CONTROL DATA
CORPORATION

CONTROL DATA®
1700
COMPUTER SYSTEM

REDUCED CORE MONITOR
REFERENCE MANUAL

REVISION RECORD

REVISION	DESCRIPTION
12/70	Original printing.
A	
10-71	This printing obsoletes previous editions. This revision includes the 1739-1 Cartridge Disk Driver and miscellaneous additions and corrections.

Publication No.
60280600

Address comments concerning this manual to:
Control Data Corporation
Technical Publications
4201 N. Lexington Ave.
St. Paul, Minnesota 55112

or use Comment Sheet in the back of this manual.

CONTENTS

INTRODUCTION 1

The CONTROL DATA® 1700 Reduced Core Monitor is a modular, core-resident standard operating monitor designed for use on a small or very specialized real-time system.

The ability to control execution of users' programs, process users' requests and handle internal and external interrupts -- all on a priority basis -- enables this monitor to form the nucleus of a process control, communications or remote batch computer terminal type of system.

The minimum system components for the 1700 Reduced Core Monitor are:

CONTROL DATA® 1704 Computer (with 4096 words of memory)
CONTROL DATA® 1711/1713 Teletypewriter
CONTROL DATA® 1721/1722 Paper Tape Reader
CONTROL DATA® 1723/1724 Paper Tape Punch

An alternate minimum system is:

CONTROL DATA® 1704 Computer (with 4096 words of memory)
CONTROL DATA® 1705 A/Q Interrupt Data Channel
CONTROL DATA® 1711/1713 Teletypewriter
CONTROL DATA® 1777 Paper Tape Station

Another alternate minimum system is:

CONTROL DATA® 1774 System Controller
CONTROL DATA® 1775 A/Q Interrupt Data Channel
CONTROL DATA® 1711/1713 Teletypewriter
CONTROL DATA® 1777 Paper Tape Station

The alternate minimum configurations have the following limitations:

The system initializer cannot be used in this configuration; therefore, the system must be loaded from absolute binary cards.

The 4K assembly system cannot be supported on this configuration; therefore, a larger system with 1700 Mass Storage Operating System must be used for Reduced Core Monitor updates.

The card reader/punch driver does not recognize binary cards.

The 1711/1713 teletypewriter may be removed from the alternate minimum configuration. The following limitations however, will be imposed:

Systems comments which are forced to the teletype are not given.

The manual interrupt processor cannot be used.

An optional alternate system is:

CONTROL DATA® 1704 Computer (with 4096 words of memory)
CONTROL DATA® 1705 A/Q Interrupt Data Channel
CONTROL DATA® 1729-2 Card Reader or 1728-430 Card Reader-Punch
CONTROL DATA® 1740-501 or 1742 Printer
CONTROL DATA® 1711/1713 Teletypewriter

Another optional alternate system is:

CONTROL DATA® 1774 System Controller
CONTROL DATA® 1775 A/Q Interrupt Data Channel
CONTROL DATA® 1729-2 Card Reader or 1728-430 Card Reader-Punch
CONTROL DATA® 1740-501 or 1742 Printer
CONTROL DATA® 1711/1713 Teletypewriter
CONTROL DATA® 1572 Programmable Sample Rate Unit Timer
CONTROL DATA® 1573 Line Synchronizered Timing Generator

MONITOR 2

The monitor sets users' programs into execution, processes user requests, and handles interrupts generated by the system hardware. Priority levels are used by the system to establish order for scheduling, queuing requests, and handling interrupts. When no user program is in process and no interrupts or requests are being processed, the system executes an idle loop at priority of -1 until an interrupt occurs.

2.1 INTERRUPT PROCESSING

All interrupts are channeled through a stacking routine or common interrupt processor, ALLIN, which saves the information necessary to resume processing of an interrupted program and switches control to an interrupt processor. Associated with each interrupt line† is a four-word interrupt trap, defined as follows:

Word 0	entry	Program address applicable at time of interrupt plus overflow indicator
1	RTJ- ($FE)	FE_{16} contains the address of the common interrupt handler (interrupt stacking routine)
2	priority level	Priority associated with this interrupt line
3	primary processor	Address of the primary processor that is to service the interrupt

Each interrupt trap is assembled in the RCMLIN program and is under complete control of the user. The common interrupt handler, for example, may be bypassed simply by changing the second location of the four-word interrupt trap. This is done on interrupt line 0, which goes directly to the primary processor.

The interrupt handler saves the following information:

Contents of Q register

Contents of A register

Memory index I

Return location and overflow indicator

Priority level prior to the interrupt

Interrupt trap delta (equals trap location - $100)

† The minimum hardware configuration stated for this system includes only two interrupt lines. However, the system may be enlarged to include a maximum of 15 interrupt lines. Consult the Reduced Core Monitor Installation Handbook, publication number 60282800, for this information.

After the required information is saved, the priority level is established for the interrupt. Priority is then used as an index to the interrupt mask table, MASKT. The mask obtained is placed in the M register.

The interrupt handler enables the interrupt system and exits to the primary processor. Two primary processors are provided - RCMPF on line 0 and LYNE1 on line 1.

RCMPF handles all internal interrupts including program protect violations resulting from monitor requests.

Memory Protect	If present, the protect processor is executed on a protect fault. Otherwise, the system prints a PF on the teletype and hangs in the idle loop.
Memory Parity or Power Failure	If either a parity fault or power failure occurs, PF appears on the teletype and the system hangs.

External interrupts may occur on:

A line connected to devices controlled by the operating system; references to such devices appear in the physical device table

A line connected to devices not controlled by the operating system, such as an external control console

A line connected to devices, some of which are controlled by the operating system

The external interrupt processor, LYNE1 (entry point EPROC), handles the first of these with multiple devices but only on line 1.

For an interrupt from a nonexistent or ghost device, the following message appears on the teletypewriter:

GI 1

where the number 1 identifies the line on which the interrupt occurred.

2.2 INPUT/OUTPUT DRIVERS

Each device in the system is operated by a driver. Each driver includes an initiator and a continuator entry (and an entry for input/output hang up errors). Data necessary to operate the device is stored in the physical device table which is separate from the driver. The initiator entry initializes physical device tables and begins input/output on an idle device. The continuator entry drives the device to complete the input/output request.

2.2.1 INITIATOR

The initiator is entered only from a request processor. Before the initiator can be entered, the read/write request processor queues the location of a new request to the logical unit waiting list. The logical unit requested then must be inserted in the physical device table (Appendix B) slot for the device; its core location is the parameter passed to the initiator by a SCHDLE request. The initiator uses the entry parameter to set up the physical device table slot for carrying out the request.

The status of the device is obtained, the operation initiated, and an exit made to the dispatcher to allow other processing until the device interrupts processing with data or an alarm condition.

2.2.2 CONTINUATOR

The continuator is either entered from the common interrupt handler or scheduled by the external interrupt processor, LYNE1. The location of the physical device table is passed to the continuator in the Q register.

The continuator determines if the I/O operation has been completed. If it has, the request completion address is scheduled and the queue of I/O requests is examined for more requests for this device. If there are more requests, the initiator is re-entered. If there are no more requests, an exit is made to the dispatcher to restore the interrupted routine.

If the interrupt indicates the operation is not complete, the continuator takes the necessary action to continue the operation. If an alarm condition is indicated, the cause of the device failure is determined and completion is scheduled as though the operation had been completed normally.

2.2.3 COMPLETION ROUTINES

The sequence of code executed upon completion of a request is no different from that of any other program. Interrupts are active. The programs are scheduled according to completion priority and transfer upon completion to the dispatcher.

2.2.4 I/O DEVICE ERROR PROCESSING

When it is impossible to correctly complete an I/O request to a device, the driver exits normally to the complete request routine which sets the appropriate bits in the Q register before jumping to the user's completion routine.

When a driver fails to get a completion interrupt on an operation which it initiated, an I/O hang-up occurs. An optional diagnostic timer has been provided to detect hang-ups.

2.3 DISPATCHER

The dispatcher is entered any time processing at a given priority level has been completed. The dispatcher compares the top entry in the scheduler stack with the top entry in the interrupt stack. If the interrupt entry is of a greater or equal priority, the interrupted routine returns to execution. If there are no entries in either stack, an idle loop is executed at a priority level of -1.

2.4 VOLATILE STORAGE

Volatile storage is a block of core managed by the program RCMVOL (entry points VOLA and VOLR). Both entry points are entered by a return jump directly or indirectly through location BA_{16} or BB_{16} in the communications area. For a description of this region, see Appendix A.

An entry to VOLA causes assignment of a block of volatile storage. Likewise, an entry to VOLR releases a block of storage. On entry to VOLA, the size of the block is contained in the word following the RTJ instruction. Interrupts should be inhibited before VOLA or VOLR is entered.

On entry to VOLA, the requested number of cells is assigned. The first three locations are filled with the contents of Q, A, and I at the time of entry to VOLA.

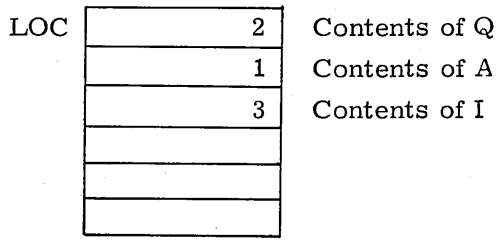
Contents of Q	Start of block = I on exit
Contents of A	
Contents of I	
Remainder of storage requested	End of block

On exit from VOLA, the I register contains the location of the contents of the Q register.

A subroutine is entered with 1 in A, 2 in Q, and 3 in I. Eight words of volatile storage are assigned as intermediate storage.

```
ENTRY                    Subroutine entry
        IIN              Inhibits interrupts
        RTJ     VOLA
        NUM     8
          .
          .
          .
        IIN
        RTJ     VOLR
```

On return from VOLA, a block of eight volatile storage locations has been assigned.



The I register contains the core location represented by LOC. The contents of A and Q are the same as on entry to VOLA. On entry to VOLR, I must contain LOC. On return from VOLR, the eight locations of volatile storage are released. The contents of A, Q and I are replaced by the contents of the first three locations of the released block. The last block of volatile requested must be the first block released.

A request for more volatile storage than is available constitutes a catastrophic condition. This results in the volatile storage assignment program entering a program with the entry point OVFVOL. OVFVOL is entered with the following in A and Q:

A Amount of overflow in words

Q Base address of the stack

OVFVOL causes the following comment to appear on the comment device

OV

The system then loops as no further action can be taken. †

Because of core limitations, it is not practical for VOLA to check for a minimum number of locations in a request. No less than three should be requested. If the user fails to comply with this restriction, the results will be unpredictable.

Interrupts must be inhibited on entry to VOLA and VOLR. Failure to do this also has unpredictable results.

2.5 RESTART ROUTINE

If the system stops because of a hung device or any of the fatal system faults, the system is restarted by master clearing and setting the RUN/STEP key temporarily to RUN. All threads are cleared and all system pointers reset to the values for initialization. This process restores the system to initialization time values, unless an application program has written over part of the monitor.

† The OV diagnostic indicates incorrect setup or use of the system.

REQUESTS 3

3.1 REQUEST MODULARITY

Eight requests are available in this system. Each request has a code between 0 and 10 which identifies the type of request. Routines are provided to process these requests. These routines have entry points which are coded as T1, T2, etc., corresponding to their request codes. All request processors are entered with the location of the request parameter list in the A register, and exit via a jump to the request exit (entry point REQXT).

3.2 ENTRY FOR REQUESTS

All programs make requests of the monitor by calling the monitor entry routine. A string of parameters follows the return jump instruction; parameters may also be placed in the A and/or Q registers. The general expansion is:

RTJ- ($F4)	Location in fixed communication region containing address of request entry processor
. . . parameters	Parameter list

The first instructions in the processor are:

Filled as a result of an RTJ
IIN

This configuration allows the processor to find the location of the request.

