.IIOS

Figure 4. IOS Service Routines and Initialization Subroutine

I/O REQUEST HANDLER

As indicated in Figure 1, the I/O Request Handler comprises:

LINK	(EP1)	Link I/O to End of Queue
LINKF	(EP2)	Link I/O to Front of QUEUE
LINKR	(EP3)	Link Reissued I/O to Front of Queue
QUEUE	(EP4)	Assign an I/O Entry
INOS	(EP5)	MME GEINOS Processor
SPEC	(EP6)	MME GESPEC Processor
ITYM	(EP7)	Master Message Processor
PTRVL		Pointer Validation
DCWCK		DCW Pointer Validation
FNDFC		Find File Code
FMTAR		Format Error Accounting Record

which are described in the following pages.

LINK I/O TO END OF QUEUE

LINK (EP1 of .MIOS) links an I/O entry which has been filled into the end of the I/O queue for the logical channel involved. The I/O entry status is set to linked.

If status returns are requested, the program request count is increased by one and the first status return word is set to zero.

If the linked I/O entry is GESPECed and the special interrupt has not occurred, no attempt is made to start the I/O.

If the linked I/O entry is not GESPECed or the special interrupt has occurred, STIO is called in an attempt to start I/O.

Additional entries are LINKF (EP2 of .MIOS) and LINKR (EP3 of .MIOS).

PRECALLING SEQUENCE

Prior to entering LINK, the registers listed must contain the data indicated.

X4	Offset to LAL of the I/O entry which has been filled
X5	LAL for program
X6	Program number
X7	Processor number

CALLING SEQUENCE

LINK can be called by any master mode system routine operating under a program number.

```
        8         16
       |.LINK    |
       |return   |
       |         |
or     |.CALL    |.MIOS,1
       |return   |
       |         |
```

OPERATING SYSTEM INTERACTION

The top entry in the stack is the location of the call to this routine. Additional stack entries are used by this routine for temporary storage. Also, routines which are called by LINK use the stack for temporary storage.

The .CRQGT gate is used when the linking is performed and when channel status is interrogated. The gate is shut by using a system shut (.SHUTS) before STIO is called and the gate is opened in that subroutine.

ROUTINE RETURNS

Return is made to the location immediately following the call to LINK. Register status:

AR Destroyed
QR Destroyed

All index registers are restored.

POSTCALLING SEQUENCE

The requestor can roadblock or relinquish following the I/O by calling .GROAD or .GRELC.

If the I/O is already completed at the time of the relinquish, the requesting program is immediately eligible for execution. A roadblock is not broken until all I/O requests for the program are completed.

SUPPORTING INFORMATION

Programming Method

LINK is reentrant and written in floatable code.

Interrupts are inhibited while gates are closed and while storing and retrieving in the stack.

Storage

Internal temporary storage is used while the .CRQGT gate is shut.

LINK occupies approximately 140 core storage locations.

Other Routines Used

Start I/O, STIO (.MIOS).
Calculate Logical Primary Channel Index, ILPCX (.MIOS).

Flowchart

See CPB-1500 for the flowchart of LINK (EP1), .MIOS module.

LINK I/O TO FRONT OF QUEUE

LINKF (EP2 of .MIOS) links an I/O entry which has been filled into the front of the I/O queue for the logical channel involved.

LINKF is an entry point in the basic LINK routine. See LINK (EP1 of .MIOS).

Flowchart

See CPB-1500 for the flowchart of LINKF (EP2), .MIOS module.

LINK REISSUED I/O TO FRONT OF QUEUE

LINKR (EP3 of .MIOS) links an I/O entry which is being reissued by GEPR into the front of the I/O queue for the logical channel involved.

LINKR is an entry point in the basic LINK routine. See LINK (EP1 of .MIOS).

Flowchart

See CPB-1500 for the flowchart of LINKR (EP3), .MIOS module.

QUEUE(EP4)
.MIOS

ASSIGN AN I/O ENTRY

QUEUE (EP4 of .MIOS) assigns an I/O entry space for a program. The I/O entries begin at .SIOQ,5 (X5=LAL) and .SNIO,5 contains the total number of entries in bits 18-35. Each 11-word entry is examined beginning at .SIOQ,5 and the first available entry is assigned to the requestor. The I/O entry status is set to building.

PRECALLING SEQUENCE

Prior to entering LINK, the registers listed contain the data indicated.

X5 LAL for program
X6 Program number
X7 Processor number

CALLING SEQUENCE

QUEUE can be called by any master mode system routine operating under a program number.

```
           8             16
          ---------------------------
          | .QUEUE     |
          | return     |
          |            |
or        | .CALL      | .MIOS,4
          | return     |
          |            |
```

OPERATING SYSTEM INTERACTION

The top entry in the stack is the location of the call to this routine. One additional stack entry is used for temporary storage.

No gates are used.

ROUTINE RETURNS

Return is made to the location immediately following the call to QUEUE. Register status:

AR Destroyed
QR Destroyed

All index registers except X4 are restored.

Index register 4 contains the response:

X4=0, no I/O entry assigned
X4≠0, I/O entry assigned; X4 contains the offset to LAL of the I/O entry which has been assigned.

If an I/O entry is assigned, words 2-10 (of 0-10) contain zeros.

POSTCALLING SEQUENCE

The response and test for an available I/O entry is made absolute by entering the following instructions after the .QUEUE call:

```
 8           16
|ADLX4      |.CRLAL,6
|TRC        |               return if entry assigned
|...        |               return if no entry assigned
```

If no entry was assigned, control can be relinquished by a call to .GRELC. When I/O terminates, an entry will be made available; the requestor should call QUEUE again.

SUPPORTING INFORMATION

Programming Method

QUEUE is reentrant and written in floatable code.

Interrupts are inhibited while searching of the entry area, while setting a selected entry status to building, and when storing temporarily in the stack.

Storage

No internal temporary storage is used.

QUEUE occupies approximately 24 core storage locations.

Other Routines Used

None

Nomenclature

I/O entry is an 11-word area used to contain information for an I/O request.

Open status indicates an I/O entry is available. Bits 30-35 of word 0=00 octal.

Building status indicates that the I/O entry is assigned and is being filled. Bits 30-35 of word 0=01 octal.

Flowchart

See CPB-1500 for the flowchart of QUEUE (EP4), .MIOS module.

MME GEINOS PROCESSOR

INOS (EP5 of .MIOS) processes an I/O request (MME GEINOS) for a slave-mode or a master-mode program. Slave-mode requests are made by a MME GEINOS fault; master-mode requests are made by a .GINOS call.

INOS checks the validity of each I/O select sequence and performs device-dependent tests by calling the channel module involved. After the I/O request is validated, an I/O entry is filled and the LINK (EP1 of .MIOS) routine is called to link the entry into the channel queue so that it is eligible for initiation.

PRECALLING SEQUENCE

Slave Mode (MME GEINOS Fault)

Prior to a slave-mode program entering INOS, the fault processor saves slave registers and sets the index registers listed below to the value indicated.

X5	LAL for program
X6	Program number
X7	Processor number

The pointers within the select sequence should be offset to the LAL.

Validity checks permit pointers only within the slave area.

Data addresses in DCW should be relative to the LAL of the slave program.

Master Mode (.GINOS Call)

Prior to a master mode program entering INOS, the registers listed must contain the data indicated:

X5	LAL for program
X6	Program number
X7	Processor number

Entering From a Slave Area - The pointers within the select sequence should be offset to the LAL of the slave area.

Validity checks permit pointers only within the slave area.

Data Addresses in DCW should be relative to the LAL of the slave program.

Entering From a Slave Service Area - The pointers within the select sequence should be offset to the LAL of the slave area.

Validity checks permit pointers within the slave service area and the slave area. The first DCW must be within the slave service area.

Data addresses in DCW must be relative to the beginning of the first slave service area.

CALLING SEQUENCE

Slave Mode (Single Command)

```
8                  16
MME              | GEINOS
I/O command      |
ZERO             | File pointer,DCW pointer
ZERO             | Status return address,courtesy call address
```

Slave Mode (Dual Command - disc, drum, typewriter)

```
8                  16
MME              | GEINOS
I/O command1     |
ZERO             | File pointer,DCW pointer1
I/O command2     |
ZERO             | File pointer,DCW pointer1
```

Master Mode (Single Command)

```
8                  16
.GINOS           |
I/O command      |
ZERO             | File pointer,DCW pointer
ZERO             | Status return pointer,courtesy call address
```

Master Mode (Dual Command - disc, drum, typewriter)

```
8                  16
.GINOS           |
I/O command1     |
ZERO             | File pointer,DCW pointer1
I/O command2     |
ZERO             | ,DCW pointer2
ZERO             | Status return pointer,courtesy call address
```

OPERATING SYSTEM INTERACTION

When control is given to INOS, the top entry in the stack is the instruction counter and indicators (IC and I) of the MME (slave mode) or the IC and I of the second word of the 2-word expansion of the .GINOS call. The stack is also used for IC and I of other routines called by INOS and for temporary storage, see LINK (EP1) and QUEUE (EP4).

INOS does not use gates; however, it calls routines that do.

The INOS routine uses the following .STEMP storage:

.STEMP, +1, +2, +3

The MME and .GINOS calls use the following .STEMP storage:

.STEMP+8,9

ROUTINE RETURNS

The return is made to the location following the select sequence.

Registers are restored if the request was made via a MME; registers are not restored following a .GINOS call.

POSTCALLING SEQUENCE

None.

SUPPORTING INFORMATION

Programming Method

INOS is reentrant and written in floatable code.

Interrupts are not inhibited.

Storage

No internal temporary storage is used.

INOS occupies approximately 285 core storage locations.

Other Routines Used

Link I/O to End of Queue, LINK (EP1 of .MIOS)
Assign an I/O Entry, QUEUE (EP4 of .MIOS)
Calculate Logical Primary Channel Index, ILPCX (.MIOS)
DCW Pointer Validation, DCWCK (.MIOS)
Pointer Validation, PTRVL (.MIOS)
Status Return, STRET (.MIOS)
Applicable Channel Modules (EP2 of .MCPIO, .MDR20, .MDS20, .MGPIO, .MMTAP, .MPRIO, .MPTAP, or .MTYPE).

Calling Sequence to EP2 - Prior to entering EP2, the registers listed must contain the data indicated:

X1	Device SCT address
X2	Logical primary channel index
X3	Select sequence address
X4	I/O entry address
X5	LAL for program
X6	Program number
X7	Processor number

If the request is from the slave area, .STEMP,5 is zero; it is nonzero if the request is from the slave service area.

The upper address limit of the program is in bits 0-17 of .STEMP+1,5, and the offset to the PAT pointer word is in bits 18-35.

EP2 is entered by the following:

```
8          16
LDA        2,DU          }
STC1       .SSTAK,5*     }   simulated .CALL
TRA        .CRCT4,2*     }
return 0                     transfer vector
  :                          (.EXIT through .EXIT 6)
return 6
```

Returns from EP2 - Any of seven returns may be made, depending upon the condition specified.

Return 0 (.EXIT) Normal data transfer, 3-word select sequence. .STEMP,5 is zero if the request is from the slave area; it is nonzero if the request is from the slave service area.

.STEMP+1,5 has the upper address limit of the program in bits 0-17 and the offset to the PAT pointer word in bits 18-35.

Registers contain:

AR	I/O command
X3	Select sequence address
X4	I/O entry address
X5	LAL for program
X6	Program number
X7	Processor number

Return 1 (.EXIT 1)	Disc/drum nondata transfer. The contents of .STEMP, .STEMP+1, and index registers are the same as for return 0. AR Command flag, REW=0, FSR=+n, BSR=-n where n is the number of records to be forward-spaced (FSR) or backspaced (BSR).
Return 2 (.EXIT 2)	Force GEPR override. The contents of .STEMP, STEMP+1, AR, and index registers are the same as for return 0.
Return 3 (.EXIT 3)	Write single character command. The contents of .STEMP, .STEMP+1, AR, and index registers are the same as for return 0.
Return 4 (.EXIT 4)	Two typewriter commands, both I/O commands stored in I/O entry. The contents of .STEMP, .STEMP+1, and index registers are the same as for return 0.
Return 5 (.EXIT 5)	Two disc/drum data transfer commands, first I/O command is stored in the I/O entry. The contents of .STEMP, .STEMP+1, and index registers are the same as for return 0. AR Second I/O command
Return 6 (.EXIT 6)	Illegal I/O command. The contents of .STEMP, .STEMP+1, and index registers are the same as for return 0.

Flowchart

See CPB-1500 for the flowchart of INOS (EP5), .MIOS module.

MME GESPEC PROCESSOR

SPEC (EP6 of .MIOS) processes a MME GESPEC for a slave-mode or a master-mode program. This request delays initiating a subsequent I/O request for a device until a special interrupt occurs on the device. Slave-mode requests are made by a MME GESPEC fault; master-mode requests are made by a .GSPEC call. In addition, SPEC processes a Minus MME GESPEC (or a Minus .GSPEC call) which removes a previous MME GESPEC (or .GSPEC call) request.

PRECALLING SEQUENCE

Slave Mode (MME GESPEC Fault)

Prior to a slave-mode program entering SPEC, the fault processor saves slave registers and sets the index registers listed below to the value indicated.

X5	LAL for program
X6	Program number
X7	Processor number

The Q-register must be loaded with a parameter indicating either a MME GESPEC or a Minus MME GESPEC.

```
          LDQ     parameter

where parameter   (bit 0)        is 0 for MME GESPEC
                       or        is 1 for Minus MME GESPEC
                  (bits 1-29)    are not examined

                  (bits 30-35)   are File code
```

Master Mode (.GSPEC Call)

Prior to a master-mode program entering SPEC, the registers listed must contain the data indicated.

X5	LAL for program
X6	Program number
X7	Processor number

The register must be loaded with a parameter indicating either a .GSPEC call or a Minus .GSPEC call. (See above for parameter definition.)

CALLING SEQUENCE

SPEC may be called by either a slave mode or a master mode program.

Slave Mode

```
         8           16
        ________________________________
        | MME        |GESPEC
        | return     |
```

The A-register contains the Response indicating what action was taken

```
where     Response = 0 if action was taken
                   = 2 if no action was taken  (File code not in PAT or
                     request is not allowed for the device.)
```

Master Mode

```
         8           16
        ________________________________
        |.GSPEC      |
        |return      |
```

The A-register contains the Response indicating what action was taken. (See above for Response definition.)

OPERATING SYSTEM INTERACTION

When control is given to SPEC, the top entry in the stack is the IC and I of the second word of the 2-word expansion of the .GSPEC call. The routine does not use the stack for internal temporary storage.

The .CRQGT gate is used by the routine when interrogating channel status and changing linking pointers.

SPEC uses the .STEMP+7 storage.

ROUTINE RETURNS

Registers are restored if request was made by a MME GESPEC. Registers are not restored if request was made by a .GSPEC call. The A-register, as indicated above, contains an indication of the response to the request. Return is made to the location following the MME or .GSPEC call.

POSTCALLING SEQUENCE

A test is made on the Response returned in the A-register to determine whether the request was denied:

```
8        16
----------------------
CMPA     | 0,DU
TZE      | "OK"
denied   |
```

SUPPORTING INFORMATION

Programming Method

SPEC is reentrant and written in floatable code.

Interrupts are inhibited while the .CRQGT gate is shut.

Storage

No internal temporary storage is used.

SPEC occupies approximately 80 core storage locations.

Other Routines Used

Calculate Logical Primary Channel Index, ILPCX (.MIOS)
Find File Code, FNDFC (.MIOS)
Unlink I/O Entry, UNLNK (.MIOS)

Flowchart

See CPB-1500 for the flowchart of SPEC (EP6), .MIOS module.

MASTER MESSAGE PROCESSOR

ITYM (EP7 of .MIOS) processes a typewriter I/O request from a system routine. This routine examines the calling sequence and fills the I/O provided, mapping the message code into file code and carriage control information. The table defining message codes resides in this program. The filled I/O entry is linked, and the I/O associated with that entry is eligible for initiation.

PRECALLING SEQUENCE

Prior to entering ITYM, the registers listed must contain the data indicated:

X0	Absolute address of calling sequence
X3	Nonzero if courtesy call address (CCA) is absolute
X4	Offset to LAL of the I/O entry which will be filled
X5	LAL for program
X6	Program number
X7	Processor number

The 3-word calling sequence referenced by X0 is as follows:

```
8        16
VFD      1/Master,1/First,34/Mesg Code
ZERO     DCWW,DCWR
ZERO     SRP,CCA
```

where:

Master (Limits Flag) = 1 Master limits requested. DCWs contain absolute data addresses.

= 0 Request from slave area. Use regular slave limits; data addresses are relative to slave LAL.

Request from slave service area. Use slave service area/slave area limits; data addresses are relative to the beginning of first slave service area.

First (Link Flag) = 1 Link I/O entry to front of channel queue.
= 0 Link I/O entry at end of channel queue.

Mesg Code (The code ITYM uses to map in file code and carriage control information):

Octal Code	Message	Mapping	
		File Code	Carr. Code
1	Mag Tape error; typewriter console reply	T*	1
2	Other error; typewriter console reply	T/	1
3	Mag tape threshold	T*	3
4	Other threshold	T/	3
5	*MNT, *DMT, *RDY	T*	2
6	Card reader, card punch information	T/	2
7	*SRT, *FIN	*T	4
10	*ABT	*T	4
11	Allocation retrieve	/T	2
12	Job delayed	*T	3
13	$ COMMENT cards	*T	2
14	GEPOP console typewriter TY1	T*	1
15	GEPOP console typewriter TY2	*T	1
16	GEPOP console typewriter TY3	T/	1
17	GEPOP console typewriter TY4	/T	1
20	Remote GEIN	T*	4
21	Time-Sharing link storage	T*	4
22	Time-Sharing invalid file system status	T*	4
23	Time-Sharing *SRT, *FIN	*T	4

DCWW (Write DCW Pointer)

Offset to LAL

First DCW must point to a 2-word buffer into which ITYM stores carriage control information.

Data addresses (see Master above)

DCWR (Read DCW Pointer)

If no read, DCWR = 0

Offset to LAL

Minimum 2-word read; ITYM blanks first two words when request is processed

Data address (see Master above)

---

*Carriage codes:
1 = One carriage return
2 = One carriage return; one tab
3 = One carriage return; two tabs
4 = One carriage return; three tabs

SRP (Status Return Pointer)

If no status requested, SRP = 0

Offset to LAL

Must point to first of two words

If operator distracted status, I/O will be reissued by GEPR.

If any GEPR override options are desired, the proper flags must be set in the I/O entry before calling ITYM.

CCA (Courtesy Call Address)

If no courtesy call, CCA = 0
If CCA ≠ 0

X3 = 0 Offset to LAL
X3 ≠ 0 Absolute

CALLING SEQUENCE

ITYM can be called by any master-mode system routine operating under a program number.

```
          8       16
          .ITYM
          return
or
          .CALL   MIOS,7
          return
```

OPERATING SYSTEM INTERACTION

When control is given to ITYM, the IC and I is the top entry in the stack. The ITYM routine temporarily stores registers in three stack entries until the LINK routine is called. The LINK routine also uses stack entries for temporary register storage.

ITYM does not use any gates but may call routines that do.

No .STEMP storage is used.

ROUTINE RETURNS

Return is made to the location immediately following the call to ITYM. Register status:

AR	Destroyed
QR	Destroyed

All index registers are restored.

POSTCALLING SEQUENCE

When return is made following the call to ITYM, the I/O has been linked. I/O may have been initiated or may have been terminated already. The status return pointer is zeroed when I/O entry is linked.

The caller can relinquish following the call. If the I/O has already terminated, the program is immediately available for processor time. A roadblock is broken when all I/O is completed.

SUPPORTING INFORMATION

Programming Method

ITYM is reentrant and written in floatable code.

Interrupts are inhibited while storing and retrieving in the stack.

Storage

No internal temporary storage is used.

ITYM occupies approximately 110 core storage locations.

Other Routines Used

None.

Flowchart

See CPB-1500 for the flowchart of ITYM (EP7), .MIOS module.

POINTER VALIDATION

PTRVL (.MIOS) checks the validity of a pointer within the I/O select sequence. Select sequences within the slave area can access only the slave area. Select sequences within the slave service area can access the first slave service area and the slave area.

PRECALLING SEQUENCE

Prior to entering PTRVL, the registers listed must contain the data indicated.

X1	Pointer (offset)
X5	LAL for program
X6	Program number
X7	Processor number

CALLING SEQUENCE

PTRVL is called from the INOS routine located in the .MIOS module.

```
8         16
TSXO      PTRVL,$
return 0
return 1
```

OPERATING SYSTEM INTERACTION

The stack is not used. PTRVL uses the following .STEMP storage:

.STEMP,5 .STEMP+1,5

No gates are used.

ROUTINE RETURNS

Return 0	(TRA 2,0).	Normal return. All registers are intact except X1 which contains Absolute Pointer -1.
Return 1	(TRA 0,0).	Error return. All registers are intact except X1 which is destroyed.

POSTCALLING SEQUENCE

None.

SUPPORTING INFORMATION

Programming Method

PTRVL is reentrant and written in floatable code.

Interrupts are not inhibited.

Storage

No internal temporary storage is used except the .STEMP locations.

PTRVL occupies approximately 10 core storage locations.

Other Routines Used

None.

Flowchart

See CPB-1500 for the flowchart of PTRVL, .MIOS module.

DCW POINTER VALIDATION

DCWCK (.MIOS) validates a DCW pointer in an I/O select sequence and checks for the presence of a transfer to DCW (TDCW) as the first DCW in the indicated string.

PRECALLING SEQUENCE

Prior to entering DCWCK, the registers listed must contain the data indicated.

X3	Address of I/O command which precedes the DCW pointer in select sequence
X4	Address of I/O entry being filled
X5	LAL for program
X6	Program number
X7	Processor number

CALLING SEQUENCE

DCWCK is called from the INOS routine located in the .MIOS module.

```
8          16
TSX2      |DCWCK,$
return 0  |
return 1  |
```

OPERATING SYSTEM INTERACTION

The stack is not used. DCWCK uses the following .STEMP storage:

.STEMP+2,5

No gates are used.

ROUTINE RETURNS

Return 0 (TRA 0,2). Normal return. Registers contain:

AR	First DCW
QR	Destroyed
X0	Destroyed
X1	DCWP+1 offset
X2	through X7 are restored

DCWP offset is stored in I/O entry SMX4, word 8.

Return 1 (TRA INABQ,$) Error return. Index registers X4 through X7 are restored.

Abort code 38 is returned if two TDCW's were given in succession. If the DCW pointer is invalid, an abort code 35 is returned.

POSTCALLING SEQUENCE

For a 3-word calling sequence:

```
          STA      5,4  Store first DCW (SMX1) in I/O entry

          STX1     6,4  Store DCWP+1 offset (SMX2) in I/O entry
```

For a 5-word calling sequence:

```
          STA      5,4  Store first DCW (SMX1) in I/O entry.
```

No action is taken on the second command.

SUPPORTING INFORMATION

Programming Method

DCWCK is reentrant and written in floatable code.

Interrupts are not inhibited.

Storage

No internal temporary storage is used, except the .STEMP locations.

DCWCK occupies approximately 25 core storage locations.

Other Routines Used

MME GEINOS Processor, INOS (EP5 of .MIOS)

Flowchart

See CPB-1500 for the flowchart of DCWCK, .MIOS module.

FIND FILE CODE

FNDFC (.MIOS) accepts a file code as input and determines if the file code is defined for the slave program. If the file code is defined, the offset to the Peripheral Assignment Table (PAT) pointer word is returned.

PRECALLING SEQUENCE

Prior to entering FNDFC, the registers listed must contain the data indicated.

AR	File code (bits 30-35; bits 0-29, not examined)
X5	LAL for program
X6	Program number
X7	Processor number

CALLING SEQUENCE

FNDFC is called from various routines in the .MIOS module.

```
8          16
TSX0       FNDFC,$
return 0
return 1
```

OPERATING SYSTEM INTERACTION

No stack or .STEMP storage is used.

No gates are used.

ROUTINE RETURNS

Return 0	(TRA 0,0)	Normal return. Index register 4 contains the offset to the PAT pointer/file code word.
Return 1	(TRA 1,0)	Error return. File code not found.

POSTCALLING SEQUENCE

None.

SUPPORTING INFORMATION

Programming Method

FNDFC is reentrant and written in floatable code.

Interrupts are not inhibited.

Storage

No internal temporary storage is used.

FNDFC occupies approximately 10 core storage locations.

Other Routines Used

None.

Flowchart

See CPB-1500 for the flowchart of FNDFC, .MIOS module.

FORMAT ERROR ACCOUNTING RECORD

FMTAR (.MIOS) formats an error accounting logical record and calls ACTFL (EP13 of .MIOS) to insert the record into an accounting buffer. FMTAR is called only by the Interrupt Handler (IOTRM). It assumes the identity of GEPOP (Program number and LAL) prior to calling ACTFL. A nonnormal status (not Channel Ready or EOF) usually causes IOTRM to call FMTAR. If the nonnormal status is a result of GEPR reissuing I/O, FMTAR is not called by IOTRM.

PRECALLING SEQUENCE

Prior to entering FMTAR, the registers listed must contain the data indicated.

QR	Status word 1
X0	Type of interrupt (TI=0, II=2, SI=4)
X1	True channel index
X2	Logical channel index
X4	I/O entry address
X5	LAL for GEPOP
X6	Program number for I/O entry
X7	Processor number

Type of interrupt may be Termination Interrupt (TI), Initiation Interrupt (II), or Special Interrupt (SI).

CALLING SEQUENCE

FMTAR is called from the IOTRM routine located in the .MIOS module.

```
8        16
TSX3     FMTAR,$
return
```

OPERATING SYSTEM INTERACTION

FMTAR does not use the stack; however, it calls routines that do.

No .STEMP storage is used.

FMTAR does not use gates; however, it calls routines that use the .CRACF and .CRQGT gates.

ROUTINE RETURNS

Return (TRA 0,3) Logical record formatting and disposal completed. Registers contain:

QR	Status word 1
X0	Type of interrupt (TI=0, II=2, SI=4)
X1	True channel index
X2	Logical channel index
X3	Transfer register
X4	I/O entry address
X5	LAL for GEPOP
X6	Program number for I/O entry
X7	Processor number

POSTCALLING SEQUENCE

None.

SUPPORTING INFORMATION

Programming Method

FMTAR is nonreentrant since IOTRM uses a software gate to ensure processing one interrupt at a time. It is written in floatable code.

Interrupts are not inhibited.

Storage

Internal temporary storage is used.

FMTAR occupies approximately 95 core storage locations.

Other Routines Used

Accounting File Request, ACTFL (EP13 of .MIOS)

Flowchart

See CPB-1500 for the flowchart of FMTAR, .MIOS module.

INTERRUPT HANDLER

The Interrupt Handler is comprised of only one routine, IOTRM. It is entered through one of four entries. The specific entry used depends upon the type of interrupt from an IOC.

IOTRM (.MIOS) performs preliminary analysis of a control processor interrupt resulting from a peripheral interrupt. After the preliminary analysis is performed, control is given to a channel module to complete further device-dependent processing. When this is completed, the channel module returns to IOTRM to complete the processing of the interrupt.

PRECALLING SEQUENCE

At startup, the .MIOS Initialization (.IIOS) initializes the interrupt vector so that pairs of instructions are provided for each interrupt type for each IOC configured. Interrupt types consist of Termination Interrupts (TI), Initiation Interrupts (II), Special Interrupts (SI), and Counterparity Interrupts (CI).

Four entries to IOTRM are provided for each IOC configured. These four entries are used for the four types of interrupts. Separation of interrupt types is for trace purposes. Once the trace is made, all interrupt types from all IOCs enter IOTRM at the same location. The CI results in a derail; the others are processed.

CALLING SEQUENCE

IOTRM gets control when a pair of instructions in the interrupt vector are executed.

```
8            16
STCI         ICI1
TRA          IOC interrupt-type routine
```

If interrupts are not inhibited at the time the peripheral terminates, the pair of instructions in the interrupt vector are executed immediately, and the IC and I of the interrupt are stored in ICI1. Then, IOTRM is given control.

If interrupts are inhibited, the pair of cells in the interrupt vector are not executed until the inhibit is removed.

OPERATING SYSTEM INTERACTION

While IOTRM is in control (by the control processor), no program is in execution. However, IOTRM uses the stack of GEPOP for calls, exits, and temporary storage.

No .STEMP storage is used.

Four gates are used by IOTRM:

- .CRQGT - when manipulating channel queues and status
- .CRCCS - when manipulating courtesy call queue
- .CRDSP - when manipulating program state
- .CRGPR - when processing GEPR-required interrupts

ROUTINE RETURNS

After IOTRM has processed all current entries in the interrupt queue, control is returned to the Dispatcher (.MDISP) either at the location of the interrupt, if no program was in execution, or at the beginning of the Dispatcher (EP1 of .MDISP) if a program was in execution.

No Program in Execution

```
     RET      ICI2
```

All registers are restored.

Program in Execution

```
     .GOTO     .MDISP,1
```

All registers are saved for the program in execution prior to processing the interrupt.

POSTCALLING SEQUENCE

None.

SUPPORTING INFORMATION

Programming Method

IOTRM is nonreentrant. The memory controller mask is set to allow all interrupts, but the routine employs a software gate to prevent reentry into the actual interrupt processing logic. It is written in floatable code.

Critical portions of the routine are interrupt inhibited.

Storage

Internal temporary storage is used while the IOTRM gate or system gates are shut.

IOTRM occupies approximately 730 core storage locations.

Other Routines Used

Status Return, STRET (.MIOS)
Unlink I/O Entry, UNLNK (.MIOS)
Start I/O, STIO (.MIOS)
Accumulate Processor Time, DACNB (EP9 of .MDISP)
Program Number at End of Queue, DSPQT (EP8 of .MDISP)
Format Error Accounting Record (FMTAR)
Applicable Channel Module (EP1 of .MCPIO, .MDR20, .MDS20, .MGPIO, .MMTAP, .MPRIO, .MPTAP, or .MTYPE)

Calling Sequence to EP1 - Prior to entering EP1, the registers listed must contain the data indicated.

