NOS 2.5.2
FEATURE NOTES

SMD131406

NOS 2.5.2 Feature Notes

NOS 2.5.2 Feature Notes

Table of Contents

NOS 2.5.2 Feature Notes

Introduction

This document contains descriptions of most of the features that are part of the NOS 2.5.2 level 678 release. The articles in this document are targeted for the site analyst, however, some of the topics may be of interest to operations or the end user so we have organized this document such that each chapter may be easily copied and distributed. The Feature Note Audience Matrix should help in distribution of the articles.

These Feature Notes were developed and written by CYBER Software Support. Questions or comments regarding them may be addressed to:

Control Data Corporation
CYBER Software Support (ARH213)
4201 Lexington Avenue North
Arden Hills, MN 55126
1-800-345-9903 (U.S. and Canada)
1-612-851-4131 (Elsewhere)

Feature Note Audience Matrix

Article Title	Site Analyst	Operations	End User
Extended Memory Enhancements	X	X	
EST Channel Reorganization	X	X	
FSE Performance Improvement	X		X
895 Disk Single PP Driver	X	X	
PSU Enhancements/ 585 Printer Support	X	X	
Tape Management System- Nos Hooks	X	X	X
Disk Display File Utility (DDF)	X		
CCL Features	X		X

CHAPTER 1

Extended Memory Enhancements

Extended Memory (EM) usage on NOS has been enhanced with the following NOS 2.5.2 level 678 features:

- NOS may now define separate EST entries for ESM (via low speed port) and unified extended memory (UEM).
- The amount of extended memory NOS can use has been expanded.

The NOS Analysis Handbook contains a new section titled Extended Memory Overview. Users should refer to this section for a more detailed description of EM and its usage.

1.1 Memory Usage on 180 Mainframes

Prior to this release, NOS was restricted to 2 million words (MW) of memory for both CM and UEM. As of NOS 2.5.2, the amount of memory NOS can access has been expanded.

Note that even with this feature, NOS is still restricted to 2 MW of central memory. The additional memory must be assigned as UEM and may be used as:

- user extended memory
- buffer space for buffered devices
- as a mass storage device (alternate system resident, rollout files, etc)

NOTE
In this article, all references to CYBER 170 class mainframes include the CYBER 865 and 875.

The new memory limits by mainframe model are as follows:

Mainframe Model	Maximum CM + UEM
815, 825, 835 865 and 875	2 MW*
810, 830, 990 and 995	8 MW**
840, 850, 860 870, 845 and 855	16 MW

* The 865 and 875 also have high speed CPU access to as much as 16 MW of ESM.

** The 810, 830 and 990 require CIP V7 for access to more than 2 MW.

1.2 CYBER 990 Considerations

The CYBER 990 requires an option to access more than 2 MW of memory. This option must be ordered through your local sales office.

Support for the larger memory was accomplished, in part, by removing the microcode support for the "fake read" option of the block copy instruction. A "fake read" was executed whenever a block copy instruction was executed and bit 21 or 22 of X0 was set. This instruction was used by CPUMTR to clear memory. CPUMTR has been modified to use a block of UEM that is set to zero at preset time when clearing memory on a 990.

This block of memory is allocated in the user extended memory portion of UEM. If a site installs the option, then they must include an 'XM' entry in their EQPdeck and allocate at least one block of user extended memory. The number of user extended memory blocks left for user access will be one less than the number allocated by the 'XM' entry. Sites allocating 'just enough' extended memory for the largest job will have to account for the block used by CPUMTR for memory clearing.

1.3 Multiple Extended Memory Equipments

Another feature at this level is the ability to configure both UEM and ESM/ECS on the same mainframe. With this feature, the site may define seperate equipments for UEM and ESM/ECS.

NOTE

It is possible with the feature to connect two separate ESM equipments to a single mainframe. It should be noted that only one of these may be defined as having a CPU access.

The CYBER 180 machines may now be configured with both UEM and ESM. The ESM is used as a link device to connect mainframes in a MMF complex. UEM is allocated for rollout, buffers and user extended memory. Note that since ESM may only by connected to CYBER 180 mainframes by the low speed port (LSP), it is not possible for ESM to function as I/O buffers or as user extended memory on CYBER 180 mainframes.

