DOS-III ERS OCT 9, 1972

Table of Contents

Oct 9/

Table of Contents (Cont.)

10/9/72

EXTERNAL REFERENCE SPECIFICATIONS

1.0 PURPOSE & SCOPE

With the introduction of new applications for disc operating systems, both within the division and without (data communications, business & data base management, EPG applications, etc.)--it had become necessary to upgrade DOS-M to meet the needs of its expanding market. The changes desired will result in a new system - DOS-III.

DOS-III provides all the features of the latest released version of DOS-M, plus the additional ones described here. Modules of DOS-III that replace modules of DOS-M are:

1) DOS-III Exec Modules
2) DOS-III Disc Monitor
3) DOS-III System Generator
4) DOS-III Bootstrap Loader
5) DOS-III Job Processor
6) DOS-III Relocating Loader
7) DOS-III I/O Drivers

(All references to extensions & additions within this ERS refer to extensions & additions to DOS-M.)

2.0 HARDWARE REQUIREMENTS

To take advantage of all features in the DOS-III system, 12K of core will be required. In addition to the hardware options described in the DOS-M manual, the following is available:

Privileged interrupt I/O card (12620A)

10/9/72

3.0 MAJOR ADDITIONAL FEATURES

3.1 Additional base page

3.1.1 System buffer moved

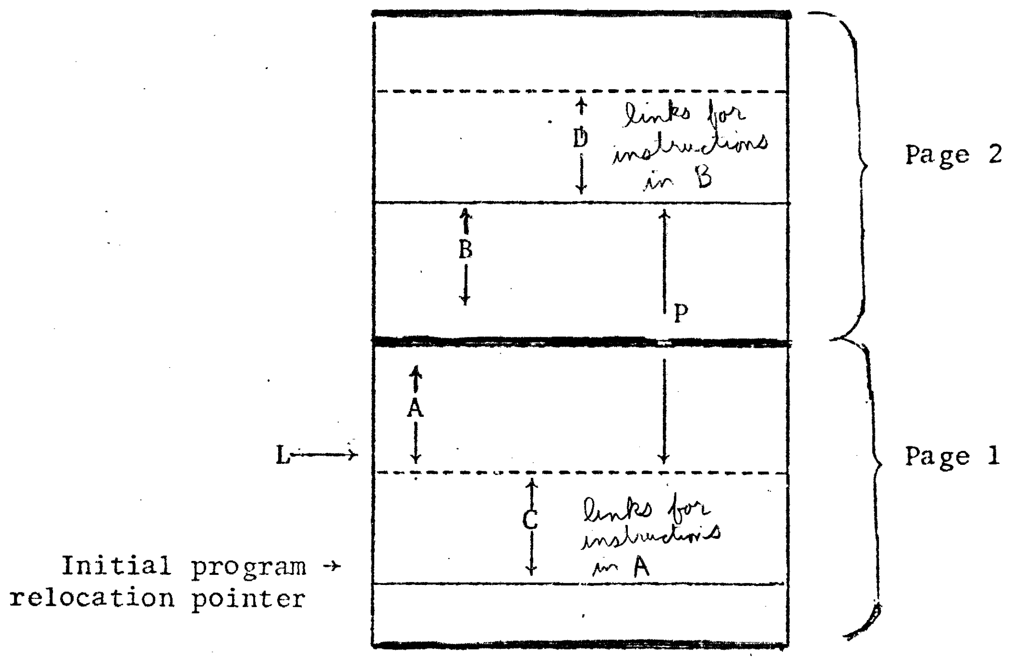
A 128 word system buffer was moved off the base page into the system area to provide more room for base page links. The total number of base page links available for system and user links is now 803_{10}.

3.1.2 Current page linking

The Relocating Loader will incorporate an algorithm to place necessary links on the current page, as opposed to the base page. (12K of core will be required for this Loader). This feature provides more area on the base page to allow complete loading of large programs in a large core system.

The algorithm is essentially that used by the Loader/2100:

Given a program that crosses a page boundary


D
links for instructions in B
Page 2
B
P
A
L
Page 1
C
links for instructions in A
Initial program relocation pointer

10/9/72

a) The NAM record provides the program length P.

b) The Loader computes a "guess" area for primary links (C) as follows: (MIN(A/2),B)/4

c) The program is relocated starting at location L. Links required for instructions in A will be placed in C; links for instructions in B will be placed in D.

d) A second relocation is done after adjusting the primary links area (C) (its size is decreased by the number of unused words). The absolute code generated from this second relocation is output to the disc.

In order to get the most effective use out of this algorithm, the user should be aware of some restrictions and considerations:

a) Current page linking may not be used on programs which, at execution time, use core following the program area for writing out data. (e.g. Assembler builds its symbol table immediately following the last word of the largest segment.)

b) Programs should be broken into subroutines of less than 2K. This stems from the fact that, using this algorithm, links are generated only at the beginning and the end of the program. (Links cannot be inserted in the middle of a program since there is no guarantee that the boundary between program and links will notfall in the middle of a skip or jump *+x sequence.) If the program spans more than two pages, the middle page will have no area available for current links, and will therefore have to default to base page links. here not [illegible]

3.2 Privileged interrupt

When the special privileged interrupt I/O card is included in the system, DOS-III allows a class of privileged interrupts to be processed independently of regular DOS-III operation with a minimal delay in responding to interrupts (privileged interrupts can be recognized within 100 microseconds).

non-priv
time from interrupt to time driver gets control -- 1st guess: ~ 2 ms

10/9/72

The I/O card separates the privileged (high priority, low channel numbers) interrupts from the regular system-controlled interrupts. When this card is used, DOS-III does not operate with the interrupt system disabled, but rather, sets control on the special I/O card to hold off lower-priority interrupts. The privileged interrupts are enabled when DOS-III is running & they can interrupt any DOS-III operation. The routines that handle privileged interrupts must be completely independent of DOS-III.