QR	Status word 1
X0	Type of Interrupt (TI=0, II=2, SI=4)
X1	True channel index
X3	Bit 0=0, bit 1=power bit, bits 2-5=major status, bits 6-17=0
X4	I/O entry address
X5	LAL for GEPOP
X6	Program number for I/O entry
X7	Processor number

EP1 is entered by

```
8             16
LDA          |1,DU      \
STC1         |.SSTAK,5*  }   simulated .CALL
TRA          |.CRCT4,2* /
return 0     |
   .         |               transfer vector
   .         |               (.EXIT through .EXIT 13)
   .         |
return 13    |
```

Returns from EP1 - Any of 13 returns may be made, depending upon the condition specified.

Return 0 (.EXIT)   Status return action. Registers contain:

X1	True channel index
X2	Logical primary channel index
X4	I/O entry address
X5	LAL for GEPOP
X6	Program number
X7	Processor number

Return 1 (.EXIT 1)	GEPR action. Register contents are the same as for return 0.
Return 2 (.EXIT 2)	GESPEC action. Registers contain: X1 True channel index X2 Logical primary channel index X5 LAL for GEPOP X7 Processor number
Return 3 (.EXIT 3)	No action on special interrupt. Registers contain: X5 LAL for GEPOP X7 Processor number
Return 4 (.EXIT 4)	Release channel, no I/O entry to release. Registers contain: X1 True channel index X2 Logical primary channel index X4 I/O entry address X5 LAL for GEPOP X7 Processor number
Return 5 (.EXIT 5)	Special interrupt, start I/O. Registers contain: X2 Logical primary channel index X5 LAL for GEPOP X7 Processor number
Return 6 (.EXIT 6)	Error threshold on DSU200 seek. Register contents are the same as for return 0.
Return 7 (.EXIT 7)	Reissue console read. Secondary mailboxes 3 and 4 are already stored, I/O entry is marked in transmission, and the channel is marked busy. Registers contain: QR Primary mailbox image for reissue X1 True channel index X2 Logical primary channel index X4 I/O entry address X5 LAL for GEPOP X6 Program number X7 Processor number
Return 8 (.EXIT 8)	Error threshold action. Register contents are the same as for return 0.

Return 9 (.EXIT 9)	GEPR action for multirecord simulation. Registers contain: QR Status word 1 X1 True channel index X2 Logical primary channel index X4 I/O entry address X5 LAL for GEPOP X6 Program number X7 Processor number
Return 10 (.EXIT 10)	Error threshold for multirecord simulation. Register contents are the same as for return 9.
Return 11 (.EXIT 11)	Status return action for multirecord simulation. Register contents are the same as for return 9.
Return 12 (.EXIT 12)	Issue next connect for multirecord simulation. Register contents are the same as for return 9.
Return 13 (.EXIT 13)	Issue next connect to CPZ100 for multirecord simulation. Secondary mailboxes 1, 2, 3, and 4 are filled. Registers contain: X1 True channel index X2 Logical primary channel index X4 I/O entry address X5 LAL for GEPOP X6 Program number X7 Processor number

Flowchart

See CPB-1500 for the flowchart of IOTRM, .MIOS module.

SERVICE SUBROUTINES USED BY BOTH HANDLERS

As shown in Figure 1, the service subroutines are used by both Handlers:

UNLNK	Unlink I/O Entry
ILPCX	Calculate Logical Primary Channel Index
STRET	Status Return
STIO	Start I/O

These subroutines are "internal;" that is, they are for use by both the I/O Request Handler and the Interrupt Handler. They are called by routines within the .MIOS module.

UNLINK I/O ENTRY

UNLNK (.MIOS) unlinks an I/O entry from an I/O queue and sets new IOS entry status.

PRECALLING SEQUENCE

Prior to entering UNLNK, the registers listed must contain the data indicated.

AR	Mask to save data in word 0 (bits 18-35) of I/O entry. (Bits 0-17 must be zero.)
QR	Data to insert in word 0 (bits 18-35) of I/O entry. (Bits 0-17 must be zero.)
X2	Logical primary channel index
X4	Address of I/O entry to be unlinked
X5	LAL for program
X6	Program number
X7	Processor number

The .CRQGT gate must be shut.

CALLING SEQUENCE

UNLNK is called by routines within the .MIOS module.

```
8        16
TSX0     UNLNK,$
return
```

OPERATING SYSTEM INTERACTION

The .CRQGT gate is shut when UNLNK is executed.

ROUTINE RETURNS

Return (TRA 0,0)	Normal return. All registers are intact except X1 and X3 which have been destroyed. The .CRQGT gate is still shut.

POSTCALLING SEQUENCE

None.

SUPPORTING INFORMATION

Programming Method

UNLNK is nonreentrant since the .CRQGT is shut. It is written in floatable code.

Storage

No internal temporary storage is used.

UNLNK occupies approximately 14 core storage locations.

Other Routines Used

None.

Flowchart

See CPB-1500 for the flowchart of UNLNK, .MIOS module.

CALCULATE LOGICAL PRIMARY CHANNEL INDEX

ILPCX (.MIOS) calculates the logical primary channel index when given the absolute SCT device address.

PRECALLING SEQUENCE

Prior to entering ILPCX, the X1 register must contain the data indicated.

X1 Absolute address of device secondary SCT

CALLING SEQUENCE

ILPCX is called by routines within the .MIOS module.

```
8       16
TSX0    ILPCX,$
return
```

OPERATING SYSTEM INTERACTION

The stack is not used.

No gates are used.

ROUTINE RETURNS

Return (TRA 0,0)	Registers contain:

AR	First word of device SCT
QR	Destroyed
X1	Absolute address of device SCT
X2	Logical primary channel index
All	other registers are restored

POSTCALLING SEQUENCE

None.

SUPPORTING INFORMATION

Programming Method

ILPCX is reentrant and written in floatable code.

Interrupts are not inhibited.

Storage

No internal temporary storage is used.

ILPCX occupies approximately 10 core storage locations.

Other Routines Used

None.

Nomenclature

Logical Primary Channel Index - The index to .CRCTX tables is used to access the channel entry for the device indicated by the SCT address. On cross-barred channels, the index is for the channel configured as the primary channel.

The index is of the form:

(256*IOC + 4*Channel No.)

If the SCT is for a device on a single device channel, the index is calculated as the difference between the beginning of the channel SCT tables (.CRCT1) and the device SCT (actually a channel SCT).

If the SCT is for a device on a multidevice channel, the device SCT contains a precalculated index in word 0, bits 24-35.

Flowchart

See CPB-1500 for the flowchart of ILPCX, .MIOS module.

STATUS RETURN

STRET (.MIOS) completes processing an I/O request that has terminated. The subroutine is called by IOS and by GEPR through GSTRT. STRET returns status to the program, disposes of the I/O entry, and checks program state to determine if it should be placed in the Dispatcher (.MDISP) queue for execution.

PRECALLING SEQUENCE

Prior to entering STRET, the registers listed must contain the data indicated.

AR	Status word 1
QR	Status word 2 (absolute data address in bits 0-17)
X2	Logical Primary channel index
X4	I/O entry address
X5	LAL for program (or LAL for GEPOP)
X6	Program number
X7	Processor number

The I/O entry referenced by X4 is in the In Transmission status. The entry is in the I/O queue when the subroutine is called by the Interrupt Handler (IOTRM).

The I/O entry is in the GEPR status and is unlinked when the routine is called by GSTRT or the I/O Request Handler.

CALLING SEQUENCE

STRET is called from routines located in the .MIOS module and by .MGEPR.

```
8         16
TSX0      STRET,$
return
```

OPERATING SYSTEM INTERACTION

One stack entry is used for the temporary storage of registers.

No .STEMP storage is used.

STRET uses the following gates:

.CRQGT - when unlinking I/O entries
.CRCCS - when linking I/O entries into the courtesy call queue
.CRDSP - when interrogating program state and putting a program into the Dispatcher queue.

If status returns are requested, the two status words will be stored via the status return pointer in the I/O entry.

If no courtesy call is requested, the I/O entry is released (set to Open status) and both I/O counts are reduced. The roadblock and relinquish state bits are checked to determine if the program is now eligible for execution.

If a courtesy call is requested, the I/O entry will be linked into the courtesy call queue to await execution at courtesy call level. The outstanding I/O count is decremented for that program. The program state is checked, and the relinquish state is reset. The program is put into the Dispatcher queue as being eligible for execution.

ROUTINE RETURNS

Return (TRA 0,0)	Registers X1 through X2 and X4 through X7 are restored; the AR, QR, and X3 registers are destroyed.

POSTCALLING SEQUENCE

None.

SUPPORTING INFORMATION

Programming Method

STRET is reentrant and written in floatable code.

Interrupts are inhibited while any gate is shut.

Storage

Internal temporary storage is used when gates are shut.

STRET occupies approximately 175 core storage locations.

Other Routines Used

Program Number in Queue Following Interrupt, DSPQM, (EP14 of .MDISP).

Nomenclature

See ILPCX (.MIOS) for a definition of logical primary channel index.

Flowchart

See CPB-1500 for the flowchart of STRET, .MIOS module.

START I/O

STIO (.MIOS) determines whether an I/O entry can be started on a channel, loads the mailboxes, and connects the I/O to the proper IOC.

Additional entries are made through STIOM, TYPER, and STGPC.

PRECALLING SEQUENCE

Prior to entering STIO, the registers listed must contain the data indicated.

X2	Logical primary channel index
X5	LAL for program or for GEPOP
X7	Processor number

The .CRQGT gate must be shut by a .SHUTS macro.

CALLING SEQUENCE

STIO is called from routines in the .MIOS module.

```
8          16
TSX3       STIO,$
return
```

OPERATING SYSTEM INTERACTION

The .CRQGT gate is shut when this subroutine is executed and while an I/O entry is selected. The gate is opened with a .OPENS macro within STIO.

STIO normally does not use the stack for temporary storage; however the trace routines do use the stack for temporary storage.

ROUTINE RETURNS

Return (TRA 0,3) Only X7 is intact. The .CRQGT gate is opened.

POSTCALLING SEQUENCE

None.

SUPPORTING INFORMATION

Programming Method

STIO is reentrant and written in floatable code.

Interrupts are inhibited; subroutines called by STIO are also interrupt inhibited.

Storage

Internal temporary storage is used while gates are shut.

STIO, STIOM, and TYPER occupy approximately 350 core storage locations.

Other Routines Used

Applicable Channel Module (EP3 of .MCPIO, .MDR20, .MDS20, .MGPIO, .MMTAP, .MPRIO, .MPTAP, or .MTYPE).

Calling Sequence to EP3 - Prior to entering EP3, the registers listed must contain the data indicated.

X0 Transfer register
X1 True channel index
X2 Logical primary channel index

The .CRQGT gate must be shut. EP3 must be interrupt inhibited.

EP3 is entered by the following calling sequence.

```
8          16
LDA        .CRCT4,2  }
ADLA       3,DU      }     simulated .CALL
TSX0       0,AU      }
return 0
                           transfer vector
return 5
```

Returns from EP3 - Any of 13 returns may be made, depending upon the condition specified.

Return 0 (TRA 0,0) Normal select routine. Registers contain:

X0 Transfer register
X1 True channel index
X2 Logical primary channel index
X7 Processor number

Return 1 (TRA 1,0) Start selected I/O entry. Registers contain:

X0	Transfer register
X1	True channel index
X2	Logical primary channel index
X4	I/O entry selected
X7	Processor number

Return 2 (TRA 2,0) No I/O entry to start. Register contents are the same as for return 0.

Return 3 (TRA 3,0) Issue seek command on DSU200. Registers contain:

QR	Primary mailbox image with device/channel inserted
X0	Transfer register
X1	True channel index
X2	Logical primary channel index
X4	I/O entry selected
X7	Processor number

Return 4 (TRA 4,0) Issue read/write on DSU200. Registers contain:

QR	Primary mailbox image with device/channel inserted
X0	Transfer register
X1	True channel index
X2	Logical primary channel index
X4	I/O entry selected
X7	Processor number

Return 5 (TRA 5,0) Multirecord select routine. Registers contain:

X0	Transfer register
X1	True channel index
X2	Logical primary channel index
X7	Processor number

Flowchart

See CPB-1500 for the flowchart of STIO, .MIOS module.

CONNECT MULTIRECORD SIMULATION DCW

STIOM (.MIOS) issues a connect for the next DCW for multirecord simulation.

STIOM is an entry point into the basic STIO subroutine. Only the precalling and calling sequences differ from those described for STIO and, therefore, are the only parameters described here. See STIO for the others.

PRECALLING SEQUENCE

Prior to entering STIOM, the registers listed must contain the data indicated.

QR	Primary mailbox image
X1	True channel index
X3	Transfer register
X4	I/O entry address
X6	Program number
X7	Processor number

The .CRQGT gate must be shut by a .SHUTS macro.

CALLING SEQUENCE

STIOM is called from the Interrupt Handler (IOTRM) (return 12). Entry is already selected and the secondary mailboxes loaded.

```
8       16
TSX3    STIOM,$
```

Flowchart

See CPB-1500 for the flowchart of STIOM, .MIOS module.

CONNECT REISSUE OF SECOND TYPEWRITER COMMAND

TYPER (.MIOS) connects the reissue of the second command in a typewriter read/write sequence.

TYPER is an entry point into the basic STIO subroutine. Only the precalling and calling sequences differ from those described for STIO and, therefore, are the only parameters described here. See STIO for the others.

PRECALLING SEQUENCE

Prior to entering TYPER, the registers listed must contain the data indicated.

QR	Primary mailbox image
X1	True channel index
X2	Logical channel index
X3	Transfer register
X4	I/O entry address
X6	Program number
X7	Processor number

The .CRQGT gate must be shut by a .SHUTS macro.

CALLING SEQUENCE

TYPER is called from the Interrupt Handler (IOTRM) (return 7). Entry is already selected.

```
8          16
TSX3       TYPER,$
```

Flowchart

See CPB-1500 for the flowchart of TYPER, .MIOS module.

CONNECT SELECTED GESPECED ENTRY

STGPC (.MIOS) connects a now eligible I/O entry previously GESPECed.

STGPC is an entry point into the basic STIO subroutine. Only the precalling and calling sequences differ from those described for STIO and, therefore, are the only parameters described here. See STIO for the others.

PRECALLING SEQUENCE

Prior to entering STGPC, the registers listed must contain the data indicated.

X1	True channel index
X2	Logical channel index
X3	Transfer register
X4	I/O entry address
X6	Program number

The .CRQGT gate must be shut by a .SHUTS macro.

CALLING SEQUENCE

STGPC is called from the IOTRM routine (return 2).

```
8          16
TSX3       STGPC,$
```

Flowchart

See CPB-1500 for the flowchart of STGPC, .MIOS module.

SERVICE ROUTINES USED EXTERNAL TO IOS

As shown in Figure 4, the service routines used by routines external to IOS are as follows:

RSMIO	(EP8)	Resume I/O for Program
ABTIO	(EP9)	Abort I/O for Program
GSTRT	(EP10)	Format I/O Status Words and Return Status
RSMCH	(EP12)	Resume I/O on Channel
ACTFL	(EP13)	Accounting File Request

RESUME I/O FOR PROGRAM

RSMIO (EP8 of .MIOS) resumes I/O for a program by resetting the stop flags in all the program I/O entries in the slave service area and by attempting to start I/O on any free channel.

PRECALLING SEQUENCE

Prior to entering RSMIO, the registers listed must contain the data indicated.

X5	LAL for program
X6	Program number
X7	Processor number

CALLING SEQUENCE

RSMIO routine is called from the ABTIO routine and can be called by any master mode program operating under a program number.

```
           8           16
          ________________________
          | .RSMIO    |
          | return    |
          |           |
or        | .CALL     | .MIOS,8
```

OPERATING SYSTEM INTERACTION

RSMIO does not use the stack for temporary storage; however, it may call routines that do.

The following .STEMP storage is used:

.STEMP+8 and .STEMP+9

The .CRQGT gate is shut by RSMIO if an attempt is made to start I/O on a channel.

ROUTINE RETURNS

Return is made after the call to RSMIO.

The AR, QR, and X0 through X4 registers are destroyed; X5 through X7 are restored.

POSTCALLING SEQUENCE

None.

SUPPORTING INFORMATION

Programming Method

RSMIO is reentrant and written in floatable code.

Interrupts are inhibited when the .CRQGT gate is shut.

Storage

No internal temporary storage is used.

RSMIO occupies approximately 50 core storage locations.

Other Routines Used

Start I/O, STIO (.MIOS).

Flowchart

See CPB-1500 for the flowchart of RSMIO (EP8), .MIOS module.

ABTIO(EP9)
.MIOS

ABORT I/O FOR PROGRAM

ABTIO (EP9 of .MIOS) aborts all program I/O requests for which the I/O has not been initiated with the exception of requests specifically marked as "don't abort" by the system. Prior to calling the ABTIO routine, all I/O requests for the program are stopped so there is no I/O in transmission. Aborting the I/O consists of unlinking and releasing I/O entries in linked and courtesy call status and adjusting the program request count. Entries in GEPR status are also checked and the associated System Configuration Table (SCT) stop flags are reset. After the I/O is aborted by ABTIO routine, the RSMIO (EP8 of .MIOS) routine is called.

PRECALLING SEQUENCE

Prior to entering ABTIO, the registers listed must contain the data indicated.

X5	LAL for program
X6	Program number
X7	Processor number

CALLING SEQUENCE

ABTIO is called from the .MBRT1 module.

```
          8          16
         ------------------------------
         | .ABTIO    |
         | return    |
         |           |
or       | .CALL     | .MIOS,9
         | return    |
         |           |
```

OPERATING SYSTEM INTERACTION

At the time the ABTIO routine gets control, IC and I of the call is the top entry in the stack.

The ABTIO routine does not use the stack for temporary storage but may call routines that do.

The .CRQGT gate is shut while unlinking linked status entries from an I/O queue. The .CRCCS gate is shut while unlinking courtesy call status entries from the courtesy call queue.

ROUTINE RETURNS

Return is made from the RSMIO routine to the location immediately following the ABTIO call.

Registers X5 through X7 are intact.

POSTCALLING SEQUENCE

None.

SUPPORTING INFORMATION

Programming Method

ABTIO is reentrant and written in floatable code.

Interrupts are inhibited while gates are closed.

Storage

No internal temporary storage is used.

ABTIO occupies approximately 135 core storage locations.

Other Routines Used

Unlink I/O Entry, UNLNK (.MIOS).
Calculate Logical Primary Channel Index, ILPCX (.MIOS).
Format I/O Status Words and Return Status, GSTRT (EP10 of .MIOS).
Resume I/O For Program, RSMIO (EP12 of .MIOS).

Flowchart

See CPB-1500 for the flowchart of ABTIO (EP9), .MIOS module.

FORMAT I/O STATUS WORDS AND RETURN STATUS

GSTRT (EP10 of .MIOS) formats status words for an I/O entry which is already unlinked from an I/O queue. Status is then returned to the program.

PRECALLING SEQUENCE

Prior to entering GSTRT, the registers listed must contain the data indicated.

X4	I/O entry address
X5	LAL for program
X6	Program number
X7	Processor number

```
Status word 1 in 1,4
Status word 2 (bits 0-17 in 0,4 (bits 0-17)
              (bits 18-35) in 8,4 (bits 18-35)
```

CALLING SEQUENCE

GSTRT is called from GEPRE (EP1 of .MGEPR) and ABTIO (EP9 of .MIOS).

```
8          16
.CALL      .MIOS,10
return
```

OPERATING SYSTEM INTERACTION

At the time the GSTRT routine gets control, IC and I of the call is the top entry in the stack.

The GSTRT routine uses the stack for temporary storage and calls routines which also use the stack.

GSTRT does not shut gates itself but calls routines which do.

ROUTINE RETURNS

Return is made to the location immediately following the GSTRT call.

All registers except X2 are restored.

POSTCALLING SEQUENCE

None.

SUPPORTING INFORMATION

Programming Method

GSTRT is reentrant and written in floatable code.

Interrupts are inhibited while storing and retrieving from the stack.

Storage

No internal temporary storage is used.

GSTRT occupies approximately 20 core storage locations.

Other Routines Used

Calculate Logical Primary Channel Index, ILPCX (.MIOS)
Status Return, STRET (.MIOS)

Flowchart

See CPB-1500 for the flowchart of GSTRT (EP10), .MIOS module.

RSMCH(EP12)
.MIOS

RESUME I/O ON CHANNEL

RSMCH (EP12 of .MIOS) examines the specified logical channel queue to determine if any eligible entries can be started.

The Main GEPR module (.MGEPR) calls this routine when it completes the processing (returning status or aborting the program) of an error. At the time GEPR is called, entries in the logical device (and channel) SCT are stopped until the I/O is reissued error free or until GEPR returns status and/or requests an abort of the program. Then the associated SCT stop bit is reset and the RSMCH routine is called.

PRECALLING SEQUENCE

Prior to entering RSMCH, the registers listed must contain the data indicated.

X1	Device SCT address
X5	LAL for program
X6	Program number
X7	Processor number

CALLING SEQUENCE

RSMCH is called from GEPRE (EP1 of .MGEPR) and RSMCH (EP12 of .MIOS).

```
8          16
EAX1       Device SCT address
.CALL      .MIOS,12
return 0                          transfer vector (.EXIT)
```

OPERATING SYSTEM INTERACTION

The RSMCH routine does not use the stack for temporary storage but calls routines which do.

No .STEMP cells are used.

The .CRQGT gate is shut prior to calling the STIO subroutine and opened in that subroutine.

ROUTINE RETURNS

Return 0 (.EXIT)	Normal return. Registers contain:

X5 LAL for program
X6 Program number
X7 Processor number

Registers X0 through X4 and AR and QR are destroyed.

POSTCALLING SEQUENCE

None.

SUPPORTING INFORMATION

Programming Method

RSMCH is reentrant and written in floatable code.

Interrupts are inhibited while the .CRQGT gate is closed.

Storage

No internal temporary storage is used.

RSMCH occupies approximately 10 core storage locations.

Other Routines Used

Calculate Logical Primary Channel Index, ILPCX (.MIOS)
Start I/O, STIO (.MIOS)

Flowchart

See CPB-1500 for the flowchart of RSMCH (EP12) .MIOS module.

ACCOUNTING FILE REQUEST

ACTFL (EP13 of .MIOS) accepts and processes a logical record for the accounting file. The logical record is moved from the requestor's area to a hard core, 320-word buffer. When a logical record cannot be inserted into the current buffer and the alternate buffer is free, the alternate buffer is designated as the current buffer and the logical record is moved into it. The filled buffer is written on the accounting file in GEFRC standard system format. This I/O is done using a GEPOP I/O entry. When the I/O is completed, a GEPOP hard core courtesy call takes any status-required action such as exchanging tapes.

A logical record is no larger than 318 words including the code word. The first word must be the code word with the following format.

bits 0-17	Record size not including code word
bits 18-35	Source code

Each type of logical record on the accounting file has a unique source code depending on the source of the data. This source code is used by the analyzer programs to determine the format of the logical record.

Source Code bits (18-35)	System Requestor	Data
000001	Termination	Job accounting
000002	GEOT	Job report data
000003	GEPR	Error statistics
000004	T and D	T and D statistics
000005	GEIN	GEIN job statistics
000006	System routines	System snaps
000007	Time-Sharing	Time-Sharing statistics
000008	Termination	Source code 01 overflow

ACTFL inserts the GEFRC record control word at the beginning of the logical record when moving the record into the hard core buffer. It also inserts a block serial number into the first word of the physical record when the record is written.

PRECALLING SEQUENCE

Prior to entering ACTFL, the registers listed must contain the data indicated.

X1	Absolute address of logical record
X5	LAL for program
X6	Program number
X7	Processor number

CALLING SEQUENCE

ACTFL is called from any master mode system routine. The return is to the first instruction following the .CALL.

```
8          16
EAX1       Absolute address of logical record
.CALL      .MIOS,13
return
```

OPERATING SYSTEM INTERACTION

No stack or .STEMP storage is used.

The .CRACF gate is shut while the tape writing switch is interrogated or set, and while the logical record is moved to the hard core buffer.

ROUTINE RETURNS

Return 0 (.EXIT)	Normal exit, logical record accepted. Index registers X0 through X7 are restored.

POSTCALLING SEQUENCE

No status check is required; the requestor may alter the logical record buffer within his area.

SUPPORTING INFORMATION

Programming Method

ACTFL is reentrant but certain actions within the routine are controlled by the .CRACF gate.

Interrupts are inhibited while the gate is shut.

Storage

ACTFL occupies approximately 820 core storage locations including the courtesy call routine, double buffers, and temporary storage cells.

Temporary storage is used while the gate is shut or when the tape writing flag is set.

Other Routines Used

Assign an I/O Entry, QUEUE (EP4 of .MIOS)
Link I/O to Front of QUEUE, LINKF (EP2 of .MIOS)
Relinquish Control, REL (EP4 of .MIOS)
Accounting Tape Switching, ACTS1 (EP1 of .MACTS)

Flowchart

See CPB-1500 for the flowchart ACTFL (EP13) of .MIOS module.

MAIN IOS MODULE INITIALIZATION

The .IIOS (.MIOS) routine performs Startup initialization when the .MIOS module is loaded into core. It initializes tables, the interrupt vector, and multirecord simulation. In addition, it builds a list of device-dependent channel modules to be loaded by Startup.

PRECALLING SEQUENCE

Prior to entering .IIOS, the registers listed must contain the data indicated.

X0	Address of bootstrap card reader definition
X2	Address of interrupt vector image
X3	Address of fault vector image
X4	Address of real-time IOC mailbox images

CALLING SEQUENCE

The .IIOS routine is called from Startup (.MINIT).

```
8          16
 ___________________________
|TSX1      |.IIOS
|return    |
|          |
```

OPERATING SYSTEM INTERACTION

None.

ROUTINE RETURNS

Return 0 (TRA 0,1)   Normal return. Registers contain:

AR	Bits 0-17. Address of device-dependent channel module table. Bits 18-29. Count of entries
QR	Next available load address
X0	Destroyed
X2	through X5 are destroyed.

POSTCALLING SEQUENCE

None.

SUPPORTING INFORMATION

Programming Method

The .IIOS routine is nonreentrant and written in floatable code.

Storage

The .IIOS routine occupies approximately 285 core storage locations and resides in the accounting file buffer area.

Other Routines Used

None.

Flowchart

See CPB-1500 for the flowchart of .IIOS, .MIOS module.

3. CHANNEL MODULES

Channel modules are provided to drive each type of I/O device configured in the GE-625/635. The modules handle all device-dependent operations, such as interrupts, I/O requests, error processing, and channel queue select strategy.

Channel modules reside in memory as part of the Hard Core Monitor and are part of the IOS/GEPR interface. Each channel module has the capability to handle as many channels as are configured with like devices; therefore, the need for separate modules for each channel is eliminated. The GECOS-III channel modules are as follows:

- .MCPIO - Card Punch
- .MDR20 - MDS200 Magnetic Drum Subsystem
- .MDS20 - DSU200 Disc Storage Subsystem
- .MGPIO - Card Reader
- .MMTAP - Magnetic Tape
- .MPRIO - Printer
- .MPTAP - Paper Tape
- .MTYPE - Typewriter

The format of a channel module is the same as any other GECOS-III module, thus preserving the homogeneity of the system.

The channel configuration tables, built by startup, provide a 4-word entry for each channel configured on each IOC. The fourth word in each entry has the absolute address of the channel module that drives all devices on that channel. The entry also provides indirect address modification to access a desired entry point in the channel module.

All channel modules are constructed so that the first four entries in the transfer vector of each module are transfers to parts of the module where four basic but device-oriented operations are performed. The last four entries are constants or variables:

Entry Point	Transfer to
EP1	Interrupt Handler
EP2	I/O Request Handler
EP3	Select Routine
EP4	Error Processing Routine
EP5	Constant. Amount of time to be exceeded before interrupt is declared lost.
EP6	Constant. Initialized to zero by the initialization routine if a multirecord command is allowed.
EP7	Constant. BCD device name in bits 0-17, bits 18-34 not used If bit 35 is 1, device command is typed in GEPR message
EP8	Variable. Accumulated IOC time for device type

In addition to the standard entry points, all mass storage device channel modules contain negative entry points used to provide storage for pertinent I/O information as well as transfer vectors to an appropriate seek-address calculation routine.

Transfer to a channel module entry point is accomplished through the use of Register Indirect address modification. An example can be used to demonstrate the use of the primary SCT entries and address modification registers. Upon encountering the instruction:

```
 8          16
|TSX1       |.CRCT4,2*
```

the AU register is set to indicate the desired entry point into a channel module,

```
          0                                 17
AU       [                                   3]
                  (set to select EP3)
```

and X2 is used to contain the logical channel index (LGX=20). Index register 2 modification is then used to access the primary SCT.

```
          0                                 17
X2       [                                  20]
                  (logical channel index
                  from .CRIO4, bits 24-35,
                  or from secondary SCT,
                  word 1, bits 24-35)
```

```
 8          16
|TSX1       |.CRCT4+20,*
```

Since the primary SCT entry contains the ..name with AU modification,

```
 8          16
|TSX1       |..name*
```

the indirect modification accomplishes the following:

```
       8          16
      |TSX1       |..name,AU
or    |TSX1       |..name+3
```

The module identification word (..name) precedes the entry vector in the channel module:

```
..name
      +1   transfer to EP1
      +2   transfer to EP2
      +3   transfer to EP3
       .
       .
       .
      +n   transfer to EPn
```

CARD PUNCH CHANNEL MODULE (.MCPIO)

The Card Punch Channel Module (.MCPIO) resides in the Hard Core Monitor. The routines given below comprise the .MCPIO module:

CPIT (EP1)	Card Punch Interrupt Handler
CPIO (EP2)	Card Punch Request
CPSL (EP3)	Card Punch Select
CPGP (EP4)	Card Punch Error
.ICPIO	Card Punch Initialization

These are described on the following pages.

CARD PUNCH INTERRUPT HANDLER

CPIT (EP1 of .MCPIO) interrogates interrupt status for channels which have a card punch configured.

If the IOC configured is a model B, multirecord simulation is initialized for all commands. If a CPZ100 card punch is configured, multirecord simulation is initialized for the CPZ100 only. The coding for simulation follows the hard core code in the .MCPIO module; it remains in hard core only if simulation is required.