1.3.1 Example EQPDECK Entries

1.3.1.1 Example 1

In this example, the site has one 170-type mainframe with ESM connected via the high speed port and a DDP. They will use ESM as an alternate system device, a secondary rollout device and as buffer space for 885-42 disk drives.

```
EQ6=DP,ET=ES,ST=ON,SZ=10000,CH=32.
.
.
MSAL,S=6.
ASR=6.
XM,AA,3000,1000.
```

These entries allocate 2 MW (decimal) of ESM. 3000000B words of the ESM are allocated as buffers and 1000000B words are used as user extended memory. The remaining space will be available as a disk device. The amount of space left on this device depends on how many routines are made alternate system resident. Notice that the equipment entry for entended memory is no longer required to be 5.

1.3.1.2 Example 2

This example shows an 865/875 with ESM used as a disk device and as buffers for 885-42 disks. UEM will be used for user extended memory.

```
EQ10=DP,ST=ON,SZ=10000,CH=7.
XM=AA,3000,1000,EM.
```

This example defines 2 MW of ESM with 3000000B words used for 885-42 buffers. If the 'EM' parameter had been left off the 'XM' entry, the user extended memory would have been taken from ESM (EQ10). By including this parameter, user extended memory is allocated in UEM while the I/O buffers are allocated in ESM.

1.3.2 Multi-Mainframe configuration Examples

When defining MMF configurations, the link device must be equipment 5.

1.3.2.1 Example 1

This example shows two 180 mainframes using ESM as a link device for MMF operations.

```
Mainframe AA

EQ5=DP,ST=ON,ET=ES/D2/NC,SZ=10000,CH=4.
EQ6=DE,ST=ON,ET=EM,SZ=2000.
XM=AA,1000,200.

Mainframe BB

EQ5=DP,ST=ON,ET=ES/D2/NC,SZ=10000,CH=4.
XM=BB,,200.
```

In this example, mainframe AA also has 895 or 887 buffered disk drives. The entry for equipment 6, allocates 2000000B words as UEM. 1000000B words of UEM are used as buffers and 200000B words are used as user extended memory. The rest could be used for secondary rollout or as an ASR device.

Mainframe BB does not have any buffered devices so there is no need for an equipment entry. The 'XM' by itself defines 200000B words of user extended memory.

1.3.2.2 Example 2

This example shows a 170-type and a 180-type mainframe using ESM as a link device.

The 170 mainframe (AA)

```
EQ5=DP,ST=ON,ET=ES,SZ=10000,CH=5.
XM=AA,1000,400.
```

The 180 mainframe (BB)

```
EQ5=DP,ST=ON,ET=ES/D2/NC,SZ=10000,CH=4.
EQ6=DE,ST=ON,ET=EM,SZ=2000.
XM=AA,1000,400.
XM=BB,1000,200.
```

In this example, the 170 mainframe allocates 1000000B words for 885-42 (DEMA) buffers and 400000B words for user extended memory.

The 180 mainframe, uses ESM as a link device and UEM as user extended memory and as buffers for 895 or 887 disk drives. Since machine AA is using ESM as both a link device and as user extended memory, the EQPDECK for mainframe BB must contain the XM entry used on mainframe AA so that both machines know the size of ESM after DEMA buffers have been allocated.

1.4 User Impact

There are two areas of potential problems for sites using the large memory feature on CYBER 990s. These problems are:

- User programs that use the "fake read" instruction must be changed to clear memory in a different manner.

- Programs written to use the block copy instructions need to ensure that X0 contains no garbage in the upper bits of the address. Prior to this feature, garbage bits in X0 were ignored by the microcode. Now these bits can produce a mode error (address out of range).

CHAPTER 2

EST Channel Reorganization

With the addition of several mass storage related features in previous releases of NOS, the channel-related data in the equipment status table (EST) has become fragmented. Since some of this information is accessed by time critical routines, the level 678 release includes a reorganization of the EST channel information.

2.1 EST Changes

With this release of NOS, the format of the EST looks like this:

```
EQDE  12/*,12/CD1,12/CD2,12/*,12/*
EQAE  12/*,12/*,12/DDD,12/*,12/*
```

* This field has not changed.

CDn Data for channel n. The format of each channel byte is:

Field (bits)	Meaning
11	Access path enabled. 0 - Access path is disabled or not present. 1 - Path is enabled.
10-9	Channel state. 0 - Up. 1 - Idle. 2 - Unused. 3 - Down.
8	Controller type. 0 - Full track. 1 - Half track.
7	Concurrent channel port number. 0 - Port A. 1 - Port B.
6	Reserved.
5	Concurrent channel flag. 0 - Non-concurrent channel. 1 - Concurrent channel.
4-0	Channel number.

DDD Device dependent data. This was formerly byte 2 of word EQDE.