3.2.1 System Preparation for Privileged Interrupt

The decision to include the privileged interrupt option in a DOS-III system is made at system generation time. If the special I/O card is to be used, the system requires two things:

1) the actual hardware location of the privileged interrupt I/O card. (This is accomplished by transferring the user's response to the "PRIV. INT. CARD CHNL?" question during system generation to a base page location - DUMMY)

2) the addresses of the privileged interrupt processing routines. Since privileged interrupts are handled independently of the system, the I/O trap cell must be filled with a subroutine jump directly to the entry point of its associated special routine. DOS-III and its central interrupt processor ($CIC) are not aware of these interrupts. (The addresses of the privileged interrupt routines are provided by the "ENT,name" option when configuring the interrupt table during system generation)

3.2.2 System Processing of Privileged Interrupts

During interrupt processing, the central interrupt processor ($CIC) interrogates the DUMMY flag to determin if privileged interrupts are possible. If the flag is zero (no privileged interrupts) the interrupt system is left disabled & normal processing continues. If the flag is non-zero (privileged interrupts possible) a STC is issued to the I/O

10/9/72

needs explanation

location containing the privileged interrupt card. (This holds off lower priority interrupts until the control on the special card is cleared). Then Bit Ø of the memory protect flag (MPTFL) is set equal to one to indicate that memory protect is off, the control flip-flop on each DMA channel is cleared to defer DMA completion interrupts, and the interrupt system is re-enabled. (The interrupt location of the special card contains a zero so that interrupts from the card are ignored).

Following completion of normal system processing, a routine ($IRT) is executed. $IRT clears control and sets the flag on the special I/O card, clears bit 0 of MPTFL to indicate that memory protect may be re-enabled and sets control on the active DMA channels if bit 15 of the DMA interrupt entries equals one.

3.2.3 Privileged Interrupt Routines

Privileged interrupt routines are responsible for saving & restoring all registers, and restoring memory protect to its original state prior to special interrupt. Because any interrupt (in hardware -- in fact, only may to) automatically disables memory protect, privileged interrupt routines must interrogate the memory protect flag (MPTFL) prior to returning to the point of interruption. If MPTFL equals zero, memory protect was on and a STC 5 must be issued. If MPTFL is not equal to zero, memory protect was off and a STC 5 must not be issued.

Privileged interrupt routines must not use any features or requests of DOS-III, nor use either DMA channel. They can communicate with normal user programs by use of the appropriate COMMON region.

4.0 MODIFICATION TO JOB PROCESSOR

Two new features will be provided within the Job Processor:

1) Creation of (and operation on) job-deck (JD) files. (JD files are source files (Type SS) with colons in columns 1 (directives) and 2 (end of source indicators)).

10/9/72

2) A batch abort feature to terminate the current batch job & continue processing in the batch mode.

4.1 Job Deck Files

4.1.1 Storing JD files

Job deck files may be stored on disc via the new store directive :ST,S,XXX,lu,C. When the colon option is included, input is terminated by a triple colon (:::). Any other statement with a colon in columns 1 and/or 2 will be interpreted as data and transferred to the designated source file. A few restrictions apply here---

a) the logical unit specified in the :ST,S directive must not be the current batch device. (This is done to protect subsequent users in a batch environment.) If it is, ILLEGAL LUN will be output and further operation will continue depending on the current mode. If the user is in the keyboard mode, an @ will be output and the system will be ready to accept a new directive. If the user is in the batch mode, a batch abort will be performed (see 4.2).

b) if the logical unit specified in the :ST,S directive is equal to 1 (ie. the system console), all input will be transferred to the designated source file until ::: is encountered with two exceptions:

:OFF

:ABORT

If either of the above are encountered when storing from the keyboard, they will be interpreted as directives and executed as such (ie. :OFF will return control to the keyboard mode without terminating the job, and action following :ABORT depends on the mode the user is in. If the :ST,S directive was entered from the system console, the current job will be aborted. If the :ST,S directive was entered from the batch device, a batch abort will be performed.)

10/9/72

NOTE: JD files containing :OFF and :ABORT can be created by storing from a device other than the system console and batch device.

4.1.2 Editing JD Files

Job deck files may be edited - with a few restrictions:

a) If the logical unit number specified in the EDIT directive is equal to the current batch device, the source statements included in the edit file may not contain a colon in column 1. (This is done to protect subsequent users in a batch environment.) If a colon is encountered in column 1 of an edit file statement from the batch device, an error message will be output and action will proceed according to the mode (ie. Keyboard mode - enter new directive; Batch mode - batch abort).

b) If the logical unit number specified in the EDIT directive is equal to 1 (ie. system console), the source statements included in the edit file may contain anything other than

:OFF

:ABORT

If either of the above two are encountered, they will be interpreted as directives and processed as such.

NOTE: JD files containing :OFF and :ABORT may be edited by entering the edit file from a device other than the system console or the current batch device.

4.2 Batch Abort

The batch abort feature has been added to provide the capability of terminating the current batch job when a given class of errors is encountered, and proceeding to the next batch job without operator intervention. The class of errors which (when encountered in the batch mode) will result in a batch abort is given in Appendix A. Action following one of these errors proceeds as follows: p50

10/9/72

1. The statement causing the termination, along with its error message, will be echoed on the list device.

2. JOB ABORTED will be output to the system console and echoed on the list device.

3. The mode will remain as batch and the batch device will be scanned until either a :JOB or an :EJOB directive is encountered.

4. Processing of either directive (:$\overset{\text{JO}}{\text{IO}}$ or :EJ) will result in terminating the original job.

NOTE: If the error resulted from encountering a statement with a colon in column 1, that statement will be checked for one of the above directives before checking further, thus minimizing the possibility of one user destroying the next user's job.

Additional capabilities are provided by expanding existing directives and incorporating several new ones.

4.3 New Directives

4.3.1 CLEAR

Purpose - To issue a clear request to an I/O device (or job binary area).

Format - :CLEAR [,logical unit]

where logical unit is the logical unit number of the device to be cleared. (If no logical unit is specified, the job binary area of the disc is cleared.)

4.3.2 RENAME

Purpose - To rename a specified user file, and optionally to change its program type. (Follows the :SS condition)

Format - :RNAME, oldname, newname [,type]

where oldname specifies the file to be renamed
newname specifies the new name for the file
type specifies the new file type for this new file

10/9/72

Comments - If any file name on one of the active subchannels is the same as newname, the message

DUPLICATE FILE NAME

is printed and the file name is not changed. If the file named oldname cannot be found on any of the active subchannels, the message

oldname UNDEFINED

is printed.

System restrictions on the 'type' parameter are:

a) it must lie in the range $0\text{-}12_{10}$.

b) file compatability must be maintained

Program types 0-5 require 11 word directory entries

types 6-12 require 5 word directory entries

Any attempt at mixing types will result in a

PARAMETER ILLEGAL error message.

4.3.3 REWIND

Purpose - To rewind a magnetic tape

Format - :RWND [,logical unit]

- where logical unit is the logical unit number of the desired magnetic tape (default is 8).