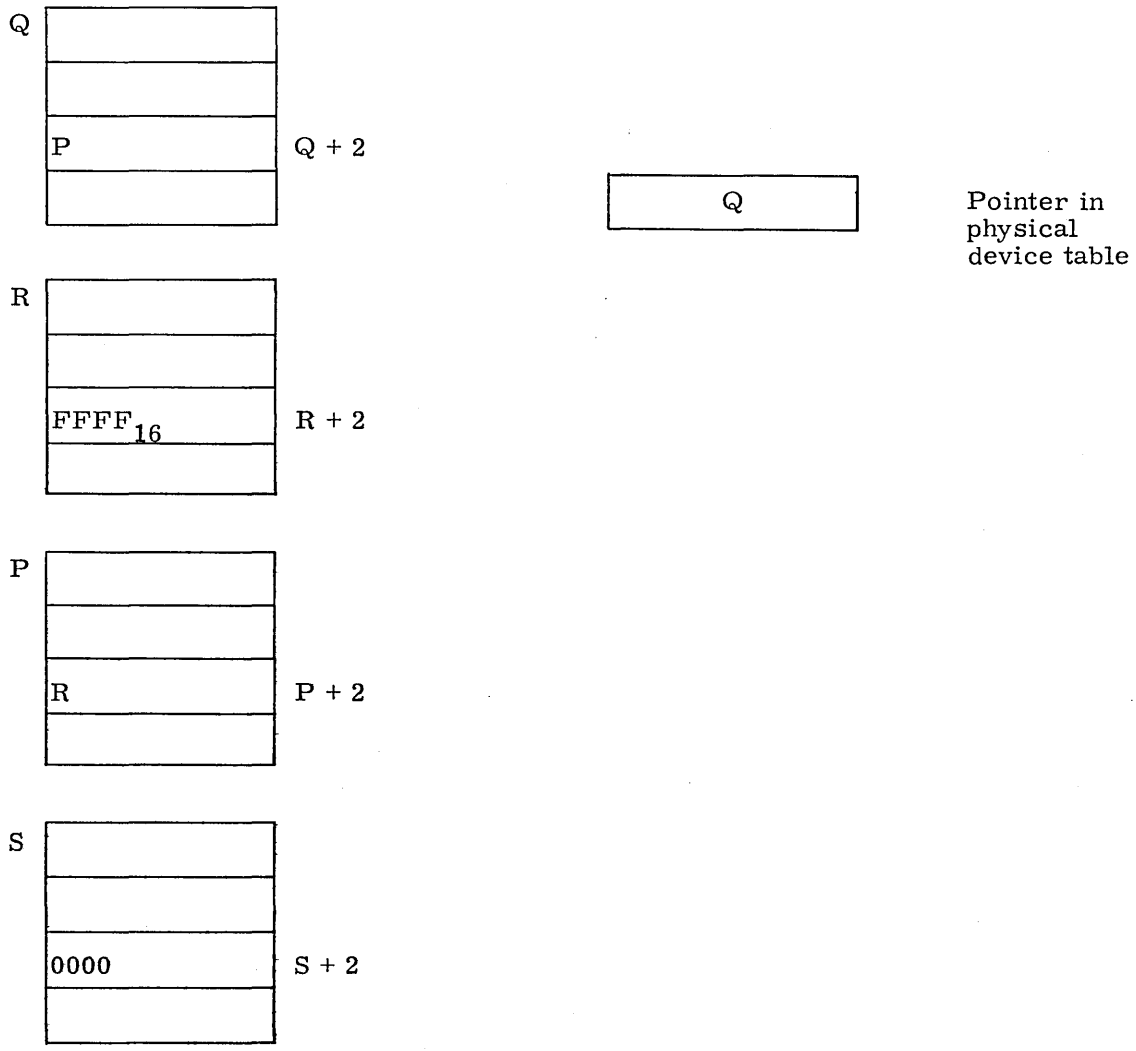
3.3 THREADING

Those monitor requests which involve a driver or schedule the use of core must wait in a queue. Such requests are processed on a first come, first served basis. All threading is performed by the appropriate request processors.

3.3.1 QUEUE FOR INPUT/OUTPUT REQUESTS

Input/output requests are queued by logical unit numbers. Requests for the same logical unit are threaded together by the third word of the parameter list. This word contains the first word address of the parameter list of the next request (zero for unqueued requests). The beginning of each queue is identified by an entry in the table of logical units which contains the address of the first word of the parameter list of the first request. The end of the queue is identified by $FFFF_{16}$ in the third word of the parameter list for the last request in the queue. There is a separate queue for each logical unit number. Each request is threaded before and during its execution. It cannot be repeated until the thread is cleared; i.e., until the request is completed. If a threaded request is repeated, it is not rethreaded but a return is made to the location following the request with bit 15 of the Q register set to 1.

Example:



REQUESTS 3

3.1 REQUEST MODULARITY

Eight requests are available in this system. Each request has a code between 0 and 10 which identifies the type of request. Routines are provided to process these requests. These routines have entry points which are coded as T1, T2, etc., corresponding to their request codes. All request processors are entered with the location of the request parameter list in the A register, and exit via a jump to the request exit (entry point REQXT).

3.2 ENTRY FOR REQUESTS

All programs make requests of the monitor by calling the monitor entry routine. A string of parameters follows the return jump instruction; parameters may also be placed in the A and/or Q registers. The general expansion is:

RTJ- ($F4)	Location in fixed communication region containing address of request entry processor
. . . parameters	Parameter list

The first instructions in the processor are:

Filled as a result of an RTJ
IIN

This configuration allows the processor to find the location of the request.

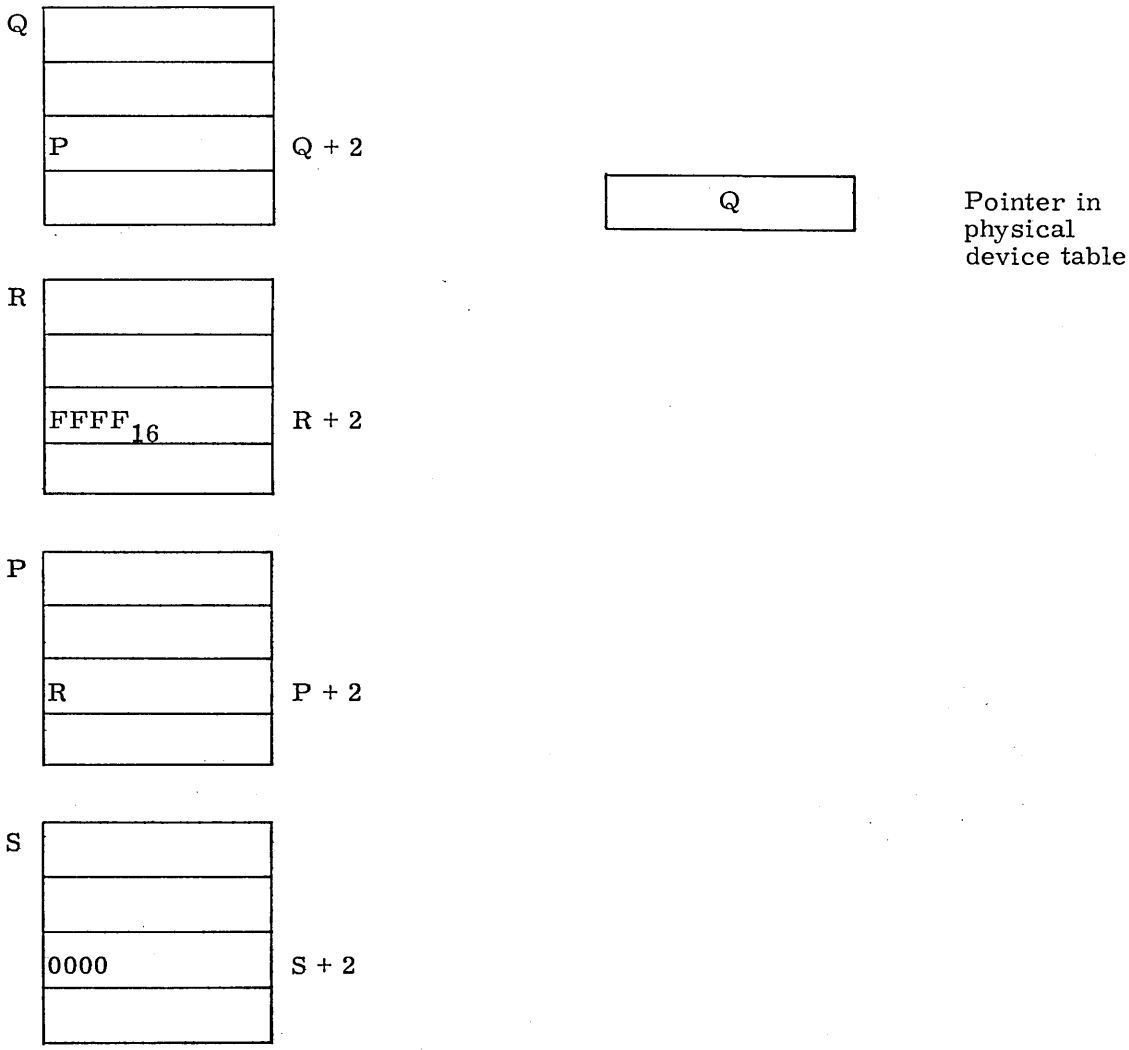
3.3 THREADING

Those monitor requests which involve a driver or schedule the use of core must wait in a queue. Such requests are processed on a first come, first served basis. All threading is performed by the appropriate request processors.

3.3.1 QUEUE FOR INPUT/OUTPUT REQUESTS

Input/output requests are queued by logical unit numbers. Requests for the same logical unit are threaded together by the third word of the parameter list. This word contains the first word address of the parameter list of the next request (zero for unqueued requests). The beginning of each queue is identified by an entry in the table of logical units which contains the address of the first word of the parameter list of the first request. The end of the queue is identified by $FFFF_{16}$ in the third word of the parameter list for the last request in the queue. There is a separate queue for each logical unit number. Each request is threaded before and during its execution. It cannot be repeated until the thread is cleared; i.e., until the request is completed. If a threaded request is repeated, it is not rethreaded but a return is made to the location following the request with bit 15 of the Q register set to 1.

Example:

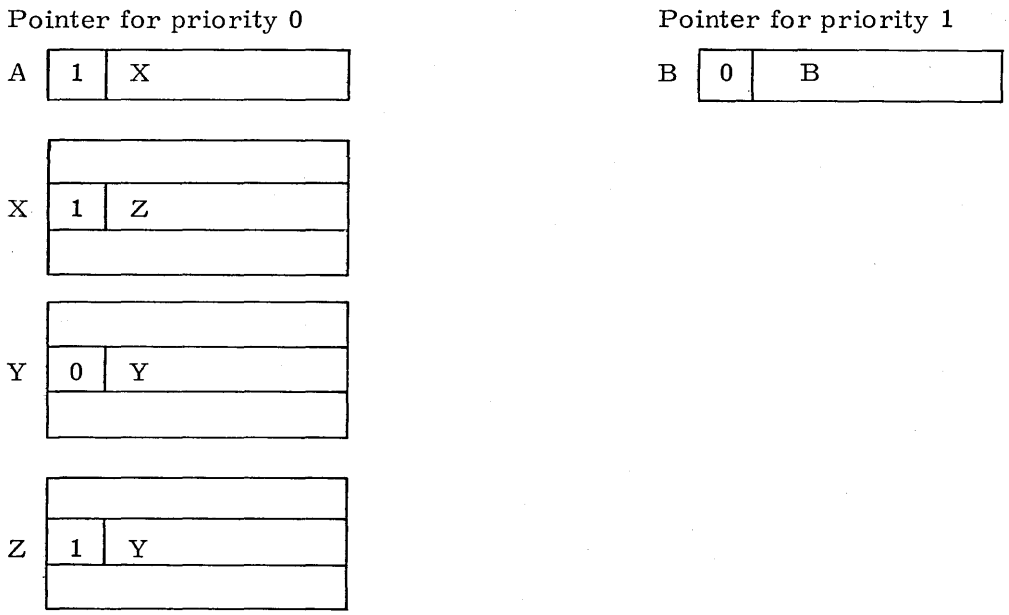


In this example, the request at Q is handled before the request at P, etc. The request at R is the last request in the queue for the device. The request at S is not in any queue.

3.3.2 QUEUE FOR SCHEDULER AND TIMER REQUESTS

Since system elements make numerous SCHDLE and TIMER requests and cannot conveniently wait for a request to be completed before making a new one, parameter lists from these requests are moved to a stack and threaded together with scheduler requests of the same priority on first come, first served basis. When idle, the top of each thread contains its own address. When a request is threaded to it, the location of the request thread word is stored in the top with the indirect bit (bit 15) set. The thread word of the last request then contains its own address, indicating the end of the thread.

Example:



Nothing is threaded to the top of the thread for level 1 at location B. The first request on priority 0 thread is located at X, the second at Z and the third at Y, which contains its own address indicating the end of the thread. Timer requests are threaded together in the same manner as the read/write requests described in Paragraph 3.4.1.

If the stack for SCHDLE or TIMER requests is filled when a new request is made, the sign bit of Q is set to 1 on return to the caller; it is set to 0 if the request is accepted. Other bits are not changed. When the stack is full, new requests are rejected until space is available.

3.4 INPUT/OUTPUT REQUESTS

The input/output requests for the Reduced Core Monitor are:

READ
WRITE
FREAD
FWRITE
STATUS

NOTE

The expanded calling sequence must be used for all requests if the utility or 4K assembler is used.

3.4.1 READ/WRITE AND FREAD/FWRITE REQUESTS

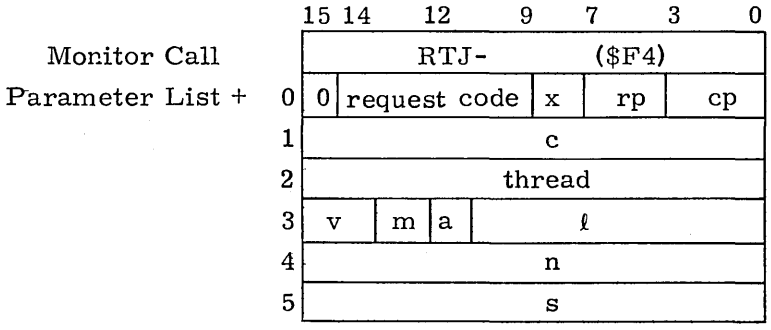
READ and WRITE instructions request processors to transfer data between the specified input/output device and core. The word count specified in the request determines the end of the transfer. The macro call is shown below:

READ
or ℓ, c, s, n, m, rp, cp, a, x
WRITE

Format read and write requests cause records to be read or written in a specific format. A particular format is associated with each device. For FREAD/FWRITE instructions n must not be zero. The macro call is shown below:

FREAD
or ℓ, c, s, n, m, rp, cp, a, x
FWRITE

The request code for READ is 1, for WRITE it is 2. The request code for FREAD is 4, and for FWRITE, it is 6. The expanded calling sequence is shown below:



The parameter descriptions which follow are the same for both forms of read and write requests with the exception noted for n.

ℓ Logical unit; an ordinal in the logical device tables (Appendix B); modified by parameter a.

c Completion address; the address of the core location to which control transfers when an input/output operation is completed. If omitted, no completion routine is scheduled and control is returned to the interrupted program.

Completion routines are operated by threading the input/output requests on the scheduler thread. A 3-bit code in the most significant three bit positions of word 3 of the request indicates completion status.