PRECALLING SEQUENCE

Prior to entering CPIT, the registers listed must contain the data indicated:

QR	Status word 1
X0	Type of interrupt (TI=0, II=2, SI=4)
X1	True channel index
X2	Logical channel index
X3	Bit 0=0, bit 1=power bit, bits 2-5=major status, bits 6-17=0 (for TI and II only)
X4	I/O entry address (for TI and II only)
X5	LAL for GEPOP
X6	Program number of I/O entry (for TI and II only)
X7	Processor number

CALLING SEQUENCE

CPIT is called from the Interrupt Handler (IOTRM) located in the .MIOS module.

```
8           16

LDA         1,DU       }
STC1        .SSTAK,5*  }    simulated .CALL
TRA         .CRCT4,2*  }
return 0
   .                        transfer vector
   .                        (.EXIT through .EXIT 13)
   .
return 13
```

OPERATING SYSTEM INTERACTION

The top entry in the stack is the IC and I of the call for the return.

No .STEMP storage is used.

No gates are used.

NORMAL ROUTINE RETURNS

Return 0 (.EXIT)	Status return action. Registers contain: X1 True channel index X2 Logical primary channel index X4 I/O entry address X5 LAL for GEPOP X6 Program number X7 Processor number
Return 1 (.EXIT 1)	GEPR action. Register contents are the same as for return 0.
Return 2 (.EXIT 2)	GESPEC action. Registers contain: X1 True channel index X2 Logical primary channel index X5 LAL for GEPOP X7 Processor number
Return 8 (.EXIT 8)	Error threshold/exchange action. Register contents are the same as for return 0.

MULTIRECORD SIMULATION RETURNS

Return 9 (.EXIT 9)	GEPR action for multirecord simulation. Registers contain: QR Status word 1 X1 True channel index X2 Logical primary channel index X4 I/O entry address X5 LAL for GEPOP X6 Program number X7 Processor number
Return 10 (.EXIT 10)	Error threshold/exchange for multirecord simulation. Register contents are the same as for return 9.
Return 11 (.EXIT 11)	Status return action for multirecord simulation. Register contents are the same as for return 9.
Return 12 (.EXIT 12)	Issue next connect for multirecord simulation. Register contents are the same as for return 9.
Return 13 (.EXIT 13)	Issue next connect for CPZ100 multirecord simulation. Register contents are the same as for return 9.

POSTCALLING SEQUENCE

None.

SUPPORTING INFORMATION

Programming Method

CPIT is nonreentrant since IOTRM uses a software gate to ensure the processing of one interrupt at a time. The routine is written in floatable code.

Interrupts are not inhibited.

Storage

Internal temporary storage is used in the simulation code.

CPIT normally occupies approximately 40 core storage locations. Multirecord simulation for the CPZ100 card punch requires 45 additional locations; another 20 are required for IOCB multirecord simulation.

Other Routines Used

Card Punch Initialization, .ICPIO (.MCPIO)

Flowcharts

See CPB-1500 for the flowchart of CPIT (EP1), .MCPIO module.

CARD PUNCH REQUEST

CPIO (EP2 of .MCPIO) validates the I/O command in a GEINOS select sequence and determines the proper return to complete the I/O entry.

PRECALLING SEQUENCE

Prior to entering CPIO, the registers listed must contain the data indicated.

X1	Device SCT address
X2	Logical primary channel index
X3	Select sequence address
X4	I/O entry address
X5	LAL for program
X6	Program number
X7	Processor number

If the request is from the slave area, .STEMP,5 is zero; it is nonzero if the request is from the slave service area.

The upper address limit of the program is in bits 0-17 of .STEMP+1,5, and the offset to the PAT pointer word is in bits 18-35.

CALLING SEQUENCE

CPIO is called from the INOS routine located in the .MIOS module.

```
8            16
LDA          |2,DU       \
STC1         |.SSTAK,5*   }   simulated .CALL
TRA          |.CRCT4,2*  /
return 0     |
  :          |                transfer vector
  .          |                (.EXIT through .EXIT 6)
return 6     |
```

OPERATING SYSTEM INTERACTION

The top entry in the stack is the IC and I of the call for the return.

No .STEMP storage is used.

No gates are used.

ROUTINE RETURNS

Return 0 (.EXIT) Normal data transfer, 3 word select sequence. .STEMP,5 is zero if the request is from the slave area; it is nonzero if the request is from the slave service area.

The upper address limit of the program is in bits 0-17 of .STEMP+1,5 and the offset to the PAT pointer word is in bits 18-35.

Registers contain:

AR	I/O command
X3	Select sequence address
X4	I/O entry address
X5	LAL for program
X6	Program number
X7	Processor number

Return 2 (.EXIT 2) Force GEPR override. The contents of .STEMP, .STEMP+1, AR, and index registers are the same as for return 0.

Return 6 (.EXIT 6) Illegal I/O command. The contents of .STEMP, .STEMP+1, and index registers are the same as for return 0.

POSTCALLING SEQUENCE

None.

SUPPORTING INFORMATION

Programming Method

CPIO is reentrant and written in floatable code.

Interrupts are not inhibited.

Storage

No internal temporary storage is used.

CPIO occupies approximately 45 core storage locations.

Other Routines Used

None.

Flowchart

See CPB-1500 for the flowchart of CPIO (EP2), .MCPIO module.

CARD PUNCH SELECT

CPSL (EP3 of .MCPIO) transfers to the multirecord select routine in STIO which selects an entry from the queue and starts the input/output.

PRECALLING SEQUENCE

Prior to entering CPSL, the registers listed must contain the data indicated.

X0 Transfer register
X1 True channel index
X2 Logical primary channel index
X7 Processor number

CALLING SEQUENCE

CPSL is called from the STIO routine located in the .MIOS module.

```
8            16
-------------------------------------------
LDA         | .CRCT4,2
ADLA        | 3.DU
TSX0        | 0,AU
return 0    |
            |              transfer vector
            |            (.EXIT through .EXIT 6)
return 6    |
            |
```

OPERATING SYSTEM INTERACTION

No stack or .STEMP storage is used.

The .CRQGT gate is shut while the routine is executed.

ROUTINE RETURNS

Return 5 (TRA 5,0) Multirecord select routine. Registers contain:

X0 Transfer register
X1 True channel index
X2 Logical primary channel index
X7 Processor number

POSTCALLING SEQUENCE

None.

SUPPORTING INFORMATION

Programming Method

CPSL is nonreentrant and written in floatable code.

Interrupts are inhibited.

Storage

No internal temporary storage is used.

CPSL consists of a single transfer instruction.

Other Routines Used

None.

CARD PUNCH ERROR

CPGP (EP4 of .MCPIO) begins processing an error status received from a channel with a card punch configured.

PRECALLING SEQUENCE

Prior to entering CPGP, the registers listed must contain the data indicated.

X2	Logical primary channel index
X3	Absolute I/O entry address
X5	LAL for program
X6	Program number
X7	Processor number

CALLING SEQUENCE

CPGP is called from the .MGEPR module.

```
8         16
LDA       |4,DU      \
STC1      |.SSTAK,5*  }   simulated .CALL
TRA       |.CRCT4,2* /
return 0  |
return 1  |               transfer vector
return 2  |               (.EXIT through .EXIT 2)
```

OPERATING SYSTEM INTERACTION

The top entry in the stack is the IC and I of the call in .MGEPR.

No .STEMP storage is used.

No gates are used.

ROUTINE RETURNS

The only exit from CPGP is by calling the exception processing SSA card punch module to continue processing the error. The returns associated with the simulated .CALL reside in the .MGEPR module and are available to GP15.

```
.GOTO     .MGP15,1
```

Register contents are the same as for the call.

POSTCALLING SEQUENCE

None.

SUPPORTING INFORMATION

Programming Method

CPGP is reentrant and written in floatable code.

Interrupts are inhibited.

Storage

No internal temporary storage is used.

CPGP consists of a single .GOTO command.

Other Routines Used

Card Punch Recovery, GP15 (.MGP15)

CARD PUNCH INITIALIZATION

Startup gives control to .ICPIO (.MCPIO) when the .MCPIO channel module is loaded into hard core. The configuration tables (.CRCT1) are examined and the module address (.CRCT4) is initialized for those channels with a card punch configured.

If a CPZ100 card punch is configured, .ICPIO initializes the Card Punch Interrupt Handler (EP1) to perform multirecord simulation. If the IOC configured is a model B, multirecord simulation is initialized for all punches.

PRECALLING SEQUENCE

Prior to entering .ICPIO, the registers listed must contain the data indicated.

X0	Address of bootstrap card reader description
X2	Address of interrupt vector image
X3	Address of fault vector image
X4	Address of real-time IOC mailbox images

CALLING SEQUENCE

This routine is called from Startup (.MINIT). The return is to the first instruction following the TSX.

```
8        16
TSX1     .ICPIO
return
```

OPERATING SYSTEM INTERACTION

None.

ROUTINE RETURNS

Return (TRA 0,1) Registers contain:

AR	Zero (no required modules)
QU	Next available load address
X0	thru X5 are destroyed

POSTCALLING SEQUENCE

None.

SUPPORTING INFORMATION

Programming Method

The .ICPIO routine is nonreentrant and written in floatable code.

Interrupts are not inhibited.

Storage

The initialization routine occupies approximately 60 core storage locations; however, it is present only at startup time. The next available load address in QU allows Startup to overlay .ICPIO. No internal temporary storage is used.

Other Routines Used

None.

Flowchart

See CPB-1500 for the flowchart of .ICPIO, .MCPIO module.

MAGNETIC DRUM SUBSYSTEM CHANNEL MODULE (.MDR20)

The Magnetic Drum Subsystem Channel Module (.MDR20) resides in the Hard Core Monitor. The routines given below comprise the .MDR20 module:

- DRIT (EP1) MDS200 Interrupt Handler
- DRIO (EP2) MDS200 Request
- DRSL (EP3) MDS200 Select
- DRGP (EP4) MDS200 Error and EOF Recovery
- Various Negative Entry Points
- .ICPIO MDS200 Initialization

These are described on the following pages.

MDS200 INTERRUPT HANDLER

DRIT (EP1 of .MDR20) interrogates interrupt status for channels which have a Magnetic Drum Subsystem (MDS200) configured.

PRECALLING SEQUENCE

Prior to entering DRIT, registers listed must contain the data indicated.

QR	Status word 1
X0	Type of interrupt (TI=0, II=2, SI=4)
X1	True channel index
X2	Logical channel index
X3	Bit 0=0, bit 1=power bit, bits 2-5=major status, bits 6-17=0 (for TI and II only)
X4	I/O entry address (for TI and II only)
X5	LAL for GEPOP
X6	Program number of I/O entry (for TI and II only)
X7	Processor number

CALLING SEQUENCE

DRIT is called from the Interrupt Handler (IOTRM) located in the .MIOS module.

```
8          16
LDA       | 1,DU      }
STC1      | .SSTAK,5* } simulated .CALL
TRA       | .CRCT4,2* }
return 0  |
   .      |              transfer vector
   .      |              (.EXIT through .EXIT 13)
   .      |
return 13 |
```

OPERATING SYSTEM INTERACTION

The top entry in the stack is the IC and I of the call for the return.

No .STEMP storage is used.

No gates are used.

ROUTINE RETURNS

Return 0 (.EXIT) Status return action. Registers contain:

X1 True channel index
X2 Logical primary channel index
X4 I/O entry address
X5 LAL for GEPOP
X6 Program number
X7 Processor number

Return 1 (.EXIT 1) GEPR action. Register contents are the same as for return 0.

Return 3 (.EXIT 3) No action on Special Interrupt (SI). Registers contain:

X5 LAL for GEPOP
X7 Processor number

Return 5 (.EXIT 5) Special interrupt, start I/O. Registers contain:

X2 Logical primary channel index
X5 LAL for GEPOP
X7 Processor number

Return 8 (.EXIT 8) Error threshold action. Register contents are the same as for return 0.

POSTCALLING SEQUENCE

None.

SUPPORTING INFORMATION

Programming Method

DRIT is nonreentrant since IOTRM uses a software gate to ensure the processing of one interrupt at a time. The routine is written in floatable code.

Interrupts are not inhibited.

Storage

No internal temporary storage is used.

DRIT occupies approximately 65 core storage locations.

Other Routines Used

None.

Flowchart

See CPB-1500 for the flowchart of DRIT (EP1), .MDR20 module.

MDS200 REQUEST

DRIO (EP2 of .MDR20) validates the I/O command in a GEINOS select sequence and determines the proper return to complete the I/O entry.

PRECALLING SEQUENCE

Prior to entering DRIO, the registers listed must contain the data indicated.

X1	Device SCT address
X2	Logical primary channel index
X3	Select sequence address
X4	I/O entry address
X5	LAL for program
X6	Program number
X7	Processor number

If the request is from the slave area, .STEMP,5 is zero; it is nonzero if the request is from the slave service area.

The upper address limit of the program is in bits 0-17 of .STEMP+1,5 and the offset to the PAT pointer word is in bits 18-35.

CALLING SEQUENCE

DRIO is called from the INOS routine located in the .MIOS module.

```
8              16
-------------------------------------------------------
|LDA           | 2,DU        )
|STC1          | .SSTAK,5*   }    simulated .CALL
|TRA           | .CRCT4,2*   )
|return 0      |
|    .         |                  transfer vector
|    .         |                  (.EXIT through .EXIT 6)
|return 6      |
```

OPERATING SYSTEM INTERACTION

The top entry in the stack is the IC and I of the call for the return.

No .STEMP storage is used.

No gates are used.

ROUTINE RETURNS

Return 1 (.EXIT 1)	Disc/drum nondata transfer. .STEMP,5 is zero if the request is from the slave area; it is nonzero if the request is from the slave service area. The upper address limit of the program is in bits 0-17 of .STEMP+1,5 and the offset to the PAT pointer word is in bits 18-35. Registers contain: AR Command flag, REW=0, FSR=+n, BSR=-n where n is the number of records to be forwardspaced (FSR) or backspaced (BSR) X3 Select sequence address X4 I/O entry address X5 LAL for program X6 Program number X7 Processor number
Return 2 (.EXIT 2)	Force GEPR override. The contents of .STEMP, .STEMP+1, and index registers are the same as for return 1. AR I/O command
Return 5 (.EXIT 5)	Two disc/drum data transfer commands, first I/O command is stored in the I/O entry. The contents of .STEMP, .STEMP+1, and index registers are the same as for return 1. AR Second I/O command
Return 6 (.EXIT 6)	Illegal I/O command. The contents of .STEMP, .STEMP+1, and index registers are the same as for return 1.

POSTCALLING SEQUENCE

None.

SUPPORTING INFORMATION

Programming Method

DRIO is reentrant and written in floatable code.

Interrupts are not inhibited.

Storage

No internal temporary storage is used.

DRIO occupies approximately 65 core storage locations.

Other Routines Used

None.

Flowchart

See CPB-1500 for the flowchart of DRIO (EP2), .MDR20 module.

DRSL(EP3)
.MDR20

MDS200 SELECT

DRSL (EP3 of .MDR20) examines an MDS200 channel queue and selects the next I/O entry to start.

If the channel is not busy, the last known, or zero, drum angular position is used to select an I/O entry from the first 12 entries. A factor of 4 is added to the known drum angle and the I/O entry selected has the closest seek angular position equal to or greater than the known angle. As an entry in the queue is passed over, its internal bypass count is checked and the entry at threshold is selected regardless of the current angular position.

PRECALLING SEQUENCE

Prior to entering DRSL, the registers listed must contain the data indicated.

X0	Transfer register
X1	True channel index
X2	Logical primary channel index
X7	Processor number

CALLING SEQUENCE

DRSL is called from the STIO routine located in the .MIOS module.

```
8           16
--------------------------------------------------------
LDA        |.CRCT4,2
ADLA       |3,DU
TSX0       |0,AU
return 0   |
   .       |                  transfer vector
   .       |                  (.EXIT through .EXIT 6)
   .       |
return 6   |
           |
```

OPERATING SYSTEM INTERACTION

No stack or .STEMP storage is used.

The .CRQGT gate is shut while the routine is executed.

ROUTINE RETURNS

Return 1 (TRA 1,0)	Start selected I/O entry. Registers contain:

X0	Transfer register
X1	True channel index
X2	Logical primary channel index
X4	I/O entry selected
X7	Processor number

Return 2 (TRA 2,0)	No I/O entry to start. Registers contain:

X0	Transfer register
X1	True channel index
X2	Logical primary channel index
X7	Processor number

POSTCALLING SEQUENCE

None.

SUPPORTING INFORMATION

Programming Method

DRSL is nonreentrant and written in floatable code.

Interrupts are inhibited.

Storage

Internal temporary storage is used while the .CRQGT gate is shut.

DRSL occupies approximately 75 core storage locations.

Other Routines Used

None.

Flowchart

See CPB-1500 for the flowchart of DRSL (EP3), .MDR20 module.

MDS200 ERROR AND EOF RECOVERY

DRGP (EP4 of .MDR20) begins processing an error status received from a channel with a magnetic drum subsystem (MDS200) configured. End of file status recovery is performed in this routine to make file segmenting invisible to the I/O requestor.

PRECALLING SEQUENCE

Prior to entering DRGP, the registers listed must contain the data indicated.

X2	Logical primary channel index
X3	Absolute I/O entry address
X5	LAL for program
X6	Program number
X7	Processor number

CALLING SEQUENCE

DRGP is called from the .MGEPR module.

```
8          16
LDA       | 4,DU        )
STC1      | .SSTAK,5*   }   simulated .CALL
TRA       | .CRCT4,2*   )
return 0  |
return 1  |                 transfer vector
return 2  |                 (.EXIT through .EXIT 2)
          |
```

OPERATING SYSTEM INTERACTION

The top entry in the stack is the IC and I of the call for the return. The stack is used for temporary storage but it is restored to its original condition before exiting.

No .STEMP storage is used.

No gates are used.

ROUTINE RETURNS

The following exits are from the EOF routine back to the .MGEPR module:

Return 0 (.EXIT)	Processing complete, continue searching I/O entry area of interrupted program.
Return 2 (.EXIT 2)	Return status to the program and then continue searching the I/O entry area of interrupted program.

Non-EOF error processing is continued by calling the first drum exception processing SSA module:

```
          .GOTO     .MGP20,1
```

Register contents for all exits are the same as for the call except that X2 is destroyed.

SUPPORTING INFORMATION

Programming Method

DRGP is reentrant and written in floatable code.

Interrupts are not inhibited.

Storage

No internal temporary storage is used.

DRGP occupies approximately 150 core storage locations.

Other Routines Used

MDS200 Recovery 1, GP20 (.MGP20)
Terminate GEPR Abort, ABREQ (EP5 of .MBRT1)
MDS200 Interrupt Handler DRIO (EP1 of .MDR20)
MDS200 Negative Entry Points (EP -1 of .MDR20)

Flowchart

See CPB-1500 for the flowchart of DRGP (EP4), .MDR20 module.

MDS200 NEGATIVE ENTRY POINTS

The following negative entry points provide device-dependent constants and seek calculation routines for the Magnetic Drum Subsystem (MDS200).

Negative EP Number	
-15	(Constant) Write and Verify command
-14	(Constant) Bypass count for select strategy
-13	(Constant) Fractional portion of new file request to reserve for later file expansion
-12	(Constant) Seek command
-11	(Constant) Read command
-10	(Constant) Write command
- 9	(Constant) Number of blocks between fences
- 8	(Constant) Number of blocks per LLINK
- 7	(Constant) Number of blocks per LINK
- 6	(Constant) Number of words per block
- 5	LLINK Seek Calculation routine with absolute block number input
- 4	LINK Seek Calculation routine with absolute block number input
- 3	Diagnostic Block Seek Calculation routine
- 2	LLINK Seek Calculation routine with PAT input
- 1	LINK Seek Calculation routine with PAT input

PRECALLING SEQUENCE

There is no precalling sequence for constants. For the seek calculation routines, negative EP -3, -4, and -5, the registers listed must contain the data indicated.

QR	Absolute block number, right justified
X1	Transfer register to routine
X3	Absolute address of word containing secondary SCT address in bits 0-17

EP -1 and EP -2

If the file is random, the QR contains the relative block number, right justified. If the file is linked, the QR contains either:

0 for the next 320 word record
or -n for backspace n records
or +n for forward space n records

X1	Transfer register to routine
X3	Absolute address of word containing secondary SCT address in bits 0-17

CALLING SEQUENCE

All negative point entries are called from master mode.

EP -6 through EP -15

The calling sequence is the same for EP -6 through EP -15 (constants) except that p2 is the applicable negative entry point number.

```
8          16
EAXn       p1           Absolute address of word containing secondary
                        SCT address in bits 0-17
LDXn       0,n          Secondary SCT address
LXLn       0,n        }
ANXn       =07777,DU  } Calculate logical channel index
EAA        p2           Negative entry point number
LDA        .CRCT4,n*    Retrieving EP transfer word constant
```

EP -1 through EP -5

The calling sequence for EP -1 through EP -5 is the same except that p2 is the applicable negative entry point number.

```
EAX3       p1           Absolute address of word containing secondary
                        SCT address in bits 0-17 (PAT for -1
                        through -3)
LDXn       0,3          Secondary SCT address
LXLn       0,1        }
ANXn       =07777,DU  } Calculate logical channel index
EAA        p2           Negative entry point number
TSX1       .CRCT4,n*    Transfer to seek calculation routine
return 0
return 1
```

EP -1 and EP -2

If the PAT is for a linked file and the seek address calculation is successful, bits 18-35 of the fourth word within the PAT (the current relative block number within the memory description) is updated to reflect the completion of the current request.

ROUTINE RETURNS

Negative EP -1, -2, -4 -5

Return 0 (TRA 0,1)	Error return. Register status:

```
AR  Destroyed
QR  Bit 0=1 error (always ON if 1, 2, or 3 is ON)
    Bit 1=1 tried to position backward too far
    Bit 2=1 tried to position forward too far
    Bit 3=1 nonexistent LLINK or LINK number
X0  Destroyed
All other registers are restored.
```

Return 1 (TRA 1,1)	Normal return. Register status:

```
AR  Record count residue for linked file
QR  Seek address
X0  Destroyed
All other registers are restored.
```

Negative EP -3

Return 0 (TRA 0,1)	Error return. Register status:

```
AR  Destroyed
QR  Bit 0=1 error
    Bit 3=1 nonexistent block number
X0  Destroyed
All other registers are restored.
```

Return 1 (TRA 1,1)	Normal return. Register status:

```
AR  Destroyed
QR  Seek address
X0  Destroyed
All other registers are restored.
```

OPERATING SYSTEM INTERACTION

The stack is used for the IC and I of the call and for temporary register storage.

No .STEMP storage is used.

A procedure gate is shut upon entry and opened just prior to returning to the caller.

POSTCALLING SEQUENCE

On an error return, the Q-register is tested for error identification. On a normal return, the Q-register (seek address) is stored in the primary mailbox image of the I/O entry (word 4) and the seek command flag in word 2 is turned on.

SUPPORTING INFORMATION

Programming Method

The routine is reentrant but a procedure gate is shut after registers are stored prior to calculating the seek address. It is written in floatable code.

Interrupts are inhibited while the gate is shut.

Storage

Internal temporary storage is used while the gate is shut.

The routine occupies approximately 250 core storage locations.

Other Routines Used

None.

Flowcharts

See CPB-1500 for the flowchart of the MDS200 negative entry points, .MDR20 module.

NOMENCLATURE

There are 768 read/write heads, 384 bands around the surface and 2 tracks per band, giving 768 total tracks. There are 32 addressable blocks per band and 64 words per addressable block.

Sizes

A LINK is 3840 words (60 blocks). An LLINK is 320 words (5 blocks) and a Diagnostic Block is 64 words (1 block).

Distribution

The first 765 addressable blocks are reserved for LLINKs. This includes the first 23 bands and part of the 24th band.

23*32+29 = 765 Blocks = 153 LLINKs

The remaining three blocks on the 24th band are reserved for Diagnostic blocks.

3 blocks = 3 Diagnostic Blocks

The remaining 360 bands are reserved for LINKs.

360*32 = 11520 blocks = 192 LINKs

SEEK ADDRESSES

BAND	BLOCK
XXXXXXXXX	XXXXX

INT (A,B) means the truncated integer value of A divided by B.

MOD (A,B) means A modulo B.

Diagnostic Block

There are only three Diagnostic Blocks beginning at block number 765.

Seek Address (Diagnostic Block)

= Block number +765

LLINKs

Each LLINK contains five blocks. As the LLINK number increases, the first 765 blocks on the drum surface are traversed, five blocks per LLINK.

Seek Address (LLINK+block number within LLINK)

= LLINK number *5+block number within LLINK

LINKs

Each LINK contains 60 blocks. As the LINK number increases, the drum is traversed beginning at Block 768, 60 blocks per LINK.

Seek Address (LINK+block number within LINK)

= LINK number *60+768+block number within LINK

BLOCK COUNT

When the seek address is received by the IOC, it is in the following form.

```
0                       18                      35
XXX XXX XXX XXX 000 000 000 000 XXX XXX XXX XXX

        BLOCK COUNT          SEEK ADDRESS
```

The block count is calculated as the number of 64 word blocks the I/O request is allowed to transmit.

MDS200 INITIALIZATION

Startup gives control to .IDR20 (.MDR20) when the .MDR20 channel module is loaded into hard core. The configuration tables (.CRCT1) are examined and the module address (.CCRT4) is initialized for those channels with an MDS200 configured.

Also appropriate commands are initialized for the type of IOC configured.

PRECALLING SEQUENCE

Prior to entering .IDR20, the registers listed must contain the data indicated.

X0	Address of bootstrap card reader description
X2	Address of interrupt vector image
X3	Address of fault vector image
X4	Address of real-time IOC mailbox images

CALLING SEQUENCE

This routine is called from Startup (.MINIT). The return is to the first instruction following the TSX.

```
8          16
--------------------------
|TSX1      |.IDR20
|return    |
|          |
```

OPERATING SYSTEM INTERACTION

None.

ROUTINE RETURNS

Return (TRA 0,1) Registers contain:

AR	Zero (no required modules)
QU	Next available load address
X2	through X5 are destroyed

POSTCALLING SEQUENCE

None.

SUPPORTING INFORMATION

Programming Method

The .IDR20 routine is nonreentrant and written in floatable code.

Interrupts are not inhibited.

Storage

This initialization routine occupies approximately 40 core storage locations, however, it is present only at startup time. The next available load address in QU allows Startup to overlay .IDR20.

No internal temporary storage is used.

Other Routines Used

None.

Flowchart

See CPB-1500 for the flowchart of .IDR20, .MDR20 module.

DISC STORAGE SUBSYSTEM CHANNEL MODULE (.MDS20)

The Disc Storage Subsystem Channel Module (.MDS20) resides in the Hard Core Monitor. The routines given below comprise the .MDS20 module:

DSIT (EP1)	DSU200 Interrupt Handler
DSIO (EP2)	DSU200 Request
DSSL (EP3)	DSU200 Select
DSGP (EP4)	DSU200 Error and EOF Recovery
Various Negative Entry Points	
.IDS20	DSU200 Initialization

These are described on the following pages.

DSIT(EP7)
.MDS20

DSU200 INTERRUPT HANDLER

DSIT (EP1 of .MDS20) interrogates interrupt status for channels which have a Disc Storage Subsystem (DSU200) configured.

PRECALLING SEQUENCE

Prior to entering DSIT, the registers listed must contain the data indicated.

QR	Status word 1
X0	Type of interrupt (TI=0, II=2, SI=4)
X1	True channel index
X2	Logical channel index
X3	Bit 0=0, bit 1=power bit, bits 2-5=major status, bits 6-17=0 (for TI and II only)
X4	I/O entry address (for TI and II only)
X5	LAL for GEPOP
X6	Program number of I/O entry (for TI and II only)
X7	Processor number

CALLING SEQUENCE

DSIT is called from the Interrupt Handler (IOTRM) located in the .MIOS module.

```
8            16
LDA          1,DU       }
STC1         .SSTAK,5*  }  simulated .CALL
TRA          .CRCT4,2*  }
return 0
   .                       transfer vector
   .                       (.EXIT through .EXIT 13)
   .
return 13
```

OPERATING SYSTEM INTERACTION

The top entry in the stack is the IC and I of the call for the return.

No .STEMP storage is used.

The .CRQGT gate is shut while the status of the completed seek I/O entry is changed.

ROUTINE RETURNS

Return 0 (.EXIT) Status return action. Registers contain:

X1	True channel index
X2	Logical primary channel index
X4	I/O entry address
X5	LAL for GEPOP
X6	Program number
X7	Processor number

Return 1 (.EXIT 1) GEPR action. Register contents are the same as for return 0.

Return 3 (.EXIT 3) No action on Special Interrupt (SI). Registers contain:

X5	LAL for GEPOP
X7	Processor number

Return 4 (.EXIT 4) Release channel, no I/O entry to release. Registers contain:

X1	True channel index
X2	Logical primary channel index
X4	I/O entry address
X5	LAL for GEPOP
X7	Processor number

Return 6 (.EXIT 6) Error threshold on DSU200 seek. Register contents are the same as for return 0.

Return 8 (.EXIT 8) Error threshold action. Register contents are the same as for return 0.

POSTCALLING SEQUENCE

None.

SUPPORTING INFORMATION

Programming Method

DSIT is nonreentrant since IOTRM uses a software gate to ensure the processing of one interrupt at a time. The routine is written in floatable code.

Interrupts are inhibited only when the .CRQGT gate is shut.

Storage

No internal temporary storage is used.

DSIT occupies approximately 90 core storage locations.

Other Routines Used

None.

Flowchart

See CPB-1500 for the flowchart of DSIT (EP1), .MDS20 module.

DSU200 REQUEST

DSIO (EP2 of .MDS20) validates the I/O command in a GEINOS select sequence and determines the proper return to complete the I/O entry.

PRECALLING SEQUENCE

Prior to entering DSIO, the registers listed must contain the data indicated.

X1	Device SCT address
X2	Logical primary channel index
X3	Select sequence address
X4	I/O entry address
X5	LAL for program
X6	Program number
X7	Processor number

If the request is from the slave area, .STEMP,5 is zero; it is nonzero if the request is from the slave service area.

The upper address limit of the program is in bits 0-17 of .STEMP+1,5 and the offset to the PAT pointer word is in bits 18-35.