4.3.4 TOP-OF-FORM

Purpose - To issue a top-of-form to a list device

Format - :TOF [,logical unit]

where logical unit is the logical unit number of the desired list device (default is 6).

4.3.5 User Exec Module Directives

Purpose - To execute user Exec modules.

Format - :EA - calls up Exec Module $EX36.

:EB - calls up Exec Module $EX37.

4.3.6 Data Communication Exec Module Directives

Purpose - To execute Data Communications routines

Format - :REMOTE,name[,P1,P2,P3,P4,P5] same as :PROG directive
:(to be announced)

A description of the functions of these directives will be provided by the Data Communications project.

10/9/72

4.4 Extensions to Existing Directives

4.4.1 :ABORT

Restrict the ABORT directive to the keyboard only mode.

4.4.2 :BATCH

The logical unit specified in a BATCH directive may not have been previously declared as a null device (See LU directive). Any attempt at assigning a null device as the batch device will result in ILLEGAL LUN.

4.4.3 :DUMP

File type 14_8 (absolute binary) may now be dumped.
The format is as follows:

File Type	Punch Device	List Device
Absolute binary	Absolute binary records	8 octal words/line

4.4.4 :EDIT

Three new edit operations are to be incorporated:

a) /S[UPPRESS] - suppress echoing of the edit file operations on the system console (provided logical unit $\neq$ 1).

b) /U[NSUPPRESS] - resume echoing of the edit file operations on the system console. (This is the default option - restored at the beginning of each :EDIT directive).

c) /M[ERGE][,k], secondary file [,m[,n]]
where

k	is the line number of the file named in the :EDIT directive at which the secondary file is to be merged. If k = 0, the secondary file (or portions thereof (as specified by the m & n parameters)) is inserted at the beginning of the primary file named in the EDIT directive. If k is omitted, the secondary file is inserted at the end of the primary file.
secondary file	is the name of the source file to be merged with the primary file named in the EDIT directive. Portions of this file may be used by specifying source statement numbers m & n.

10/9/72

m,n — specify sequence numbers of the source statements in the secondary file to be used. All former restrictions on m & n still apply.

NOTES:

1) The types of source statements (statements following /I or /R in an edit file) allowed in an edit file are dependent upon the device from which the edit file comes.

   a) From the current batch device - no source statement may have a colon in column one.

   b) From the system console - a source statement may have a colon in column one and be interpreted as data as long as it is not

      :OFF
      :ABORT

      (The above two will be interpreted as directives).

   c) From any other device - a source statement may contain anything other than a / in column one. Input is terminated only by /E.

2) Action following detection of an error in the edit file is dependent upon which device the edit files come from.

   a) If the edit file is coming from the system console, and either of the following errors is encountered:

      PARAMETER ILLEGAL
      NO SOURCE

      The user may re-enter the statement in error.

   b) If the edit file is coming from a device other than the system console, the edit directive (keyboard mode) or the entire job (batch mode) will be aborted.

4.4.5 :EJOB

A top-of-form will be issued to the list device prior to printing the END JOB message instead of following it.

4.4.6 :JOB

A top-of-form will be issued to the list device prior to printing the JOB message.

10/9/72

4.4.7 :LIST

A top-of-form will be issued to the device specified by the logical unit number prior to execution of the LIST directive.

4.4.8 :LU [,n_1[,n_2]]

This directive has been expanded to allow $n_2 = 0$. If $n_2 = 0$, the logical unit specified by n_1 will be interpreted as a null device and any request to perform I/O on that device will be ignored. (This option is not allowed for $n_1 = 1,2,3$ or the current batch device).

NOTE: All forms of the LU directive will be allowed under operation attention.

4.4.9 :PDUMP & :ADUMP

The output resulting from these two directives will be labeled so that they may be distinguished.

4.4.10 :STORE

a) Type X files

A new file type - absolute binary (AB) - type 14_8 may be stored. The format is

```
:STORE,X,file,logical unit
```

where file is the name of the user file to be filled with absolute binary programs from the logical unit specified.

When an end-of-tape is encountered,

```
DONE?
```

will be asked. If there are more tapes, the operator places the next tape in the input device and replies NO , otherwise, the answer's YES.

b) Type S files

An optional parameter has been added to the :ST,S directive. The format is

```
:STORE,S,file,logical unit [,C]
```

If the colon parameter (C) is included, <u>and</u> the logical unit does not refer to the current batch device, a colon in column one will

10/9/72

not be interpreted as the end of the source file. In this mode, a triple colon will be required to terminate source file input.

4.4.11 :OFF

Restrict the OFF directive to the keyboard only mode.

5.0 MODIFICATIONS TO EXEC CALLS

Additional capabilities are provided by expanding existing exec calls and incorporating several new ones.

5.1 New Exec Modules and Calls

Four new exec modules (along with four request codes to access them) will be recognized by DOS-III. Two of the exec modules are reserved for Data Communications use, and the other two are available to the user. If any of these are not included at system generation time, DSGEN will not interpret them as undefined externals, but any attempt to reference them during system execution will result in an RQ error.

5.1.1 Data Communications Exec Modules

The two exec modules reserved for Data Communications will be $EX34 and $EX35. They may be executed by an exec call with request codes 25_{10} and 26_{10} respectively. A description of the functions of these two exec modules will be provided by the Data Communications project.

5.1.2 User Exec Modules

The two exec modules available for the user will be $EX36 and $EX37. They may be executed by an exec call with request codes 27_{10} and 28_{10}

10/9/72

respectively. The user's particular exec module will determine the meaning of the parameters - the only restriction is that the number of parameters in the exec call may not exceed 8. really 6 parameters

Assembly Language

```
           EXT EXEC
              :
           JSB EXEC               (Transfer control to DOS-M)
           DEF *+2 (to 8)         (Determine # of parameters)
           DEF RCODE              (Request code)
           DEF PRAM1              (First optional parameter)
              :
           DEF PRAM6              (Sixth optional parameter)
              :
   RCODE   DEC 27 (or 28)         (Request code)
   PRAM1   ---                    (Up to 6 words of parameter information)
   PRAM6   ---
```

Fortran

```
   IRCD = 27 (or 28)
   CALL EXEC (IRCD[,P1,...P6])
```

5.1.3 Program Segment Return

Purpose - To return from a segment to the main program at the instruction immediately following the program segment load call. (ie. to provide a subroutine-like return from a segment to a main program)

Assembly Language

```
           EXT EXEC
              :
           JSB EXEC               (Transfer control to DOS-M)
           DEF *+2 (to 7)         (Determine # of parameters)
           DEF RCODE              (Request code)
```

10/9/72

```
          DEF PRAM1            (First optional parameter)
            :
          DEF PRAM5            (Fifth optional parameter)
            :
RCODE     DEC 29               (Request code = 29)
PRAM1     ---                  (Up to 5 words of parameter information;
PRAM5     ---                   passed to the main program as parameters
                                are passed to a suspended program.  See
                                PROGRAM SUSPEND)
```

Fortran

```
IRCDE = 29
CALL EXEC (IRCDE[,P1,...P5])
```

5.1.4 Control of Memory Protect

Purpose - To control memory protect from a user program.