These 3 bits have the following significance:

Bit 15 = 1 Failed

Bit 14 = 1 Short read

Bit 13 = 1 Device ready

The completion address is entered with these bits set in similar positions in Q. If fewer than n words were transferred, the address of the location following the last word filled is placed in a location computed as follows:

s + n - 1 s = starting address

n = number of words requested

NOTE

The word count (n) must not exceed the buffer length.

s Starting address; address of first block location to be transferred (see the description of parameter x).

n Number of words to transfer. If n, the number of words is determined by the x parameter. If n = 0, the minimum amount of information is transferred (one word or one character) depending on the device. For FWRITE and FREAD, n cannot be 0. See parameter x for further discussion.

m Mode; determines the operating condition (ASCII or binary) of a driver. In binary mode, (m = B), data is transferred as it appears in core or on an input/output device. In ASCII mode, (m = A), data is converted from ASCII to external form for output or from external form to ASCII for input.

rp Request priority (0-5, 0 lowest) with respect to other requests. The priority level establishes the position in the input/output device queue.

cp Completion priority (0-5); the level at which the sequence of code, specified by parameter c, is executed.

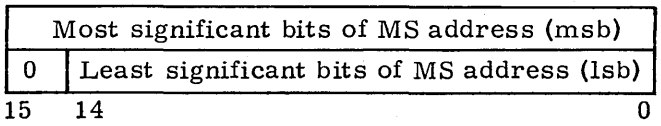
a Relative/indirect indicator for logical unit. If a is blank, the first parameter, ℓ, specifies the logical unit. If a = R, ℓ is a signed increment ($-1FF_{16} \leq \ell \leq +1FF_{16}$) added to the address of the first word of the parameter list to obtain the core location containing the logical unit number. If a = I, ℓ is the address of the core location number ($\ell \leq 3FF_{16}$).

x Absolute/relative indicator. This parameter affects parameters c, s and n as shown below. Because of the wraparound feature, computed addresses may be before or after the parameter list.

x	c	Definition of c
=0 or blank	c	Completion address; absolute address of completion routine.
≠ 0 and ≠ blank	c	A positive increment added to the address of the first word of the parameter list to form the completion address.

x	s	Definition of s
=0 or blank	s	The starting address.
≠ 0 and ≠ blank	s	A positive increment added to the address of the first word of the parameter list to form the starting address.
=0 or blank	(s)	A core location which contains the actual starting address.
≠ 0 and ≠ blank	(s)	A positive increment added to the address of the parameter list to form the address of a location containing another positive increment. The second increment is added to the address of the first word of the parameter list to obtain the starting address.

x	n	Definition of n
-	n	The actual length of the block to be transferred; here x has no meaning.
=0 or blank	(n)	A core location containing the block size.
≠ 0 and ≠ blank	(n)	A positive increment added to the address of the first word of the parameter list to obtain the address of the location containing the block size. For mass storage, the location containing the second increment is immediately followed by two words which contain the MS address in the format as seen below.



The mass storage address specifies a drum address or a sector on a disk, whichever is applicable.

The parameter v is the error code passed to the completion address in bits 15-13 of Q and set in the request by the system at completion.

Settings for a are as follows:

0	ℓ is a logical unit number
1	ℓ is a signed increment ($\pm 511_{10}$)
2	ℓ is a core location

Settings for m are as follows:

0	Binary
1	ASCII

Threaded requests cannot be executed again until they are completed and the thread is zero.

If this is attempted, Q is negative on return from the request processor. Otherwise, Q remains unchanged.

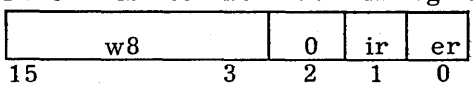
3.4.2 UNIT STATUS REQUEST

The STATUS request enables the user to determine the immediate status of a particular logical unit.

STATUS ℓ, sl, a

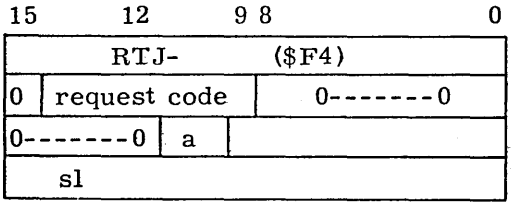
- ℓ Logical unit; may be actual logical unit or absolute address of location containing the actual logical unit ($\leq 3FF_{16}$).
- sl Level of status; 2 for level 2 status; 0 or blank not level 2 status
- a Indirect indicator for ℓ; if ℓ is actual logical unit; I if ℓ is absolute location containing the actual logical unit

The unit status processor takes the immediate status of the unit specified. Director status level 1 or 2 is returned in the A register. The Q register contains the following:



- ir An internal reject on the input of status operation
- er An external reject on the input of status operation
- w8 Bits 15-3 of word 8 of the physical device table

The request code for STATUS is 3 and the expanded code is as follows:



3.5 OTHER SYSTEM REQUESTS

The three monitor requests not associated with input/output operations are:

INDIR

SCHDLE

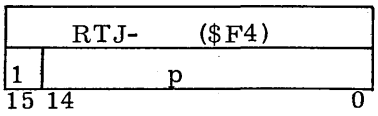
TIMER

3.5.1 INDIR REQUEST

INDIR allows indirect execution of any other request including I/O requests as determined by the parameter list referenced by p.

INDIR (p)

INDIR has no request code. The calling sequence is as follows:



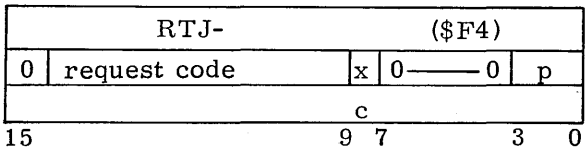
(p) Address of the first word of the parameter list of any other request; p must be enclosed in parentheses.

3.5.2 SCHDLE REQUEST

With the SCHDLE request, programs are queued on a priority basis. A program requested by SCHDLE is executed only when it is the oldest waiting task with the highest priority. All programs specified by SCHDLE requests are entered by a simple jump and exited by a jump to entry point DISP. The value in Q is passed to the requested program on entry.

SCHDLE c, p, x

The SCHDLE request code is 9 and the expanded calling sequence is as follows:



c Address to be executed as described under x.

p Priority level at which the program, specified by c and x, is executed.

x Absolute/relative indicator.

Parameters c and x work together as defined below.

x	c	Definition of c
=0 or blank	c	The location to be executed is c.
≠ 0 and ≠ blank	c	A positive increment added to the address of the first word of the parameter list to obtain the execution location. Because of memory wrap-around, the execution location may be before or after the SCHDLE request.

Example:

```
ENQ       1
SCHDLE    COMP,6
```

The core location associated with COMP is executed at priority level 6 and is entered with 0001_{16} in Q.

If a new program is at a priority level higher than the current level, the request is not stacked, but is immediately executed.

If the program priority level is less than or equal to the current level, the parameter list from the request is moved to the SCHDLE stack and threaded by priority, and on a first come, first served basis within priority. The stacking subroutines make entries to the stack and the dispatcher removes entries.

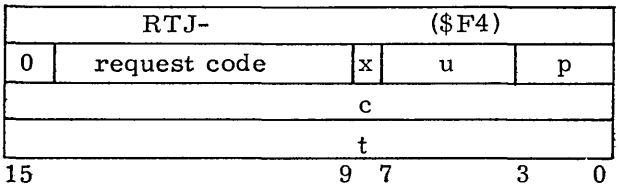
When an input/output request is completed, the driver causes the completion routine to be executed by threading the input/output request to the scheduler stack. This process avoids filling the stack with input/output completion addresses eliminating possible system SCHDLE requests.

3.5.3 TIMER REQUEST

The TIMER request is a delayed SCHDLE request. With TIMER, a SCHDLE request is made after a specified time delay.

```
TIMER c,p,x,t,u
```

The TIMER request code is 8 and the expanded calling sequence is as follows:



c	Address to be executed (completion address)
p	Priority level of program
x	Absolute/relative indicator
t	Time delay
u	Units of delay

Parameters c and x word together as defined below.

x	c	Definition of c
=0 or blank	c	The location to be executed is c.
≠ 0 and ≠ blank	c	A positive increment added to the address of the first word of the parameter list to obtain the execution location. Because of memory wrap-around, the execution location may be before or after the SCHDLE request.

Parameter u determines the unit in which the time delay t is measured.

u	t
0 or blank	Basic units of timing device
1	1/10 seconds
2	Seconds
3	Minutes

TIMER requests are stacked in the schedule request stack but are not threaded with them. Instead, they are threaded together on the basis of time until activation. When the delay for a TIMER request has expired, a SCHDLE request is made with Q = contents of location E8, the core clock counter at the time the SCHDLE request was made. An external parameter in the Systems Tables and Parameters program, RCMDEV, specifies the number of simultaneous TIMER expirations permitted to prevent loss of interrupts if time is insufficient to process the number of TIMER requests expiring at one time. The TIMER request can only be used on those systems that have timers.

3.6 SYSTEM INPUT/OUTPUT DEVICES

These I/O devices may have significance at installation. The core location contains the logical unit numbers for the devices. The operator can change these values via the K processor. All programs can, therefore, address these particular units indirectly or determine their values by interrogating the communications region.

Core Location	Contents
$FD	Logical unit number of input comment device
$FC	Logical unit number of output comment device
$FB	Logical unit number of standard print output device
$FA	Logical unit number of standard binary output device
$F9	Logical unit number of standard input device

MANUAL INTERRUPT PROCESSING 4

A manual interrupt notifies the Reduced Core Monitor that a control statement is being entered through the console teletypewriter. The manual interrupt processor accepts the control statement for determination of the next operation.

4.1 SCHEDULING CORE RESIDENT ROUTINES

If an application routine is scheduled through a manual interrupt, the user sets up the manual interrupt processor, RCMINT, as described in the Reduced Core Monitor Installation Handbook. When it is set up correctly, the user may manually interrupt the operation in process and type in the two character code associated with the applications routine. The routine is then scheduled.

4.2 SCHEDULING AN ABSOLUTE LOADER

To load a program into core for execution, the program must be in absolute formatted binary on paper tape. The operator may manually interrupt and type:

LD, hhhh, P

The absolute binary program is read into core beginning at location hhhh. If only LD is typed the read operation begins at the first location following the Reduced Core Monitor package. After being read in, the first location of the program is scheduled at priority P. If P is not present, the program is scheduled at priority 0.

4.3 SETTING SYSTEM I/O DEVICES

The system input/output devices may be re-assigned by manually interrupting and typing:

K,mℓ, mℓ

m is one of the following:

I	Standard input ($F9)
P	Standard binary output ($FA)
O	Standard print output ($FB)
M	Message unit (standard output comment) ($FC)
C	Standard input comment ($FD)
R	Restore units to initialized value

ℓ is the logical unit being assigned; no more than two decimal digits may be entered.

USER OPTIONS 5

The following modules in this chapter are not part of the basic 2K monitor, but are included with the Reduced Core Monitor system to be initialized with the system if the user desires.

5.1 TIMER PACKAGE

The timer package consists of the timer request processor and the diagnostic timer.

5.1.1 TIMER REQUEST PROCESSOR

The timer request processor, T8, processes the timer request described in Section 3.5.3. This module also processes timer interrupts and checks the timer threads for completion of time delays.

5.1.2 DIAGNOSTIC TIMER

Two features must be present for proper operation of the diagnostic timer:

1750 controller and 1572 programmable sample rate generator or a 1573 line synchronized timing generator

TIMER request module of the Reduced Core Monitor which includes the timer driver

NOTE

If these conditions are not met, hangup errors cannot be detected.

A driver computes a time differential for each input/output operation that it starts. This differential is placed in the physical device table slot for that device. The diagnostic timer module then decrements the time differential each time the diagnostic timer operates. If the time differential is found to be negative before being decremented, it is assumed that no operation is in progress. When the differential becomes negative, after being decremented, a hang-up is assumed. If a hang-up is detected, the diagnostic timer accesses the physical device table for the device to obtain the diagnostic entry in the driver. The diagnostic entry is then scheduled at the driver's priority. A normal request completion is made with the error bit set in the Q register.

The diagnostic timer is first operated after system initialization. Thereafter, it causes itself to be operated periodically by the TIMER request. The frequency of operation is dependent on a parameter internal to the diagnostic timer program.

The devices to be supervised by the diagnostic timer are specified by a table of physical device table addresses. This table resides within the timer package. References to devices may be added to or deleted from this table.

5.2 ENGINEERING FILE

The engineering file records hardware errors detected by the Reduced Core Monitor. When an abnormal condition occurs, the driver determines whether it is an error and then takes action. The error is then added to the device counter for errors.

5.2.1 ERRORS

All errors as found by the driver are logged. This includes rejects, alarms, and timer hang-ups. There is one counter for each device. There is also a word containing the last status at the last error time.

5.2.2 ERROR TABLES

Error tables are in the logging routine. There are two words for each logical unit. For example:

Word 1	0000	0000	0001	0001	0011_{16}
Word 2	0000	0011	0001	0001	0311_{16}

Word 1 is a counter for errors on this unit. Word 2 is the status word as defined at error time for the unit. It is a status taken by the Reduced Core Monitor and may vary slightly from the true hardware status. The error tables in the logging routine are always the same distance from the beginning of that routine; they are in LU number order. The address of the error tables is found in location $F3.

5.2.3 USAGE OF ERROR FILES

The error files indicate possible hardware errors. By dumping the error words and inspecting them, the customer engineer can determine faulty equipment. Location $F3 contains the address of error tables. The easiest way to interrogate the file is by stepping through the table in Sweep mode.