CALLING SEQUENCE

DSIO is called from the INOS routine located in the .MIOS module.

```
8           16
LDA        |2,DU      \
STC1       |.SSTAK,5*  }   simulated .CALL
TRA        |.CRCT4,2* /
return 0   |
   .       |               transfer vector
   .       |               (.EXIT through .EXIT 6)
   .       |
return 6   |
```

OPERATING SYSTEM INTERACTION

The top entry in the stack is the IC and I of the call for the return.

No .STEMP storage is used.

No gates are used.

ROUTINE RETURNS

Return 1 (.EXIT 1) Disc/drum nondata transfer. .STEMP,5 is zero if the request is from the slave area; it is nonzero if the request is from the slave service area.

The upper address limit of the program is in bits 0-17 of .STEMP+1,5, and the offset to the PAT pointer word is in bits 18-35.

Registers contain:

AR	Command flag, REW=0, FSR=+n, BSR=-n where n is the number of records to be forward-spaced (FSR) or backspaced (BSR).
X3	Select sequence address
X4	I/O entry address
X5	LAL for program
X6	Program number
X7	Processor number

Return 2 (.EXIT 2) Force GEPR override. The contents of .STEMP, .STEMP+1, and index registers are the same as for return 1.

AR I/O command

Return 5 (.EXIT 5) Two disc/drum data transfer commands, first I/O command is stored in the I/O entry. The contents of .STEMP, .STEMP+1, and index registers are the same as for return 1.

AR Second I/O command

Return 6 (.EXIT 6) Illegal I/O command. The contents of .STEMP, .STEMP+1, and index registers are the same as for return 1.

POSTCALLING SEQUENCE

None.

SUPPORTING INFORMATION

Programming Method

DSIO is reentrant and written in floatable code.

Interrupts are not inhibited.

Storage

No internal temporary storage is used.

DSIO occupies approximately 60 core storage locations.

Other Routines Used

None.

Flowchart

See CPB-1500 for the flowchart of DSIO (EP2), .MDS20 module.

DSU200 SELECT

DSSL (EP3 of .MDS20) examines a DSU200 channel queue and selects the next I/O entry to start.

If the channel is not busy and an I/O entry is found that does not have a seek command started, a return is made via .EXIT 3 to start the seek command. If all seek commands are started, DSSL looks for an entry with the first seek command completed. When it finds this entry, a return is made via .EXIT 4 to start the second command.

PRECALLING SEQUENCE

Prior to entering DSSL, the registers listed must contain the data indicated.

X0	Transfer register
X1	True channel index
X2	Logical primary channel index
X7	Processor number

CALLING SEQUENCE

DSSL is called from the STIO routine located in the .MIOS module.

```
8          16
LDA        .CRCT4,2
ADIA       3,DU
TSX0       0,AU
return 0                       transfer vector
   .                           (.EXIT through .EXIT 6)
   .
return 6
```

OPERATING SYSTEM INTERACTION

No stack or .STEMP storage is used.

The .CRQGT gate is shut while the routine is executed.

ROUTINE RETURNS

Return 1 (TRA 1,0)   Start selected I/O entry. Registers contain:

X0	Transfer register
X1	True channel index
X2	Logical primary channel index
X4	I/O entry selected
X7	Processor number

Return 2 (TRA 2,0)	No I/O entry to start. Registers contain: X0 Transfer register X1 True channel index X2 Logical primary channel index X7 Processor number
Return 3 (TRA 3,0)	Issue seek command on DSU200. Registers contain: QR Primary mailbox image with device/channel inserted X0 Transfer register X1 True channel index X2 Logical primary channel index X4 I/O entry selected X7 Processor number
Return 4 (TRA 4,0)	Issue read/write on DSU200. Register contents are the same as for return 3.

POSTCALLING SEQUENCE

None.

SUPPORTING INFORMATION

Programming Method

DSSL is nonreentrant and written in floatable code.

Interrupts are inhibited.

Storage

Internal temporary storage is used while the .CRQGT gate is shut.

DSSL occupies approximately 65 core storage locations.

Other Routines Used

None.

Flowchart

See CPB-1500 for the flowchart of DSSL (EP3), .MDS20 module.

DSU200 ERROR AND EOF RECOVERY

DSGP (EP4 of .MDS20) begins processing an error status received from a channel with a disc storage subsystem (DSU200) configured. End of file status recovery is performed in this routine to make file segmenting and internal hardware fences invisible to the I/O requestor.

PRECALLING SEQUENCE

Prior to entering DSGP, the registers listed must contain the data indicated.

X2	Logical primary channel index
X3	Absolute I/O entry address
X5	LAL for program
X6	Program number
X7	Processor number

CALLING SEQUENCE

DSGP is called from the .MGEPR module.

```
8           16
-------------------------------------------------------
LDA        |4,DU       )
STC1       |.SSTAK,5*  }    simulated .CALL
TRA        |.CRCT4,2*  )
return 0   |
return 1   |                transfer vector
return 2   |                (.EXIT through .EXIT 2)
           |
```

OPERATING SYSTEM INTERACTION

The top entry in the stack is the IC and I of the call in .MGEPR. The stack is used for temporary storage but it is restored to its original condition before exiting.

No .STEMP storage is used.

No gates are used.

ROUTINE RETURNS

The following exits are from the EOF routine to the .MGEPR module:

Return 0 (.EXIT)	Processing complete, continue searching I/O entry area of interrupted program.
Return 2 (.EXIT 2)	Return status to the program and then continue searching the I/O entry area of interrupted program.

Non-EOF error processing is continued by calling the first drum exception processing SSA module:

```
          .GOTO     .MGP17,1
```

Register contents for all exits are the same as for the call.

SUPPORTING INFORMATION

Programming Method

DSGP is reentrant and written in floatable code.

Interrupts are inhibited while storing and retrieving from the stack.

Storage

No internal temporary storage is used.

DSGP occupies approximately 200 core storage locations.

Other Routines Used

DSU200 Recovery 1, GP17 (.MGP17)
Terminate GEPR Abort, ABREQ (EP5 of .MBRT1)
DSU200 Interrupt Handler DSIT (EP1 of .MDS20)
DSU200 Negative Entry Points (EP -1 of .MDS20)

Flowchart

See CPB-1500 for the flowchart of DSGP (EP4), .MDS20 module.

DSU200 NEGATIVE ENTRY POINTS

The following negative entry points provide device-dependent constants and seek calculation routines for the Disc Storage Subsystem (DSU200).

Negative EP Number	
-15	(Constant) Write and Verify command
-14	(Constant) Bypass count for select strategy
-13	(Constant) Fractional portion of new file request to reserve for later file expansion
-12	(Constant) Seek command
-11	(Constant) Read command
-10	(Constant) Write command
- 9	(Constant) Number of blocks between fences
- 8	(Constant) Number of blocks per LLINK
- 7	(Constant) Number of blocks per LINK
- 6	(Constant) Number of words per block
- 5	LLINK Seek Calculation routine with absolute block number input
- 4	LINK Seek Calculation routine with absolute block number input
- 3	Diagnostic Block Seek Calculation routine
- 2	LLINK Seek Calculation routine with PAT input
- 1	LINK Seek Calculation routine with PAT input

PRECALLING SEQUENCE

There is no precalling sequence for constants. For the seek calculation routines, negative EP -3, -4, and -5, the registers listed must contain the data indicated.

QR	Absolute block number, right justified
X1	Transfer register to routine
X3	Absolute address of word containing secondary SCT address in bits 0-17

EP -1 and EP -2

If the file is random, the QR contains the relative block number, right justified. If the file is linked, the QR contains either:

0 for the next 320 records
or -n for backspace n records
or +n for forward space n records

X1	Transfer register to routine
X3	Absolute address of word containing secondary SCT address in bits 0-17

CALLING SEQUENCE

All negative point entries called are from master mode.

EP -6 through EP -15

The calling sequence is the same for EP -6 through EP -15 (constants) except that p2 is the applicable negative entry point number.

```
8          16
EAXn      |pl             Absolute address of word containing secondary
          |               SCT address in bits 0-17
LDXn      |0,n            Secondary SCT address
LXLn      |0,n        }
ANXn      |=07777,DU  }   Calculate logical channel index
EAA       |p2             Negative entry point number
LDA       |.CRCT4,n*      Retrieving EP transfer word constant
          |
```

EP -1 through EP -5

The calling sequence for entry points EP -1 through EP -5 is the same except that p2 is the applicable negative entry point number.

```
EAX3      |pl             Absolute address of word containing secondary
          |               SCT address in bits 0-17 (and PAT for -1
          |               through -3)
LDXn      |0,3            Secondary SCT address
LXLn      |0,1        }
ANXn      |=07777,DU  }   Calculate logical channel index
EAA       |p2             Negative entry point number
TSX1      |.CRCT4,n*      Transfer to seek calculation routine
return 0  |
return 1  |
          |
```

EP -1 and EP -2

If the PAT is for a linked file and the seek address calculation is successful, bits 18-35 of the fourth word within the PAT (the current relative block number within the memory description) is updated to reflect the completion of the current request.

ROUTINE RETURNS

Negative EP -1, -2, -4, -5

Return 0 (TRA 0,1) Error return. Register status:

QR	Bit 0=1 error (always ON if 1, 2, or 3 is ON) Bit 1=1 tried to position backward too far Bit 2=1 tried to position forward too far Bit 3=1 nonexistent LLINK or LINK number
AR	Destroyed
X0	Destroyed

All other registers are restored.

Return 1 (TRA 1,1) Normal return. Register status:

AR	Record count residue for linked file
QR	Seek address
X0	Destroyed

All other registers are restored.

Negative EP -3

Return 0 (TRA 0,1) Error return. Register status:

AR	Destroyed
QR	Bit 0=1 error Bit 3=1 nonexistent block number
X0	Destroyed

All other registers are restored.

Return 1 (TRA 1,1) Normal return. Register status:

AR	Destroyed
QR	Seek address
X0	Destroyed

All other registers are restored.

OPERATING SYSTEM INTERACTION

The stack is used for the IC and I of the call and for temporary register storage.

No .STEMP storage is used.

A procedure gate is shut upon entry and opened just prior to returning to the caller.

POSTCALLING SEQUENCE

On an error return, the Q-register is tested for error identification. On a normal return, the Q-register (seek address) is stored in the primary mailbox image of the I/O entry (word 4) and the seek command flag in word 2 is turned on.

SUPPORTING INFORMATION

Programming Method

The routine is reentrant but a procedure gate is shut after registers are stored prior to calculating the seek address. It is written in floatable code.

Interrupts are inhibited while the gate is shut.

Storage

Internal temporary storage is used while the gate is shut.

The routine occupies approximately 190 core storage locations.

Other Routines Used

None.

Flowchart

See CPB-1500 for the flowchart of the DSU200 negative entry points, .MDS20 module.

NOMENCLATURE

A module is minimum configuration (4 discs).
A maximum file unit has 4 modules (16 discs).
The minimum addressable segment is 40 words.

There are 16 actuators in a maximum configuration (1 actuator per every two surfaces).

There are 64 positions (tracks) for each actuator, 8 logical read/write heads per actuator, and 3840 words (96 segments) can be accessed from one actuator position (track).

Each disc has 3 zones:

0 = inner
1 = top surface outer
2 = bottom surface outer

Of the 96 segments accessible for one track, 32 are in the inner zone and 64 are in the outer zone.

Sizes

A LINK is 3840 words (96 segments). An LLINK is 320 words (8 segments) and a Diagnostic Block is 40 words (1 segment).

Distribution

The first four of the 64 tracks are reserved for Diagnostic Blocks and LLINKs.

For one module (4 discs):

4*128 segments (inner zone) = 512 Diagnostic Blocks
4*256 segments (outer zone) = 128 LLINKs

The last 60 tracks are reserved for LINKs.

For one module (4 discs):

4*5760 segments (inner and outer zone) = 240 LINKs.

For a maximum configuration file unit (16 discs):

2048	Diagnostic Blocks	(2.08 percent)
512	LLINKs	(4.17 percent)
960	LINKs	(93.75 percent)

SEEK ADDRESSES

ZONE	DISC	TRACK	SEGMENT
XX	XXXX	XXXXXX	XXXXX

INT (A,B) means the truncated integer value of A divided by B.

MOD (A,B) means A modulo B.

Diagnostic Block

As the block number increases, the Diagnostic Blocks are distributed as follows:

BLOCK NUMBER	DISTRIBUTION (OCTAL VALUES)			
	ZONE	DISC	TRACK	SEGMENTS
0-37	0	0	0	0-37
40-77	0	0	0	0-37
100-137	0	0	2	0-37
140-177	0	0	3	0-37
200-237	0	1	0	0-37
•				
•				
•				
3740-3777	0	17	3	0-37

Seek Address (diagnostic block)

= Disc No. *2048+Track*32+Segment No.

$$= \mathrm{INT}\left(\frac{\mathrm{BLK\ No.}}{128}\right)*2048+\mathrm{MOD}\ \left(\mathrm{INT}\left(\frac{\mathrm{Blk\ No.}}{32}\right),4\right)*32+\mathrm{MOD}(\mathrm{Blk\ No.},32)$$

LLINKs

Each LLINK contains 8 segments and there are 32 LLINKs per disc.

As the LLINK number increases on a disc, the track number changes every fourth LLINK, ranging from 0 to 1 to 2 to 3 to 0 and so forth. At each track position for the first four groups of four LLINKs on a disc, 32 addressable segments in the upper outer zone are accessed (8 segments per LLINK). At each track position for the second four groups of four LLINKs on a disc, 32 addressable segments in the lower outer zone are accessed (8 segments per LLINK).

	LLINK NUMBER	DISTRIBUTION (OCTAL VALUES)			
		ZONE	DISC	TRACK	SEGMENTS
first	0	1	0	0	0-7
half of	1	1	0	0	10-17
disc 0)	2	1	0	0	20-27
	3	1	0	0	30-37
	4	1	0	1	0-7
	5	1	0	1	10-17
	•				
	•				
	•				
	17				

	LLINK NUMBER		DISTRIBUTION (OCTAL VALUES)		
		ZONE	DISC	TRACK	SEGMENTS
(second half of disc 0)	20	2	0	0	0-7
	21	2	0	0	10-17
	.				
	.				
	.				
	37	2	0	3	30-37
(first half of disc 1)	40	1	1	0	0-7
	.				
	.				
	.				
(last half of disc 17)	.				
	.				
	.				
	777	2	17	3	30-37

Seek Address (LLINK +block number within LLINK)

$$= \text{Zone} *32768+\text{Disc}*2048+\text{Track}*32+\text{Segment No.}$$

$$= (\text{MOD}\ (\text{INT}\left(\frac{\text{LLINK No.}}{16}\right),2)+1)\ *32768$$

$$+\ \text{INT}\left(\frac{\text{LLINK No.}}{32}\right)*2048+\ (\text{MOD}\ (\text{INT}\left(\frac{\text{LLINK No.}}{4}\right),4))\ *32$$

$$+\ \text{MOD}\ (\text{LLINK},4)*8+\text{Block}$$

LINKs

Each link contains 96 segments (3840 words) which is the total storage accessible at one track position.

Links are assigned on a disc unit beginning on track 4, disc 0. As the LINK number increases, the link is assigned to the next increasing disc on the same track. It then moves to the next track, disc 0, and up again.

There are a total of 960 links on a fully configured file unit (60 tracks *16 discs) of four modules.

The first 32 segments in a link reside on the inner zone (0), the next 32 on the upper outer zone (1), and the last 32 on the lower outer zone (2).

Link Number (Octal)	Distribution (Octal values)	
	Disc	Track
0	0	4
1	1	4
2	2	4
.		
.		
.		
17	17	4
20	0	5
.		
.		
.		
37	17	5
40	0	6
.		
.		
.		
740	0	42
.		
.		
.		
777	17	43
1000	0	44
.		
.		
.		
1660	0	77
.		
.		
.		
1677	17	77

Seek Address (LINK + block number within LINK)

$$= \text{Disc No. }*2048+\text{Track No. }*32+\text{Displacement}$$

$$= \text{MOD (LINK No.,16)}*2048+ \left(\text{INT}\left(\frac{\text{LINK No.}}{4*\text{No. of modules}}\right) +4\right) *32$$

$$+ \text{INT}\left(\frac{\text{Block No. within Link}}{32}\right) *32768+\text{MOD (BLOCK No. Within Link,32)}$$

DSU200 INITIALIZATION

Startup gives control to .IDS20 (.MDS20) when the .MDS20 channel module is loaded into hard core. The configuration tables (.CRCT1) are examined and the module address (.CRCT4) is initialized for those channels with a DSU200 configured.

PRECALLING SEQUENCE

Prior to entering .IDS20, the registers listed must contain the data indicated.

X0	Address of bootstrap card reader description
X2	Address of interrupt vector image
X3	Address of fault vector image
X4	Address of real-time IOC mailbox images

CALLING SEQUENCE

This routine is called from Startup (.MINIT). The return is to the first instruction following the TSX.

```
8          16
TSX1       .IDS20
return
```

OPERATING SYSTEM INTERACTION

None.

ROUTINE RETURNS

Return (0,1) Registers contain:

AR	Zero (no required modules)
QU	Next available load address
X2	through X5 are destroyed

POSTCALLING SEQUENCE

None.

SUPPORTING INFORMATION

Programming Method

The .IDS20 routine is nonreentrant and written in floatable code.

Interrupts are not inhibited.

Storage

No internal temporary storage is used.

This initialization routine occupies approximately 22 core storage locations; however, it is present only at startup time. The next available load address in QU allows Startup to overlay .IDS20.

Other Routines Used

None.

Flowchart

See CPB-1500 for the flowchart of .IDS20, .MDS20 module.

CARD READER CHANNEL MODULE (.MGPIO)

The Card Reader Channel Module (.MGPIO) resides in the Hard Core Monitor. The routines given below comprise the .MGPIO module:

CRIT (EP1)	Card Reader Interrupt Handler
CRIO (EP2)	Card Reader Request
CRSL (EP3)	Card Reader Select
CRGP (EP4)	Card Reader Error
.IGPIO	Card Reader Initialization

These are described on the following pages.

CARD READER INTERRUPT HANDLER

CRIT (EP1 of .MGPIO) interrogates interrupt status for channels which have a card reader configured.

If the IOC configured is a model B, multirecord simulation is initialized for all commands. The coding for simulation follows the hard core code in the .MCPIO module; it remains in hard core only if simulation is required.

PRECALLING SEQUENCE

Prior to entering CRIT, registers listed must contain the data indicated.

QR	Status word 1
X0	Type of interrupt (TI=0, II=2, SI=4)
X1	True channel index
X2	Logical channel index
X3	Bit 0=0, bit 1=power bit, bits 2-5=major status, bits 6-17=0 (for TI and II only)
X4	I/O entry address (for TI and II only)
X5	LAL for GEPOP
X6	Program number of I/O entry (for TI and II only)
X7	Processor number

CALLING SEQUENCE

CRIT is called from the Interrupt Handler (IOTRM) located in the .MIOS module.

```
8          16
LDA        | 1,DU      \
STC1       | .SSTAK,5*  >   simulated .CALL
TRA        | .CRCT4,2* /
return 0   |
   .       |               transfer vector
   .       |               (.EXIT through .EXIT 13)
   .       |
return 13  |
           |
```

OPERATING SYSTEM INTERACTION

The top entry in the stack is the IC and I of the call for the return.

No .STEMP storage is used.

No gates are used.

NORMAL ROUTINE RETURNS

Return 0 (.EXIT)	Status return action. Registers contain: X1 True channel index X2 Logical primary channel index X4 I/O entry address X5 LAL for GEPOP X6 Program number X7 Processor number
Return 1 (.EXIT 1)	GEPR action. Register contents are the same as for return 0.
Return 2 (.EXIT 2)	GESPEC action. Registers contain: X1 True channel index X2 Logical primary channel index X5 LAL for GEPOP X7 Processor number
Return 8 (.EXIT 8)	Error threshold exchange action. Register contents are the same as for return 0.

MULTIRECORD SIMULATION RETURNS

Return 9 (.EXIT 9)	GEPR action for multirecord simulation. Registers contain: QR Status word 1 X1 True channel index X2 Logical primary channel index X4 I/O entry address X5 LAL for GEPOP X6 Program number X7 Processor number
Return 10 (.EXIT 10)	Error threshold exchange for multirecord simulation. Register contents are the same as for return 9.
Return 11 (.EXIT 11)	Status return action for multirecord simulation. Register contents are the same as for return 9.
Return 12 (.EXIT 12)	Issue next connect for multirecord simulation. Register contents are the same as for return 9.

POSTCALLING SEQUENCE

None.

SUPPORTING INFORMATION

Programming Method

CRIT is nonreentrant since IOTRM uses a software gate to ensure the processing of one interrupt at a time. The routine is written in floatable code.

Interrupts are not inhibited.

Storage

Internal temporary storage is used in the simulation code.

CRIT normally occupies approximately 45 core storage locations. Multirecord simulation requires approximately 40 additional locations.

Other Routines Used

Card Reader Initialization, .IGPIO, (.MGPIO)

Flowchart

See CPB-1500 for the flowchart of CRIT (EP1), .MGPIO module.

CARD READER REQUEST

CRIO (EP2 of .MGPIO) validates the I/O command in a GEINOS select sequence and determines the proper return to complete the I/O entry.

PRECALLING SEQUENCE

Prior to entering CRIO, the registers listed must contain the data indicated.

X1	Device SCT address
X2	Logical primary channel index
X3	Select sequence address
X4	I/O entry address
X5	LAL for program
X6	Program number
X7	Processor number

If the request is from the slave area, .STEMP,5 is zero; it is nonzero if the request is from the slave service area.

The upper address limit of the program is in bits 0-17 of .STEMP+1,5 and the offset to the PAT pointer word is in bits 18-35.

CALLING SEQUENCE

CRIO is called from the INOS routine located in the .MIOS module.

```
8            16
LDA          2,DU       \
STC1         .SSTAK,5*   }    simulated .CALL
TRA          .CRCT4,2*  /
return 0
   :                          transfer vector
   :                          (.EXIT through .EXIT 6)
return 6
```

OPERATING SYSTEM INTERACTION

The top entry in the stack is the IC and I of the call for the return.

No .STEMP storage is used.

No gates are used.

ROUTINE RETURNS

Return 0 (.EXIT)	Normal data transfer, 3 word select sequence. .STEMP,5 is zero if the request is from the slave area; it is nonzero if the request is from the slave service area. The upper address limit of the program is in bits 0-17 of .STEMP+1,5, and the offset to the PAT pointer word is in bits 18-35. Registers contain: AR I/O command X3 Select sequence address X4 I/O entry address X5 LAL for program X6 Program number X7 Processor number
Return 2 (.EXIT 2)	Force GEPR override. The contents of .STEMP, .STEMP+1, AR, and index registers are the same as for return 0.
Return 6 (.EXIT 6)	Illegal I/O command. The contents of .STEMP, .STEMP+1, and index registers are the same as for return 0.

POSTCALLING SEQUENCE

None.

SUPPORTING INFORMATION

Programming Method

CRIO is reentrant and written in floatable code.

Interrupts are not inhibited.

Storage

No internal temporary storage is used.

CRIO occupies approximately 30 core storage locations.

Other Routines Used

None.

Flowchart

See CPB-1500 for the flowchart of CRIO (EP2), .MGPIO module.

CARD READER SELECT

CRSL (EP3 of .MGPIO) transfers to the multirecord select routine in STIO which selects an entry from the queue and starts the input/output.

PRECALLING SEQUENCE

Prior to entering CRSL, the registers listed must contain the data indicated.

X0 Transfer register
X1 True channel index
X2 Logical primary channel index
X7 Processor number

CALLING SEQUENCE

CRSL is called from the STIO routine located in the .MIOS module.

```
8          16
-----------------------------------------------------
|LDA       |.CRCT4,2
|ADLA      |3,DU
|TSX0      |0,AU
|return 0  |
|   .      |                transfer vector
|   :      |                (.EXIT through .EXIT 6)
|return 6  |
|          |
|          |
```

OPERATING SYSTEM INTERACTION

No stack or .STEMP storage is used.

The .CRQGT gate is shut while the routine is executed.

ROUTINE RETURNS

Return 5 (TRA 5,0) Multirecord select routine. Registers contain:

X0 Transfer register
X1 True channel index
X2 Logical primary channel index
X7 Processor number

POSTCALLING SEQUENCE

None.

SUPPORTING INFORMATION

Programming Method

CRSL is nonreentrant and written in floatable code.

Interrupts are inhibited.

Storage

No internal temporary storage is used.

CRSL consists of a single transfer instruction.

Other Routines Used

None.

CARD READER ERROR

CRGP (EP4 of .MGPIO) begins processing an error status received from a channel with a card reader configured.

PRECALLING SEQUENCE

Prior to entering CRCP, the registers listed must contain the data indicated.

X2	Logical primary channel index
X3	Absolute I/O entry address
X5	LAL for program
X6	Program number
X7	Processor number

CALLING SEQUENCE

CRGP is called from the .MGEPR module.

```
8          16
LDA       | 4,DU       \
STC1      | .SSTAK,5*   > simulated .CALL
TRA       | .CRCT4,2*  /
return 0  |
return 1  |               transfer vector
return 2  |               (.EXIT through .EXIT 2)
```

OPERATING SYSTEM INTERACTION

The top entry in the stack is the IC and I of the call in .MGEPR.

No .STEMP storage is used.

No gates are used.

ROUTINE RETURNS

The only exit from CRGP is by calling the first exception processing SSA card reader module to continue processing the error. The returns associated with the simulated .CALL reside in the .MGEPR module and are available to GP09.

```
.GOTO     | .MGP09,1
```

Register contents are the same as for the call.

POSTCALLING SEQUENCE

None.

SUPPORTING INFORMATION

Programming Method

CRGP is reentrant and written in floatable code.

Interrupts are inhibited.

Storage

No internal temporary storage is used.

CRGP consists of a single .GOTO command.

Other Routines Used

Card Reader Recovery 1, GP09 (.MGP09)

CARD READER INITIALIZATION

Startup gives control to .IGPIO (.MGPIO) when the .MGPIO channel module is loaded into hard core. The configuration tables (.CRCT1) are examined and the module address (.CRCT4) is initialized for those channels with a card reader configured.

If the IOC configured is a model B, .IGPIO initializes the Card Reader Interrupt Handler (EP1) to perform multirecord simulation.

PRECALLING SEQUENCE

Prior to entering .IGPIO, the registers listed must contain the data indicated.

X0	Address of bootstrap card reader description
X2	Address of interrupt vector image
X3	Address of fault vector image
X4	Address of real-time IOC mailbox images

CALLING SEQUENCE

This routine is called from Startup (.MINIT). The return is to the first instruction following the TSX.

```
8           16
TSX1       | .IGPIO
return     |
           |
           |
```

OPERATING SYSTEM INTERACTION

None.

ROUTINE RETURNS

Return (TRA 0,1) Registers contain:

AR	Zero (no required modules)
QU	Next available load address
X0	through X5 are destroyed

POSTCALLING SEQUENCE

None.

SUPPORTING INFORMATION

Programming Method

The .IGPIO routine is nonreentrant and written in floatable code.

Interrupts are not inhibited.

Storage

The .IGPIO routine occupies approximately 40 core storage locations; however, it is present only at startup time. The next available load address in QU allows Startup to overlay .IGPIO.

Other Routines Used

None.

Flowchart

See CPB-1500 for the flowchart of .IGPIO, .MGPIO module.

MAGNETIC TAPE CHANNEL MODULE (.MMTAP)

The Magnetic Tape Channel Module (.MMTAP) resides in the Hard Core Monitor. The routines given below comprise the .MMTAP module:

- MTIT (EP1) Magnetic Tape Interrupt Handler
- MTIO (EP2) Magnetic Tape Request
- MTSL (EP3) Magnetic Tape Select
- MTGP (EP4) Magnetic Tape Error
- .IMTAP Magnetic Tape Initialization

These are described on the following pages.

MAGNETIC TAPE INTERRUPT HANDLER

MTIT (EP1 of .MMTAP) interrogates interrupt status for channels which have a magnetic tape configured. This includes (1) updating positioning information, (2) enabling the peripheral allocator if its flag is set, and (3) attempting to start I/O, if an earlier attempt resulted in a device busy status and the I/O entry was bypassed until the device became ready.

PRECALLING SEQUENCE

Prior to entering MTIT, registers listed must contain the data indicated.

QR	Status word 1
X0	Type of interrupt (TI=0, II=2, SI=4)
X1	True channel index
X2	Logical channel index
X3	Bit 0=0, bit 1=power bit, bits 2-5=major status, bits 6-17=0 (for TI and II only)
X4	I/O entry address (for TI and II only)
X5	LAL for GEPOP
X6	Program number of I/O entry (for TI and II only)
X7	Processor number

CALLING SEQUENCE

MTIT is called from the Interrupt Handler (IOTRM) located in the .MIOS module.

```
8           16
------------------------------------------------------------
|LDA        |1,DU      ⎫
|STC1       |.SSTAK,5* ⎬  simulated .CALL
|TRA        |.CRCT4,2* ⎭
|return 0   |
|   .       |             transfer vector
|   .       |             (.EXIT through .EXIT 13)
|   .       |
|return 13  |
|           |
```

OPERATING SYSTEM INTERACTION

The top entry in the stack is the IC and I of the call for the return.

No .STEMP storage is used.

The .CRQGT is shut while interrogating the channel queue for GESPECed entries.

ROUTINE RETURNS

Return 0 (.EXIT)	Status return action. Registers contain: X1 True channel index X2 Logical primary channel index X4 I/O entry address X5 LAL for GEPOP X6 Program number X7 Processor number
Return 1 (.EXIT 1)	GEPR action. Register contents are the same as for return 0.
Return 5 (.EXIT 5)	Special interrupt, start I/O. Registers contain: X2 Logical primary channel index X5 LAL for GEPOP X7 Processor number
Return 8 (.EXIT 8)	Error threshold exchange action. Register contents are the same as for return 0.

POSTCALLING SEQUENCE

None.

SUPPORTING INFORMATION

Programming Method

MTIT is nonreentrant since IOTRM uses a software gate to ensure the processing of one interrupt at a time. The routine is written in floatable code.

Interrupts are inhibited only when the .CRQGT gate is shut.

Storage

Internal temporary storage to save registers is used while updating positioning information.

MTIT occupies approximately 240 core storage locations.