Assembly Language

```
          EXT EXEC
            :
          JSB EXEC             (Transfer control to DOS-M)
          DEF *+3              (Return point)
          DEF RCODE            (Request code)
          DEF MPTK             (Memory protect parameter)
            :
RCODE     DEC 30               (Request code = 30)
MPTK      DEC n                (n = 0; memory protect on
                                n ≠ 0; memory protect off)
```

Fortran

```
IRCDE = 30
MPTK = 0 (or 1)
CALL EXEC (IRCDE, MPTK)
```

NOTE: Any program termination (whether normal or aborted) will restore memory protect

10/9/72

Jan 19, 1973

5.2 Extensions to Existing Exec Calls

see p 3-5

5.2.1 Changes to CONWD for Read/Write

	0	0	W	J		A		K	V	M	Logical Unit #					
Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0

Field	Function
M	Determines the mode of data transfer. If M=Ø, transfer is in ASCII character format, and if M = 1, binary format. (Disc is always binary).
V	1) When reading variable length records from punched tape devices in binary format (M=1), if V=0 the record length is determined by buffer length. If V=1, the record length is determined by the word count in the first non-zero character which is read in. 2) When outputting ASCII records to a list device (M=0), if V=0 the 1st character in the buffer will be interpreted as a carriage control character (see Line Printer Formatting). If V=1, single spacing is assumed, & the entire buffer (including the first character) is output to the list device.
K	Used with keyboard input, specifies printing the input as received if K=1. If K=0, "no printing" is specified.
A	When inputting or outputting variable length binary records (M=V=1), the A bit specifies absolute binary format.
W	If W=1, control is returned to the calling program after starting the I/O transfer. If W=Ø, the system waits until the transfer is complete before returning.
J	If J=1 and the logical unit refers to the disc, a backward track increment will be performed (i.e. JBIN read/write). At this time, the system makes no checks to assure that the track and sector count will not overflow.

Jan 19, 1973

5.2.2 I/O Control see p 3-10

The function code field in the CONWD has been expanded to include code=0. If the function code=0, a clear request is issued to the I/O device specified by the logical unit #.

5.2.3 Work Area Limits see p 3-13

An optional parameter will be added to allow the user to obtain work area limits on the current user disc. If the parameter is omitted, or its value is equal to zero, the work area limits returned will refer to the system disc.

```
                        JSB EXEC
                        DEF *+5 (or 6)
                        DEF RCODE       (=17)
                        DEF FTRAK
                        DEF LTRAK
                        DEF SIZE
                        DEF DISC   (optional parameter)
         DISC    DEC    n       (n=0→ system disc; n≠0⇒current user disc)
```

5.2.4 File Read/Write see p 3-7

An optional parameter will be added to provide the user with information concerning a file read/write overflow. If the parameter is omitted and an overflow occurs, an IT error will occur. If the parameter is included and an overflow occurs, control will be returned to the user program with the optional parameter set equal to the number of words (+) or characters (-) (as defined by BUFFL) remaining in the file. If the optional parameter is included and no overflow occurs, the value of the parameter is set equal to zero.

```
         JSB EXEC         (Transfer control to system)
         DEF *+7(or 8)    (Point of return from system)
         DEF RCODE        (Request code)   (= 14/15)
         DEF CONWD        (Control information)
         DEF BUFFR        (Buffer location)
         DEF BUFFL        (Buffer length)
         DEF FNAME        (File name)
         DEF RSECT        (Relative sector within file)
         DEF PARAM        (area left in file if overflow)
```

10/9/72

5.2.5 Program Loading

An alternate entry point flag (AEPF, location 135_8) will be incorporated into the system. During a program load (segment (request code = 8) or main (request code = 10)) this flag will be interrogated. If its contents are zero, control will be transferred to the programs main entry point. If the contents of AEPF are non-zero, it will be interpreted as the address of the alternate entry point of the desired program. After clearing this flag, control will be transferred to the program via JMP AEPF, I.

5.2.6 File Search   see 3-21

An optional parameter will be added to allow the user to search the user or system directory without wait. If the parameter is omitted, the file search will be performed on the user directory with wait.

```
      JSB   EXEC        (Transfer control to system)
      DEF   *+4(or 5)   (Point of return from system)
      DEF   RCODE       (Request code)
      DEF   FNAME       (File name)
      DEF   NSECT       (Number of sectors)
      DEF   PARAM       (Optional parameter)
PARAM DEC   n           (n=0 user search with wait
                         n=1 user search without wait
                         n=2 system search with wait
                         n=3 system search without wait)
```

Jan 19, 1973

6.0 CHANGES IN DSGEN (SYSTEM GENERATOR) OPERATING PROCEDURES

6.1 Initialization Phase

6.1.1 Privileged interrupt information

Following the 'TIME BASE GEN CHNL?' question,

PRIV INT CARD CHNL?

will be asked. Respond with the octal channel number of the privileged interrupt card (and all devices in channels below the card become privileged) or zero if the card is not used.

6.1.2 DMA channel information

The 'IS 2114?' question will be reworded as follows:

"# DMA CHANNELS?"

Respond with the number of DMA channels available in the system (1 or 2).

6.2 PROGRAM INPUT PHASE

The action following ERRØ2, ERRØ3, and ERRØ4 has been changed to:

Action - computer halts; to try again, reposition tape to beginning of program and press RUN.

6.3 DISC LOADING PHASE

6.3.1 Base Page Link Area

The two questions '# SYSTEM LINKS?' and '# USER LINKS?' asked at the beginning of the Disc Loading Phase will be replaced by the single question:

LINKS?

The operator responds with a decimal number of links. The number of links must not exceed 803_{10}. If the operator responds with a "blank", the maximum number of links (803) will be assumed.