5.3 PROTECT PROCESSOR

The protect processor allows debugging of an application package without destroying the monitor. When this processor is used, it should be initialized at the end of the Reduced Core Monitor, but before the application programs. The program protect switch must be set. All core locations from location 0 to the end of the of the protect processor, are protected with the exception of certain communication region locations, entries to the request entry module and entries to the dispatcher. When an illegal protect violation occurs, the location of the violation is given on the comment device followed by PF. Program flow is discontinued and any outstanding completion addresses are executed.

When the applications package is debugged, the PROTECT switch may be off. If more core is required, the applications package may be reloaded over the protect processor.

DRIVERS 6

The drivers for the 1700 Reduced Core Monitor are combined into this chapter for convenient reference.

The following is a list of the drivers grouped according to devices. Section numbers of the particular driver are indicated in parentheses.

TELETYPEWRITER

1711, 1713 Teletypewriter driver (Section 6.1)

CARD READER and/or PUNCH

1728-430 Card Reader/Punch driver (Section 6.2)

PRINTER

1742 Line Printer driver (Section 6.3)

PAPER TAPE READER and/or PUNCH

1721/1722 Paper Tape Reader and
1723/1724 Paper Tape Punch Driver
or
1777 Paper Tape Station Driver (Section 6.4)

6.1 TELETYPEWRITER DRIVER (1711, 1713)

Four types of requests (READ, WRITE, FREAD, and FWRITE) are honored from this driver. Mode has no meaning and is assumed to be ASCII. Each request specifies the core location being read into or written from, number of words, and completion address.

6.1.1 READ

The number of words specified by the READ request is filled starting at the specified core location. Two characters fill one word with the first character being put into the upper half of the word. Bit 7 of each character is an even parity bit and is set to zero after checking it and before packing the character. If the parity bit is incorrect, a hardware error is indicated. If zero number of words is specified, only one character is read into the upper half of the specified core location. The lower character is filled with binary ones.

6.1.2 FREAD

Words in core are filled starting at the specified core location and continuing until the number of words specified is filled or a carriage return is encountered. Two characters are packed in each word. Bit 7 of each character is interpreted as an even parity bit before being cleared to zero. Line feed characters are ignored. If a cancel character is encountered, characters are passed with no information being stored until a carriage return is detected. The request is then repeated from the beginning.† If a carriage return is not encountered before the specified number of characters are read, characters are passed until a carriage return is encountered. A cancel character has the same effect as a carriage return.

6.1.3 WRITE

The number of words specified is printed starting at the specified core location. Each word causes two characters to be printed with the upper half being printed first. If zero number of words is specified, only one character is printed from the upper half of the specified core location.

6.1.4 FWRITE

This operation is the same as WRITE except that before any words are printed, a carriage return and line feed function are executed by the teletypewriter driver.

6.1.5 MANUAL INTERRUPT

A manual interrupt is caused by pressing the manual interrupt button on the teletypewriter. If a manual interrupt is detected by the teletypewriter driver, the manual interrupt processor is entered.

6.1.6 ERROR CONDITIONS

The driver recognizes the following categories of errors.

Internal reject on input or output instruction

External reject on input or output instruction

Failure to interrupt (if diagnostic timer is in use)

Parity error on input

Lost data

The above errors are considered irrecoverable by the driver. The driver sets the error fields in the physical device table and the error parameter in the request. Upon entry to the completion program, the Q register is negative, which indicates an irrecoverable error to the user.

† The caller's buffer is backgrounded with ones prior to repeating the request.

6.1.7 STATUS RESPONSE

The following responses appear in the A register of the 1711/1712/1713 teletypewriter when a status request is issued.

A Register	Condition
Bit 0	Ready
1	Busy
2	Interrupt
3	Data
4	End of operation
5	Alarm
6	Lost data
7-8	Unused
9	Read mode
10	Motor on
11	Manual interrupt
12-15	Unused

6.2 1728-430 CARD READER/PUNCH DRIVER

This driver reads and punches ASCII Hollerith only. This driver may be used for the 1729-2 card reader first by deleting the instructions in the driver set off by a double line of asterisks and then reassembling.

6.2.1 READ ASCII

Words in core are filled starting at a given address until the number of requested words are filled. Each column is converted from Hollerith to a seven-bit equivalent before being stored. These ASCII characters are stored two per word, leaving bits 7 and 15 as zero. If the number of words being read is zero, one column is read and the character is placed in the upper half of the word with ones placed in the lower half.

6.2.2 FREAD ASCII

Columns are read in ASCII mode until either one entire card is read or the number of words requested is filled.

If the number of words requested is depleted prior to reading one card, then the remainder of the card is unavailable and the read operation is in READ ASCII mode.

6.2.3 WRITE ASCII

Each word in core is converted to two Hollerith columns. Characters not within the range $20 through $5F are converted to blanks. The character in the upper half of the word is punched first, followed by the character in the lower half of the word. This continues until the number of words specified is reached.

6.2.4 FWRITE ASCII

ASCII FWRITE capabilities of the 1728-430 are the same as WRITE in ASCII mode except that a maximum of one card is punched.

6.2.5 ERROR CONDITIONS

The following error conditions are detected by the 1728 (1729-2) driver.

Internal or external reject on input or output instruction

Failure to interrupt (if diagnostic timer is in use)

Alarm interrupt

Illegal Hollerith punch detected during read

When the driver detects an irrecoverable failure, it sets the error field in the physical device table and the error parameter of the request. Upon entry to the completion routine the Q register is negative, which indicates an irrecoverable error to the user.

End of file is flagged as an illegal Hollerith punch and a question mark is put in the input buffer.

6.2.6 DIRECTOR STATUS

The following status bits are the hardware status bits returned in A when a system status request to the 1728-430/1729-2 is made.

A Register	Title	Description
A0=1	Ready	Card reader operational
A1=1	Busy	Card reader busy
A2=1	Interrupt	Indicates interrupt response generated by card reader. Other status bits must be monitored to determine the cause of the interrupt
A3=1	Data	Indicates data transfer may occur. Reader Data: the data hold register contains information ready for transfer to the computer.

A Register (cont'd)	Title (cont'd)	Description (cont'd)
A4=1	End of operation	Indicates the card reader completed operation.
A5=1	Alarm	Indicates presence of alarm condition.
A6=1	Lost data	Indicates data not transferred out of the holding register before the next column being read appeared. The status drops when a clear (A0=1) is sent to the controller. NOTE When lost data occurs, no further transfers occur from that card, and an end of operation status is generated.
A7=1	Protected	Indicates the PROTECT switch on the card reader is in protect position. When in this position, the card reader only accepts instructions with a 1 on the program protect line. All other instructions are rejected. A protected instruction is used with either a protected or unprotected card reader.
A8=1	Error	Indicates a pre-read error occurred.
A9=1	Feed alert	Indicates that during a card cycle, the transport of the card failed.
A10=1	End of file	Indicates end-of-file switch is on.
A11=1	Chip box error	Indicates chip box is full.

Level 2 director status returns the following:

A Register	Condition
A0=1	Hopper empty
A1=1	Stacker full
A2=1	Fail to feed
A3=1	Reader area jam
A4=1	Punch area jam
A5=1	Stacker area jam
A6=1	Pre-read error
A7=1	Punch error
A8=1	Manual
A10=1	Interlock

6.3 1742 AND 1740-501 LINE PRINTER DRIVER

The 1742 driver recognizes ASCII FWRITE only. It prints up to 136 characters per line and ignores any additional characters.

6.3.1 CONTROL CHARACTERS

FORTRAN compatible ASCII directives are the only control characters used. These occur in the first character of the output buffer.

Character	Action before printing
blank	Space 1 line
0	Space 2 lines
+	No space
1	Page eject
all others	Space one line

6.3.2 ILLEGAL CHARACTERS

Only ASCII characters $20 through $5F are legal; all others are ignored.

6.3.3 ERROR CONDITIONS

The following hardware errors are detected by the driver:

Internal or external reject

Failure to interrupt (if diagnostic timer is in use)

Alarm

When the completion routine is entered, bit 15 of the Q register is set thus notifying the requestor of the error.

6.3.4 DIRECTOR STATUS

Status Bits	Title
A0=1	Ready
A1=1	Busy
A2=1	Interrupt
A3=1	Data
A4=1	End of Operation
A5=1	Alarm
A7=1	Protected
A9=1	6/8 lines coincident
All others	Undefined

Alarm may be caused by paper out, paper tear, fuse alarm, or open interlock.

6.4 PAPER TAPE DEVICES

This driver operates both the 1721/1722 paper tape reader and 1723/1724 paper tape punch or the 1777 paper tape station.

This driver honors READ, FREAD, WRITE, and FWRITE requests in both binary and ASCII modes.

6.4.1 READ BINARY

Starting at the address specified in the calling sequence, core is filled with the number of words specified in the calling sequence. Two frames of tape fill one word. The first frame is packed into the upper half of a word. If zero words are specified, only one frame is read into the upper half of the word, and the lower half is filled with ones.

6.4.2 READ ASCII

Starting at the address specified in the calling sequence, core is filled with the requested number of words. Two tape frames fill one word. Bit 7 of each frame is an even parity bit, and is set to zero for each frame. If zero words are specified, only one frame is read into the upper half of the word, and the lower half is filled with ones.

6.4.3 FREAD ASCII

Words in core are filled beginning at the starting address and continuing until the specified number of words is read or an ASCII carriage return character is encountered. If the number of words is read before the carriage return is reached, tape frames are passed without storing any information in core until a carriage return is encountered. Two tape frames are packed in each word with bit 7 of each frame interpreted as an even parity bit before being set to zero. Line feed, cancel, and null characters are ignored. If a carriage return is read before the word count is depleted, no further data is transferred. The address of the next location to be filled is placed at the end of the specified block. This address is stored in location s+n-1 (s is the starting address and n is the number of words requested). If an odd number of characters is read, the lower half of the last word is filled with ones.

6.4.4 FREAD BINARY

If an ASCII asterisk is the first frame read during a FREAD binary, the request is changed to ASCII mode. If not, the first word is interpreted as the complement of the number of words in this formatted record. This number or the number of words specified in the calling sequence, whichever is smaller, determines the number of words to be filled After the entire record is read, the succeeding word is a checksum which balances the sum of the header word and the information in the record to zero. If the sum is incorrect, a hardware error message is issued. If the word count is depleted before the end of the record, tape is passed until the end of the record is found and the checksum is checked.

If the end of the record is found before the word count is depleted, no further data is transferred and the address of the next word is stored in s+n-1 (s is the starting address and n is the number of words requested).

The system cannot recognize word counts between 21760 and 22015 (these appear as asterisks). If the word count of the binary format record is greater than 22016, then 256 is subtracted to obtain the actual word count. The system adds 256 to word counts exceeding 21760 for paper tape format records on output.

6.4.5 WRITE BINARY

Starting at the specified address in the calling sequence, the requested number of words is punched. Each word is punched in two frames of tape. The upper eight bits are punched first. If zero words are specified, the upper half of the word is punched only in one frame. If the specified number of words cannot be punched, a hardware error is indicated.

6.4.6 WRITE ASCII

Starting at the specified address, the requested number of words is punched. Each word is punched in two tape frames. Bit 7 of each frame is changed to provide even parity. If zero words are specified, the upper half of the specified core location is punched into only one frame.

6.4.7 FWRITE ASCII

This request functions in the same way as WRITE ASCII except that after all words have been punched, a carriage return and line feed character are punched or when a carriage is encountered, a line feed character is inserted following the carriage return.

6.4.8 FWRITE BINARY

The first word punched is the complement of the number of words to be punched. The requested number of words is then punched starting at the specified address. Each word is punched in two tape frames. After the entire record is written, the succeeding word is a checksum which balances the sum of the header word and the information in the record to zero. If the proposed format record length is greater than 21760, then 256 is added to the word count prior to punching. Record lengths between 21760 and 22015 appear as asterisks and are avoided in reading the record.

6.4.9 ERROR CONDITIONS

The driver recognizes the following categories of errors:

Internal reject on input or output statement

External reject on input or output statement

Alarm condition
Data lost on read
Parity error on read
Checksum error on read
Failure to interrupt (if diagnostic timer is in use)

These errors are considered irrecoverable by the driver. The driver sets the error field in the physical device table and the error parameter in the request. The Q register is set negative upon entry to the completion routine and indicates an irrecoverable error to the user.

6.4.10 STATUS RESPONSE

When a status request is issued, the following is returned in the A register from the paper tape reader. There is only one level of status for paper tape devices.

A Contents	Condition
A0=1	Ready
A1=1	Busy
A2=1	Interrupt
A3=1	Data
A4=1	Unused
A5=1	Alarm
A6=1	Lost data
A7=1	Protected
A8=1	Existence code
A9=1	Paper motion failure
A10=1	Power on
A11-A15	Unused

The paper tape punch is the same except for bit 6 which is unused and bit 11 which indicates tape supply is low. On the 1777 paper tape station, bit 6 set denotes a validation error on punch.

6.5 1700 CARTRIDGE DISK DRIVER

The 1739-1 Cartridge Disk Drive (CDD) utilizes a single fixed disk and a single removable disk in a cartridge case. Both disks have two recording surfaces and are standard on the 1739-1. The disks are referred to as Disk 0 and Disk 1, and are individually addressed by the hardware controller by a single bit designator within the file address word. Ordinarily Disk 0 is the removable disk and Disk 1 is the fixed disk. A toggle switch on the maintenance panel is used to reverse addressing which causes Disk 0 to reference the fixed disk and Disk 1 to reference the removable disk. Software users reference the entire Cartridge Disk Drive as a single logical unit with Disk 0 always containing the lowest sector address and Disk 1 the highest sector address. The disk referenced is dependent upon the toggle switch position for disk addressing.

The 1700 cartridge disk driver operates under the control of the 1700 Reduced Core Monitor to provide the capability for data transfer to and from the disk as a mass storage device.