Other Routines Used

Enable Program, ENB (EP6 of .MDISP)

Additional Input/Output

Tape Positioning information is updated in the device SCT. The information consists of:

Number of records passed since last EOF (RECCNT)
Number of end of files passed on tape (EOF)
Density (HI or LO)
Lost indicator when positioning data is not valid (LOST)
End of tape status since last rewind (EOT)

Flowchart

See CPB-1500 for the flowchart of MTIT (EP1), .MMTAP module.

MAGNETIC TAPE REQUEST

MTIO (EP2 of .MMTAP) validates the I/O command in a GEINOS select sequence and determines the proper return to complete the I/O entry.

PRECALLING SEQUENCE

Prior to entering MTIO, the registers listed must contain the data indicated.

X1	Device SCT address
X2	Logical primary channel index
X3	Select sequence address
X4	I/O entry address
X5	LAL for program
X6	Program number
X7	Processor number

If the request is from the slave area, .STEMP,5 is zero; it is nonzero if the request is from the slave service area.

The upper address limit of the program is in bits 0-17 of .STEMP+1,5 and the offset to the PAT pointer word is in bits 18-35.

CALLING SEQUENCE

MTIO is called from the INOS routine located in the .MIOS module.

```
8             16
LDA           2,DU        )
STC1          .SSTAK,5*   }  simulated .CALL
TRA           .CRCT4,2*   )
return 0
   .                         transfer vector
   .                         (.EXIT through .EXIT 6)
return 6
```

OPERATING SYSTEM INTERACTION

The top entry in the stack is the IC and I of the call for the return.

No .STEMP storage is used.

No gates are used.

ROUTINE RETURNS

Return 0 (.EXIT)	Normal data transfer, 3 word select sequence. .STEMP,5 is zero if the request is from the slave area; it is nonzero if the request is from the slave service area. The upper address limit of the program is in bits 0-17 of .STEMP+1,5 and the offset to the PAT pointer word is in bits 18-35. Registers contain: AR I/O command X3 Select sequence address X4 I/O entry address X5 LAL for program X6 Program number X7 Processor number
Return 2 (.EXIT 2)	Force GEPR override. The contents of .STEMP, .STEMP+1, AR, and index registers are the same as for return 0.
Return 3 (.EXIT 3)	Write single character command. The contents of .STEMP and .STEMP+1, AR, and index registers are the same as for return 0.
Return 6 (.EXIT 6)	Illegal I/O command. The contents of .STEMP, .STEMP+1, and index registers are the same as for return 0.

POSTCALLING SEQUENCE

None.

SUPPORTING INFORMATION

Programming Method

MTIO is reentrant and written in floatable code.

Interrupts are not inhibited.

Storage

No internal temporary storage is used.

MTIO occupies approximately 65 core storage locations.

Other Routines Used

None.

Flowchart

See CPB-1500 for the flowchart of MTIO (EP2), .MMTAP module.

MAGNETIC TAPE SELECT

MTSL (EP3 of .MMTAP) transfers to the general select routine in STIO which selects an entry from the queue and starts the input/output.

PRECALLING SEQUENCE

Prior to entering MTSL, the registers listed must contain the data indicated.

X0	Transfer register
X1	True channel index
X2	Logical primary channel index
X7	Processor number

CALLING SEQUENCE

MTSL is called from the STIO routine located in the .MIOS module.

```
8          16
LDA        .CRCT4,2
ADLA       3,DU
TSX0       0,AU
return 0
   .                         transfer vector
   .                         (.EXIT through .EXIT 6)
   .
return 6
```

OPERATING SYSTEM INTERACTION

No stack or .STEMP storage is used.

The .CRQGT gate is shut while the routine is executed.

ROUTINE RETURNS

Return 0 (TRA 0,0) Normal select routine. Registers contain:

X0	Transfer register
X1	True channel index
X2	Logical primary channel index
X7	Processor number

POSTCALLING SEQUENCE

None.

SUPPORTING INFORMATION

Programming Method

MTSL is nonreentrant and written in floatable code.

Interrupts are inhibited.

Storage

No internal temporary storage is used.

MTSL consists of a single transfer instruction.

Other Routines Used

None.

MAGNETIC TAPE ERROR

MTGP (EP4 of .MMTAP) begins processing an error status received from a channel with a magnetic tape configured.

PRECALLING SEQUENCE

Prior to entering MTGP, the registers listed must contain the data indicated.

X2	Logical primary channel index
X3	Absolute I/O entry address
X5	LAL for program
X5	Program number
X7	Processor number

CALLING SEQUENCE

MTGP is called from the .MGEPR module.

```
8           16
LDA        | 4,DU       )
STC1       |.SSTAK,5*   }  simulated .CALL
TRA        | .CRCT4,2*  )
return 0   |
return 1   |                 transfer vector
return 2   |                 (.EXIT through .EXIT 2)
           |
```

OPERATING SYSTEM INTERACTION

The top entry in the stack is the IC and I of the call in .MGEPR.

No .STEMP storage is used.

No gates are used.

ROUTINE RETURNS

The only exit from MTGP is by calling the first exception processing SSA magnetic tape module to continue processing the error. The returns associated with the simulated .CALL reside in the .MGEPR module and are available to GP30.

```
.GOTO      | .MGP30,1
```

Register contents are the same as for the call.

POSTCALLING SEQUENCE

None.

SUPPORTING INFORMATION

Programming Method

MTGP is reentrant and written in floatable code.

Interrupts are inhibited.

Storage

No internal temporary storage is used.

MTGP consists of a single .GOTO command.

Other Routines Used

Magnetic Tape Recovery, GP30 (.MGP30)

.IMTAP
.MMTAP

MAGNETIC TAPE INITIALIZATION

Startup gives control to .IMTAP (.MMTAP) when the .MMTAP channel module is loaded into hard core. The configuration tables (CRCT1) are examined and the module address (.CRCT4) is initialized for those channels with magnetic tapes configured.

Also, addresses within the tables used by the Magnetic Tape Interrupt Handler (EP1) are made absolute.

PRECALLING SEQUENCE

Prior to entering .IMTAP, the registers listed must contain the data indicated.

X0	Address of bootstrap card reader description
X2	Address of interrupt vector image
X3	Address of fault vector image
X4	Address of real-time IOC mailbox images

CALLING SEQUENCE

This routine is called from Startup (.MINIT). The return is to the first instruction following the TSX.

```
8         16
TSX1      .IMTAP
return
```

OPERATING SYSTEM INTERACTION

None.

ROUTINE RETURNS

Return (TRA 0,1) Registers contain:

AR	Zero (no required modules)
QU	Next available load address
X2	through X5 are destroyed

POSTCALLING SEQUENCE

None.

SUPPORTING INFORMATION

Programming Method

The .IMTAP routine is nonreentrant and written in floatable code.

Interrupts are not inhibited.

Storage

No internal temporary storage is used.

This initialization routine occupies approximately 47 core storage locations; however, it is present only at startup time. The next available load address in QU allows Startup to overlay .IMTAP.

Other Routines Used

None.

Flowchart

See CPB-1500 for the flowchart of .IMTAP, .MMTAP module.

PRINTER CHANNEL MODULE (.MPRIO)

The Printer Channel Module (.MPRIO) resides in the Hard Core Monitor. The routines given below comprise the .MPRIO module:

• PRIT (EP1)	Printer Interrupt Handler
• PRIO (EP2)	Printer Request
• PRSL (EP3)	Printer Select
• PRGP (EP4)	Printer Error
• .IPRIO	Printer Initialization

These are described on the following pages.

PRINTER INTERRUPT HANDLER

PRIT (EP1 of .MPRIO) interrogates interrupt status for channels which have a printer configured.

If the IOC configured is a model B, multirecord simulation is initialized for all commands. The coding for simulation follows the hard core code which is in the .MCPIO module; it remains in hard core only if simulation is required.

PRECALLING SEQUENCE

Prior to entering PRIT, the registers listed must contain the data indicated:

QR	Status word 1
X0	Type of interrupt (TI=0, II=2, SI=4)
X1	True channel index
X2	Logical channel index
X3	Bit 0=0, bit 1=power bit, bits 2-5=major status, bits 6-17=0 (for TI and II only)
X4	I/O entry address (for TI and II only)
X5	LAL for GEPOP
X6	Program number of I/O entry (for TI and II only)
X7	Processor number

CALLING SEQUENCE

PRIT is called from the Interrupt Handler (IOTRM) located in the .MIOS module.

```
 8            16
LDA          |1,DU      )
STC1         |.SSTAK,5*  }  simulated .CALL
TRA          |.CRCT4,2* )
return 0     |
   .         |              transfer vector
   .         |              (.EXIT through .EXIT 13)
   .         |
return 13    |
```

OPERATING SYSTEM INTERACTION

The top entry in the stack is the IC and I of the call for the return.

No .STEMP storage is used.

No gates are used.

NORMAL ROUTINE RETURNS

Return 0 (.EXIT)	Status return action. Registers contain: X1 True channel index X2 Logical primary channel index X4 I/O entry address X5 LAL for GEPOP X6 Program number X7 Processor number
Return 2 (.EXIT 2)	GESPEC action. Registers contain: X1 True channel index X2 Logical primary channel index X5 LAL for GEPOP X7 Processor number
Return 8 (.EXIT 8)	Error threshold/exchange action. Register contents are the same as for return 0.

MULTIRECORD SIMULATION RETURNS

Return 9 (.EXIT 9)	GEPR action for multirecord simulation. Registers contain: QR Status word 1 X1 True channel index X2 Logical primary channel index X4 I/O entry address X5 LAL for GEPOP X6 Program number X7 Processor number
Return 10 (.EXIT 10)	Error threshold/exchange for multirecord simulation. Register contents are the same as for return 9.
Return 11 (.EXIT 11)	Status return action for multirecord simulation. Register contents are the same as for return 9.
Return 12 (.EXIT 12)	Issue next connect for multirecord simulation. Register contents are the same as for return 9.

POSTCALLING SEQUENCE

None.

SUPPORTING INFORMATION

Programming Method

PRIT is nonreentrant since IOTRM uses a software gate to ensure the processing of one interrupt at a time. The routine is written in floatable code.

Interrupts are not inhibited.

Storage

Internal temporary storage is used in the simulation code.

PRIT normally occupies approximately 45 core storage locations. Multirecord simulation requires approximately 40 additional locations.

Other Routines Used

Printer Initialization, .IPRIO (.MPRIO)

Flowchart

See CPB-1500 for the flowchart of PRIT (EP1), .MPRIO module.

PRIO(EP2)
.MPRIO

PRINTER REQUEST

PRIO (EP2 of .MPRIO) validates the I/O command in a GEINOS select sequence and determines the proper return to complete the I/O entry.

PRECALLING SEQUENCE

Prior to entering PRIO, the registers listed must contain the data indicated.

X1	Device SCT address
X2	Logical primary channel index
X3	Select sequence address
X4	I/O entry address
X5	LAL for program
X6	Program number
X7	Processor number

If the request is from the slave area, .STEMP,5 is zero; it is nonzero if the request is from the slave service area.

The upper address limit of the program is in bits 0-17 of .STEMP+1,5 and the offset to the PAT pointer word is in bits 18-35.

CALLING SEQUENCE

PRIO is called from the INOS routine located in the .MIOS module.

```
8              16
LDA            2,DU       )
STC1           .SSTAK,5*  }  simulated .CALL
TRA            .CRCT4,2*  )
return 0
   .                        transfer vector
   .                        (.EXIT through .EXIT 6)
   .
return 6
```

OPERATING SYSTEM INTERACTION

The top entry in the stack is the IC and I of the call for the return.

No .STEMP storage is used.

No gates are used.

ROUTINE RETURNS

Return 0 (.EXIT) Normal data transfer, 3 word select sequence. .STEMP,5 is zero if the request is from the slave area; it is nonzero if the request is from the slave service area.

The upper address limit of the program is in bits 0-17 of .STEMP+1,5 and the offset to the PAT pointer word is in bits 18-35.

Registers contain:

AR	I/O command
X3	Select sequence address
X4	I/O entry address
X5	LAL for program
X6	Program number
X7	Processor number

Return 2 (.EXIT 2) Force GEPR override. The contents of .STEMP, .STEMP+1, AR, and index registers are the same as for return 0.

Return 6 (.EXIT 6) Illegal I/O command. The contents of .STEMP, .STEMP+1, and index registers are the same as for return 0.

POSTCALLING SEQUENCE

None.

SUPPORTING INFORMATION

Programming Method

PRIO is reentrant and written in floatable code.

Interrupts are not inhibited.

Storage

No internal temporary storage is used.

PRIO occupies approximately 25 core storage locations.

Other Routines Used

None.

Flowchart

See CPB-1500 for the flowchart of PRIO (EP2), .MPRIO module.

PRSL(EP3)
.MPRIO

PRINTER SELECT

PRSL (EP3 of .MPRIO) transfers to the multirecord select routine in STIO which selects an entry from the queue and starts the input/output.

PRECALLING SEQUENCE

Prior to entering PRSL, the registers listed must contain the data indicated.

X0	Transfer register
X1	True channel index
X2	Logical primary channel index
X7	Processor number

CALLING SEQUENCE

PRSL is called from the STIO routine located in the .MIOS module.

```
8           16
LDA        | .CRCT4,2
ADLA       | 3,DU
TSX0       | 0,AU
return 0   |
   .       |                transfer vector
   .       |                (.EXIT through .EXIT 6)
return 6   |
           |
```

OPERATING SYSTEM INTERACTION

No stack or .STEMP storage is used.

The .CRQGT gate is shut while the routine is executed.

ROUTINE RETURNS

Return 5 (TRA 5,0)   Multirecord select routine. Registers contain:

X0	Transfer register
X1	True channel index
X2	Logical primary channel index
X7	Processor number

POSTCALLING SEQUENCE

None.

SUPPORTING INFORMATION

Programming Method

PRSL is nonreentrant and written in floatable code.

Interrupts are inhibited.

Storage

No internal temporary storage is used.

PRSL consists of a single transfer instruction.

Other Routines Used

None.

**PRGP(EP4)
.MPRIO**

PRINTER ERROR

PRGP (EP4 of .MPRIO) begins processing an error status received from a channel with a printer configured.

PRECALLING SEQUENCE

Prior to entering PRGP, the registers listed must contain the data indicated.

X2	Logical primary channel index
X3	Absolute I/O entry address
X5	LAL for program
X6	Program number
X7	Processor number

CALLING SEQUENCE

PRGP is called from the .MGEPR module.

```
8           16
LDA        | 4,DU
STC1       | .SSTAK,5*  }   simulated .CALL
TRA        | .CRCT4,2*
return 0   |
return 1   |                transfer vector
return 2   |                (.EXIT through .EXIT 2)
```

OPERATING SYSTEM INTERACTION

The top entry in the stack is the IC and I of the call in .MGEPR.

No .STEMP storage is used.

No gates are used.

ROUTINE RETURNS

The only exit from PRGP is by calling the first exception processing SSA printer module to continue processing the error. The returns associated with the simulated .CALL reside in the .MGEPR module and are available to GP13.

```
 .GOTO     | .MGP13,1
```

Register contents are the same as for the call.

POSTCALLING SEQUENCE

None.

SUPPORTING INFORMATION

Programming Method

PRGP is reentrant and written in floatable code.

Interrupts are inhibited.

Storage

No internal temporary storage is used.

PRGP consists of a single .GOTO command.

Other Routines Used

Printer Recovery 1, GP13 (.MGP13)

PRINTER INITIALIZATION

Startup gives control to .IPRIO (.MGPIO) when the .MGPIO channel module is loaded into hard core. The configuration tables (.CRCT1) are examined and the module address (.CRCT4) is initialized for those channels with a printer configured.

If the IOC configured is a model B, .ICPIO initializes the Printer Interrupt Handler (EP1) to perform multirecord simulation.

PRECALLING SEQUENCE

Prior to entering .IPRIO, the registers listed must contain the data indicated.

X0	Address of bootstrap card reader description
X2	Address of interrupt vector image
X3	Address of fault vector image
X4	Address of real-time IOC mailbox images

CALLING SEQUENCE

This routine is called from Startup (.MINIT). The return is to the first instruction following the TSX.

```
8          16
TSX1       .IPRIO
return
```

OPERATING SYSTEM INTERACTION

None.

ROUTINE RETURNS

Return (TRA 0,1) Registers contain:

AR	Zero (no required modules)
QU	Next available load address
X0	through X5 are destroyed

POSTCALLING SEQUENCE

None.

SUPPORTING INFORMATION

Programming Method

The .IPRIO routine is nonreentrant and written in floatable code.

Interrupts are not inhibited.

Storage

No internal temporary storage is used.

The .IPRIO routine occupies approximately 35 core storage locations; however, it is present only at startup time. The next available load address in QU allows Startup to overlay .IPRIO.

Other Routines Used

None.

Flowchart

See CPB-1500 for the flowchart of .IPRIO, .MGPIO module.

PAPER TAPE CHANNEL MODULE (.MPTAP)

The Paper Tape Channel Module (.MPTAP) resides in the Hard Core Monitor. The routines given below comprise the .MPTAP module:

PTIT	Paper Tape Interrupt Handler
PTIO	Paper Tape Request
PTSL	Paper Tape Select
PTGP	Paper Tape Error
.IPTAP	Paper Tape Initialization

These are described on the following pages.

PAPER TAPE INTERRUPT HANDLER

PTIT (EP1 of .MPTAP) interrogates interrupt status for channels which have a paper tape reader/punch configured.

PRECALLING SEQUENCE

Prior to entering PTIT, registers listed must contain the data indicated:

QR	Status word 1
X0	Type of interrupt (TI=0, II=2, SI=4)
X1	True channel index
X2	Logical channel index
X3	Bit 0=0, bit 1=power bit, bits 2-5=major status, bits 6-17=0 (for TI and II only)
X4	I/O entry address (for TI and II only)
X5	LAL for GEPOP
X6	Program number of I/O entry (for TI and II only)
X7	Processor number

CALLING SEQUENCE

PTIT is called from the Interrupt Handler (IOTRM) located in the .MIOS module.

```
8            16
LDA          1,DU       \
STC1         .SSTAK,5*   }  simulated .CALL
TRA          .CRCT4,2*  /
return 0
   .                        transfer vector
   .                        (.EXIT through .EXIT 13)
   .
return 13
```

OPERATING SYSTEM INTERACTION

The top entry in the stack is the IC and I of the call for the return.

No .STEMP storage is used.

No gates are used.

ROUTINE RETURNS

Return 0 (.EXIT)	Status return action. Registers contain: X1 True channel index X2 Logical primary channel index X4 I/O entry address X5 LAL for GEPOP X6 Program number X7 Processor number
Return 1 (.EXIT 1)	GEPR action. Register contents are the same as for return 0.
Return 3 (.EXIT 3)	No action on special interrupt. Registers contain: X5 LAL for GEPOP X7 Processor number
Return 8 (.EXIT 8)	Error threshold action. Register contents are the same as for return 0.

POSTCALLING SEQUENCE

None.

SUPPORTING INFORMATION

Programming Method

PTIT is nonreentrant since the IOTRM uses a software gate to ensure the processing of one interrupt at a time. The routine is written in floatable code.

Interrupts are not inhibited.

Storage

No internal temporary storage is used.

PTIT occupies approximately 50 core storage locations.

Other Routines Used

None.

Flowchart

See CPB-1500 for the flowchart of PTIT (EP1), .MPTAP module.

PAPER TAPE REQUEST

PTIO (EP2 of .MPTAP) validates the I/O command in a GEINOS select sequence and determines the proper return to complete the I/O entry.

PRECALLING SEQUENCE

Prior to entering PTIO, the registers listed must contain the data indicated.

X1	Device SCT address
X2	Logical primary channel index
X3	Select sequence address
X4	I/O entry address
X5	LAL for program
X6	Program number
X7	Processor number

If the request is from the slave area, .STEMP,5 is zero; it is nonzero if the request is from the slave service area.

The upper address limit of the program is in bits 0-17 of .STEMP+1,5 and the offset to the PAT pointer word is in bits 18-35.

CALLING SEQUENCE

PTIO is called from the INOS routine located in the .MIOS module.

```
8             16
LDA           2,DU      ⎫
STC1          .SSTAK,5* ⎬  simulated .CALL
TRA           .CRCT4,2* ⎭
return 0
   .                       transfer vector
   .                       (.EXIT through .EXIT 6)
   .
return 6
```

OPERATING SYSTEM INTERACTION

The top entry in the stack is the IC and I of the call for the return.

No .STEMP storage is used.

No gates are used.

ROUTINE RETURNS

Return 0 (.EXIT)	Normal data transfer, 3 word select sequence. .STEMP,5 is zero if the request is from the slave area; it is nonzero if the request is from the slave service area. The upper address limit of the program is in bits 0-17 of .STEMP+1,5 and the offset to the PAT pointer word is in bits 18-35. Registers contain: AR I/O command X3 Select sequence address X4 I/O entry address X5 LAL for program X6 Program number X7 Processor number
Return 2 (.EXIT 2)	Force GEPR override. The contents of .STEMP, .STEMP+1, AR, and index registers are the same as for return 0.
Return 6 (.EXIT 6)	Illegal I/O command. The contents of .STEMP, .STEMP+1, and index registers are the same as for return 0.

POSTCALLING SEQUENCE

None.

SUPPORTING INFORMATION

Programming Method

PTIO is reentrant and written in floatable code.

Interrupts are not inhibited.

Storage

No internal temporary storage is used.

PTIO occupies approximately 30 core storage locations.

Other Routines Used

None.

Flowchart

See CPB-1500 for the flowchart of PTIO (EP2), .MPTAP.

PAPER TAPE SELECT

PTSL (EP3 of .MPTAP) transfers to the general select routine in STIO which selects an entry from the queue and starts the input/output.

PRECALLING SEQUENCE

Prior to entering PTSL, the registers listed must contain the data indicated.

X0 Transfer register
X1 True channel index
X2 Logical primary channel index
X7 Processor number

CALLING SEQUENCE

PTSL is called from the STIO routine located in the .MIOS module.

```
8          16
LDA        .CRCT4,2
ADLA       3,DU
TSX0       0,AU
return 0
   .                   transfer vector
   .                   (.EXIT through .EXIT 6)
   .
return 6
```

OPERATING SYSTEM INTERACTION

No stack or .STEMP storage is used.

The .CRQGT gate is shut while the routine is executed.

ROUTINE RETURNS

Return 0 (TRA 0,0) Normal select routine. Registers contain:

X0 Transfer register
X1 True channel index
X2 Logical primary channel index
X7 Processor number

POSTCALLING SEQUENCE

None.

SUPPORTING INFORMATION

Programming Method

PTSL is nonreentrant and written in floatable code.

Interrupts are inhibited.

Storage

No internal temporary storage is used.

PTSL consists of a single transfer instruction.

Other Routines Used

None.

PAPER TAPE ERROR

PTGP (EP4 of .MPTAP) begins processing an error status received from a channel with a paper tape configured.

PRECALLING SEQUENCE

Prior to entering PTGP, the registers listed must contain the data indicated.

X2	Logical primary channel index
X3	Absolute I/O entry address
X5	LAL for program
X6	Program number
X7	Processor number

CALLING SEQUENCE

PTGP is called from the .MGEPR module.

```
8          16
LDA       |4,DU       )
STC1      |.SSTAK,5*  }    simulated .CALL
TRA       |.CRCT4,2*  )
return 0  |
return 1  |                transfer vector
return 2  |                (.EXIT through .EXIT 2)
```

OPERATING SYSTEM INTERACTION

The top entry in the stack is the IC and I of the call in .MGEPR.

No .STEMP storage is used.

No gates are used.

ROUTINE RETURNS

The only exit from PTGP is by calling the first exception processing SSA device module to continue processing the error. The returns associated with the simulated .CALL reside in the .MGEPR module and are available to GP11.

```
.GOTO     |.MGP11,1
```

Register contents are the same as for the call.

POSTCALLING SEQUENCE

None.

SUPPORTING INFORMATION

Programming Method

PTGP is reentrant and written in floatable code.

Interrupts are not inhibited.

Storage

No internal temporary storage is used.

PTGP consists of a single .GOTO command.

Other Routines Used

Paper Tape Recovery, GP11 (.MGP11)

PAPER TAPE INITIALIZATION

Startup gives control to .IPTAP (.MMTAP) when the .MPTAP channel module is loaded into hard core. The configuration tables (.CRCT1) are examined and the module address (.CRCT4) is initialized for those channels with paper tape configured.

PRECALLING SEQUENCE

Prior to entering .IPTAP, the registers listed must contain the data indicated.

X0	Address of bootstrap card reader description
X2	Address of interrupt vector image
X3	Address of fault vector image
X4	Address of real-time IOC mailbox images

CALLING SEQUENCE

This routine is called from Startup (.MINIT). The return is to the first instruction following the TSX.

```
8          16
TSX1      |.IPTAP
return    |
```

OPERATING SYSTEM INTERACTION

None.

ROUTINE RETURNS

Return (TRA 0,1) Registers contain:

AR	Zero (no required modules)
QU	Next available load address
X2	thru X5 are destroyed

POSTCALLING SEQUENCE

None.

SUPPORTING INFORMATION

Programming Method

The .IPTAP routine is nonreentrant and written in floatable code.

Interrupts are not inhibited.

Storage

No internal temporary storage is used.

The .IPTAP routine occupies approximately 20 core storage locations; however, it is present only at startup time. The next available load address in QU allows Startup to overlay .IPTAP.

Other Routines Used

None.

Flowchart

See CPB-1500 for the flowchart of .IPTAP, .MPTAP module.

TYPEWRITER CHANNEL MODULE (.MTYPE)

The Typewriter Channel Module (.MTYPE) resides in the Hard Core Monitor. The routines given below comprise the .MTYPE module:

TYIT	Typewriter Interrupt Handler
TYIO	Typewriter Request
TYSL	Typewriter Select
TYGP	Typewriter Error
.ITYPE	Typewriter Initialization

These are described in the following pages.

TYPEWRITER INTERRUPT HANDLER

TYIT (EP1 of .MTYPE) interrogates interrupt status for channels which have a typewriter configured.

PRECALLING SEQUENCE

Prior to entering TYIT, registers listed must contain the data indicated:

QR	Status word 1
X0	Type of interrupt (TI=0, II=2, SI=4)
X1	True channel index
X2	Logical channel index
X3	Bit 0=0, bit 1=power bit, bits 2-5=major status, bits 6-17=0 (for TI and II only)
X4	I/O entry address (for TI and II only)
X5	LAL for GEPOP
X6	Program number of I/O entry (for TI and II only)
X7	Processor number

CALLING SEQUENCE

TYIT is called from the Interrupt Handler (IOTRM) located in the .MIOS module.

```
8           16
LDA         1,DU         }
STC1        .SSTAK,5*    }  simulated .CALL
TRA         .CRCT4,2*    }
return 0
   .                        transfer vector
   .                        (.EXIT through .EXIT 13)
   .
return 13
```

OPERATING SYSTEM INTERACTION

The top entry in the stack is the IC and I of the call for the return.

No .STEMP storage is used.

No gates are used.

ROUTINE RETURNS

Return 0 (.EXIT)	Status return action. Registers contain: X1 True channel index X2 Logical primary channel index X4 I/O entry address X5 LAL for GEPOP X6 Program number X7 Processor number
Return 1 (.EXIT 1)	GEPR action. Register contents are the same as for return 0.
Return 2 (.EXIT 2)	GESPEC action. Registers contain: X1 True channel index X2 Logical primary channel index X5 LAL for GEPOP X7 Processor number
Return 7 (.EXIT 7)	Reissue console read. Secondary mailboxes 3 and 4 are already stored, I/O entry is marked in transmission, and the channel is marked busy. Registers contain: QR Primary mailbox image for reissue X1 True channel index X2 Logical primary channel index X4 I/O entry address X5 LAL for GEPOP X6 Program number X7 Processor number
Return 8 (.EXIT 8)	Error threshold action. Register contents are the same as for return 0.

POSTCALLING SEQUENCE

None.

SUPPORTING INFORMATION

Programming Method

TYIT is nonreentrant since IOTRM uses a software gate to ensure the processing of one interrupt at a time. The routine is written in floatable code.

Interrupts are not inhibited.

TYIT(EP1)
.MTYPE

Storage

Internal temporary storage is used in processing Data Alert, Operator Alert.

TYIT occupies approximately 80 core storage locations.

Other Routines Used

None.

Flowchart

See CPB-1500 for the flowchart of TYIT (EP1), .MTYPE module.

TYPEWRITER REQUEST

TYIO (EP2 of .MTYPE) validates the I/O command in a GEINOS select sequence and determines the proper return to complete the I/O entry.

PRECALLING SEQUENCE

Prior to entering TYIO, the registers listed must contain the data indicated.

X1	Device SCT address
X2	Logical primary channel index
X3	Select sequence address
X4	I/O entry address
X5	LAL for program
X6	Program number
X7	Processor number

If the request is from the slave area, .STEMP,5 is zero; it is nonzero if the request is from the slave service area.

The upper address limit of the program is in bits 0-17 of .STEMP+1,5 and the offset to the PAT pointer word is in bits 18-35.

CALLING SEQUENCE

TYIO is called from the INOS routine located in the .MIOS module.

```
8               16
LDA            |2,DU       )
STC1           |.SSTAK,5*  }  simulated .CALL
TRA            |.CRCT4,2*  )
return 0       |
   .           |              transfer vector
   .           |              (.EXIT through .EXIT 6)
   .           |
return 6       |
```

OPERATING SYSTEM INTERACTION

The top entry in the stack is the IC and I of the call for the return.

No .STEMP storage is used.

No gates are used.

ROUTINE RETURNS

Return 0 (.EXIT)	Normal data transfer, 3 word select sequence. .STEMP,5 is zero if the request is from the slave area; it is nonzero if the request is from the slave service area. The upper address limit of the program is in bits 0-17 of .STEMP+1,5 and the offset to the PAT pointer word is in bits 18-35. Registers contain: AR I/O command X3 Select sequence address X4 I/O entry address X5 LAL for program X6 Program number X7 Processor number
Return 2 (.EXIT 2)	Force GEPR override. The contents of .STEMP, .STEMP+1, AR, and index registers are the same as for return 0.
Return 4 (.EXIT 4)	Two typewriter commands, which are stored in I/O entry. The contents of .STEMP, .STEMP+1, and index registers are the same as for return 0.
Return 6 (.EXIT 6)	Illegal I/O command. The contents of .STEMP, .STEMP+1, and index registers are the same as for return 0.

POSTCALLING SEQUENCE

None.