6.3.2 Interrupt Table Entries

DSGEN requests the interrupt table entries ... INTERRUPT TABLE

Operator responds with an entry for each I/O location which may interrupt, in ascending order, and in this format ... n_1, option

10/9/72

where

n_1 is the octal channel number (high number--lower priority--for two board interfaces) between 10_8-37_8 inclusive (must be entered in ascending order)

'option' - directs the system in handling the interrupt:

EQT, n_2 - relates channel to EQT entry n_2

ENT,entry - causes control to transfer to the 'entry' point of a user written system program (privileged interrupt I/O driver) upon interrupt

ABS,value - places an absolute octal value in the interrupt location (may be NOP, CLC, etc.)

[illegible]
[illegible]
p 15

Three additional errors are possible during input of interrupt table entries:

Message	Meaning	Action
ERR30	Invalid INT mnemonic	Same as ERR28
ERR33	Invalid entry point	Same as ERR28
ERR34	Invalid absolute value	Same as ERR28

6.3.3 Setting user base page address

Following loading of the disc resident modules (if any) DSGEN reports the last address plus 1 of the area on base page used for system links

LWA SYS LINKS LLLLL

It then requests the first word address for user base page links.

FWA USER LINKS?

Operator responds with an octal address greater than or equal to LLLLL, but less than 2000_8.

6.4 Sample System Generation

```
SYS GEN CODE?
0000

SYS DISC CHNL?
14

# SECTORS/TRACK?
24
```

10/9/72

```
SYS DISC SIZE?
200
# DRIVES?
2
FIRST SYSTEM TRACK?
0
FIRST SYSTEM SECTOR?
3
SYS DISC SUBCHNL?
0
USER DISC SUBCHNL?
0
TIME BASE GEN CHNL?
0
PRIV INT CARD CHNL?
13
# DMA CHANNELS?
2
LWA MEM?
37677
ALLOW :SS?
YES
PRGM INPT?
DF
INPUT DISC SUBCHNL?
1
LIBR INPT?
PT
PRAM INPT?
TY
*EOT
NO UNDEF EXTS
ENTER PROG PARAMETERS
/E
```

10/9/72

```
# LINKS?
803

SYSTEM
DISCM                      02000
  :
*EQUIPMENT TABLE ENTRY
  :
*DEVICE REFERENCE TABLE
  :
*INTERRUPT TABLE
11,EQT,1
12,ENT,C.XX
24,ABS,102044
  :
/E
EXEC SUPERVISOR MODULES
  :
I/O DRIVER MODULES
  :
LWA SYS LINKS              LLLLL
FWA USER LINKS?
LLLLL
LWA SYS XXXXX
FWA USER?
XXXXX
USER SYSTEM PROGRAMS
  :
*SYSTEM STORED ON DISC
```

7.0 CHANGES IN RELOCATING LOADER OPERATING PROCEDURES

10/9/72

p 4-26

7.1 Modification to parameters of :PROG, LOADR directive

Format

:PROG,LOADR[,P_1,P_2,P_3,P_4,P_5]

P_1 determines the relocatable object program input combination:
= 0 for loading from jbin and relocatable library
= 2 for loading from jbin, user files, and relocatable library
= n for loading from jbin, user files, tape (paper or magnetic, specified by logical unit n), and relocatable library.

P_2 list device logical unit

P_3 = 0 for no DEBUG
≠ 0 for DEBUG

P_4 ≠ 0 for current page linking
= 0 for base page linking

P_5 = 0 for system default program bounds (i.e. UBFWA-UBLWA and UMFWA-UMLWA)
= 1 for user specified program bounds

Default conditions:

$P_1 = 0$

$P_2 = 6$

$P_3 = P_4 = P_5 = 0$

7.2 Checksum error recoverability

A checksum error encountered when loading relocatable programs from paper tape will no longer cause the Loader to abort. After the LØ1 error message is printed, the loader returns to the paper tape load point:

```
LOAD TAPE
LOADR SUSP
@
```

and suspends. The operator may then restart loading at the beginning of the program, by replacing the tape in the input device and typing :GO.

Jan 19, 1973

7.3 Disc Resident Device Drivers (Type 4) Accepted

The Loader will now accept Type 4 programs and store them as such in the user directory. Type 4 programs may not be combined with any other type during any given load operation.

7.4 Program Bounds Specification

If parameter 5 in the :PROG, LOADR directive is zero, the program bounds of the relocatable modules being loaded will be determined by the system pointers:

UBFWA	- lower base page bound
UBLWA	- upper base page bound
UMFWA	- lower main memory bound
UMLWA	- upper main memory bound

If parameter 5 is equal to one, the user may specify his base page and main memory bounds. Two records will be read from the input device. The first record will be interpreted as the lower and upper base page bounds (two octal numbers separated by a comma). An error encountered in the first record will result in an L18 error message. The second record will be interpreted as the lower and upper main memory bounds (two octal numbers separated by a comma). An error encountered in the second record will result in an L19 error message. If any of the bounds are omitted, systems default values will be assumed. In the keyboard mode, input of these two records will be prompted by the following two messages:

```
BP      BND [L,U]?
PROG    BND [L,U]?
```

In the batch mode, these two records should immediately follow the :PROG, LOADR Directive.

An L18 or L19 error message in the keyboard mode will allow the user to re-enter the desired program bound information. Either error message in the batch mode will cause the Loader to abort.

7.5 Loader Error Messages

7.5.1 LØ1

The LØ1 (checksum error) no longer causes the Loader to abort. The user may re-position his input tape to the beginning of the program, then type :GO.

Jun 19, 1973

7.5.2 LØ6

The LØ6 (duplicate main or segment name) error will no longer occur if the user program name is the same as a program name in the system directory.

7.5.3 L18

An L18 error message will result from any of the following:

a) an illegal octal digit in the base page bounds specification
b) the lower base page bound specified is greater than the upper base page bound specified.
c) either the lower or the upper base page bound is greater than 2000_8.

In the keyboard mode, the user may re-enter new base page bounds. In the batch mode, the loader aborts.

7.5.4 L19

An L19 error message will result from either of the following:

a) an illegal octal digit in the main memory bounds specifications.
b) the lower program bound is greater than the upper program bound.

In the keyboard mode, the user may re-enter new program bounds. In the batch mode, the loader aborts.