The cartridge disk controller interfaces with the A/Q channel and the direct storage access bus. The A/Q channel transmits functions and address information to the controller and status information to the computer. The direct storage access bus transfers all data between the controller and the computer.

6.5.1 DISK DRIVER REQUESTS

The disk driver processes requests for data transfer to and from mass memory (READ, WRITE, FREAD, or FWRITE) made by user programs and provides a program overlay capability.

The calling sequence specifies the number of words to be transferred to or from core, beginning at the specified starting address and sector number. Sectors are read or written sequentially until the requested number of words has been transferred. If zero words are requested, the disk driver transfers one word to or from core.

The disk driver with word address capability provides two algorithms for addressing information on the disk. These algorithms permit word addressing and sector addressing. Each disk sector contains 96 words. When sector addressing is specified, information is transferred to or from a sector starting at the first word of the sector. When word addressing is specified, information is transferred starting at the specified disk word location.

6.5.2 DATA TRANSFER REQUEST FORMATS

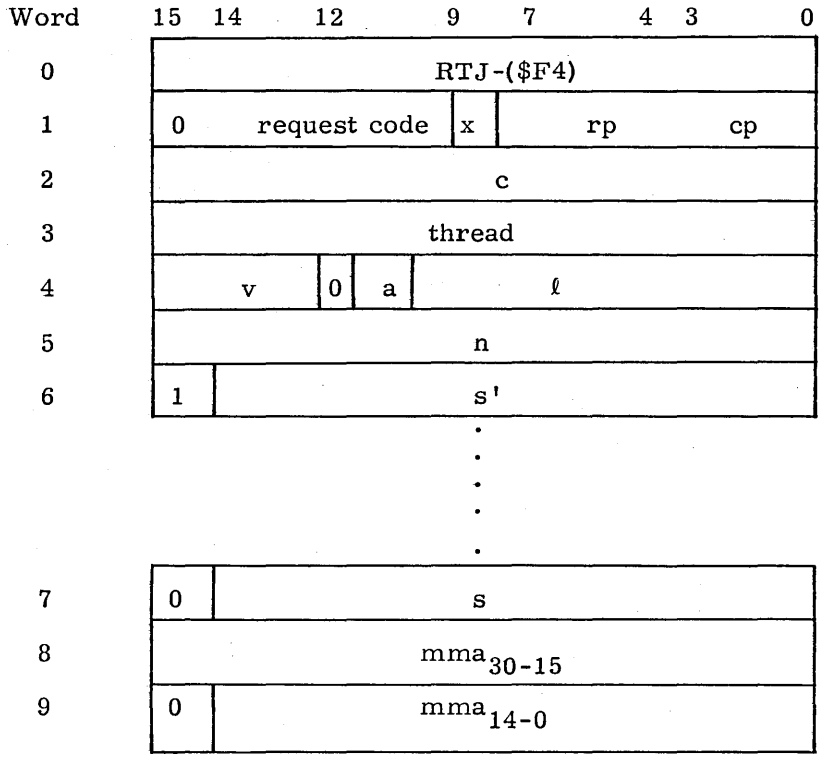
The execution of a data transfer request transfers n words from mass memory (READ or FREAD) or to mass memory (WRITE or FWRITE) starting at any core first word address indicated by s and mass memory address indicated by mma. If n is zero, one word is transferred. No data formatting is involved since corresponding core and mass memory locations contain identical 16-bit images.

The disk driver data transfer request differs from conventional requests in that the two-word mass memory address must be included. To accommodate this, two formats are provided. All parameters not described in the following paragraphs can be found in Section 3.4.

The first format provides a seven-word format (Section 3.4) which is consistent with normal requests (x parameter not set).

If the x parameter is set, an indirect reference to the first word address increment found in the s parameter is made. This positive increment added to the address of the parameter list forms the address of a location containing another positive increment. The second increment added to the address of the parameter list gives the starting address. Two words containing the mass memory address (mma) immediately follow the second increment.

The following format must be used:



If parameter x is zero, both s and s' are absolute addresses. Otherwise, they are 15-bit positive increments which are added to the address of the first request parameter (word 1) to form absolute addresses. Control returns to the location following word 6 after the request is made.

The second request format adds two additional words containing mma in line with the conventional data transfer format, identified by a direct reference to s (bit 15 equals 0), as follows:

Word	15	14	12	9	7	4	3	0
0	RTJ- ($F4)							
1	0	request code		x	rp		cp	
2	c							
3	thread							
4	v		0	a	ℓ			
5	n							
6	0	s						
7	mma_{30-15}							
8	0	mma_{14-0}						

Control returns to the location following word 8 after the request is made.

6.5.3 READ AND WRITE REQUESTS

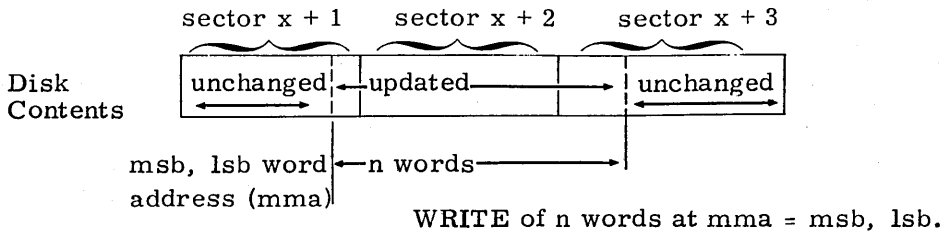
The expanded calling sequence described in Section 3.4 must be used for all requests if the utility of 4K assembler is used.

READ and WRITE requests provide the ability to simulate the word address by allowing the mma to be any word address within the size range of the disk. The driver converts the word address to sector and word in the sector by dividing by 96. A partial sector WRITE request causes the entire sector to be read into a buffer in the driver, the user's data to be moved into the appropriate portion of that buffer, and the entire buffer to be written onto the disk.

The READ request fills core, starting at a specified address, with the specified number of words. If zero words are requested, one word is transferred. Transfer is initiated from the disk word address specified by the msb (most significant bits) and lsb (least significant bits) of the request. The lsb is a 15-bit value. A carry into bit 15 of lsb should be treated as an overflow condition, and msb should be incremented by 1.

WRITE transfers the requested number of words from core to disk. The driver interprets the disk starting address (msb, lsb) as a word address. When writing on disk in this mode, the remainder of partially updated sectors is preserved.

An example of a word oriented write across several sectors follows:



WRITE of n words at mma = msb, lsb.

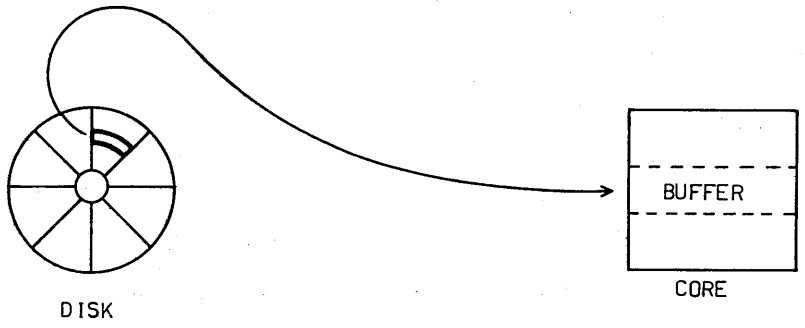
An example of an indirect READ request:

```
R       READ    8,C-R-1,(PARAM-R-1),15,B,4,4,,1
          :
PARAM   ADC     BUFFER-R-1,$1,$6D59
        BSS     BUFFER(15)
```

where C is the completion address.

As a result of this request, 15 words are read from logical unit 8 (disk) starting from disk word address $1,6D59_{16}$, in to core starting at first word address BUFFER. Disk word address 1,6D59 is the same as sector 974, word 25 (divide $16D59_{16}$ by 96_{10} for sector and word.

From disk word address 1,6D59 To core first word address BUFFER



For a similar WRITE request, 15 words are written on the disk at sector 974, word 25. Other words in the sector are unchanged.

6.5.4 FREAD AND FWRITE REQUESTS

FREAD and FWRITE requests utilize the sector orientation of the disk. The format for FREAD and FWRITE is the same as READ and WRITE. The mma represents a sector number; n represents the number of words to be transferred. If it is not a multiple of 96 for an FWRITE request, the unused words of the last sector are set to zero.

FREAD fills core, starting at a requested address, with the specified number of words. If zero words are requested, one word is transferred. Only zero is permitted in the most significant bits (msb) field of mma in an FREAD request. The driver interprets the least significant bits (lsb) field as a disk sector address.

This request transfers the specified number of words from core to disk. The driver interprets the starting disk address as a sector address. The msb must be zero. If zero words are requested, one word is transferred. The remainder of partially updated sector is not preserved.

Using the same symbolic conventions as the previous example, a normal FWRITE request appears as:

```
FWRITE          8,COMP,BUFFER,113,B,4,4,,1
   :
   :
ADC             0,103
   :
   :
BSS             BUFFER(113)
```

In this case, 113 words are written from the core first word address BUFFER onto the disk, starting at sector 103.

6.5.5 OVERLAY REQUESTS

In some cases, it is desirable for a mass-memory-resident program to be able to overlay itself with another program segment. Control transfers to the latter when it is available in core. A normal READ or FREAD request to the disk driver cannot handle the situation, because the request parameters must remain intact through completion of the request. To accommodate this situation, RTJ+OVRLAY replaces a READ or FREAD monitor call RTJ- ($F4).

OVRLAY must be a routine accessable to the requesting program. The following sequence of code is recommended as the OVRLAY routine.

```
OVRLAY   000      000          Calling sequence for typical disk read
         IIN      0            RTJ OVRLAY
         LDA*     OVRLAY       ADC $200,COMP,0,$8C2,N,BUF,0,ADR
         EOR-     ONEBIT+15    Set indirect bit in request word of parameter list
         STA*     OVR1         Store request in indirect monitor call
         RTJ-     ($F4)        Execute indirect monitor call
OVR1     NUM      0
         JMP-     ($EA)        Exit to dispatcher
```

Control is not returned to the requesting program after the request; however, the completion address specified in the parameter list is honored. Control passes to the overlaying program. The disk driver moves the parameters to the physical device table prior to the start of the mass memory transfer.

6.5.6 BAD SECTOR COMPATIBILITY

The 1700 Cartridge Disk Driver provides for the utilization of disk cartridges (packs) which have defective areas. Manufacturing standards permit a small percentage of bad sectors per cartridge.

The software routines that provide the bad sector capability perform the following functions:

Test all surfaces of the disk prior to system initialization cataloging bad sectors and assigning alternates

Provide recovery from accessing a previously defined bad sector by accessing its alternate

Request a list of the bad sector directory with the bad sectors and alternates

Every new pack entered in the system must have its bad sector directory initialized. This initialization must be performed off-line prior to system initialization time.

BDSINT -- Bad Sector Initializer

The bad sector initializer module (BDSINT) is a stand alone program that is executed prior to the execution of the system initializer. Any cartridge disk used on the system must have been previously initialized, or the RCM execution must be stopped and BDSINT must be loaded and executed.

The three primary functions of this routine are:

Perform surface testing of the cartridge disk

Create the bad sector directory on the device (pack)

Allocate and assign good sectors as alternates for bad sectors

BDSINT Messages

The control statement

*B,e,t

calls BDSINT to perform a surface test and create the bad sector directory. The parameters are defined as follows:

e Single character hexadecimal equipment number (0-F) of the cartridge disk controller

t Initialization operation

- 0 Initialize disk 0 only
- 1 Initialize disk 1 only
- 2 Initialize both

This function writes over any data on the surface areas of the designated cartridges and creates a bad sector directory including a user specified number of alternate sectors.

This number of alternate sectors that may be reserved on disk 0 and on disk 1, is an assembly time option; e.g., EQU NUMALT (30). This necessitates reassembling modules of the system initializer BDSINT and RCMCRT which contain the directories after BDSINT is executed. The system is released with NUMALT equated to an arbitrary number of 30 alternate sectors.

The RCM routine RCMTBL must also be reassembled to provide room in the cartridge disk physical device table for the bad sector directory and the table of alternates.

A *P statement requests the bad sector directory to be printed according to the following format:

```
BAD SECTOR DIRECTORY
BAD SEC. = SYSTEM SEC. xxxx = FILE ADDRESS hhhh ALT. = FILE ADDRESS hhhh
```

xxxx	RCM system hexadecimal sector address
hhhh	Physical disk address (file address)

A * (cr) statement terminates the bad sector initialization and permits the user to start system initialization.

Messages to the Operator

BDSINT IN	The Bad Sector Initializer is ready to accept control statements.
BDSINT ERR. N	Initialization error specified by N. Initialization is restarted.
DISK 0 SECTOR ZERO BAD	Sector zero is bad. Disk cannot be used.
DISK 1 SECTOR ZERO BAD	Sector zero is bad. Disk cannot be used.
EXCEEDED THE MAX. NO. OF. ALTERNATES ON DISK 0/1	Disk must be refurbished or more alternate sectors specified.
FORMAT ERROR	Invalid control statement. Re-enter statement.
INTERNAL/EXTERNAL REJECT	Cannot communicate with the device.