2.2 Operational Changes

With the definition of the access path enabled flag, the requirement that channel zero always be the primary channel has been removed. When the access path flag is clear (DISABLED), DSD does not display the channel.

2.3 Changed Monitor Functions

The monitor functions CCHM and REQM have been changed.

2.3.1 CCHM

The CCHM function has been changed to allow only one channel. The new format of the function request is:

```
12/CCHM,12/CH,36/UNUSED
```

CH Channel data.

Bits	Meaning
11-6	Reserved (zero)
5	Concurrent flag. 0 - Non-current channel. 1 - Concurrent channel.
4-0	Channel number.

2.3.2 RCHM

The RCHM function has been changed to allow only 2 channels in the request. The request format is:

```
12/RCHM,12/CH1,12/CH2,24/UNUSED
```

CHn Channel n data.

Bits	Meaning
11-6	Reserved (zero)
5	Concurrent flag. 0 - Non-concurrent channel. 1 - Concurrent channel.
4-0	Channel number.

NOTE

The old format of the EST supported four channels for tape drives although the tape driver did not support this. This feature removes the four channel format for tape ESTs.

CHAPTER 3

NOS FSE Performance Improvement

NOS 2.5.2 L678 contains code to improve the performance of the NOS Full Screen Editor (FSE) when painting a terminal screen. FSE will now send a direct cursor addressing sequence when painting unmarked text lines that contain a number of consecutive blanks rather than sending the (larger) number of consecutive blanks directly. This is accomplished each time a screen is painted by calculating in advance the length of the direct cursor addressing sequence for each line and sending this sequence in lieu of the blanks themselves.

Direct cursor addressing lessens the time required to paint a screen by reducing the total number of characters sent downline. For example, the number of characters sent to the screen on a CDC 721 terminal in 132 column mode is reduced by 50% for an average COMPASS listing. Improvement will be less dramatic for any terminal type when editing text files possessing a dense character spacing or for those terminals that have long direct cursor addressing codes.

Improvement will occur in both single-user mode (FSE) or the multi-user mode (SMF).

3.1 Requirements

This feature works only on terminals that possess an ERASE_TO_END_OF_LINE capability. Note also that it applies only to unmarked text lines; blanks in marked lines must be sent individually for the highlighting.

CHAPTER 4

895 Disk Single PP Driver

Prior to this release, NOS required 2 peripheral processors (PPs) per channel on an 895 disk subsystem. Now, a hardware enhancement, known as the 170 Direct Memory Access (DMA), along with a new 895 driver, 1XY, make it possible for only one Concurrent I/O (CIO) PP to drive the 895s via a concurrent channel (CCH). The 1XY driver interfaces with the system via the physical unit tables (PUT) like the other buffered I/O drivers.

(For an overview of the terminology and hardware components of the CIO system, reference the NOS 2.5.1 L664 Feature Notes publication number SMD110168.)

4.1 Configuration Specifications

1. 1XY requires one dedicated CPP versus 1XM which requires 2 dedicated NPPs per channel. 1XY's resource usage (CPU/CM/UEM) is comparable to that of 1XM.

2. The 895 disks may be configured with intermixed NCHs and CCHs to the same disk unit; that is, any 895 disk may be driven by both the NIO and CIO drivers concurrently.

3. The performance of the CIO path matches the performance of the NIO path.

4. Disk deadstart is only supported via the NIO path.

5. Disk maintenance may occur via the CIO or the NIO path.

6. The 895 disks still may not be shared by way of any multimainframe variants.

7. Error detection and device verification in 1XY is identical to the implementation in 1XM.

8. 1XY error logging is identical to 1XM with one exception: the channel control registers will be logged by 1XY.

4.2 1XY/1XM Differences

1. CIP cannot be installed to any 895s that do not have an NIO path defined. This means that those 895s accessible only via a CIO path will have all cylinders formatted in large sectors.

2. At this time, EDD cannot dump firmware from CCHs. This deficiency will be addressed at a future date.

3. 1XY does not reload firmware internally. Instead, it downs the CCH and drops out. After the necessary maintenance, LOADBC should be used to reload the firmware.

4.3 External Changes

4.3.1 895 Equipment Definition

The equipment definition for the 895 disks is unchanged with the exception of the channel entry. Differentiation between CCHs and NCHs is accomplished by adding a "C" to any CCHs. For example:

```
EQ020=DC,ST=ON,UN=0,CH=02/C03.
```

This example defines the disk paths to be NCH 02 and CCH 03.