8.0 MODIFICATIONS TO I/O DRIVERS

8.1 Clear request

All I/O Drivers have been modified to accept a clear request (request code = 3, function code = 0).

8.2 EQT10 usage

No input driver that can be used as a batch device may destroy EQT10. (The manual should be changed to state that no driver should store into EQT10, since it is used by the system.)

10/9/72

8.3 Line Printer Drivers

The three line printer drivers (DVR12) have been modified to interpret bit 7 in the CONWD as carriage control specification. If bit 7 = 0, the first character in the output buffer will determine carriage control. If bit 7 = 1, single spacing will be assumed and all characters in the output buffer will be output as data.

In addition, the three DVR12's will return a top-of-form status (Bit 6 = 1) when top of form is executed.

8.4 Absolute program consideration

The photoreader driver (DVR01) and the teleprinter driver (DVR00) have been modified to accept absolute binary data. If bits 6,7, and 10 are set, data will be transferred as absolute binary. The driver checks the bits (6,7, & 10) in ascending order. If any low order bit (6 or 7) is zero, higher order bits (7 & 10) are not checked.

9.0 USER PROGRAM CHANGES

9.1 System Base Page see DOS p A-2

The Base Page System Communication Area has been modified as follows:

LOCATION	NAME		CONTENTS
40-267			(Same as before)
270	DUMMY		I/O Address of Privileged Interrupt Card
271	MPTFL		Memory Protect Flag
272	$GOPT		Point of Suspension Continuation Address
273	$LDCD	$IDCD	Input Request Code Check
274	$MDBF		Exec Module Data Buffer (2 words)
276	TEMP		Data Communication Temporary (7 Word Buffer)
305	TEMPØ	}	Temporaries
306	TEMP1		
307	TEMP2		
310	UTMP0		
311	UTMP1		
312	UTMP2		
313	MSECT		Negative Number of Sectors/Track
314	VADR		Address of Instruction Causing Memory Protect Violation

10/9/72

LOCATION	NAME	CONTENTS
315	IODMD	Current Resident I/O Driver Module Flag
316	RCODE	Current Request Code Value
317	SXA	Operator Attention Restore A Register Value
320	SXB	Operator Attention Restore B Register Value
321	SXEO	Operator Attention Restore E & O Register Value
322	SXSUS	Operator Attention Return Address
323	EFMP	Extended File Management Package Flag
324	DSCLB	Disc Track/Sector of Relocatable Library
325	DSCL#	Number of Relocatable Library Routines
326	LSTCH	Last Disc Referenced
327	TRAC#	User File Table Validity Flag
330	XFLG	Entry Address for Disc not Ready
331	SSFLG	System Search Flag
332	CHARC	Batch Input Character Count
333	TYEQT	System TTY EQT4 Address

Any user programs referencing the variables named above should be modified to reflect the new addresses.

9.2 Capability to turn off interrupt

The system will now accept a CLF 0 instruction from a user. Interrupt and memory protect will be off on return to the user program. (This instruction does not set MPTFL.)

9.3 Privileged mode

A mode flag (MDFLG, location 133_8) will be incorporated into the system. This flag will be set and cleared by data communications routines, and interrogated by the system. When properly set, it performs the following functions:

a) allows execution of a user subroutine on an I/O completion where the I/O was initiated without wait.

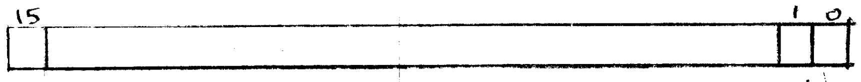
b) returns to the user program after detection of an I/O or request code error.

c) allows the user to maintain control while Exec calls following File Search are being held off by the system.

Suggested use: 1) The user can use the comp. subr. to tell him when an I/O without wait has completed. He can't start another I/O request within the comp. subr. but he could set a flag being tested elsewhere.

2) He can do his own error processing / keep system from printing error messages. Jan 19, 1973

Completion bit
0 = normal return
1 = call completion routine



Return bit
0 = normal return/ non-return
1 = always return

File Search Bit
0 = Wait
1 = Let user execute

The system interrogates MDFLG as follows: (location 133)

A. Before processing an Exec call, the system determines whether or not a file search without wait is in progress. If it is, further Exec calls may not be processed. At this time, bit 1 of MDFLG is checked. If it is not set, control is passed to the wait loop until the file search is completed. (As soon as the file search is complete, the pending Exec call is processed.) If bit 1 of MDFLG is set, control is returned to the point following the Exec call with the A-Register set to 8 (Exec busy).

What if it isn't? at normal place ?

B. After I/O initiation, Bit 0 is checked. If set, control is returned to the user with the A-~~r~~egister set:

(This is supposed to be I/O without wait.)

only where you <u>wouldnt</u> return to user in error

<u>A-reg after initiation</u>

same as before (0–4)

CONTENTS	MEANING
0	Operation initiated
1	Read/Write illegal
2	Control request ignored
3	Device down
4	Immediate completion
5	DMA busy
6	Driver busy
7	Driver overlay area busy
8	Exec overlay area busy
9	Operation rejected
10	Memory protect error
11	Request code error
12	Execution time exceeded
13	Not used
14	Illegal logical unit
15	Logical unit unassigned
16	Illegal buffer address
17	Core wrap around
18	Illegal track address
19	File Cannot be found

see 5-9 and 5-12 for regular information in A

also see p 28

Jan 19, 1973

C. After I/O completion, Bit 15 is checked. If set, control is temporarily passed to a user subroutine immediately following the EXEC call. Upon entry to this routine, the B-register will contain the transmission log from the driver; the A-register will contain status as follows:

It was necessary to make these neg. to avoid ambiguity. (These are passed (in pos. form) by driver to DOS.

CONTENTS	MEANING
0	I/O completed without errors
-1	Device not ready
-2	End of tape
-3	Parity error
-4	Batch input detected ':'

NOTE: Processing within the completion subroutine is done with interrupt and memory protect off. The user may not use any routines which may be called by other software modules unless these routines are re-entrant.

int on for priv devices ←

[illegible]

If the I/O completion resulted from an I/O error (device not ready), end of tape, or parity error) and the violating device is not the system teletype or the disc, bit 14 of EQT4 is set to indicate that the device is down.

Bit 0 of the MDFLG is then checked. If it is set, control will be returned to the user (bypassing system processing of the error).