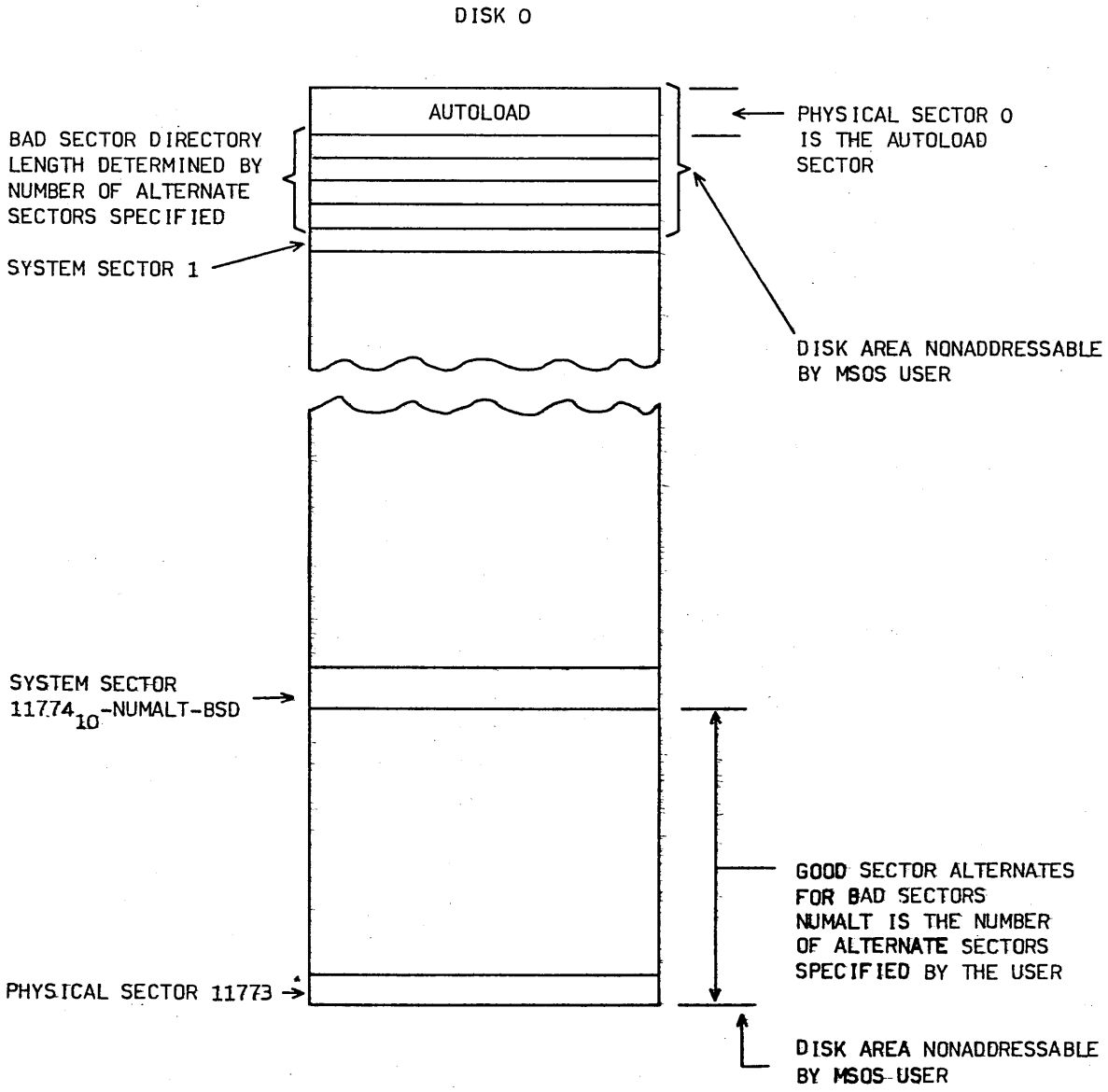
N

1	Compare Error	Data received by the computer does not compare with data read from disk.
2	Lost Data	Loss of data on DSA transfer
3	Address Error	Controller has detected an illegal file address

4	Controller Seek Error	Controller is unable to find the file address specified.
5	Parity Error	Controller has received a parity error signal from DSA buss.
6	Protect Fault	An attempt was made to write into protected core.
7	Drive Seek Error	Drive Unit has moved beyond the legal limits of the Disk Drive.

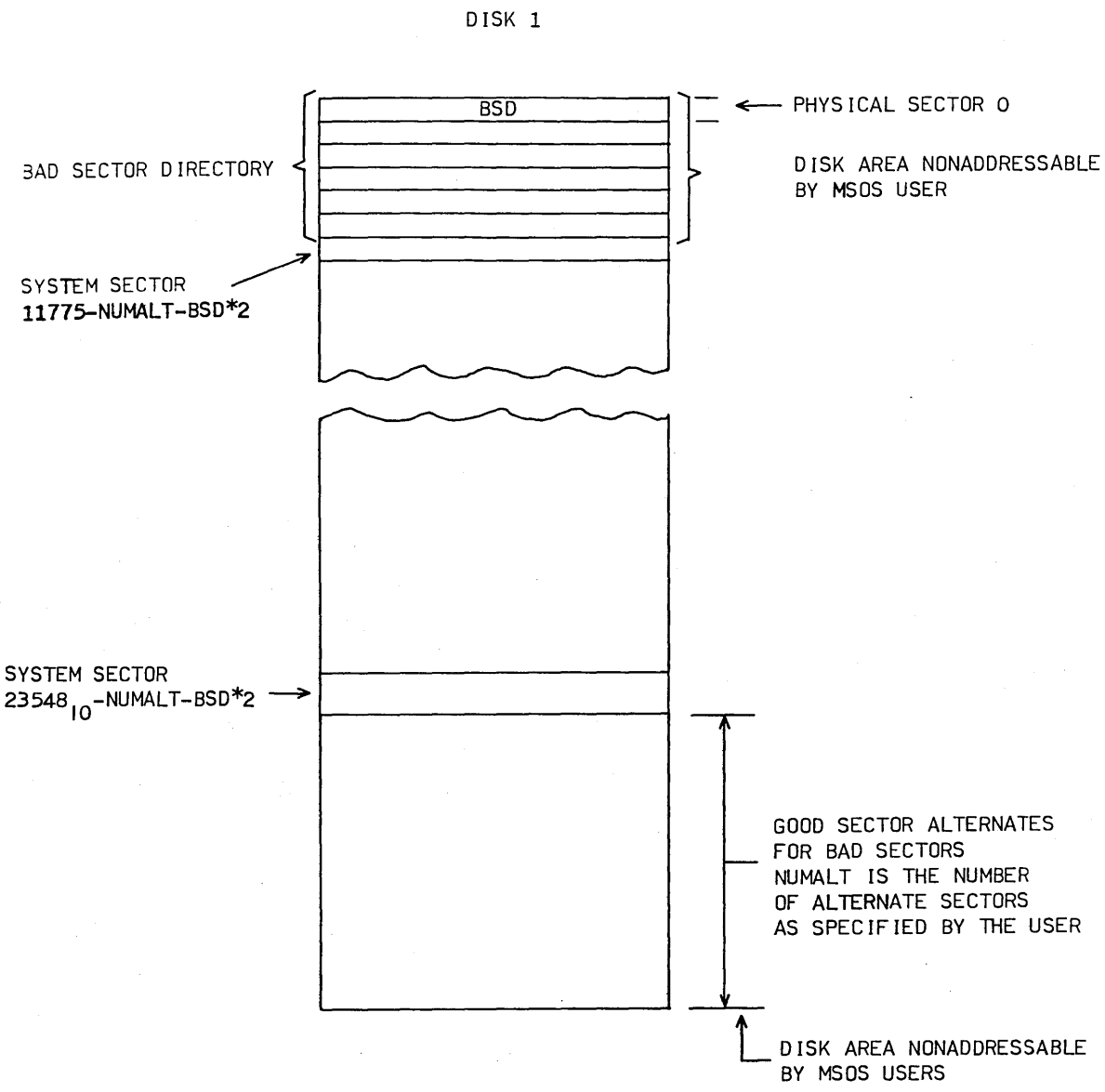
Each system sector represents a physical position on disk 0 or disk 1 which may be equated to a file address by referencing the algorithm. Although each system sector address has a corresponding file address, the opposite is not true; i.e., the alternate sectors reserved for allocation to bad sectors are not addressable and do not have a system sector address.

The conversion algorithm accounts for the sectors reserved for the bad sector directory and alternate sectors and assumes allocation according to the following diagrams.

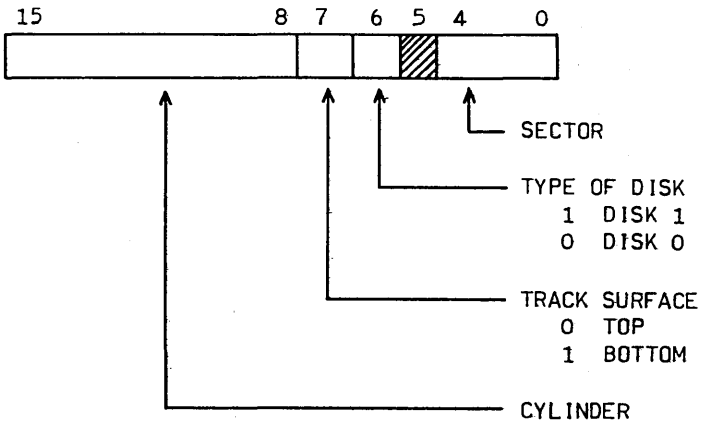


NOTE

THE NEXT CONTIGUOUS SYSTEM SECTOR IS PHYSICAL SECTOR 0 + BSD OF THE FIXED DISK.


DISK 1
BSD
PHYSICAL SECTOR 0
BAD SECTOR DIRECTORY
DISK AREA NONADDRESSABLE
BY MSOS USER
SYSTEM SECTOR
11775-NUMALT-BSD*2
SYSTEM SECTOR
23548$_{10}$-NUMALT-BSD*2
GOOD SECTOR ALTERNATES
FOR BAD SECTORS
NUMALT IS THE NUMBER
OF ALTERNATE SECTORS
AS SPECIFIED BY THE USER
DISK AREA NONADDRESSABLE
BY MSOS USERS

The file address word has the following format:



Multiple Disks

Multiple disks may be initialized by recalling BDSINT with a *B,e,t control statement prior to loading the RCM initializer. If initialization of several cartridges is required for a single drive, loading of a new cartridge should be done prior to issuing the *B,e,t control statement. Multiple disk drives are handled by the proper equipment assignment for the e parameter for each initialization.

6.5.7 ERROR CONDITIONS

The driver recognizes the following categories of errors:

Internal reject on input or output instruction
External reject on input or output instruction
Parity
Seek
Address
Lost data
Protect fault
Checkword
Defective sector
Compare
Time-out

No error recovery is attempted for parity errors, protect faults, or time-out errors.

Several methods of error detection are used during disk transfers. After data reads and writes, a hardware compare function is issued to compare the data read or written with the data found on the file. When an error is detected, a reposition and retry is attempted up to ten times.

Following a data transfer, the current word address status register is monitored to determine if the read/write was of proper length. Ten attempts are made to recover from the error.

If a controller seek error is detected, the driver determines the address to which it has mis-positioned. A computation determines the proper increment of forward or reverse seek. Up to ten retries are made to reposition.

When an irrecoverable error occurs, the driver sets the error field of word 13 of the disk physical device table and the error parameter in the request. The Q register is negative upon entry to the completion program and indicates an irrecoverable error to the user. No information about the nature of the error is passed to the user.

An assembly time option is provided for the user which allows dynamically downing a sector which has gone bad or setting the error field in the disk physical device table. If the option to dynamically down a sector is selected, an alternate is provided if one exists, the request is reissued, and the information is written on the alternate sector. The bad sector with its alternate is entered into the bad sector directory in the PHYSTB and on the disk.

Recovery from a read error due to a bad sector is not possible.

When an irrecoverable error occurs, bits 5-0 of ERRCOD, word 13 of the disk physical device table, contains one of the following error codes.

Contents	Description
0	I/O hang-up; no completion interrupt occurred as a result of initiation of a device operation
1	Internal or external reject
3	Parity error; the controller recives a parity error signal from the direct storage access bus while transferring data or control information
4	Checkword error; the controller logic detects an incorrect checkword in data read from file storage during a read, compare, or checkword operation
5	Internal reject; the system cannot communicate with the device because it is not available
6	External reject; the decive is busy or not ready and cannot perform the specified request
7	Compare error; data read or written does not compare with original data
13	Controller seek error; the controller has been unable to obtain the file address selected during a read, write, compare, or checkword operation (usually an indication of a positioning error)
14	Drive seek error; the drive unit has detected that the cylinder positioner has moved beyond the legal limits of the device during a load address, write, read, compare, checkword check, or write address function
15	Address error; detection of an illegal file address obtained from the computer or the controller has advanced beyond the limits of file storage
16	Protect fault; an unprotected controller operation attempts to write in a protected core location
17	Defective sector error; one of the following conditions occurs Error on write attributed to a defective surface Sector not previously downed and assigned an alternate Dynamic sector downing option not selected
18	Dynamic sector downing error; failed because no free entries exist in the bad sector directory or an attempt was made to down a sector in the bad sector directory

6.5.8 ADDRESSING ALGORITHMS

Two algorithms provide the means of disk sector addressing (FREAD and FWRITE) and disk word addressing (READ and WRITE).

Disk Sector Addressing Algorithm (FREAD and FWRITE)

The system sector number (s) specified by the request parameter list is converted to a seek function as follows:

$$\frac{s}{s_1} = q_1 + r_1$$

The divisor s_1 equals 11774 (the largest physical address on Disk 0) minus NUMALT (the number of alternate sectors the user specified). This operation determines whether disk 0 or disk 1 is to be used. The quotient q_1 equals zero for disk 0 and one for disk 1; q_1 is placed in bit 6 of the file address word. The next dividend r_1 contains the system sector address. This value is added to a BIAS which is the number of sectors required for the bad sector directory. This value varies according to the number of sectors specified as alternates (NUMALT). The BIAS is a minimum of one sector and is incremented for each block of 96 sectors specified as alternates.

$$\frac{r_1 + \text{BIAS}}{29} = q_2 + r_2$$

Decimal 29 is the number of sectors per track. The remainder r_2 is the sector within the track and is placed in bits 4-0 of the file address word. The next dividend q_2 determines the track within the cylinder.

$$\frac{q_2}{2} = q_3 + r_3$$

The divisor 2 is the number of tracks per cylinder. The quotient q_3 is the cylinder and is placed in bits 15-8 of the file address word. If r_3 is 0, the upper track is being used. Otherwise, if r_3 is 1, the lower track is being used. The remainder r_3 is placed in bit 7 of the file address word.