SUPPORTING INFORMATION

Programming Method

TYIO is reentrant and written in floatable code.

Interrupts are not inhibited.

Storage

No internal temporary storage is used.

TYIO occupies approximately 40 core storage locations.

Other Routines Used

None.

Flowchart

See CPB-1500 for the flowchart of TYIO (EP2), .MTYPE module.

TYPEWRITER SELECT

TYSL (EP3 of .MTYPE) selects an I/O entry from the channel queue to start.

PRECALLING SEQUENCE

Prior to entering TYSL, the registers listed must contain the data indicated.

X0	Transfer register
X1	True channel index
X2	Logical primary channel index
X7	Processor number

CALLING SEQUENCE

TYSL is called from the STIO routine located in the .MIOS module.

```
8          16
LDA        .CRCT4,2
ADLA       3,DU
TSX0       0,AU
return 0
   .                        transfer vector
   .                        (.EXIT through .EXIT 6)
   .
return 6
```

OPERATING SYSTEM INTERACTION

No stack or .STEMP storage is used.

The .CRQGT gate is shut while the routine is executed.

ROUTINE RETURNS

Return 1 (TRA 1,0) Start selected I/O entry. Registers contain:

X0	Transfer register
X1	True channel index
X2	Logical primary channel index
X4	I/O entry selected
X7	Processor number

POSTCALLING SEQUENCE

None.

SUPPORTING INFORMATION

Programming Method

TYSL is nonreentrant and written in floatable code.

Interrupts are inhibited.

Storage

Internal temporary storage is used while the .CRQGT gate is shut.

TYSL occupies approximately 25 core storage locations.

Other Routines Used

None.

Flowchart

See CPB-1500 for the flowchart of TYSL (EP3), .MTYPE module.

TYGP(EP4)
.MTYPE

TYPEWRITER ERROR

TYGP (EP4 of .MTYPE) begins processing an error status received from a channel with a typewriter configured.

PRECALLING SEQUENCE

Prior to entering TYGP, the registers listed must contain the data indicated.

X2 Logical primary channel index
X3 Absolute I/O entry address
X5 LAL for program
X6 Program number
X7 Processor number

CALLING SEQUENCE

TYGP is called from the .MGEPR module.

```
8           16
LDA        |4,DU        }
STC1       |.SSTAK,5*   }   simulated .CALL
TRA        |.CRCT4,2*   }
return 0   |
return 1   |                transfer vector
return 2   |                (.EXIT through .EXIT 2)
```

OPERATING SYSTEM INTERACTION

The top entry in the stack is the IC and I of the call in .MGEPR.

No .STEMP storage is used.

No gates are used.

ROUTINE RETURNS

The only exit from TYGP is by calling the exception processing SSA typewriter module to continue processing the error. The returns associated with the simulated .CALL reside in the .MGEPR module and are available to GP22.

```
.GOTO      | .MGP22,1
```

Register contents are the same as for the call.

POSTCALLING SEQUENCE

None.

SUPPORTING INFORMATION

Programming Method

TYGP is reentrant and written in floatable code.

Interrupts are not inhibited.

Storage

No internal temporary storage is used.

TYGP consists of a single .GOTO command.

Other Routines Used

Console Recovery, GP22 (.MGP22)

.ITYPE
.MTYPE

TYPEWRITER INITIALIZATION

Startup gives control to .ITYPE(.MTYPE) when the .MTYPE channel module is loaded into hard core. The configuration tables (.CRCT1) are examined and the module address (.CRCT4) is initialized for those channels with a typewriter configured.

PRECALLING SEQUENCE

Prior to entering .ITYPE, the registers listed must contain the data indicated.

X0	Address of bootstrap card reader description
X2	Address of interrupt vector image
X3	Address of fault vector image
X4	Address of real-time IOC mailbox images

CALLING SEQUENCE

This routine is called from Startup (.MINIT). The return is to the first instruction following the TSX.

```
8         16
TSX1      .ITYPE
return
```

OPERATING SYSTEM INTERACTION

None.

ROUTINE RETURNS

Return (TRA 0,1) Registers contain:

AR	Zero (no required modules)
QU	Next available load address
X2	thru X5 are destroyed

POSTCALLING SEQUENCE

None.

SUPPORTING INFORMATION

Programming Method

The .ITYPE routine is nonreentrant and written in floatable code.

Interrupts are not inhibited.

Storage

No internal temporary storage is used.

The .ITYPE routine occupies approximately 20 core storage locations; however, it is present only at startup time. The next available load address in QU allows Startup to overlay .ITYPE.

Other Routines Used

None.

Flowchart

See CPB-1500 for the flowchart of .ITYPE, .MTYPE module.

4. ACCOUNTING FILE TAPE SWITCHING

Tape switching for the accounting file is performed by the Accounting Tape Switching Module (.MACTS). It differs from the other IOS modules in that it does not reside permanently in the Hard Core Monitor. It is called into hard core only when required. When required, it is called into the slave service area of the Main GEPOP module .MPOPM.

ACCOUNTING TAPE SWITCHING MODULE (.MACTS)

The .MACTS module contains three entry points:

- ACTS1 (EP1 of .MACTS)
- ACTS2 (EP2 of .MACTS)
- ACTS3 (EP3 of .MACTS)

which are discussed on the following pages.

NORMAL CLOSE OF ACCOUNTING FILE

ACTS1 (EP1 of .MACTS) closes the current accounting file preparatory to performing normal accounting tape switching. An end of file is written on the current accounting tape, it is rewound, and a dismount message is issued. The Peripheral Dispenser Module (.MALC2) is then called to assign another handler on the same channel. A ready handler is requested first. If the request is denied, a standby handler is requested. If neither request is granted, ACTS1 waits for the currently assigned handler to come to standby.

After a handler is assigned, applicable system configuration tables (SCT) are modified if a device change was made. The exchange method is identical to that used by the GE-625/635 Error Processing (GEPR) routines. A typewriter message notifies the operator to dismount the completed accounting tape and identifies the now-current collector.

PRECALLING SEQUENCE

Prior to entering ACTS1, the registers listed must contain the data indicated.

X5	LAL for the program
X6	Program number
X7	Processor number

In addition, the tape writing flag is set by the courtesy call routine in ACTFL (EP13 of .MIOS). This ensures that the accounting file cannot be accessed by any other program until it is reset.

CALLING SEQUENCE

ACTS1 is called from the courtesy call routine in ACTFL which is executed by GEPOP. ACTFL has encountered an end of tape. Return is to the first instruction following the call.

```
8          16
.CALL      .MACTS,1
return 0
```

OPERATING SYSTEM INTERACTION

No stack or .STEMP storage is used.

The current block serial number in .CRACF is reset to zero when the tape exchange is completed.

POSTCALLING SEQUENCE

The SCT referred to by bits 7-17 of .CRACF has assigned status and contains the physical location of the current accounting tape. If an exchange was made, the previously assigned physical device SCT has unassigned status.

ROUTINE RETURNS

Return 0 (.EXIT) Processing completed. Register status:

AR Destroyed
QR Destroyed
X0 through X4 Destroyed
X5 through X7 Restored

SUPPORTING INFORMATION

Programming Method

ACTS1 is reentrant, written in floatable code, and operates in the slave service area or in the hard core monitor.

Only one copy of the routine can be executing the actual tape switching; this is controlled by the .CRACF gate and the communications cells.

Storage

ACTS1, ACTS2, and ACTS3 occupy approximately 300 core storage locations.

Internal temporary storage is used while the .CRACF gate is closed.

Other Routines Used

Assign an I/O Entry, QUEUE (EP4 of .MIOS)
Link I/O in Front of Queue, LINKF (EP2 of .MIOS)
Master Message Processor, ITYM (EP7 of .MIOS)
Process Specific Channel Request, ENT2 (EP2 of .MALC2)

Flowchart

See CPB-1500 for the flowchart of ACTS1, .MACTS module.

ERROR CLOSE OF ACCOUNTING FILE

ACTS2 (EP2 of .MACTS) closes the current accounting file preparatory to performing tape switching as a result of encountering an error. ACTS2 is called when ACTFL encounters an unrecoverable error while writing on the current collector tape. A typewriter message notifies the operator of the error and states the number of good records written on the tape. Normal tape closing is then performed (see ACTS1).

With the exception of the calling sequence, all parameters for ACTS2 are the same as ACTS1 and, therefore, are not repeated here.

CALLING SEQUENCE

ACTS2 is called from the courtesy call routine in ACTFL (EP13 of .MIOS) which is executed by GEPOP. ACTFL has encountered an unrecoverable error of the current collector tape. Return is to the first instruction following the call.

```
8       16
.CALL   .MACTS,2
return 0
```

SUPPORTING INFORMATION

Flowchart

See CPB-1500 for the flowchart of ACTS2, .MACTS module.

EXTERNAL REQUEST FOR ACCOUNTING FILE CLOSE

ACTS3 (EP3 of .MACTS) processes external requests for accounting tape switching. The tape writing flag in .CRACF is checked. If the flag is zero, it is set to nonzero and the closing procedure is performed. Before exiting, the tape writing flag is reset to zero. If the tape writing flag is nonzero, ACTS3 waits until the flag is reset to zero by the courtesy call routine in ACTSFL (EP13 of .MIOS) before the closing procedure is begun.

PRECALLING SEQUENCE

Prior to entering ACTS3, the registers listed must contain the data indicated.

X5	LAL for program
X6	Program number
X7	Processor number

CALLING SEQUENCE

ACTS3 is called by an external request.

```
8        16
.CALL    .MACTS,3
```

OPERATING SYSTEM INTERACTION

No stack or .STEMP storage is used.

The .CRACF gate is shut while the tape writing flag is interrogated.

The current block serial number in .CRACF is reset to zero when the tape exchange is completed.

POSTCALLING SEQUENCE

The SCT referred to by bits 7-17 of .CRACF has assigned status and contains the physical location of the current accounting tape. If an exchange was made, the previously assigned physical device SCT has unassigned status.

ROUTINE RETURNS

Return 0 (.EXIT) Processing requested by an external program is completed. Register status:

AR	Destroyed
QR	Destroyed
X0	through X7 are restored

SUPPORTING INFORMATION

ACTS3 is reentrant, written in floatable code, and operates in the slave service area of hard core.

Interrupts are inhibited while the .CRACF gate is shut.

Only one copy of the routine can be executing the actual tape switching; this is controlled by the .CRACF gate and the communication cells.

Storage

ACTS1, ACTS2, and ACTS3 occupy approximately 300 core storage locations.

Internal temporary storage is used while the .CRACF gate is closed.

Other Routines Used

Assign an I/O Entry, QUEUE (EP4 of .MIOS)
Link I/O in Front of Queue, LINKF (EP2 of .MIOS)
Master Message Processor, ITYM (EP7 of .MIOS)
Process Specific Channel Request, ENT2 (EP2 of .MALC2)

5. REMOTE I/O SUPERVISION

Remote input/output supervision within the GE-625/635 GECOS-III Comprehensive Operating Supervisor is performed by the following two modules:

- .MDNET Remote Interrupt Processor
- .MROUT Remote Access MME Processor

The module .MDNET processes the interrupts that occur on DATANET-30 communications processor channels as a result of some activity at a remote terminal. In addition to processing the interrupts, .MDNET makes certain checks on the validity of the commands sent to the DATANET-30 communications processor. Response to some action at a remote terminal by a specific program within the GECOS-III operating environment is formulated by developing the proper calling sequence to a MME GEROUT for the action required. The module .MROUT performs this function.

REMOTE INTERRUPT PROCESSOR MODULE (.MDNET)

The Remote Interrupt Processor Module (.MDNET) is one of the device-dependent modules called upon by the main IOS module, .MIOS, to interface with GERTS/30, the program used to drive the DATANET-30 communications processor. In addition, .MDNET interfaces with a number of GECOS-III modules: .MRGIN (Remote GEIN); .MROUT (Remote Access I/O to GECOS-III); .MGEOT (SYSOUT Disperser); and a portion of program GEPOP. Hence, in a sense, .MDNET and its related modules can be considered a subsystem within GECOS-III to handle the problems associated with remote processing.

Within the remote subsystem are two general modes of operation:

- Direct Access
- Batch

When a terminal communicates directly with a slave program it is considered direct access processing. The Time-Sharing System is a slave program that operates in this mode.

Batch mode on the other hand consists of reading of a job input (under control of .MRGIN) and printing of subsequent job output (under control of .MGEOT) in addition to normal user-initiated requests for status or program aborts initiated from a remote terminal. Both .MRGIN and .MGEOT are designated for remote terminals. The module .MDNET selects the I/O and processes the interrupts that occur on DATANET-30 communications processor channels for direct access and batch modes of operation.

A special sequence of commands must be executed in order to perform remote I/O. Data passed to and from the DATANET-30 communications processor always consists of two parts: an intercomputer message (ICM) and data. The ICM describes the nature of the actual data that is to follow. The receiving computer can either reject or accept this data. The format of the ICM consists of 12 six-bit characters defined as indicated below:

0	1	2	3	4	5	6	7	8	9	10	11

where the character fields define the following:

0-1 Station ID
2 Line number expressed in binary
3 Terminal type

4 Hardware status when terminal is DATANET-760:

01 - NAK: transmission to/from DATANET-760 failed after 16 attempts

02 - PRT: print button depressed on DATANET-760

5 ICM operation code defined as follows:

From GE-625/635:

00 Input accepted
01 Accept output
02 Accept final output
03 Output not available
04 Terminate input
05 Accept no more calls
06 Accept calls

From DATANET-30:

07 Terminate all calls
10 Accept direct output
11 Accept direct output, wait for output
12 ICM rejection
13 Wait for output
14 DAC output, prepare for PPT input
15 Accept DAC PPT input
24 Final output accepted
25 Accept input
26 Accept input, last of current SNUMB
27 Send output
30 Backspace output
31 Break condition
32 Send status
33 Abort (job, activity)
34 Connect to slave program
35 Accept direct input
36 Line disconnect
37 ICM rejection

6 DATANET-760 screen number
7-8 Word count (number of six-character words of data following this ICM expressed in binary; must always contain at least one word of data).

9 Reject reason if the ICM operation code in character 5 indicates ICM reject:

01 Undefined or inconsistent line number
02 Undefined operation code
03 Undefined or inconsistent station ID
04 Undefined or inconsistent word count
05 Checksum error

10 Not used

11 Checksum (exclusive OR of characters 0-10)

When sending information to the DATANET-30 communications processor .MDNET does the following:

- Issues a write command with the DCW(s) pointing to the ICM and the subsequent data.

- Determines if the ICM is accepted without error by examining the status return occurring with a terminate interrupt. (Status return is channel ready if accepted; status return is intermediate major status if the DATANET-30 communications processor wishes to reject the ICM.)

If the status return is channel ready, the output operation is considered complete. If not, .MDNET initiates a read sequence to the DATANET-30 communications processor to determine the reject reason. The read sequence consists of the following:

- A special interrupt on a DATANET-30 communications processor channel triggers the read sequence and causes .MDNET to read the ICM.

- Acceptance or rejection of the data following the ICM is determined by .MDNET after a terminate interrupt and intermediate status following the read command.

Acceptance causes a second read command to be issued with the second DCW constructed to accept the number of words indicated by the ICM. The second read normally terminates channel ready.

Rejection of the ICM by .MDNET results in the preparation of an ICM with a reject operation code, and a write command to the DATANET-30 communications processor. A point to remember is that the GE-625/635 system or the DATANET-30 communications processor will not reject an ICM with a reject operation code, but will accept the ICM then retry the original write command. Issuance of a write command from the GE-625/635 takes precedence over a write command received simultaneously from the DATANET-30 communications processor.

The .MDNET module will handle a maximum of three DATANET-30s controlling up to 31 lines each. To do this .MDNET maintains a set of tables to describe what lines are performing what process in conjunction with what program. These tables are defined and maintained by .MDNET, and formats are described along with other system tables in GE-625/635 GECOS-III Introduction and System Tables, CPB-1488.

A driving force in .MDNET is the line interface mode, an element of the table .CRMSC, used among other things to determine the legality of ICM operation codes from the DATANET-30. A line is in one of the interface modes for the following reasons:

0 Disconnect	When line is disconnected it is initially in this mode.
1 Idle	A line goes into Idle mode during the period of time between the acceptance of an Accept Input, Last of Current SNUMB ICM and a subsequent Accept Input ICM, Disconnect ICM, Send Output ICM, or one of the other ICM-designated operations.
2 Wait for Activity Termination	Activity for this line has terminated and line table is waiting for I/O to complete. When I/O is complete, the line will be put back in a Connect to Slave or Disconnect line mode.
3 Direct Access	A line is in Direct Access mode when the terminal to which it is connected communicates directly with a slave program. When the slave program issues an Accept Direct Output, Wait for Input ICM, two things happen: ● Upon completion of the write to the DATANET-30 communications processor the slave program is treated as if no input is pending as far as I/O is concerned. ● When the terminal responds with the input, the slave program awaiting input may have been swapped out of core, resulting in a rejection of the ICM. However, the second try in response to the ICM rejection should result in success.
4 GEOUT	A line in the GEOUT mode is essentially sending batch output or job status to a remote terminal. It should be noted that GEOUT need only be concerned with obtaining a block of output for one line at a time per DATANET-30 communications processor, but more than one line at a time can be in the GEOUT mode.

5 Remote GEIN	A line is in the Remote GEIN mode when a job input is being read into .MRGIN from the DATANET-30. No other job input will be sent over this line until the GE-625/635 accepts the input and responds with an Input Accepted ICM. Many lines can be in this mode at the same time, since .MRGIN is written to multiplex buffers and accepts several inputs at the same time.
6 Time-Sharing	This mode is essentially the same as Direct Access as far as line discipline is concerned. However, Time-Sharing is a special direct access program, requiring special queues for communication and may or may not be enabled by .MDNET, depending upon operator option.
7 Connect to Slave	A line is in this mode when a terminal has called in and requested to be connected to a slave program but the slave program has not executed a corresponding remote inquiry. If the slave program is Time-Sharing, .MDNET will either enable Time-Sharing, make the connection because Time-Sharing is already enabled, or inform the user at the terminal that Time-Sharing is not available and disconnect him. However, if the slave program is not Time-Sharing, there is no communication with the terminal telling the user where the program is, unless there is a program in the system that will do a remote inquiry. If such is the case (no program in the system to do remote inquiry), the particular line is in a sort of limbo waiting for something to happen.
8 Not used	
9 GEPOP Processing ABORT Request	A line is in this mode when a request to abort a program has occurred and program GEPOP is in the process of issuing the abort. The line goes into Idle mode when status regarding the abort has been transmitted to the terminal.

The Remote Interrupt Processor (.MDNET) contains 15 entry points (EP1-EP15) designated as follows:

● RHNDLR (EP1)	Remote Interrupt Handler
● DNIO (EP2)	I/O Request Routine
● D30SEL (EP3)	Channel Select Routine
● GEEPR (EP4)	GEPR Entry Point
● IEND (EP5)	Location containing time allowed before interrupt is considered lost. This time is set at 15 seconds.

● IEND (EP6)	Multirecord command allowed/not allowed
● IEND (EP7)	Bits 0-17 contain device name; 18-34 are not used; bit 35 contains 1 if device command is to be typed in GEPR message.
● IEND (EP8)	Accumulated IOC time for device type
● DNGEPQ (EP9)	DATANET-30 channel GEEPR flag
● SWAP (EP10)	Called by .MSWAP to inform .MDNET that a program waiting to connect to slave is now back in execution.
● .RLOG (EP11)	Contains pointer to remote logical channel designator
● .RRDIQ (EP12)	Contains address of pointer to absolute address of assigned GEPOP I/O queue
● .RGINQ (EP13)	Contains address of pointer to I/O queue entry used by Remote GEIN to read DATANET-30 output
● .RSOQ (EP14)	Contains address of pointer to SYSOUT queue indexed by DATANET-30
● .RINQ (EP15)	Contains pointer to direct access remote inquiry queue

Entry points 11-15 define the location of various tables that must be referenced by other modules. These tables are set up and maintained by .MDNET but may be accessed by other modules for information. The development of an address takes place in the calling module. Entry points 11-15 are consistent with entry points defined by IOS in general.

REMOTE INTERRUPT HANDLER

RHNDLR (EP1 of .MDNET) processes all interrupts, initiates any error recovery procedures that may be required, and selects any I/O for each configured DATANET-30 communications processor channel. To do this, RHNDLR maintains a set of line interface tables for each DATANET-30 communications processor containing the current status of each channel.

There is also a well-defined interaction between Remote GEIN (.MRGIN), Remote SYSOUT (.MEGOT), program GEPOP, Time-Sharing System (.MTIMS), and any slave program requesting direct access and .MDNET, since each system requires special processing for queuing of requests and handling of any subsequent I/O. RHNDLR handles these functions.

PRECALLING SEQUENCE

Prior to entering RHNDLR, the following registers must contain the data indicated:

Q Status word 1 (.CRSM,1 contains status word 2). Status word 1 is for initiation and terminate interrupts only. Bit usage is as follows:

0	Sync bit (1)
1	Power bit (1 indicates power off)
2-5	Major status
6-11	Substatus
12-15	Must be zero
16	Type of interrupt (1 indicates terminate interrupt)
17	Abort condition (1 indicates bad status)
18-23	IOC status
24-29	Must be zero
30-35	Record count residue

X0 Type of interrupt (TI=0, II=2, SI=4)

X1 True channel index (IOC*256+4*CHN0 for active channel)

X2 Logical channel index (IOC*256+CHN0 for logical channel)

X3 Status. Bits are defined as follows: 0 = 0, 1 = power, 2-5 = major status, 6-17 = 0.

X4 Absolute address of active I/O entry. (TI and II only)

X5 LAL for GEPOP (need for .CALL, .EXIT, etc.)

X6 Program number of I/O entry (TI and II)

X7 Control processor number

CALLING SEQUENCE

RHNDLR is called from IOTRM located in the .MIOS module.

```
        8            16
   -----------------------------------------------
   |    LDA        |  1,DU
   |    STC1       |  .SSTAK, 5*
   |    TRA        |  .CRCT 4,2*
   |               |
   |    return 0   |  Status return
   |    return 3   |  No action
   |    return 4   |  No I/O queue entry
   |    return 5   |  Special interrupt, start I/O
```

OPERATING SYSTEM INTERACTION

The top entry in the stack is the IC and I of the call for the returns.

No .STEMP storage is used.

The following gates are used:

.CRD30	For tables in .MDNET that pertain to DATANET-30 communications processor information. (It is also used by .MROUT.)
.CRLAL-1	Gates LAL entries. Allows .MDNET to determine whether a slave program is being aborted, enabled or is swappable.
.CRPOQ	Gates GEPOP's queue, allowing entries to be put into the queue to enable .MRGIN.
.CRQGT	Gates I/O queues. Used to unlink an I/O queue for a line for which a disconnect has been received. (This occurs only when the line was in direct access mode.)
.CRRGQ	Gates the .MRGIN queue when putting in new line requests or disconnects.
.CRDSP	Dispatcher gate. Used when determining if a program is in memory and when putting a program into the Dispatcher queue after breaking relinquish.

ROUTINE RETURNS

Return 0 Normal status return. Registers contain:

X1	True channel index.
X2	Logical primary channel index.
X4	I/O entry address
X5	LAL for GEPOP
X6	Program number of I/O entry
X7	Processor number

Return 1 (.EXIT 1) GEPR exit. Registers same as for return 0.

Return 3 (.EXIT 3) No action. Registers contain:

X5	LAL for GEPOP
X7	Processor number

Return 4 No I/O entry to return. Registers contain:

X1	True channel index
X2	Logical primary channel index
X4	I/O entry address
X5	LAL for GEPOP
X7	Processor number

Return 5 (.EXIT 5) Special interrupt, start I/O. Registers contain:

X1	True channel index
X2	Logical primary channel index
X5	LAL for GEPOP
X7	Processor number

POSTCALLING SEQUENCE

None.

SUPPORTING INFORMATION

Programming Method

RHNDLR is nonreentrant, since .IOTRM of .MIOS uses a software gate to ensure the processing of one interrupt at a time. RHNDLR is written in floatable code.

Storage

No .STEMP storage is used.

RHNDLR occupies approximately 1026 core storage locations.

Other Routines Used

Enable Program, ENB (EP6 of .MDISP)
Channel Select Routine, D30SEL (EP3 of .MDNET)

I/O REQUEST

DNIO (EP2 of .MDNET) checks for the legality of a request status or reset status command for the DATANET-30 communications processor in the calling sequence to a MME GEINOS. This MME must have been issued by GEPOP.

PRECALLING SEQUENCE

Prior to entering DNIO the following registers must contain the data indicated:

X1	SCT address
X3	Select sequence address (command 1)
X4	Address of I/O queue
X5	LAL of requesting program
X6	Program number
X7	Processor number

CALLING SEQUENCE

DNIO is called from INOS located in the .MIOS module.

```
8          16
LDA        2,DU
STL1       .SSTAK, 5*
TRA        .CRCT4,2*

Return 2
Return 8
```

OPERATING SYSTEM INTERACTION

The top entry in the stack is the IC and I of the call for the return.

No gates are used.

ROUTINE RETURNS

Return 2 Force GEPR override. Registers contain:

X3	Address of command
X4	I/O entry address
X5	LAL for GEPOP
X6	Program number
X7	Processor number

Return 6	Illegal command, abort K8. Registers contain:

X3	Address of command
X4	I/O entry address
X5	LAL for GEPOP
X6	Program number
X7	Processor number

POSTCALLING SEQUENCE

None.

SUPPORTING INFORMATION

Programming Method

DNIO is nonreentrant and written in floatable code.

Storage

DNIO occupies approximately 14 core storage locations.

Other Routines Used

None.

CHANNEL SELECT

D30SEL (EP3 of .MDNET) selects the next I/O to perform and makes the final preparations in the I/O queue before issuing a connect to the indicated DATANET-30 communications processor channel.

PRECALLING SEQUENCE

Prior to entering D30SEL the following registers must contain the data indicated:

X0 Address of return
X2 Logical primary channel index

In addition, the cell .RNXRS (if nonzero) contains the absolute address of the I/O queue and the type I/O to start as indicated by the format below:

0 17	18 35
I/O Address	I/O Type

CALLING SEQUENCE

D30SEL is called from STIO located in the .MIOS module.

```
8          16
LDA        3,DU
ADLA       .CRCT4,2
TSX0       Q,AU

Return     Start I/O
Return     Don't start I/O
```

OPERATING SYSTEM INTERACTION

The top entry in the stack is the IC and I of the call for the return.

No .STEMP storage is used.

The following gate is used:

.CRQGT	This gate is closed when D30SEL is in execution. Opening and closing is a function of the main level interrupt processor IOTRM (EP1 of .MIOS), not of the .MDNET module.

ROUTINE RETURNS

Return (TRA 1,0) Start I/O return. Registers contain:

X1	True channel index
X2	Logical primary channel index
X4	I/O entry address

Return (TRA 2,0) Do not start I/O return.

POSTCALLING SEQUENCE

None.

SUPPORTING INFORMATION

Programming Method

D30SEL is nonreentrant and must not be executed at the same time as RHNDLR (EP1 of .MDNET), as both routines use the cell .RNXRS. D30SEL is written in floatable code.

Interrupts are inhibited.

Storage

No temporary storage is used.

D30SEL occupies approximately 171 core storage locations.

Other Routines Used

None.

PROCESS DATANET-30 ERROR STATUS RETURN

GEEPR (EP4 of .MDNET) begins processing an error status return received from a channel with a DATANET-30 communications processor configured.

This entry point has two functions:

- To process lost interrupts
- To process GEPR requests

Which of the two functions is to be performed is determined by a test of the contents of location .CRPRG,7. If the contents are zero, it indicates lost interrupts; nonzero indicates GEPR is in control.

PRECALLING SEQUENCE

Prior to entering GEPR the registers listed must contain the data indicated:

For lost interrupts:

X1	True channel index
X2	Logical channel index
X4	Location of I/O queue or contains a -1, indicating that the channel was marked busy to allow the DATANET-30 to time out
X5	LAL for program GEPOP
X6	Program number of I/O queue entry (undetermined when C(X4) = -1)
X7	Processor number

For GEPR requests:

X2	Logical channel index
X3	Location of I/O queue in question
X5	LAL of program with I/O queue
X7	Processor number

CALLING SEQUENCE

GEEPR is called from IOTRM of .MIOS for lost interrupt and from .MGEEPR for GEPR processing.

```
       8       16
       LDA     4,DU
       STC1    .SSTAK,5*
       TRA     .CRCT4,2*
```

OPERATING SYSTEM INTERACTION

The top entry in the stack is used for the IC and I of the call.

No .STEMP storage is used.

No gates are used.

ROUTINE RETURNS

Return is to the .MGP12 module for GEEPR actions:

8	16
GOTO	.MGP12

Return is to the .MIOS module when there is no I/O queue location and the DATANET-30 is forced to time out:

8	16
.EXIT	4

Return is to RHNDLR (EP1 of .MDNET) to process the bad interrupt:

8	16
TRA	RHNDLR,$

POSTCALLING SEQUENCE

None.

SUPPORTING INFORMATION

Programming Method

GEEPR is reentrant and written in floatable code as part of .MDNET.

Storage

No temporary storage is used.

GEEPR uses approximately 11 core storage locations.

Other Routines Used

None.

CHECK FOR WAITING CONNECT TO SLAVE

SWAP (EP10 of .MDNET) determines if any lines are in the Connect to Slave mode waiting to connect to the calling program. If a line is in this mode, the connection is made; and the line is placed in the Direct Access/Time-Sharing mode.

PRECALLING SEQUENCE

Prior to entering SWAP, the registers listed must contain the data indicated:

X1	Relative location of remote inquiry queue entry (obtained from the lower half of location .SREMT)
X5	LAL of calling program
X6	Calling program number
X7	Processor number

CALLING SEQUENCE

SWAP is called from the .MSWAP module:

```
       8        16
       LXL1     .SRMENT,5
       TZE      NOCALL,$
       .CALL    .MDNET,10
NOCALL NULL
```

OPERATING SYSTEM INTERACTION

The top of the stack contains the IC and I of the call for the return.

No .STEMP storage is used.

The .CRD30 gate is shut when scanning the .CRMSC table.

ROUTINE RETURNS

Return is to the .MSWAP module via the .EXIT macro.

POSTCALLING SEQUENCE

None.