[illegible]

D. Calling sequences operating in the privileged mode might look something like this

```
        JSB EXEC
        DEF END
        DEF RCODE
        DEF CONWD    (must be I/O without wait)
         .
         .
         .
END     JMP INIT   }
COMP    NOP        }  Executed following I/O completion.
         .         }  Includes checks for completion errors.
         .         }
        JMP COMP,I }
INIT     .         }  Executed following I/O initiation.
         .         }  Includes checks for initiation errors.
```

Jan 19, 1973

10.0 DISC BOOTLOADERS

In addition to the existing Bootstrap Loader, three core-resident bootloaders will be available (ISS Bootloader, IOMEC Bootloader, and 7900/7901 Bootloader). These bootloaders will support multi-drive systems. Due to the limited amount of memory (64 words) a few restrictions will have to be imposed:

1. PRESET must be pressed prior to execution.
2. The disc must reside in the same I/O channel as that at system generation time.
3. There must be a DOS-III system present on Drive Ø, Head Ø. (As long as there is a system on Drive Ø, Head Ø, a system from any subchannel may be brought up.)

Jan 19, 1973

APPENDIX A

forward ref p7

Message	Description
BAD CONTROL STATE	An illegal directive (or :SS when not allowed) has been encountered.
CHECKSUM ERROR	Checksum error encountered while storing a relocatable or absolute file (:ST,R or :ST,X)
DUPLICATE FILE NAME	Doubly defined file name found in a STORE directive (other than :ST,P); an EDIT directive with a new file name; or a RENAME directive.
END FILE	During :EDIT- End of source file occurred before end of edit file During :ST,S- Colon encountered in column 1 of a source statement
file ILLEGAL	1. A file name begins with a non-alphabetic character. 2. The file name is a :LI,S directive was not a source file.
ILLEGAL DIGIT	1. In a decimal number, character is other than 0-9. 2. In an octal number, character is other than 0-7.
ILLEGAL LUN	1. Logical unit requested is equal to zero, greater than the number of logical units in the system, not the correct type (i.e. input type for output type), etc. 2. In a :ST,S,XXX,lu,C directive, the lu refers to the current batch device. 3. In a :BA,lu directive, the lu referenced is a null or invalid device.
ILLEGAL PROGRAM TYPE	Program requested in a RUN or PROG directive is not legal.

Jan 19, 1973

Message	Description
LIMIT ERROR	During EDIT - source statement numbers are out of order - beginning source statement number is greater than final source statement number During PDUMP, ADUMP - dump limits are incompatible During DUMP - sector numbers are illegal
LUN UNASSIGNED	Logical unit requested in a directive is unassigned
MISSING PARAMETER	A parameter is missing in a directive
NAME *IGNORED	Illegal JOB name; non-alphabetic first character
NO BIN END	No END record detected when storing a relocatable binary program
NO SOURCE	No source statements following a /R or /I in an EDIT directive (either / or : was encountered.)
NUMBER OVERFLO	An integer is too large
OVERFLOW JBIN	There is not enough room in the JBIN for storing the relocatable binary output from the assembler or compilers
PARAMETER ILLEGAL	A parameter of a directive is illegal. During EDIT - /X was encountered (X not I,R,D,E,M,U,S) - Colon in column 1 following /D
TRAC # TOO BIG	Track requested is higher than last available disc track (track may be in JBIN area)
file name UNDEFINED	Undefined file name in RUN, PROG, DUMP, EDIT, or RENAME directive
WRONG INPUT	Relocatable binary input furnished for a source file request or vice-versa

Jan 19, 1973

8/31/73

II. DOS-III-B

The following modifications to DOS-III are designed to support data communications and data management projects. Release to systems integration is scheduled for February 1, 1974 assuming the availability of a new hardware privileged interrupt I/O board.

A. Data Communications Support (Available November 1, 1973)

1. Privileged driver completion interface

A system subroutine ($PCOM) will be included in the core-resident portion of the system. When a privileged driver detects a completion, it should call this routine (with the interrupt system disabled to prevent reentrancy problems). Control will be returned immediately. When the system is capable of handling the completion, it will enter the privileged driver at the normal completion entry point (C.XX). The calling sequence for notifying the system of a privileged device completion is:

```
EXT  $PCOM
   .
   .
   .
LDA  EQT1        Pass saved EQT address
CLF  0           Disable interrupts
JSB $PCOM        Notify system of completion
   .
   .
   .
```

NOTE: System processing within $PCOM assumes the availability of the new hardware privileged interrupt I/O board.

2. System Timing Capabilities

System timing capabilities will be provided through two system routines:

$TIME	To set, reset, or release timers
$CLCK	To update timers whenever a clock interrupt occurs (every 100 msec) and transfer control to the appropriate processor if a time-out occurs.

- $TIME will exist as a library subroutine. With modifications to the system generator, this subroutine will be available to "system" routines in either the system or user area of memory. It can be called to set, reset or release timers as follows:

<u>Set or Reset</u>

```
         EXT  $TIME
           :
         LDA  VALUE    (time in -100 ms)
         LDB  ADBUF    (address of timer buffer)
         CLF  0        Disable interrupt
         JSB  $TIME    Set timer
           :
ADBUF    DEF  *+1      Timer Buffer
         <ID>          16-bit identifier
         <time-out processor address>
         NOP
                       Used by system
         NOP
```

<u>Release</u>

```
         EXT  $TIME
           :
         CLA           (indicates release)
         LDB  ADBUF    (identifies timer)
         CLF  0        Disable interrupts
         JSB  $TIME    Release Timer
```

NOTE: Since the system uses a link-list mechanism for updating these timers, all routines requesting timers must remain core-resident during program execution.

- $CLCK will exist as a portion of the core-resident system. It will be executed every time the clock (ie TBG) interrupts (currently every 100 milliseconds). This routine updates all timers, calling appropriate time-out processers if a time-out occurs. The time-out processor will be called with the "A" register set to the ID of the timer causing the time-out.

NOTE: Since execution of the time-out processors will be included in the processing time for clock interrupts, their lengths should be kept at a minimum.

3. Expanded EQT Unit Field

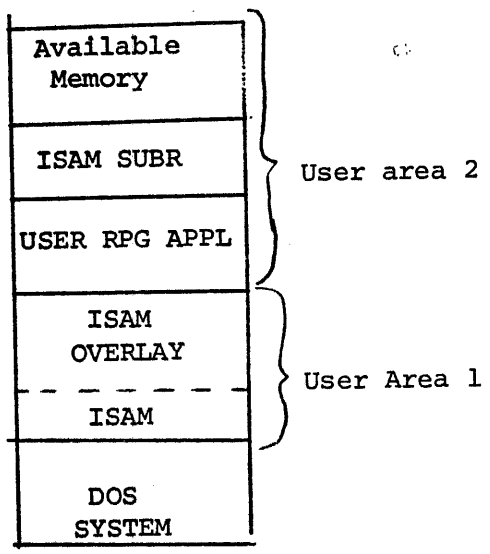
The unit field in EQT3 will be expanded from three bits (Bits 8-6) to five bits (Bits 10-6). This will allow up to 32 units per device.