Disk Word Addressing Algorithm (READ and WRITE)

The word number (a) specified by the request's msb and lsb is treated as a double-precision disk word address and is converted as follows.

$$\frac{a}{96} = s + w$$

The word number (a) is msb (15-0) and lsb (14-0). The divisor 96 is the number of words in a sector. The resulting computation yields a sector and word within a sector address. The disk sector algorithm is then used to compute a seek function from s. Transmission of information starts with word w of sector s.

6.5.9 CARTRIDGE REPLACEMENT

On the first read or write of a cartridge disk, the core-resident bad sector directory is updated with the bad sector directory associated with the pack (resident on the pack on sector 0).

Since a user may wish to exchange packs to add or save data, a function ENTR allows the reading and updating of the bad sector directory associated with the newly loaded cartridge. This capability allows the driver to work with a new bad sector directory for present transferring of data.

The control statement is issued following a manual interrupt and has the following format.

ENTR,ℓ,t

ℓ Logical unit (decimal)

t Type of disk

0 Disk 0
1 Disk 1

INITIALIZATION 7

Initialization is the procedure which initializes mass storage and the 1700 system. If the cartridge disk drive is installed, the bad sector initialization module (BDSINT), a stand-alone program must be executed before the system initialization is executed (see Section 6.5.6 for this procedure).

7.1 SYSTEM INITIALIZER

The system initializer loads the Reduced Core Monitor into core from relocatable binary paper tape. Initialization begins at the lowest location (towards 0) of core.

The initializer is composed of the following modules:

BDSINT	Bad sector initialization module (for cartridge disk drive only)
RCMCON	Control module
RCMLDR	Loader module
RCMIDR	Paper tape reader driver
RCMPDR	Paper tape punch driver
RCMCDR	Teletypewriter comment driver

The control module must be the first module read in and the rest of the modules must be loaded in higher core locations. All modules are in run-anywhere form.

7.2 SYSTEM LOADING

When the system initializer is executed and ready to accept control statements, SI is printed on the teletype. Whenever the system initializer is ready to accept another control statement, Q is printed.

The following control statements are acceptable to the system initializer.

*L, hhhh (cr)

Loads from the paper tape reader beginning at location 0, at the location immediately following the last location filled, or at location hhhh if specified, where hhhh is a hexadecimal number greater than the last location filled. The initializer overlays itself only by blank storage.

*D (cr)

When the *D statement precedes an *L statement, the loader information, concerning any previous data block declaration, is deleted. A subsequent data storage declaration causes a new block of core to be assigned as storage for data from subsequent programs.

* (cr)

Continues an interrupted operation; e. g., when the input station empties, the initializer gives a Q. This indicates operator intervention is necessary. To continue loading, the next paper tape is mounted and * (cr) is typed.

*P (cr)

Punches an absolute record of the programs which have been loaded by previous *L statements. The last location + 1 of the last program punched is preserved as the base for absolutizing any programs loaded by subsequent *L statements. Any externals which are not patched when *P is detected are printed at this time.

*T (cr)

Terminates loading. The initializer first gives the names of the unpatched externals. It then punches a paper tape of the system and terminates initialization.

7.3 INITIALIZING THE SYSTEM IN TWO PARTS

If the system with the applications programs is too large for available core, the Reduced Core Monitor and the applications programs may be initialized in two parts. The Reduced Core Monitor through LINK should be loaded first with a *L and punched out with a *P statement in absolute form. The remaining applications and monitor programs are then loaded and punched. It is not necessary to write the applications programs in run-anywhere form. Both absolute tapes may then be loaded by the absolute loader in the order of initialization.

External references made by the Reduced Core Monitor to the applications package cannot be patched; therefore, a method of accessing entry points in the applications package from RCM modules must be devised. One method is to make the last word of the Reduced Core Monitor an entry point (as in LINK) and preface the applications package with a vector table of all applications package entry points. Drivers and their device tables could be grouped with the applications package and the LOG1A table put at the beginning of the a applications package. By making LOG1A an entry point defined as the last location of the monitor, the LOG1A table can be accessed normally by the monitor. External references made by the applications routines to the Reduced Core Monitor module are patched in the two-part initialization.

7.4 ABSOLUTE LOADER

An absolute loader is provided to load the absolutized system from binary cards or paper tape. Details of installing with the absolute loader are found in the installation handbook.

COMMUNICATIONS REGION A

The area of core below FF_{16} is used as a communication region because it can be addressed directly by a one-word instruction. Its contents are defined by the following table. All locations are protected except as noted.

Location	Contents	HEX Equivalent
0 & 1	RTJ to restart routine	(GO1700)
2	0 0	0
3	0 01	1
4	0 011	3
5	0 0111	7
6	0 01111	F
7	0 01 1	1F
8	0 01 1	3F
9	0 01 1	7F
A	0 01 1	FF
B	0 01 1	1FF
C	0 01 1	3FF
D	0 01 1	7FF
E	00001 1	FFF
F	0001 1	1FFF
10	001 1	3FFF
11	01 1	7FFF
12	1 1	FFFF
13	1 10	FFFE
14	1 100	FFFC
15	1 1000	FFF8
16	1 10000	FFF0
17	1 10 0	FFE0
18	1 10 0	FFC0
19	1 10 0	FF80
1A	1 10 0	FF00
1B	1 10 0	FE00
1C	1 10 0	FC00
1D	1 10 0	F800
1E	1 10 0	F000
1F	1110 0	E000
20	110 0	C000
21	10 0	8000
22	0 0	0000
23	0 1	1
24	0 10	2

Location	Contents	HEX Equivalent
25	0 100	4
26	0 1000	8
27	0 10000	10
28	0 10 0	20
29	0 10 0	40
2A	0 10 0	80
2B	0 10 0	100
2C	0 10 0	200
2D	0 10 0	400
2E	0 10 0	800
2F	0 10 0	1000
30	0 10 0	2000
31	010 0	4000
32	10 0	8000
33	1 10	FFFE
34	1 101	FFFD
35	1 1011	FFFB
36	1 10111	FFF7
37	1 101 1	FFEF
38	1 101 1	FFDF
39	1 101 1	FFBF
3A	1 101 1	FF7F
3B	1 101 1	FEFF
3C	1 101 1	FDFF
3D	1 101 1	FBFF
3E	1 101 1	F7FF
3F	11101 1	EFFF
40	1101 1	DFFF
41	101 1	BFFF
42	01 1	7FFF
43	0 101	5
44	0 110	6
45	0 1001	9
46	0 1010	A
47-64	Interrupt stack (expandable)	
65-A2	Reserved for interrupt stack expansion or for a process (unprotected)	
A3-A9	Tops of threads of scheduler stack	
AA-B3	Available for process or for expansion of scheduler thread tops	
B4	Top of thread of empties in schedule stack	
B5	Location of Find Next Request subroutine	
B6	Address of Complete Request subroutine used by drivers	
B7	Address of MASKT	
B8	Core location of top of interrupt stack	
B9	Address of request exit	
BA	Address of volatile storage release routine	
BB	Address of volatile storage assignment routine	
BC	Address of absolutizing routine for logical unit number	
BD	Address of absolutizing routine for starting address	
BE	Unused	
BF	Address of absolutizing routine for number of words	

Location	Contents
C0-C7	Mask table
C8-D0	Available for expansion of mask table
D1-D6	LOG1 table
D7-DC	LOG2 table
DD-E2	LOG1A table
E3-E7	Available for table expansion or process
E8	Real time clock counter
E9	Manual interrupt processor busy flag
EA	Location of dispatcher
EB-EE	Available for process
EF	Current priority level
F0	Core location of first available volatile storage
F1	Last location of initialized system
F2	Largest location of core
F3	Location of error logging file when it is used
F4	Location of entry for systems requests
F5-F7	Used by System Initializer
F8	Location of protect processor when it is in core
F9	Logical unit number of standard input device
FA	Logical unit number of standard binary output device
FB	Logical unit number of standard print output device
FC	Logical unit number of output comment device
FD	Logical unit number of input comment device
FE	Location of common interrupt handler
FF	Memory index register I (unprotected)

PHYSICAL DEVICE TABLES B

The DEVICE TABLES are included in the device's program, RCMDEV, and in the communications area.

PHYSICAL DEVICE TABLE

A Physical Device Table is provided for each I/O device to be operated by a driver. The table includes all the necessary parameters to specify a device and allow it to be operated by its associated driver.

The size of the table is at least 16_{10} words. Words 0 through 15 have a standard function for all devices.

PHYSICAL DEVICE TABLE

		15 14 13 12 11 10 9 8 7 6 5 4 3 2 1 0	Symbolic Name	
Word	0	0 | 0 0 1 0 0 1 | 0 | 0 0 0 0 LEVEL	ELVL	↑ STANDARD FOR ALL DEVICES
	1	Driver Initiator Address	EDIN	
	2	Driver Continuator Address	EDCN	
	3	Driver I/O Hang-up Diagnostic Address	EDPGM	
	4	Diagnostic Clock	EDCLK	
	5	Logical Unit Currently Assigned to this Device	ELU	
	6	Current Request Parameter List Loc.	EPTR	
	7	Converter | Equip. | Station Code	EWES	
	8	Request Status Bits (see table)	EREQST	
	9	Status Bits (see detail)	ESTAT1	
	10	Current Location for Driver	ECCOR	
	11	Last Location+1 for Driver	ELSTWD	
	12	Last Equipment Status Read	ESTAT2	
	13	Logical Unit | Error Code	ERRCOD	
	14	Used by Driver		
	15	Return Addr. for Complete Request	RETURN	↓

Word	Description
0	ELVL $120X. A scheduler request to operate the driver initiator address at level X, the driver priority level.
1	EDIN Driver initiator address.
2	EDCN Driver continuator address. Control is transferred to EDCN on interrupt at the priority level assigned to the interrupt trap region. This priority level must be the same as the priority level specified by word 0.
3	EDPGM Driver error routine address. Control is transferred here when the diagnostic clock is counted down to negative by the diagnostic timer at the driver priority level.
4	EDCLK Diagnostic clock. This location is set by the driver and is counted down by the diagnostic timer to time-out a hardware completion interrupt. Set idle by complete request. (idle =-1)
5	ELU Logical unit currently assigned to the device. Zero if the device is not in use. Set by the request processor and may be reassigned by find next request. Cleared by find next request or complete request.
6	EPTR Call parameter list location for current request. Set by find next request.
7	EWES Hardware Address Bits — Code 0-6 Station 7-10 Equipment 11-15 Converter The equipment status is obtained by loading this word into Q, followed by input. Status is saved in word 12, ESTAT2.
8	EREQST Request Status Bits 15 1 if operation is in progress 0 if operation is complete 14 1 if driver detects I/O hang up

Word	Description		
	13, 12, 11	Equipment class. See table for equipment type.	
	10-4	Equipment type constant (T). See table for equipment type.	
	3	1 if equipment table includes words 18-33 for message buffering	
	2-0	Unused	
9	ESTAT1 Status Word Number 1		
	Bits		
	15	1 if error condition and/or end of file detected	Driver
	14	1 if fewer words read than requested	Driver
	13	1 if device remains ready after detecting an error or end of file	Driver
	12-11	Unused	
	10	1 if parity error occurred	Driver
	9	Unused	
	8, 7, 6	Reserved for special use by individual drivers	
	5	0 if this is a control character	Driver
	4	0 if this is first character	Driver
	3	1 if ASCII, 0 if Binary mode	Driver
	2	1 if lower character, 0 if upper unformatted	Driver
	1	1 if format read or write, 0 if unformatted	RCMFNR
	0	1 if write request, 0 if read request	RCMFNR
10	ECCOR Location where the driver will next store or obtain data. Set initially by Find Next Request.		
11	ELSTWD Location where the driver is to store or obtain data to satisfy the request.		
12	ESTAT2 Status word 2 Last value of equipment status (see word 7).		
13	ERRCOD Error code word for request completion resulting from an error.		
	Bits		
	0-5	Error code. See table.	
	6-15	Logical unit	
14	Used by drivers.		

15 Return address from complete request routine is stored here.

16 and on These words may be added to the device table if they are required for special purposes. For example, they can be used to count the lines per page of output or to link several tables together all using the same driver. The following list contains the Standard Equipment Type Codes, Equipment Class Codes and Error Codes.

Standard Equipment Type Codes

Word 8, EREQST Bits 10-4

0†	1711/1712 Teletypewriter
1	1721/1722 Paper Tape Reader
2	1723/1724 Paper Tape Punch
3	Unassigned
4	1739-1 Cartridge Disk Unit
5	1738-853 Disk Unit
6	1751 Drum Unit
7	1729 Card Reader
8	1738-854 Disk Unit
9	601 Magnetic Tape Unit
10	Software Buffering Device
11	1742 Line Printer
12	1728-430 Card Reader/Punch
13	Software Core Allocator
14	210 CRT Display Station
15	1558 Latching Relay Output
16	1553 External Register Output
17	311B/312B Data Set Terminal
18	322/323 Teletype Terminal
19	501 Line Printer
20	166 Line Printer
21	1612 Line Printer
22	415 Card Punch
23	405 Card Reader
24	608 Magnetic Tape Unit
25	609 Magnetic Tape Unit
26	1713 Teletype Keyboard
27	1713 TTY Paper Tape Punch
28	1713 TTY Paper Tape Reader
29	1729-2 Card Reader
30	1797 Buffered I/O Interface
31	Software Dummy Alternate
32	1584 Selectric I/O Typewriter
33	1582 Flexowriter I/O Typewriter
34	1716 Coupling Data Channel
35	1718 Satellite Coupler
36	Unassigned
37	8000 Series Magnetic Tape Unit

† Numbers are given in decimal and must be converted to hexadecimal before inserting in bits 10-4.