(Implementation of this change occurred in conjunction with the removal of the HT parameter from all buffered I/O devices. This parameter was ignored at NOS 2.5.1 L664 and L670.)

4.3.1.1 LBC Entry

The LBC entry uses the "C" convention described above. To specify controlware loads for the above example, the entry would be:

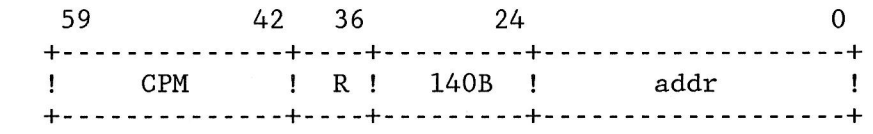
```
LBC,CC,02,C03.
```

4.3.2 PPB Entry

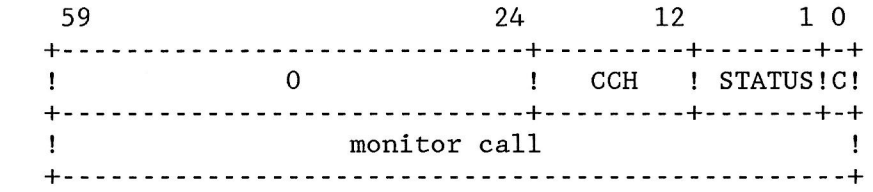
Support of the PPB entry has been removed. The entry was ignored at NOS 2.5.1 L664 and L670.

4.3.3 CPM Function 140

MALET and LOADBC need to be able to request a CPP. A new CPM function has been implemented to assign a CPP for SSJ= type programs. The call format is:



where addr is the parameter block address. The parameter block format is:



where CCH is the CCH required by the CPP, STATUS is the reply status (0 if CPP assigned, 1 if not), and C is the completion bit.

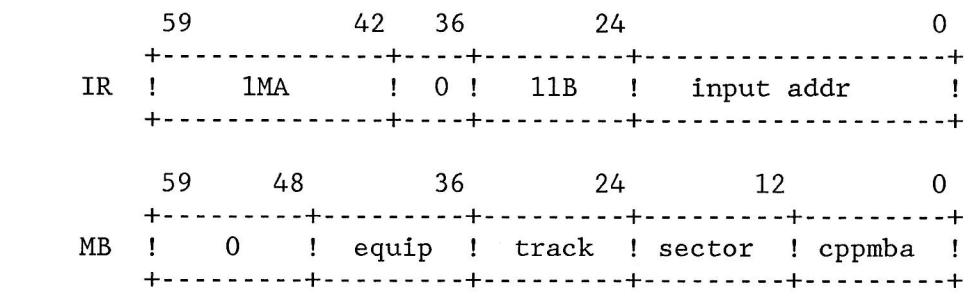
4.3.4 CVL

The CVL interface has been changed to use bit 2**5 of the channel number to indicate that the request is for a CCH.

4.3.5 1MA Function 11

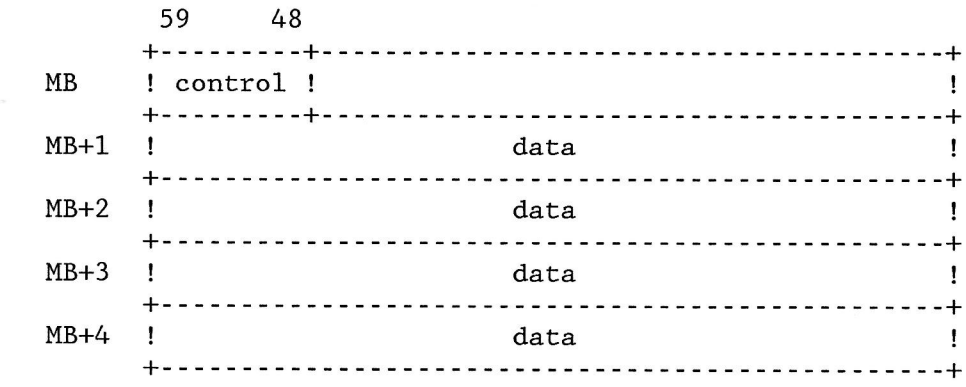
1MA function 11 assists the CPP in the loading of mass storage resident 12-bit overlays. This function is called by a new driver, 6DD, which is loaded into the CPP if a mass storage overlay load is necessary.

The format for function 11 is:



where input addr is the input register address of the CPP, equip is the equipment to read the overlay from, track is the track to read the overlay from, sector is the starting sector of the overlay and cppmba is the message buffer address of the CPP.