4. Control Requests with no DMA

Any control request (request code = 3) with a subfunction less than 100 octal will no longer have a DMA channel assigned for it.

B. ISAM and RPG Support (Available January 1, 1974)

ISAM and RPG capabilities will enhance 21XX market potential over the next several years. It is desired that ISAM and RPG operate unchanged regardless of the hardware environment, i.e., the system would handle any new hardware requirements transparent to the application. The following diagram illustrates the relationship envisioned for RPG application, ISAM and DOS using current memory:



User application is initiated from console or batch. The user RPG application requests ISAM via a call to the ISAM subroutine which in turn translates into a DOS system call.

The system transfers control to ISAM and its overlays. When ISAM is completed, it returns to the system with a completion call which in turn returns back to the original return address of the ISAM subroutine call.

For current memory, it appears that calls need not go through the system. However, when extended memory becomes available with ISAM and user RPG applications on separate memory maps, it would require ISAM and user RPG application program modifications if calls did not go through the DOS system.

Both the loader and system generator will be modified to allow setting of the separate unique main loading addresses, one main and one sub-main.

which would be
sub main

1. Multiple related user programs

a. Submain entry request

```
      JSB  EXEC
      DEF  *+N+1
      DEF  RCODE
      DEF  NAME
      DEF  SUBF
      DEF  P1
       .
       .
       .
      DEF  Pn
RCODE DEC  31          Request Code Number
NAME  ASC  3,XXXXX     Submain Name
SUBF  DEC  p           Submain Request Code
P1
P2                     Parameters in pairs (buffer address followed
.                      by buffer length)
.
.
P8
```

Control is transferred to SUBMAIN entry address with the B register pointing to a buffer which contains the SOBF and the buffer addresses.

b. Submain return request

```
      JSB  EXEC
      DEF  *+N+1
      DEF  RCODE
      DEF  P1[,I]
      DEF  P2[,I]
       .
       .
       .
      DEF  Pn[,I]
```

```
RCODE   DEC   29
P1
P2
 .                  Parameters
 .
 .
P5
```

It is recommended that parameters P1 to Pn be used for passing error conditions or status information back to the caller. The address of the parameter buffer is in B register upon return to the caller.

[illegible]

2. Memory Allocation/Deallocation Capability

A forthcoming hardware enhancement is the memory extension capability. This memory enhancement will consist of each user being able to uniquely map to a maximum of 32 - 1K memory increments. To optimally utilize this capability requires dynamic memory allocation. In order to allow the same programs to operate with and without the memory extension module, the system will provide software routines to handle dynamic memory allocation of the remaining available memory after the end of the user program.

a. Memory allocation

```
            JSB EXEC
            DEF *+5
            DEF RCODE
            DEF BUFFL
            DEF BADR
            DEF SADR
    RCODE   DEC 32            Request code
    BADR    DEC m             Requested starting memory address;
                              zero if don't care.
    BUFFL   DEC n or(-2n)     Number of words (or characters)
returned:    SADR    DEC +m            for starting memory address
                -p            for: number requested is not available
                              and p words are available.
                 0            for illegal BADR value; 0<BADR<Z where
                              Z is the last available user memory address
```

NOTE: SADR is returned by the system.

b. Memory deallocation

```
             JSB  EXEC
             DEF  *+3
             DEF  RCODE
             DEF MADR
    RCODE    DEC  33          Request Code
    MADR     DEC  0 or M      0 - For release all memory allocated
                              M - Starting address of buffer to be
                                  released.
```

C. <u>Miscellaneous Upgrades</u> (Available January 1, 1974)

1. Programmatic control of files

   Exec calls will be provided to allow the user to create, delete, and rename user disc files under program control. Calling sequences are as follows:

   - To Create

```
             JSB  EXEC
             DEF  *+6
             DEF RCODE
             DEF FNAME
             DEF TYPE
             DEF DSKLN
             DEF ERRTN
```

   where:

```
    RCODE   DEC 34
    FNAME   ASC 3,XXXXX     (5 character file name)
    TYPE    OCT n           Bit 6 = 0 for permanent
                                    1 for temporary
                            Bits 5-0 = program type
    DSKLN   OCT X           Length in sectors
```

```
              ERRTN     -3 if invalid file name
                        -2 if invalid type
                        -1 if insufficient disc space
                           (directory or user)
                         0 if normal return
                        >0 if duplicate file name (contents
                           is address of old directory entry)
```

° To Delete:

```
          JSB EXEC
          DEF *+3
          DEF RCODE
          DEF FNAME
     where:

     RCODE DEC 35
     FNAME ASC 3,XXXXX   (5 character file name)
```

° To Rename

```
          JSB EXEC
          DEF *+5 (or 6)
          DEF RCODE
          DEF ONAME     old file name
          DEF NNAME     new file name
          DEF ERRTN     (returned by system)
          DEF NTYPE     optional new type

 where

 RCODE   DEC   36
 ONAME   ASC   3,XXXXX
 NNAME   ASC   3,XXXXX
 ERRTN   OCT   n      i = -3 invalid new file name
                          -2 invalid new program type
                          -1 undefined old file name
                           0 normal return
                          >0 duplicate new file name (contents are
                             address of duplicate file directory
                             (entry)
 NTYPE   OCT   m      Bit 6 = 0 for permanent
                      Bits 5-0 = program type
```

2. Selective System Generation Library Loading Capability
   The system generator will be modified to support an "optional" library load mode. In this mode, duplicate program and entry point names will be ignored, and new routines will replace old ones.

   NOTE: To accompany this operating system capability the current libraries should be consolidated and updated.

3. Optional Cyclic Check on Disc Write
   The control word for disc write requests will be expanded as follows:

   CONWD

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
							K			logical unit #					

   where:

   K = 0 → execute cyclic check after disc write
   K = 1 → eliminate cyclic check after disc write

   This allows the user to programmatically bypass cyclic checking after disc write requests, thus speeding up execution time.