38	1732/608 Tape Driver (D606*2)
39	1732/609 Tape Driver (D606*2)
40	1530 A/D Converter 30/40 PPS
41	1534 A/D Converter 200 PPS
42	1538 A/D Converter High Speed
43, 44	Unassigned
45-99	Reserved for future standard equipment
100-127	Open for user assignment

Equipment Class Codes

Word 8, EREQST Bits 11-13

0	Class not defined
1	Magnetic tape device
2	Mass storage device
3	Card device
4	Paper Tape device
5	Printer device
6	Teletype device
7	Reserved for future use

Error Codes

Word 13, ERRCOD Bits 0-5

0	I/O hangup
1	Internal or external reject
2	Alarm
3	Parity error
4	Checksum error
5	Internal reject
6	External reject
7	Echo check error on punch
8	Illegal Hollerith punch
9	Attempt to read or punch cards in binary
10	Unused
11	Change from read mode to punch mode or vice versa (cards)

LOGICAL DEVICE TABLES

Word 0 in each of the LOG tables contains the largest logical unit number on the system.

LOG1 — The subsequent locations of the LOG1 table correspond in order with the logical unit numbers; each location contains an index to the LOG1 table. This means that a logical unit number may be used as a pointer to the LOG1 table to pick up the pointer to the LOG1A table location which is pertinent to that logical unit.

LOG1A Each subsequent location of LOG1A contains the address of the PHYSTB slot corresponding to the logical unit number used as a pointer to the LOG1 table to pick up the pointer to this location. The entries on the LOG1A are grouped according to the interrupt line they use; i. e., all devices which interrupt on line 1 are first, all devices which interrupt on line 2 are next, etc. The line groups must be in order but the logical unit numbers need not be.

LOG2 Each subsequent location of LOG2 contains the top of the thread for the requests of a particular device. The order of information is the same as in the LOG1 table.

SYSTEM COMMENTS AND DIAGNOSTICS C

System comments indicate to the operator that an error has occurred or advise him that a system program has completed one operation and is ready for the next one.

Initializer Comments and Diagnostics

The following comments are issued to indicate operator action is needed:

SI	System initializer ready to begin operation; output comment device
Q	System initializer has completed previous operation and the initializer is waiting for the next directive

The following are diagnostics:

E1	Binary input record checksum error
E2	Incorrect control statement
E3	Binary record incorrect or out of order
E4	Incorrect common storage reservation
E5	Program too long, causing core overflow
E6	Attempt to load program below top of system
E7	Data storage assigned beyond storage limit
E8	Duplicate entry point
E9	Unused
E10	Unpatched external
E11	Unused
E12	Two programs reference the same external name; one with absolute addressing and one with relative addressing
SLEW	Indicates an irrecoverable error is occurring while loading a program. The remainder of that program is bypassed.

System Diagnostics and Messages

These diagnostics, issued by the RCM, indicate a catastrophic condition which hangs the system:

PF	Power failure, parity fault in core, or protect fault if the protect processor is not in core and the PROTECT switch is activated
OV	Overflow of volatile storage

These comments or diagnostics are not catastrophic and do not hang the system:

GI n	Ghost interrupt from a non-existent device on line n
RC	A request was issued with an illegal request code
hhhh, PF	Protect fault from the protect processor while in the system; hhhh is the hexadecimal location of the error
LnnFmm	Logical unit nn failed with error mm; message is printed by the LD absolute loader if the input unit fails. (Error code mm is the same as those listed in Appendix B, Word 13 of Physical Device Table)
EOL	End of load message put out by the absolute loader when the paper tape unit is empty
MI	Manual interrupt processor ready for input

ASCII CONVERSION TABLES D

The 1963 American Standard Code for Information Interchange (ASCII) is used by the 1700 Reduced Core Monitor. The three different ASCII options (ASCII 63, ASCII 68 and CDC ASCII subset of ASCII 68) used by the 1700 Reduced Core Monitor are tabulated in the following tables. These are all a 64-character subset. ASCII code uses 8 bits. Bit 8, which is always zero, is omitted in the following table. Bits 1 through 4 contain the low order 4 bits of code for the character in that row. Bits 5-7 contain the high order 3 bits of the code for the character in that column. The first table details the ASCII 63 standard.

ASCII Symbol	Bit Configuration	Hexadecimal Number	Hollerith Punch
Δ	010 0000	20	No punch
!	010 0001	21	11-8-2
"	010 0010	22	8-7
#	010 0011	23	12-8-7
$	010 0100	24	11-8-3
%	010 0101	25	0-8-5
&	010 0110	26	8-2
'(APOS)	010 0111	27	8-4
(	010 1000	28	0-8-4
)	010 1001	29	12-8-4
*	010 1010	2A	11-8-4
+	010 1011	2B	12
, (Comma)	010 1100	2C	0-8-3
-	010 1101	2D	11
.	010 1110	2E	12-8-3
/	010 1111	2F	0-1
0	011 0000	30	0
1	011 0001	31	1
2	011 0010	32	2
3	011 0011	33	3
4	011 0100	34	4
5	011 0101	35	5
6	011 0110	36	6
7	011 0111	37	7
8	011 1000	38	8
9	011 1001	39	9
:	011 1010	3A	8-5
;	011 1011	3B	11-8-6
<	011 1100	3C	12-8-6
=	011 1100	3D	8-3
>	011 1110	3E	8-6
?	011 1111	3F	12-8-2

ASCII Symbol	Bit Configuration	Hexadecimal Number	Hollerith Punch
@	100 0000	40	0-8-7
A	100 0001	41	12-1
B	100 0010	42	12-2
C	100 0011	43	12-3
D	100 0100	44	12-4
E	100 0101	45	12-5
F	100 0110	46	12-6
G	100 0111	47	12-7
H	100 1000	48	12-8
I	100 1001	49	12-9
J	100 1010	4A	11-1
K	100 1011	4B	11-2
L	100 1100	4C	11-3
M	100 1101	4D	11-4
N	100 1110	4E	11-5
O	100 1111	4F	11-6
P	101 0000	50	11-7
Q	101 0001	51	11-8
R	101 0010	52	11-9
S	101 0011	53	0-2
T	101 0100	54	0-3
U	101 0101	55	0-4
V	101 0110	56	0-5
W	101 0111	57	0-6
X	101 1000	58	0-7
Y	101 1001	59	0-8
Z	101 1010	5A	0-9
[	101 1011	5B	12-8-5
\	101 1100	5C	0-8-2
]	101 1101	5D	11-8-5
↑	101 1110	5E	11-8-7
←	101 1111	5F	0-8-6

By an assembly option, 1968 ASCII Standard may be used on the RCM.

Hexadecimal Number	ASCII Symbol	ASCII 68 Hollerith	Hollerith Punch for Conversion
21	!	12-8-7	11-8-2
23	#	8-3	12-8-7
25	%	0-8-4	0-8-5
26	&	12	8-2
27	'	8-5	8-4
28	(	12-8-5	0-8-4
29	)	11-8-5	12-8-4
2B	+	12-8-6	12
2D	-	11	11 or 8-4
3A	:	8-2	8-5
3C	<	12-8-4	12-8-6
3D	=	8-6	8-3
3E	>	0-8-6	8-6
3F	?	0-8-7	12-8-2
40	@	8-4	0-8-7
5B	[	12-8-2	12-8-5
5D	]	11-8-2	11-8-5
5F	_	0-8-5	0-8-6

By an assembly option the CDC ASCII subset may be used. The ASCII 68 standard is modified to reflect the Control Data subset. Hexadecimal code 5B (12-8-2 punch) is replaced with 7B; and 5D (11-8-2) is replaced with 7D. The 12-0 punch is interpreted as 7B; and the 11-0 punch is interpreted as 7D.

GLOSSARY E

Absolute Program	A core image of a program which may be read directly into core by a checksum read and executed.
Applications Package	All programs which are not part of the RCM package.
Completion Routine	A segment of coding which is to be executed on completion of an I/O request.
Data Block	Storage area proceeding a program or group of programs which may be accessed by more than one program. Equivalent to labeled common.
Entry Point	A location within a program which is meant to be accessed by other programs.
External	A location which has been defined as an entry point in another program and is to be referenced.
Unpatched External	An external for which no entry point has been found by the System Initializer.
Hang	The system stops normal meaningful processing and begins repetitious processing because it becomes caught in a loop of instructions.
Interrupt	A signal coming in on one of the interrupt lines temporarily stops the program being executed and causes some other routine to begin execution.
Internal Interrupt	An interrupt generated by the mainframe (parity fault, protect fault, power failure).
External Interrupt	An interrupt generated by some peripheral device.
Interruptible	An interruptible operation is one which can be stopped by an interrupt from a higher priority level and then restarted later after the interrupting operation has finished.
Idle Loop	A do-nothing loop in the system which is executed as priority -1 when there are no other programs to be executed.
Loop	A segment of coding which is repeated by the last instruction is a jump to the beginning of the segment.

Overlay	Load or read in a new program or data over all or part of a program already in core.
Priority Level	All programs are assigned a priority level; the use of the central processor is determined by priority level. The highest program priority is 6; the lowest is -1.
Process	A process or process control application is a function external to the machine which is to be monitored or controlled by the 1700. Programs performing operations with respect to a process are referred to as process programs.
RCM Package	This refers to the Reduced Core Monitor with its drivers and any of the optional modules which are being used.
Re-entrant	This refers to a program which may be interrupted, called by the interrupting program and resumed at the interrupted point, all with no loss of continuity.
Relocatable	A program which includes control information regarding program name, entries, externals, transfer address, and command sequence storage. It may be loaded anywhere in absolute form by a relocating loader.
Run-anywhere	Programs that can be moved and successfully operated elsewhere in core after being loaded by a relocating loader.
Schedule	Put into operation another program at a given priority by means of a monitor request.
Schedule Up	Schedule a program at a higher priority.
Schedule Down	Schedule a program at a lower or equal priority.
System Initializer	Routine to install the RCM system (with application programs) from relocatable binary tape.
Task	A discrete piece of work to be done by a program.
User	A programmer or process or job that uses the 1700 Reduced Core Monitor.
Volatile Storage	Protected core memory reserved for intermediate storage of data for re-entrant system programs.
Wrap-around	If an address referenced is beyond the largest location of core, the location referenced is relocated by hardware to a lower existing memory bank location according to the machines memory bank addressing algorithm.

FIFTEEN BIT ARITHMETIC

F

Fifteen bit arithmetic is implied with all relative address calculations in the parameter list. Fifteen bit arithmetic is required to make the backward relative addressing mode possible. Fifteen bit ones complement arithmetic on a 16 bit ones complement adder simply means that a borrow from bit 14 of A must be end-a-round.

For addition the following rules apply:

1. Truncate both arguments to 15 bits.
2. Set bit 15 of either argument. Bit 15 of both arguments must be different.
3. Add the two arguments and take the result modulo 15 bits.

Example:

Calculate $L_1 + L_2$, where L_1 is 5_{16} and L_2 is 20_{16}

	8005	L_1 is A with bit 15 set
subtract	FFDF	$-L_2$ from storage (bit 15 of $+L_2$ is zero)
	0026	$L_1 + L_2$
subtract	1	end around borrow from first result
	0025	result modulo 15 bits

For subtraction the following rules apply:

1. Truncate both arguments to 15 bits. Bit 15 of both arguments must be the same.
2. Subtract the two arguments and take the result modulo 15 bits.

Example:

Calculate $L_1 - L_2$, where L_1 is 5 and L_2 is -1B

	0005	L_1 is A modulo 15 bits
subtract	7FE4	L_2 from storage modulo 15 bits
	8021	
subtract	1	End around borrow from first subtraction
	8020	
	0020	Result modulo 15 bits

INDEX

COMMENT SHEET

MANUAL TITLE 1700 Reduced Core Monitor Reference Manual

PUBLICATION NO. 60280600 REVISION A

FROM: NAME:

BUSINESS ADDRESS:

COMMENTS:

This form is not intended to be used as an order blank. Your evaluation of this manual will be welcomed by Control Data Corporation. Any errors, suggested additions or deletions, or general comments may be made below. Please include page number references and fill in publication revision level as shown by the last entry on the Record of Revision page at the front of the manual. Customer engineers are urged to use the TAR.

CUT ALONG LINE

AA3419 REV. 11/69 PRINTED IN U.S.A.

NO POSTAGE STAMP NECESSARY IF MAILED IN U. S. A.

FOLD ON DOTTED LINES AND STAPLE

STAPLE

STAPLE

FOLD

FOLD

FIRST CLASS
PERMIT NO. 8241

MINNEAPOLIS, MINN.

BUSINESS REPLY MAIL
NO POSTAGE STAMP NECESSARY IF MAILED IN U.S.A.

POSTAGE WILL BE PAID BY

CONTROL DATA CORPORATION
Technical Publications
4201 N. Lexington Ave.
St. Paul, Minnesota 55112

ARH220

CUT ALONG LINE

FOLD

FOLD

>>> CUT OUT FOR USE AS LOOSE-LEAF BINDER TITLE TAB

CONTROL DATA
CORPORATION

8100 34th AVE. SO., MINNEAPOLIS, MINN. 55440

LITHO IN U.S.A.