SUPPORTING INFORMATION

Programming Method

SWAP is nonreentrant and written in floatable code.

Interrupts are inhibited.

Storage

No temporary storage is used.

SWAP occupies approximately 23 core storage locations.

Other Routines Used

None.

INITIALIZE .MDNET MODULE

The .IDNET routine performs the startup initialization when the .MDNET module is loaded into core storage.

PRECALLING SEQUENCE

Prior to entering .IDNET, the registers listed must contain the data indicated:

X0 Address of bootstrap card reader description

X2 Address of interrupt vector image

X3 Address of fault vector image

X4 Address of real-time IOC mailbox images

CALLING SEQUENCE

This routine is called from Startup (.MINIT). The return is to the first instruction following the TSX.

```
 8        16
 TSX1    .IDNET
 Return
```

OPERATING SYSTEM INTERACTION

None.

ROUTINE RETURNS

Return (0,1) Registers contain:

AR Zero (no required modules)

QU Next available load address

X0 through X5 are destroyed

POSTCALLING SEQUENCE

None.

SUPPORTING INFORMATION

Programming Method

The .IDNET routine is nonreentrant and written in floatable code.

Interrupts are not inhibited.

Storage

No internal temporary storage is used.

This initialization routine occupies approximately 22 core storage locations; however, it is present only at Startup time. The next available load address in QU allows Startup to overlay .IDNET.

Other Routines Used

None.

REMOTE ACCESS MME PROCESSOR MODULE (.MROUT)

The module designated .MROUT is the device-dependent module that controls remote access I/O from DATANET-30 communications processor channels to GECOS-III. The DATANET-30 communications processor is in turn tied to various remote stations (teletypewriter, GE-115). Internally, .MROUT works with Interrupt Processor), .MIOS (Input/Output Supervisor), and .MBRT1 (Activity Terminator).

User entry into .MROUT is via one entry point designated GROUT using a calling sequence initiated by a MME GEROUT, followed by a three-word call. The 17 possible actions currently defined by the calling sequence accomplish the following:

- Respond to a remote inquiry by performing the necessary functions of checking station ID, program name, valid operation code, and type of access. Any illegal activities attempted are aborted.
- Set up PAT pointers and entries for different types of I/O and generate I/O queue entries in preparation for DATANET-30 communications processor I/O.
- Link, unlink, and fill I/O queue entries, via .MIOS, for DATANET-30 communications processor requests.

The entry point into .MROUT is designated as follows:

- GROUT (EP1) .MROUT MME GEROUT Processor

It is described on the following pages.

.MROUT MME GEROUT PROCESSOR

GROUT (EP1 of .MROUT) performs the functions designated by the operation code field of the VFD pseudo operation in the calling sequence to a MME GEROUT. There are 17 defined operations at this time for dealing with I/O through a DATANET-30 communications processor to a terminating station.

PRECALLING SEQUENCE

None.

CALLING SEQUENCE

The calling sequence for the functions defined by GROUT has the following general form:

```
8          16
MME        GEROUT
VFD        18/A, 06/Operation code, H12/Station ID
ZERO       status return, cc
Normal return
```

where:

ZERO can vary depending upon operation code.
cc is courtesy call address in users program when I/O is complete.

The fields of the VFD pseudo operation are defined as follows:

A   Defined by operation code field (See below.)

Operation Code:

00-02 Invalid, not assigned.
03 Terminal direct access output. Calling sequence is:

```
8          16
MME        GEROUT
VFD        18/A, 06/3, H12/Station ID
ZERO       status return, cc
```

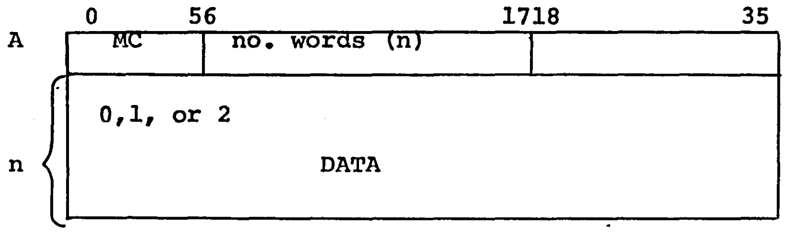
where:

A (VFD instruction) is the location of data in the following form:

		0 17	18 35
For TTY	A	no. words (n)	0
	A+1	no. characters (≤4n)	
n≤78	n	DATA	

For GE-115

n≤319



04 Terminal direct access output, then input. Calling sequence is:

```
8           16
MME         GEROUT
VFD         18/A, 06/4, H12/Station ID
ZERO        status return, cc
Normal return
```

The format of A (VFD) is the same as operation code 03, but word-1 bits 18-35 contain the address of an input area.

05 Remote inquiry. Calling sequence is:

```
8           16
MME         GEROUT
VFD         18/0, 06/5, 12/0
BCI         1, Name
Normal return
```

Bits 24-35 of the VFD field are set to station ID if a match is made.

The execution of a remote inquiry call is considered I/O because control can be relinquished or "roadblocked" until a terminal calls in and creates a match.

"Name" of BCI is a six-BCD-character name to be used for linking a program with a line, using the same name as the direct access channel name.

06 Remote terminal type. Calling sequence is:

```
8           16
MME         GEROUT
VFD         6/Type, 12/0, 06/6, H12/Station ID
Return      Terminal not connected
Normal return
```

"Type" field of the VFD instruction is set to one of the following codes when a terminal exists of the specified station ID:

00	1004, voice-grade line(s)
01	GE-115, voice-grade line(s)
02	not used
03	GE-115, Telpak A line(s)
04	teletypewriter
05	DATANET-760, 4-line screen
06	DATANET-760, 8-line screen
07	DATANET-760, 16-line screen
10	DATANET-760, 26-line screen

07 Undefined at this time.

10 Input accepted with respect to Accept Input, Last of Current SNUMB (Remote GEIN). Calling sequence is:

```
8         16
MME       | GEROUT
VFD       | 18/0, 06/10, H12/Station ID
ZERO      | status return, cc
Normal return
```

11 Terminate input (Remote GEIN). Calling sequence is:

```
8         16
MME       | GEROUT
VFD       | 18/0, 06/10, H12/Station ID
ZERO      | status return, cc
Normal return
```

Q register bits 0-17 contain a three-character reason code.

12 Input accepted (Remote SYSOUT). Calling sequence is:

```
8         16
MME       | GEROUT
VFD       | 18/A, 06/12, H12/Station ID
ZERO      | status return, cc
Normal return
```

where:

A (VFD field) points to the input buffer to be used for the next block of input data from the DATANET-30 communications processor.

13 Output not available (Remote GEOUT). Calling sequence is:

```
8         16
MME       | GEROUT
VFD       | 18/0, 06/13, H12/Station ID
ZERO      | status return
Normal return
```

14 Accept final output (Remote GEOUT). Calling sequence is:

```
8          16
MME        GEROUT
VFD        18/A, 06/15, H12/Station ID
ZERO
Normal return
```

where:

A (VFD field) points to a string of DCWs that describe records to be outputted. DCWs are IOTDs that point to the data portion of each logical record in the block to be transmitted.

15 Accept output (Remote GEOUT). Calling sequence and comments are the same as for operation code 14, above.

16 Unlink Remote GEIN output queue. Calling sequence is:

```
8          16
MME          GEROUT
VFD        18/A, 06/16, H12/Station ID
Not used
Normal return
```

where:

A (VFD field) contains the DATANET-30 communications processor index (0, 1, 2).

17 Line disconnect. Calling sequence is:

```
8          16
MME        GEROUT
VFD        18/0, 06/17, H12/Station ID
ZERO       status return
Normal return
```

20 Current line status. Calling sequence is:

```
8          16
MME        GEROUT
VFD        18/0, 06/17, H12/Station ID
ZERO       status return
Normal return
```

where:

A (VFD field) is set to a terminal-type code when a terminal exists of the specified station ID. (See "Type" codes at top of previous page.)

"Status return" (ZERO) is set to "1" in the following bits for the reasons indicated:

31 Terminal idle
32 Direct access I/O in progress for that terminal
34 Terminal disconnected
35 Break condition at remote terminal

21 Direct access output/prepare for PPT input. Calling sequence is the same as for operation code 04.

22 Accept direct access PPT input. Calling sequence is:

```
8          16
MME        GEROUT
VFD        18/A, 06/11, H12/Station ID
ZERO       status return
Normal return
```

where:

A (VFD field) is the address where the next block of PPT input is to be transmitted. The first block of PPT input is defined by operation code 21.

23 Wait for output (Remote GEOUT). Calling sequence is the same as operation code 13.

Station ID

A two-character identification that defines the station or terminal used in the "HELLO" sequence by the terminal operator. Each code is checked for uniqueness by the DATANET-30 communications processor.

The status return field of the ZERO pseudo-op points to one word in the user program where status is to be returned upon completion of the I/O involved.

OPERATING SYSTEM INTERACTION

The top entry in the stack is the IC and I of the call for the return as well as the reference to the parameters that follow the call (MME).

.STEMP, +1, +2, +3, +4, +5, and +8 are used for temporary storage.

The .CRD30 gate is used to gate the DATANET-30 communications processor tables defined and used in GEEPR (EP4 of .MDNET).

ROUTINE RETURNS

Return (.EXIT) Status return action. Registers contain:

X1 True channel index
X2 Logical primary channel index
X4 I/O entry address
X5 LAL for GEPOP
X6 Program number
X7 Processor number

The IC and I in the stack are incremented according to the calling sequence prior to .EXIT.

POSTCALLING SEQUENCE

None.

SUPPORTING INFORMATION

Programming Method

GROUT is reentrant and written in floatable code. This module is also an HCM routine and MME processor.

Storage

No internal temporary storage is used.

GROUT occupies approximately 768 core storage locations.

Other Routines Used

Terminate Error Entry, FALT (EP3 of .MBRT1)
Link I/O to End of Queue, LINK (EP1 of .MIOS)
Assign an I/O Entry, QUEUE (EP4 of .MIOS)
GEPR Entry Point, GEEPR (EP4 of .MDNET)
Time Allowed Before Interrupt Lost Location, IEND (EP5 of .MDNET)
Multirecord Command Allowed/Not Allowed Location, IEND (EP6 of .MDNET)
BCD Device Name, Command Location, IEND (EP7 of .MDNET)

GLOSSARY

EOF	End of File
EP	Entry Point
GECOS	GE-625/635 Comprehensive Operating Supervisor
GEPR	Exception Processor
GESPEC	Request to delay I/O until special interrupt occurs
HCM	Hard Core Monitor
II	Initiation Interrupt
I/O	Input/Output
IOC	Input/Output Controller
IOS	Input/Output Supervisor
LAL	Lower Address Limit
LLINK	Little Link
.MACTS	Accounting Tape Switching module
.MCPIO	Card Punch channel module
.MDR20	Magnetic Drum Subsystem channel module
.MDS20	Disc Storage Subsystem channel module
.MGPIO	Card Reader channel module
.MIOS	Main IOS Module
.MMTAP	Magnetic Tape channel module
.MPRIO	Printer channel module
.MPTAP	Paper Tape channel module
.MTYPE	Typewriter channel module
PMX	Primary Mailbox
SA	Slave Area
SCT	System Configuration Table
SI	Special Interrupt
SMX	Secondary Mailbox
SSA	Slave Service Area
TI	Termination Interrupt

INDEX

ABORT
  ABORT I/O FOR PROGRAM 56
  Terminate GEPR Abort ABREQ 109

ABREQ
  Terminate GEPR Abort ABREQ 109

ABTIO
  ABTIO 6
  ABTIO (EP9 of .MIOS) 56

ACCOUNTING
  accounting tape switching module 2
  FORMAT ERROR ACCOUNTING RECORD 33
  Accounting File Request ACTFL 34
  Format Error Accounting Record (FMTAR) 37
  ACCOUNTING FILE REQUEST 62
  Accounting Tape Switching ACTS1 64
  Accounting Tape Switching Module (.MACTS) 181
  NORMAL CLOSE OF ACCOUNTING FILE 182
  accounting tape switching 182
  ERROR CLOSE OF ACCOUNTING FILE 184
  EXTERNAL REQUEST FOR ACCOUNTING FILE CLOSE 185

ACCUMULATE
  Accumulate Processor Time DACNB 37

ACTFL
  ACTFL 6
  ACTFL (EP13 of .MIOS) 33
  Accounting File Request ACTFL 34
  ACTFL (EP13 of .MIOS) 62
  ACTFL (EP13 of .MIOS) 182
  ACTFL (EP13 of .MIOS) 184

ACTS1
  Accounting Tape Switching ACTS1 64
  ACTS1 (EP1 of .MACTS) 181
  ACTS1 (EP1 of .MACTS) 182

ACTS2
  ACTS2 (EP2 of .MACTS) 181
  ACTS2 (EP2 of .MACTS) 184

ACTS3
  ACTS3 (EP3 of .MACTS) 181
  ACTS3 (EP3 of .MACTS) 185

ACTUATORS
  actuators 113

ADDRESS
  CCA (Courtesy Call Address) 25

ADDRESSABLE
  minimum addressable segment 113

- ADDRESSES
  - SEEK ADDRESSES 96
  - SEEK ADDRESSES 114

- ASSIGN
  - ASSIGN AN I/O ENTRY 12
  - Assign an I/O Entry QUEUE 18
  - Assign an I/O Entry QUEUE 64
  - Assign an I/O Entry QUEUE 183
  - Assign an I/O Entry QUEUE 186

- ASSIGNMENT
  - Peripheral Assignment Table (PAT) 31

- BLOCK
  - Diagnostic Block 96
  - BLOCK COUNT 96

- CALCULATE
  - Calculate Logical Primary Channel Index ILPCX 9
  - Calculate Logical Primary Channel Index ILPCX 18
  - Calculate Logical Primary Channel Index ILPCX 22
  - CALCULATE LOGICAL PRIMARY CHANNEL INDEX 43
  - Calculate Logical Primary Channel Index ILPCX 57
  - Calculate Logical Primary Channel Index ILPCX 59
  - Calculate Logical Primary Channel Index ILPCX 61

- CALCULATION
  - seek calculation routines 92
  - seek calculation routines 110

- CALL
  - .GINOS Call 15
  - CCA (Courtesy Call Address) 25
  - GEPOP hard core courtesy call 62

- CARD
  - Card Punch 2
  - Card Reader 2
  - Card Punch 67
  - Card Reader 67
  - CARD PUNCH INTERRUPT HANDLER 70
  - Card Punch Initialization .ICPIO 72
  - CARD PUNCH REQUEST 73
  - CARD PUNCH SELECT 75
  - CARD PUNCH ERROR 77
  - Card Punch Recovery GP15 78
  - CARD PUNCH INITIALIZATION 79
  - CARD READER INTERRUPT HANDLER 121
  - Card Reader Initialization .IGPIO 123
  - CARD READER REQUEST 124
  - CARD READER SELECT 126
  - CARD READER ERROR 128
  - Card Reader Recovery 1 GP09 129
  - CARD READER INITIALIZATION 130

- CARRIAGE
  - Carriage codes 24

CCA
   CCA (Courtesy Call Address) 25

CHANNEL
   channel primary System Configuration Table 1
   device-dependent channel module 1
   channel modules 2
   Calculate Logical Primary Channel Index ILPCX 9
   Calculate Logical Primary Channel Index ILPCX 18
   Channel Modules 18
   Calculate Logical Primary Channel Index ILPCX 22
   Channel Module 37
   CALCULATE LOGICAL PRIMARY CHANNEL INDEX 43
   Logical Primary Channel Index 44
   Channel Module 48
   Calculate Logical Primary Channel Index ILPCX 57
   Calculate Logical Primary Channel Index ILPCX 59
   RESUME I/O ON CHANNEL 60
   logical channel queue 60
   Calculate Logical Primary Channel Index ILPCX 61
   CHANNEL MODULES 67
   Transfer to a channel module entry point 68
   Process Specific Channel Request ENT2 183
   Process Specific Channel Request ENT2 186

CI
   Counterparity Interrupts (CI) 35

CLOSE
   NORMAL CLOSE OF ACCOUNTING FILE 182
   ERROR CLOSE OF ACCOUNTING FILE 184
   EXTERNAL REQUEST FOR ACCOUNTING FILE CLOSE 185

CODE
   Find File Code FNDFC 22
   Mesg Code 24
   FIND FILE CODE 31

CODES
   Carriage codes 24

COMMAND
   CONNECT REISSUE OF SECOND TYPEWRITER COMMAND 51

CONFIGURATION
   channel primary System Configuration Table 1
   system configuration tables (SCT) 182

CONNECT
   CONNECT MULTIRECORD SIMULATION DCW 50
   CONNECT REISSUE OF SECOND TYPEWRITER COMMAND 51
   CONNECT SELECTED GESPECED ENTRY 52

CONSOLE
   Console Recovery GP22 177

CONSTANTS
   device-dependent constants 92
   device-dependent constants 110

- CONTROL
  - Relinquish Control REL 64

- CORE
  - hard core monitor (HCM) 2
  - GEPOP hard core courtesy call 62

- COUNT
  - BLOCK COUNT 96

- COUNTER
  - instruction counter and indicators (IC and I) 17

- COUNTERPARITY
  - Counterparity Interrupts (CI) 35

- COURTESY
  - CCA (Courtesy Call Address) 25
  - GEPOP hard core courtesy call 62

- CPGP
  - CPGP (EP4 of .MCPIO) 77

- CPIO
  - CPIO (EP2 of .MCPIO) 73

- CPIT
  - CPIT (EP1 of .MCPIO) 70

- CPSL
  - CPSL (EP3 of .MCPIO) 75

- CPZ100
  - CPZ100 70

- CRCT1
  - CRCT1 142

- CRGP
  - CRGP (EP4 of .MGPIO) 128

- CRIO
  - CRIO (EP2 of .MGPIO) 124

- CRIT
  - CRIT (EP1 of .MGPIO) 121

- CRSL
  - CRSL (EP3 of .MGPIO) 126

- DACNB
  - Accumulate Processor Time DACNB 37

- DCW
  - DCW Pointer Validation DCWCK 18
  - DCWW (Write DCW Pointer) 24
  - DCWR (Read DCW Pointer) 24
  - DCW POINTER VALIDATION 29
  - CONNECT MULTIRECORD SIMULATION DCW 50

DCWCK
  DCWCK 3
  DCW Pointer Validation DCWCK 18
  DCWCK (.MIOS) 29

DCWR
  DCWR (Read DCW Pointer) 24

DCWW
  DCWW (Write DCW Pointer) 24

DEVICE
  logical device SCT 1
  device secondary SCT 43

DEVICE-DEPENDENT
  device-dependent channel module 1
  device-dependent constants 92
  device-dependent constants 110

DIAGNOSTIC
  Diagnostic blocks 96
  Diagnostic Block 96
  Diagnostic Blocks 114

DISC
  DSU200 Disc Storage Subsystem 2
  DSU200 Disc Storage Subsystem 67
  Disc Storage Subsystem (DSU200) 110

DISPATCHER
  Dispatcher (.MDISP) 36
  Dispatcher (.MDISP) 45

DRGP
  DRGP (EP4 of .MDR20) 90

DRIO
  DRIO (EP2 of .MDR20) 85
  MDS200 Interrupt Handler DRIO 91

DRIT
  DRIT (EP1 of .MDR20) 82

DRSL
  DRSL (EP3 of .MDR20) 88

DRUM
  MDS200 Magnetic Drum Subsystem 2
  MDS200 Magnetic Drum Subsystem 67
  Magnetic Drum Subsystem (MDS200) 92

DSGP
  DSGP (EP4 of .MDS20) 108

DSIO
   DSIO (EP2 of .MDS20) 103

DSIT
   DSIT (EP1 of .MDS20) 100
   DSU200 Interrupt Handler DSIT 109

DSPQH
   Program Number at Front of Queue DSPQH 37

DSPQM
   Program Number in Queue Following Interrupt DSPQM 46

DSPQT
   Program Number at End of Queue DSPQT 37

DSSL
   DSSL (EP3 of .MDS20) 106

DSU200
   DSU200 Disc Storage Subsystem 2
   DSU200 Disc Storage Subsystem 67
   DSU200 INTERRUPT HANDLER 100
   DSU200 REQUEST 103
   STORAGE SUBSYSTEM (DSU200) REQUEST 103
   DSU200 SELECT 106
   DSU200 Recovery 1 GP17 109
   DSU200 Interrupt Handler DSIT 109
   DSU200 Negative Entry Points 109
   DSU200 NEGATIVE ENTRY POINTS 110
   Disc Storage Subsystem (DSU200) 110
   DSU200 INITIALIZATION 118

ENABLE
   Enable Program ENB 134

ENB
   Enable Program ENB 134

END
   LINK I/O TO END OF QUEUE 8
   Link I/O to End of Queue LINK 18
   Program Number at End of Queue DSPQT 37

ENT2
   Process Specific Channel Request ENT2 183
   Process Specific Channel Request ENT2 186

ENTRY
  entry points (EP) 3
  ASSIGN AN I/O ENTRY 12
  Assign an I/O Entry QUEUE 18
  Unlink I/O Entry UNLNK 22
  Unlink I/O Entry UNLNK) 37
  UNLINK I/O ENTRY 41
  CONNECT SELECTED GESPECED ENTRY 52
  Unlink I/O Entry UNLNK 57
  GEPOP I/O entry 62
  Assign an I/O Entry QUEUE 64
  Transfer to a channel module entry point 68
  Terminate Error Entry FALT 91
  MDS200 Negative Entry Points 91
  MDS200 NEGATIVE ENTRY POINTS 92
  DSU200 Negative Entry Points 109
  DSU200 NEGATIVE ENTRY POINTS 110
  Assign an I/O Entry QUEUE 183
  Assign an I/O Entry QUEUE 186

EOF
  MDS200 ERROR AND EOF RECOVERY 90

EP
  entry points (EP) 3

ERROR
  FORMAT ERROR ACCOUNTING RECORD 33
  Format Error Accounting Record (FMTAR) 37
  CARD PUNCH ERROR 77
  MDS200 ERROR AND EOF RECOVERY 90
  Terminate Error Entry FALT 91
  CARD READER ERROR 128
  MAGNETIC TAPE ERROR 140
  PRINTER ERROR 152
  PAPER TAPE ERROR 163
  TYPEWRITER ERROR 176
  Error Processing (GEPR) 182
  ERROR CLOSE OF ACCOUNTING FILE 184

FALT
  Terminate Error Entry FALT 91

FAULT
  MME GEINOS Fault 15
  MME GESPEC Fault 20

FILE
  Find File Code FNDFC 22
  FIND FILE CODE 31
  Accounting File Request ACTFL 34
  ACCOUNTING FILE REQUEST 62
  maximum file unit 113
  NORMAL CLOSE OF ACCOUNTING FILE 182
  ERROR CLOSE OF ACCOUNTING FILE 184
  EXTERNAL REQUEST FOR ACCOUNTING FILE CLOSE 185

FIND
  Find File Code FNDFC 22
  FIND FILE CODE 31

FIRST
  First (Link Flag) 24

FLAG
  Master (Limits Flag) 23
  First (Link Flag) 24

FMTAR
  FMTAR 3
  FMTAR (.MIOS) 33
  Format Error Accounting Record (FMTAR) 37

FNDFC
  FNDFC 3
  Find File Code FNDFC 22
  FNDFC (.MIOS) 31

FORMAT
  FORMAT ERROR ACCOUNTING RECORD 33
  Format Error Accounting Record (FMTAR) 37
  Format I/O Status Words and Return Status GSTRT 57
  FORMAT I/O STATUS WORDS AND RETURN STATUS 58

FRONT
  LINK I/O TO FRONT OF QUEUE 10
  LINK REISSUED I/O TO FRONT OF QUEUE 11
  Program Number at Front of Queue DSPQH 37
  Link I/O to Front of QUEUE LINKF 64
  Link I/O in Front of Queue LINKF 183
  Link I/O in Front of Queue LINKF 186

GEINOS
  MME GEINOS PROCESSOR 15
  MME GEINOS Fault 15
  MME GEINOS Processor INOS 30

GEPOP
  GEPOP I/O entry 62
  GEPOP hard core courtesy call 62
  Main GEPOP module .MPOPM 181

GEPR
  GEPR reissuing I/O 33
  Main GEPR module (.MGEPR) 60
  Terminate GEPR Abort ABREQ 109
  Error Processing (GEPR) 182

GESPEC
  MME GESPEC PROCESSOR 20
  MME GESPEC Fault 20

GESPECED
  CONNECT SELECTED GESPECED ENTRY 52

GP09
  Card Reader Recovery 1 GP09 129

GP11
  Paper Tape Recovery GP11 164

GP13
  Printer Recovery 1 GP13 153

GP15
  Card Punch Recovery GP15 78

GP17
  DSU200 Recovery 1 GP17 109

GP20
  MDS200 Recovery 1 GP20 91

GP22
  Console Recovery GP22 177

GP30
  Magnetic Tape Recovery GP30 141

GSTRET
  GSTRET 45

GSTRT
  GSTRT 6
  Format I/O Status Words and Return Status GSTRT 57
  GSTRT (EP10 of .MIOS) 58

HANDLER
  I/O Request Handler 4
  Interrupt Handler (IOTRM) 5
  I/O REQUEST HANDLER 7
  Interrupt Handler (IOTRM) 33
  INTERRUPT HANDLER 35
  I/O Request Handler 40
  Interrupt Handler 40
  Interrupt Handler (IOTRM) 45
  I/O Request Handler 45
  Interrupt Handler (IOTRM) 51
  CARD PUNCH INTERRUPT HANDLER 70
  Interrupt Handler (IOTRM) 70
  MDS200 INTERRUPT HANDLER 82
  Interrupt Handler (IOTRM) 82
  MDS200 Interrupt Handler DRIO 91
  DSU200 INTERRUPT HANDLER 100
  Interrupt Handler (IOTRM) 100
  DSU200 Interrupt Handler DSIT 109
  CARD READER INTERRUPT HANDLER 121
  Interrupt Handler (IOTRM) 121
  MAGNETIC TAPE INTERRUPT HANDLER 133
  Interrupt Handler (IOTRM) 133
  PRINTER INTERRUPT HANDLER 145
  Interrupt Handler (IOTRM) 145
  PAPER TAPE INTERRUPT HANDLER 157
  Interrupt Handler (IOTRM) 157
  TYPEWRITER INTERRUPT HANDLER 168
  Interrupt Handler (IOTRM) 168

HARD
- hard core monitor (HCM) 2
- GEPOP hard core courtesy call 62

HCM
- hard core monitor (HCM) 2

II
- Initiation Interrupt (II) 33
- Initiation Interrupts (II) 35

ILPCX
- ILPCX 3
- Calculate Logical Primary Channel Index ILPCX 9
- Calculate Logical Primary Channel Index ILPCX 18
- Calculate Logical Primary Channel Index ILPCX 22
- ILPCX (.MIOS) 43
- Calculate Logical Primary Channel Index ILPCX 57
- Calculate Logical Primary Channel Index ILPCX 59
- Calculate Logical Primary Channel Index ILPCX 61

INDEX
- Calculate Logical Primary Channel Index ILPCX 9
- Calculate Logical Primary Channel Index ILPCX 18
- Calculate Logical Primary Channel Index ILPCX 22
- CALCULATE LOGICAL PRIMARY CHANNEL INDEX 43
- Logical Primary Channel Index 44
- Calculate Logical Primary Channel Index ILPCX 57
- Calculate Logical Primary Channel Index ILPCX 59
- Calculate Logical Primary Channel Index ILPCX 61

INDICATORS
- instruction counter and indicators (IC and I) 17

INITIALIZARION
- Printer Initializarion .IPRIO 147

INITIALIZATION
- INITIALIZATION SUBROUTINE 6
- .MIOS Initialization (.IIOS) 35
- MAIN IOS MODULE INITIALIZATION 65
- Card Punch Initialization .ICPIO 72
- CARD PUNCH INITIALIZATION 79
- MDS200 INITIALIZATION 97
- DSU200 INITIALIZATION 118
- Card Reader Initialization .IGPIO 123
- CARD READER INITIALIZATION 130
- MAGNETIC TAPE INITIALIZATION 142
- PRINTER INITIALIZATION 154
- PAPER TAPE INITIALIZATION 165
- TYPEWRITER INITIALIZATION 178

INITIATION
- Initiation Interrupt (II) 33
- Initiation Interrupts (II) 35

INOS
   INOS 3
   INOS (EP5 of .MIOS) 15
   INOS 27
   INOS 29
   MME GEINOS Processor INOS 30

INTERRUPT
   Interrupt Handler (IOTRM) 5
   Interrupt Handler (IOTRM) 33
   Termination Interrupt (TI) 33
   Initiation Interrupt (II) 33
   Special Interrupt (SI) 33
   INTERRUPT HANDLER 35
   Interrupt Handler 40
   Interrupt Handler (IOTRM) 45
   Program Number in Queue Following Interrupt DSPQM 46
   Interrupt Handler (IOTRM) 51
   CARD PUNCH INTERRUPT HANDLER 70
   Interrupt Handler (IOTRM) 70
   MDS200 INTERRUPT HANDLER 82
   Interrupt Handler (IOTRM) 82
   MDS200 Interrupt Handler DRIO 91
   DSU200 INTERRUPT HANDLER 100
   Interrupt Handler (IOTRM) 100
   DSU200 Interrupt Handler DSIT 109
   CARD READER INTERRUPT HANDLER 121
   Interrupt Handler (IOTRM) 121
   MAGNETIC TAPE INTERRUPT HANDLER 133
   Interrupt Handler (IOTRM) 133
   PRINTER INTERRUPT HANDLER 145
   Interrupt Handler (IOTRM) 145
   PAPER TAPE INTERRUPT HANDLER 157
   Interrupt Handler (IOTRM) 157
   TYPEWRITER INTERRUPT HANDLER 168
   Interrupt Handler (IOTRM) 168

INTERRUPTS
   handle interrupts 3
   Termination Interrupts (TI) 35
   Initiation Interrupts (II) 35
   Special Interrupts (SI) 35
   Counterparity Interrupts (CI) 35

IOS
   INTRODUCTION to IOS 1
   I/O Supervisor (IOS) 1
   MAIN IOS MODULE (.MIOS) 3
   MAIN IOS MODULE INITIALIZATION 65

IOTRM
- IOTRM 3
- Interrupt Handler (IOTRM) 5
- Interrupt Handler (IOTRM) 33
- IOTRM (.MIOS) 35
- Interrupt Handler (IOTRM) 45
- Interrupt Handler (IOTRM) 51
- Interrupt Handler (IOTRM) 70
- Interrupt Handler (IOTRM) 82
- Interrupt Handler (IOTRM) 100
- Interrupt Handler (IOTRM) 121
- Interrupt Handler (IOTRM) 133
- Interrupt Handler (IOTRM) 145
- Interrupt Handler (IOTRM) 157
- Interrupt Handler (IOTRM) 168

ITYM
- ITYM 3
- ITYM (EP7 of .MIOS) 23
- Master Message Processor ITYM 183
- Master Message Processor ITYM 186

I/O
- I/O Supervisor (IOS) 1
- I/O queues 1
- handle I/O requests 3
- I/O Request Handler 4
- I/O REQUEST HANDLER 7
- LINK I/O TO END OF QUEUE 8
- Start I/O STIO 9
- LINK I/O TO FRONT OF QUEUE 10
- LINK REISSUED I/O TO FRONT OF QUEUE 11
- ASSIGN AN I/O ENTRY 12
- I/O select sequence 15
- Link I/O to End of Queue LINK 18
- Assign an I/O Entry QUEUE 18
- Unlink I/O Entry UNLNK 22
- typewriter I/O request 23
- I/O select sequence 27
- I/O select sequence 29
- GEPR reissuing I/O 33
- Unlink I/O Entry UNLNK) 37
- Start I/O STIO) 37
- I/O Request Handler 40
- UNLINK I/O ENTRY 41
- I/O Request Handler 45
- START I/O 47
- RESUME I/O FOR PROGRAM 54
- Start I/O STIO 55
- ABORT I/O FOR PROGRAM 56
- Unlink I/O Entry UNLNK 57
- Format I/O Status Words and Return Status GSTRT 57
- Resume I/O For Program RSMIO 57
- FORMAT I/O STATUS WORDS AND RETURN STATUS 58
- RESUME I/O ON CHANNEL 60
- Start I/O STIO 61
- GEPOP I/O entry 62
- Assign an I/O Entry QUEUE 64
- Link I/O to Front of QUEUE LINKF 64
- Assign an I/O Entry QUEUE 183
- Link I/O in Front of Queue LINKF 183

I/O (cont.)
   Assign an I/O Entry QUEUE 186
   Link I/O in Front of Queue LINKF 186

LIMITS
   Master (Limits Flag) 23

LINK
   LINK 3
   LINK I/O TO END OF QUEUE 8
   LINK (EP1 of .MIOS) 8
   LINK I/O TO FRONT OF QUEUE 10
   LINK (EP1 of .MIOS). 10
   LINK REISSUED I/O TO FRONT OF QUEUE 11
   LINK (EP1 of .MIOS) 11
   LINK (EP1 of .MIOS) 15
   Link I/O to End of Queue LINK 18
   Link I/O to End of Queue LINK 18
   First (Link Flag) 24
   Link I/O to Front of QUEUE LINKF 64
   LINK 95
   LINK 114
   Link I/O in Front of Queue LINKF 183
   Link I/O in Front of Queue LINKF 186

LINKF
   LINKF 3
   LINKF (EP2 of .MIOS) 8
   LINKF (EP2 of .MIOS) 10
   Link I/O to Front of QUEUE LINKF 64
   Link I/O in Front o f Queue LINKF 183
   Link I/O in Front of Queue LINKF 186

LINKR
   LINKR 3
   LINKR (EP3 of .MIOS) 8
   LINKR (EP3 of .MIOS) 11

LINKS
   LINKs 96
   LINKs 116

LLINK
   LLINK 95
   LLINK 114

LLINKS
   LLINKs 96
   LLINKs 115

LOGICAL
   logical device SCT 1
   Calculate Logical Primary Channel Index ILPCX 9
   Calculate Logical Primary Channel Index ILPCX 18
   Calculate Logical Primary Channel Index ILPCX 22
   CALCULATE LOGICAL PRIMARY CHANNEL INDEX 43
   Logical Primary Channel Index 44
   Calculate Logical Primary Channel Index ILPCX 57
   Calculate Logical Primary Channel Index ILPCX 59
   logical channel queue 60
   Calculate Logical Primary Channel Index ILPCX 61

MAGNETIC
MDS200 Magnetic Drum Subsystem 2
Magnetic Tape 2
MDS200 Magnetic Drum Subsystem 67
Magnetic Tape 67
Magnetic Drum Subsystem (MDS200) 92
MAGNETIC TAPE INTERRUPT HANDLER 133
MAGNETIC TAPE REQUEST 136
MAGNETIC TAPE SELECT 138
MAGNETIC TAPE ERROR 140
Magnetic Tape Recovery GP30 141
MAGNETIC TAPE INITIALIZATION 142

MAIN
MAIN IOS MODULE (.MIOS) 3
Main GEPR module (.MGEPR) 60
MAIN IOS MODULE INITIALIZATION 65
Main GEPOP module .MPOPM 181

MASTER
MASTER MESSAGE PROCESSOR 23
Master (Limits Flag) 23
Master Message Processor ITYM 183
Master Message Processor ITYM 186

MDR20
MDR20 18

MDS20
MDS20 37

MDS200
MDS200 Magnetic Drum Subsystem 2
MDS200 Magnetic Drum Subsystem 67
MDS200 INTERRUPT HANDLER 82
MDS200 REQUEST 85
MDS200 SELECT 88
MDS200 ERROR AND EOF RECOVERY 90
MDS200 Recovery 1 GP20 91
MDS200 Interrupt Handler DRIO 91
MDS200 Negative Entry Points 91
MDS200 NEGATIVE ENTRY POINTS 92
Magnetic Drum Subsystem (MDS200) 92
MDS200 INITIALIZATION 97

MESG
Mesg Code 24

MESSAGE
MASTER MESSAGE PROCESSOR 23
Master Message Processor ITYM 183
Master Message Processor ITYM 186

MME
MME GEINOS PROCESSOR 15
MME GEINOS Fault 15
MME GESPEC PROCESSOR 20
MME GESPEC Fault 20
MME GEINOS Processor INOS 30

- MODULE
  - device-dependent channel module 1
  - accounting tape switching module 2
  - MAIN IOS MODULE (.MIOS) 3
  - Channel Module 37
  - Channel Module 48
  - Main GEPR module (.MGEPR) 60
  - MAIN IOS MODULE INITIALIZATION 65
  - Transfer to a channel module entry point 68
  - Accounting Tape Switching Module (.MACTS) 181
  - Main GEPOP module .MPOPM 181

- MONITOR
  - hard core monitor (HCM) 2

- MTGP
  - MTGP (EP4 of .MMTAP) 140

- MTIO
  - MTIO (EP2 of .MMTAP) 136

- MTIT
  - MTIT (EP1 of .MMTAP) 133

- MTSL
  - MTSL (EP3 of .MMTAP) 138

- MULTIRECORD
  - CONNECT MULTIRECORD SIMULATION DCW 50
  - multirecord simulation 70

- NEGATIVE
  - MDS200 Negative Entry Points 91
  - MDS200 NEGATIVE ENTRY POINTS 92
  - DSU200 Negative Entry Points 109
  - DSU200 NEGATIVE ENTRY POINTS 110

- NORMAL
  - NORMAL CLOSE OF ACCOUNTING FILE 182

- NUMBER
  - Program Number at End of Queue DSPQT 37
  - Program Number at Front of Queue DSPQH 37
  - Program Number in Queue Following Interrupt DSPQM 46
  - track number 115

- PAPER
  - Paper Tape 2
  - Paper Tape 67
  - PAPER TAPE INTERRUPT HANDLER 157
  - PAPER TAPE REQUEST 159
  - PAPER TAPE SELECT 161
  - PAPER TAPE ERROR 163
  - Paper Tape Recovery GP11 164
  - PAPER TAPE INITIALIZATION 165

PAT
   Peripheral Assignment Table (PAT) 31

PERIPHERAL
   Peripheral Assignment Table (PAT) 31

POINTER
   DCW Pointer Validation DCWCK 18
   Pointer Validation PTRVL 18
   DCWW (Write DCW Pointer) 24
   DCWR (Read DCW Pointer) 24
   SRP (Status Return Pointer) 25
   POINTER VALIDATION 27
   DCW POINTER VALIDATION 29

POSITIONING
   Tape Positioning 135

PRGP
   PRGP (EP4 of .MPRIO) 152

PRIMARY
   channel primary System Configuration Table 1
   primary SCT 1
   Calculate Logical Primary Channel Index ILPCX 9
   Calculate Logical Primary Channel Index ILPCX 18
   Calculate Logical Primary Channel Index ILPCX 22
   CALCULATE LOGICAL PRIMARY CHANNEL INDEX 43
   Logical Primary Channel Index 44
   Calculate Logical Primary Channel Index ILPCX 57
   Calculate Logical Primary Channel Index ILPCX 59
   Calculate Logical Primary Channel Index ILPCX 61

PRINTER
   Printer 2
   Printer 67
   PRINTER INTERRUPT HANDLER 145
   Printer Initialization .IPRIO 147
   PRINTER REQUEST 148
   PRINTER SELECT 150
   PRINTER ERROR 152
   Printer Recovery 1 GP13 153
   PRINTER INITIALIZATION 154

PRIO
   PRIO (EP2 of .MPRIO) 148

PRIT
   PRIT (EP1 of .MPRIO) 145

PROCESS
   Process Specific Channel Request ENT2 183
   Process Specific Channel Request ENT2 186

PROCESSING
   Error Processing (GEPR) 182

PROCESSOR
MME GEINOS PROCESSOR 15
MME GESPEC PROCESSOR 20
MASTER MESSAGE PROCESSOR 23
MME GEINOS Processor INOS 30
Accumulate Processor Time DACNB 37
Master Message Processor ITYM 183
Master Message Processor ITYM 186

PROGRAM
Program Number at End of Queue DSPQT 37
Program Number at Front of Queue DSPQH 37
Program Number in Queue Following Interrupt DSPQM 46
RESUME I/O FOR PROGRAM 54
ABORT I/O FOR PROGRAM 56
Resume I/O For Program RSMIO 57
Enable Program ENB 134

PRSL
PRSL (EP3 of .MPRIO) 150

PTGP
PTGP (EP4 of .MPTAP) 163

PTIO
PTIO (EP2 of .MPTAP) 159

PTIT
PTIT (EP1 of .MPTAP) 157

PTRVL
PTRVL 3
Pointer Validation PTRVL 18
PTRVL (.MIOS) 27

PTSL
PTSL (EP3 of .MPTAP) 161

PUNCH
Card Punch 2
Card Punch 67
CARD PUNCH INTERRUPT HANDLER 70
Card Punch Initialization .ICPIO 72
CARD PUNCH REQUEST 73
CARD PUNCH SELECT 75
CARD PUNCH ERROR 77
Card Punch Recovery GP15 78
CARD PUNCH INITIALIZATION 79

QUEUE
QUEUE 3
LINK I/O TO END OF QUEUE 8
LINK I/O TO FRONT OF QUEUE 10
LINK REISSUED I/O TO FRONT OF QUEUE 11
QUEUE (EP4 of .MIOS) 12
Link I/O to End of Queue LINK 18
Assign an I/O Entry QUEUE 18
Program Number at End of Queue DSPQT 37
Program Number at Front of Queue DSPQH 37
Program Number in Queue Following Interrupt DSPQM 46

QUEUE (cont.)
- logical channel queue 60
- Assign an I/O Entry QUEUE 64
- Link I/O to Front of QUEUE LINKF 64
- Assign an I/O Entry QUEUE 183
- Link I/O in Front o f Queue LINKF 183
- Assign an I/O Entry QUEUE 186
- Link I/O in Front of Queue LINKF 186

QUEUES
- I/O queues 1

READ
- DCWR (Read DCW Pointer) 24

READER
- Card Reader 2
- Card Reader 67
- CARD READER INTERRUPT HANDLER 121
- Card Reader Initialization .IGPIO 123
- CARD READER REQUEST 124
- CARD READER SELECT 126
- CARD READER ERROR 128
- Card Reader Recovery 1 GP09 129
- CARD READER INITIALIZATION 130

RECORD
- FORMAT ERROR ACCOUNTING RECORD 33
- Format Error Accounting Record (FMTAR) 37

RECOVERY
- Card Punch Recovery GP15 78
- MDS200 ERROR AND EOF RECOVERY 90
- MDS200 Recovery 1 GP20 91
- DSU200 Recovery 1 GP17 109
- Card Reader Recovery 1 GP09 129
- Magnetic Tape Recovery GP30 141
- Printer Recovery 1 GP13 153
- Paper Tape Recovery GP11 164
- Console Recovery GP22 177

REISSUE
- CONNECT REISSUE OF SECOND TYPEWRITER COMMAND 51

REISSUED
- LINK REISSUED I/O TO FRONT OF QUEUE 11

REISSUING
- GEPR reissuing I/O 33

REL
- Relinquish Control REL 64

RELINQUISH
- Relinquish Control REL 64

REQUEST
    I/O Request Handler 4
    I/O REQUEST HANDLER 7
    typewriter I/O request 23
    Accounting File Request ACTFL 34
    I/O Request Handler 40
    I/O Request Handler 45
    ACCOUNTING FILE REQUEST 62
    CARD PUNCH REQUEST 73
    MDS200 REQUEST 85
    DSU200 REQUEST 103
    STORAGE SUBSYSTEM (DSU200) REQUEST 103
    CARD READER REQUEST 124
    MAGNETIC TAPE REQUEST 136
    PRINTER REQUEST 148
    PAPER TAPE REQUEST 159
    TYPEWRITER REQUEST 171
    Process Specific Channel Request ENT2 183
    EXTERNAL REQUEST FOR ACCOUNTING FILE CLOSE 185
    Process Specific Channel Request ENT2 186

RESUME
    RESUME I/O FOR PROGRAM 54
    Resume I/O For Program RSMIO 57
    RESUME I/O ON CHANNEL 60

RETURN
    Status Return STRET 18
    SRP (Status Return Pointer) 25
    Status Return STRET 37
    STATUS RETURN 45
    Format I/O Status Words and Return Status GSTRT 57
    FORMAT I/O STATUS WORDS AND RETURN STATUS 58
    Status Return STRET 59

ROUTINES
    seek calculation routines 92
    seek calculation routines 110

RSMCH
    RSMCH 6
    RSMCH (EP12 of .MIOS) 60

RSMIO
    RSMIO 6
    RSMIO (EP8 of .MIOS) 54
    RSMIO (EP8 of .MIOS) 56
    Resume I/O For Program RSMIO 57

SCT
    SCT 1
    logical device SCT 1
    primary SCT 1
    device secondary SCT 43
    system configuration tables (SCT) 182

SECONDARY
    device secondary SCT 43

SEEK
  seek calculation routines 92
  SEEK ADDRESSES 96
  seek calculation routines 110
  SEEK ADDRESSES 114

SEGMENT
  minimum addressable segment 113

SELECT
  I/O select sequence 15
  I/O select sequence 27
  I/O select sequence 29
  CARD PUNCH SELECT 75
  MDS200 SELECT 88
  DSU200 SELECT 106
  CARD READER SELECT 126
  MAGNETIC TAPE SELECT 138
  PRINTER SELECT 150
  PAPER TAPE SELECT 161
  TYPEWRITER SELECT 174

SELECTED
  CONNECT SELECTED GESPECED ENTRY 52

SEQUENCE
  I/O select sequence 15
  I/O select sequence 27
  I/O select sequence 29

SI
  Special Interrupt (SI) 33
  Special Interrupts (SI) 35

SIMULATION
  CONNECT MULTIRECORD SIMULATION DCW 50
  multirecord simulation 70

SPEC
  SPEC 3
  SPEC (EP6 of .MIOS) 20

SPECIAL
  Special Interrupt (SI) 33
  Special Interrupts (SI) 35

SPECIFIC
  Process Specific Channel Request ENT2 183
  Process Specific Channel Request ENT2 186

SRP
  SRP (Status Return Pointer) 25

START
  Start I/O STIO 9
  Start I/O STIO) 37
  START I/O 47
  Start I/O STIO 55
  Start I/O STIO 61

STATUS
Status Return STRET 18
SRP (Status Return Pointer) 25
Status Return STRET 37
STATUS RETURN 45
Format I/O Status Words and Return Status GSTRT 57
Format I/O Status Words and Return Status GSTRT 57
FORMAT I/O STATUS WORDS AND RETURN STATUS 58
FORMAT I/O STATUS WORDS AND RETURN STATUS 58
Status Return STRET 59

STGPC
STGPC 3
STGPC (.MIOS) 52

STIO
STIO 3
STIO 8
Start I/O STIO 9
Start I/O STIO) 37
STIO (.MIOS) 47
STIO subroutine 51
Start I/O STIO 55
Start I/O STIO 61

STIOM
STIOM 3
STIOM (.MIOS) 50

STORAGE
DSU200 Disc Storage Subsystem 2
DSU200 Disc Storage Subsystem 67
STORAGE SUBSYSTEM (DSU200) REQUEST 103
Disc Storage Subsystem (DSU200) 110

STRET
STRET 3
Status Return STRET 18
Status Return STRET 37
STRET (.MIOS) 45
Status Return STRET 59

SUBROUTINE
INITIALIZATION SUBROUTINE 6
STIO subroutine 51

SUBSYSTEM
MDS200 Magnetic Drum Subsystem 2
DSU200 Disc Storage Subsystem 2
MDS200 Magnetic Drum Subsystem 67
DSU200 Disc Storage Subsystem 67
Magnetic Drum Subsystem (MDS200) 92
STORAGE SUBSYSTEM (DSU200) REQUEST 103
Disc Storage Subsystem (DSU200) 110

SUPERVISOR
I/O Supervisor (IOS) 1

SWITCHING
- accounting tape switching module 2
- Accounting Tape Switching ACTS1 64
- Accounting Tape Switching Module (.MACTS) 181
- accounting tape switching 182

SYSTEM
- channel primary System Configuration Table 1
- system configuration tables (SCT) 182

TABLE
- channel primary System Configuration Table 1
- Peripheral Assignment Table (PAT) 31

TABLES
- system configuration tables (SCT) 182

TAPE
- Magnetic Tape 2
- Paper Tape 2
- accounting tape switching module 2
- Accounting Tape Switching ACTS1 64
- Magnetic Tape 67
- Paper Tape 67
- MAGNETIC TAPE INTERRUPT HANDLER 133
- Tape Positioning 135
- MAGNETIC TAPE REQUEST 136
- MAGNETIC TAPE SELECT 138
- MAGNETIC TAPE ERROR 140
- Magnetic Tape Recovery GP30 141
- MAGNETIC TAPE INITIALIZATION 142
- PAPER TAPE INTERRUPT HANDLER 157
- PAPER TAPE REQUEST 159
- PAPER TAPE SELECT 161
- PAPER TAPE ERROR 163
- Paper Tape Recovery GP11 164
- PAPER TAPE INITIALIZATION 165
- Accounting Tape Switching Module (.MACTS) 181
- accounting tape switching 182

TERMINATE
- Terminate Error Entry FALT 91
- Terminate GEPR Abort ABREQ 109

TERMINATION
- Termination Interrupt (TI) 33
- Termination Interrupts (TI) 35

TI
- Termination Interrupt (TI) 33
- Termination Interrupts (TI) 35

TIME
- Accumulate Processor Time DACNB 37

TRACK
- track number 115

TRANSFER
- Transfer to a channel module entry point 68

TYGP
   TYGP (EP4 of .MTYPE) 176

TYIO
   TYIO (EP2 of .MTYPE) 171
TYIT
   TYIT (EP1 of .MTYPE) 168

TYPER
   TYPER 3
   TYPER (.MIOS) 51

TYPEWRITER
   Typewriter 2
   typewriter I/O request 23
   CONNECT REISSUE OF SECOND TYPEWRITER COMMAND 51
   Typewriter 67
   TYPEWRITER INTERRUPT HANDLER 168
   TYPEWRITER REQUEST 171
   TYPEWRITER SELECT 174
   TYPEWRITER ERROR 176
   TYPEWRITER INITIALIZATION 178

TYSL
   TYSL (EP3 of .MTYPE) 174

UNIT
   maximum file unit 113

UNLINK
   UNLINK 3
   Unlink I/O Entry UNLNK 22
   Unlink I/O Entry UNLNK) 37
   UNLINK I/O ENTRY 41
   Unlink I/O Entry UNLNK 57

UNLNK
   Unlink I/O Entry UNLNK 22
   Unlink I/O Entry UNLNK) 37
   UNLNK (.MIOS) 41
   Unlink I/O Entry UNLNK 57

VALIDATION
   DCW Pointer Validation DCWCK 18
   Pointer Validation PTRVL 18
   POINTER VALIDATION 27
   DCW POINTER VALIDATION 29

WORDS
   Format I/O Status Words and Return Status GSTRT 57
   FORMAT I/O STATUS WORDS AND RETURN STATUS 58

WRITE
   DCWW (Write DCW Pointer) 24

.CCRT4
   .CCRT4 97
   .CCRT4 130
   .CCRT4 142

- .CRACF
  - .CRACF 33
  - .CRACF 63
  - .CRACF 182
  - .CRACF 185

- .CRCCS
  - .CRCCS 36
  - .CRCCS 45
  - .CRCCS 56

- .CRCT1
  - .CRCT1 44
  - .CRCT1 79
  - .CRCT1 97
  - .CRCT1 118
  - .CRCT1 130
  - .CRCT1 154
  - .CRCT1 165
  - .CRCT1 178

- .CRCT4
  - .CRCT4 79
  - .CRCT4 118
  - .CRCT4 154
  - .CRCT4 165
  - .CRCT4 178

- .CRCTX
  - .CRCTX 44

- .CRDSP
  - .CRDSP 36
  - .CRDSP 45

- .CRGPR
  - .CRGPR 36

.CRQGT
  .CRQGT 8
  .CRQGT 9
  .CRQGT 21
  .CRQGT 33
  .CRQGT 36
  .CRQGT 41
  .CRQGT 45
  .CRQGT 47
  .CRQGT 47
  .CRQGT 50
  .CRQGT 51
  .CRQGT 52
  .CRQGT 54
  .CRQGT 56
  .CRQGT 60
  .CRQGT 61
  .CRQGT 75
  .CRQGT 88
  .CRQGT 89
  .CRQGT 100
  .CRQGT 101
  .CRQGT 106
  .CRQGT 107
  .CRQGT 126
  .CRQGT 133
  .CRQGT 134
  .CRQGT 138
  .CRQGT 150
  .CRQGT 161
  .CRQGT 175

.GINOS
  .GINOS Call 15

.GSPEC
  .GSPEC 20

.ICPIO
  Card Punch Initialization .ICPIO 72
  .ICPIO (.MCPIO) 79

.IDR20
  .IDR20 (.MDR20) 97

.IDS20
  .IDS20 (.MDS20) 118

.IGPIO
  Card Reader Initialization .IGPIO 123
  .IGPIO (.MGPIO) 130

.IIOS
  .IIOS 6
  .MIOS Initialization (.IIOS) 35
  .IIOS (.MIOS) 65

.IMTAP
  .IMTAP (.MMTAP) 142

- .IPRIO
  - Printer Initializarion .IPRIO 147
  - .IPRIO (.MGPIO) 154

- .IPTAP
  - .IPTAP (.MMTAP) 165

- .ITYPE
  - .ITYPE (.MTYPE) 178

- .MACTS
  - .MACTS 2
  - Accounting Tape Switching Module (.MACTS) 181
  - ACTS1 (EP1 of .MACTS) 181
  - ACTS2 (EP2 of .MACTS) 181
  - ACTS3 (EP3 of .MACTS) 181
  - ACTS1 (EP1 of .MACTS) 182
  - ACTS2 (EP2 of .MACTS) 184
  - ACTS3 (EP3 of .MACTS) 185

- .MCPIO
  - .MCPIO 2
  - .MCPIO 18
  - .MCPIO 37
  - .MCPIO 48
  - .MCPIO 67
  - CPIT (EP1 of .MCPIO) 70
  - .MCPIO 70
  - CPIO (EP2 of .MCPIO) 73
  - CPSL (EP3 of .MCPIO) 75
  - CPGP (EP4 of .MCPIO) 77
  - .ICPIO (.MCPIO) 79
  - .MCPIO 79

- .MDISP
  - Dispatcher (.MDISP) 36
  - Dispatcher (.MDISP) 45

- .MDR20
  - .MDR20 2
  - .MDR20 37
  - .MDR20 48
  - .MDR20 67
  - DRIT (EP1 of .MDR20) 82
  - DRIO (EP2 of .MDR20) 85
  - DRSL (EP3 of .MDR20) 88
  - DRGP (EP4 of .MDR20) 90
  - .IDR20 (.MDR20) 97
  - .MDR20 97

- .MDS20
  - .MDS20 2
  - .MDS20 18
  - .MDS20 48
  - .MDS20 67
  - DSIT (EP1 of .MDS20) 100
  - DSIO (EP2 of .MDS20) 103
  - DSSL (EP3 of .MDS20) 106
  - DSGP (EP4 of .MDS20) 108
  - .IDS20 (.MDS20) 118
  - .MDS20 118

.MGEPR
  Main GEPR module (.MGEPR) 60

.MGPIO
  .MGPIO 2
  .MGPIO 18
  .MGPIO 37
  .MGPIO 48
  .MGPIO 67
  CRIT (EP1 of .MGPIO) 121
  CRIO (EP2 of .MGPIO) 124
  CRSL (EP3 of .MGPIO) 126
  CRGP (EP4 of .MGPIO) 128
  .IGPIO (.MGPIO) 130
  .MGPIO 130
  .IPRIO (.MGPIO) 154
  .MGPIO 154

.MINIT
  .MINIT 79
  .MINIT 97
  .MINIT 118
  .MINIT 130
  .MINIT 142
  .MINIT 154
  .MINIT 165
  .MINIT 178

.MIOS
  .MIOS 2
  MAIN IOS MODULE (.MIOS) 3
  LINK (EP1 of .MIOS) 8
  LINKF (EP2 of .MIOS) 8
  LINKR (EP3 of .MIOS) 8
  LINKF (EP2 of .MIOS) 10
  LINKR (EP3 of .MIOS) 11
  LINK (EP1 of .MIOS) 11
  QUEUE (EP4 of .MIOS) 12
  INOS (EP5 of .MIOS) 15
  LINK (EP1 of .MIOS) 15
  SPEC (EP6 of .MIOS) 20
  ITYM (EP7 of .MIOS) 23
  PTRVL (.MIOS) 27
  DCWCK (.MIOS) 29
  FNDFC (.MIOS) 31
  FMTAR (.MIOS) 33
  ACTFL (EP13 of .MIOS) 33
  IOTRM (.MIOS) 35
  .MIOS Initialization (.IIOS) 35
  UNLNK (.MIOS) 41
  ILPCX (.MIOS) 43
  STRET (.MIOS) 45
  STIO (.MIOS) 47
  STIOM (.MIOS) 50
  TYPER (.MIOS) 51

MIOS (cont.)
- STGPC (.MIOS) 52
- RSMIO (EP8 of .MIOS) 54
- ABTIO (EP9 of .MIOS) 56
- RSMIO (EP8 of .MIOS) 56
- GSTRT (EP10 of .MIOS) 58
- RSMCH (EP12 of .MIOS) 60
- ACTFL (EP13 of .MIOS) 62
- .IIOS (.MIOS) 65
- ACTFL (EP13 of .MIOS) 182
- ACTFL (EP13 of .MIOS) 184

.MIOS.
- LINK (EP1 of .MIOS). 10

.MMTAP
- .MMTAP 2
- .MMTAP 18
- .MMTAP 37
- .MMTAP 48
- .MMTAP 67
- MTIT (EP1 of .MMTAP) 133
- MTIO (EP2 of .MMTAP) 136
- MTSL (EP3 of .MMTAP) 138
- MTGP (EP4 of .MMTAP) 140
- .IMTAP (.MMTAP) 142
- .MMTAP 142
- .IPTAP (.MMTAP) 165

.MMTYPE
- .MMTYPE 2

.MPOPM
- Main GEPOP module .MPOPM 181

.MPRIO
- .MPRIO 2
- .MPRIO 18
- .MPRIO 37
- .MPRIO 48
- .MPRIO 67
- PRIT (EP1 of .MPRIO) 145
- PRIO (EP2 of .MPRIO) 148
- PRSL (EP3 of .MPRIO) 150
- PRGP (EP4 of .MPRIO) 152

.MPTAP
- .MPTAP 2
- .MPTAP 18
- .MPTAP 37
- .MPTAP 48
- .MPTAP 67
- PTIT (EP1 of .MPTAP) 157
- PTIO (EP2 of .MPTAP) 159
- PTSL (EP3 of .MPTAP) 161
- PTGP (EP4 of .MPTAP) 163
- .MPTAP 165

.MTYPE
  .MTYPE 18
  .MTYPE 37
  .MTYPE 48
  .MTYPE 67
  TYIT (EP1 of .MTYPE) 168
  TYIO (EP2 of .MTYPE) 171
  TYSL (EP3 of .MTYPE) 174
  TYGP (EP4 of .MTYPE) 176
  .ITYPE (.MTYPE) 178
  .MTYPE 178

DOCUMENT REVIEW SHEET

TITLE: GE-625/635 GECOS-III I/O Supervision

CPB #: 1494

FROM·

Name:

Position:

Address:

Comments concerning this publication are solicited for use in improving future editions. Please provide any recommended additions, deletions, corrections, or other information you deem necessary for improving this manual. The following space is provided for your comments.

COMMENTS:

Please cut along this line

NO POSTAGE NECESSARY IF MAILED IN U.S.A.
Fold on two lines shown on reverse
side, staple, and mail.

STAPLE STAPLE

FOLD

FIRST CLASS
PERMIT, No. 4332
PHOENIX, ARIZONA

BUSINESS REPLY MAIL

NO POSTAGE STAMP NECESSARY IF MAILED IN THE UNITED STATES

POSTAGE WILL BE PAID BY

GENERAL ELECTRIC COMPANY
COMPUTER EQUIPMENT DEPARTMENT
13430 NORTH BLACK CANYON HIGHWAY
PHOENIX, ARIZONA - 85029

ATTENTION: DOCUMENTATION STANDARDS AND PUBLICATIONS B-90

FOLD

INFORMATION SYSTEMS

GENERAL ELECTRIC

LITHO U.S.A.