The following communication protocol is used:



where control is: 0, if CPP is ready for data

n, if n words of data in buffer for CPP

2000, if EOR encountered

4000, if unrecovered read error occurred.

4.3.6 LOADBC Change

LOADBC has been modified to accept the "C" prefix for a controlware load on a CCH. For example:

```
LOADBC,C=C03.
```

to load controlware on CCH number 03.

4.4 Internal Change

4.4.1 Physical Unit Table Change

1XY has to process auxiliary requests for other system routines since it cannot accommodate them itself. These requests are passed through the physical unit table (PUT) in word PILL. The request definitions are:

bit 2**17	log pack serial number
bit 2**16	process disk flaw map
bit 2**15	log firmware identifier for channel defined by bits 2**23 - 2**18
bit 2**14	format maintenance cylinder
bit 2**12	PUT request failed flag

CHAPTER 5

PSU Enhancements/585 Printer Support

Support for the 585 printer on NOS is not being announced with the 2.5.2 release. It will be announced at a later date. This feature note is being included at this time so that the information will be readily available at the time of the announcement.

The 585 printer will be supported on NOS via the PSU utility. The only change to the NOS operating system for support of the 585 printer was a modification to DSP to add two selection bits to the BANNER function call. This change allows changing forms control functions at the beginning and end of BANNER page construction.

To the user, the 585 printer functions like previously available NOS line printers. Routing files to print on the 585 printer is by default the same as routing to a 533/536 PSU printer or to a 580 printer. The 585 printer is unlike the 580 and 533/536 printers in that it is driven by PSU over CDCNET rather than by PSU over the NAM/CCP network or by BIO over a channel.

5.1 Configuration Files

When installing PSU at level 678, the configuration file(s) for the mainframe(s) on which PSU is installed must be purged before bringing PSU up for the first time. These files reside under user name SYSTEMX (UI=377777) and have the names of PCFxx where xx is the mainframe id. There are new options and commands at level 678 which cause the contents of this file to be different from previous releases. After PSU has been brought up the first time and all attributes established, use the PSU SAVE command to capture the attributes for subsequent PSU startups.

5.2 PSU Commands

Three new commands have been added to PSU - TRAIN, SUPPRES and BANNERS.

5.2.1 TRAIN

PSU now selects files for printing based on train, in a manner similar to BATCHIO. The format of the TRAIN command is:

```
TRAIN,pn,tr
```

pn the printer name
tr 64 or 95

A new column on the K-display shows which train is in effect for each printer logged in. The train attribute for each printer defaults to 95. For those printers which have 64-character bands and should not print A9 files, the train attribute should be changed to 64.

5.2.2 SUPPRES

The SUPPRES command causes suppression of carriage control action on the file being printed. It may be entered only while a file is being printed and its effect terminates when the file completes printing. The format is:

```
SUPPRES,pn.
```

pn the printer name

5.2.3 BANNERS

The BANNERS command allows selection of 0, 1, or 2 banners to be printed on any printer. Format for the command is:

```
BANNERS,pn,nb
```

pn the printer name
nb the number of banners required - 0, 1, or 2

The ENABLE,pn,BANNER and DISABLE,pn,BANNER commands have been superseded by the BANNERS command, so they have been eliminated.

5.2.4 Saving Attributes

Train, number of banners and the other attributes (ID, FC, etc.) associated with each printer are saved with the PSU SAVE command and restored at the next PSU startup.

5.2.5 File Positioning

All file positioning commands (SKIPRU, BKSPRU, END, REPRINT) may be given from the ACTIVE state or from the STOP state. However, when a file positioning command is given from the STOP state, the state will be changed to ACTIVE upon completion of the command. Note that 585 connections go through a WAIT state between when a STOP command is given and when the STOP state

occurs, and between when a file positioning command is given and when it completes if not given from the STOP state.

5.3 K Displays

The K-display has been enhanced with three new columns that show the size of the form in the printer (set with the PRSIZE command), the current default density of the printer (also set with the PRSIZE command), and the number of banners that will print on the printer (set with the BANNERS command). The column for the form size is headed SZ, the density is headed DE and the number of banners is headed BA. These columns are in the order described and are placed between the MAX and STATUS columns. The PCT column (percent of file printed) (blank) for 585 printers.

5.4 Printer Support

PSU now supports up to 12 printers, containing any mix of 533/536 and 585 printers. However, if the 533/536 printers are configured for other than user names PRINT01-PRINT08 and/or the 585 printers are configured for user names other than PRINT09-PRINT12, the EVFU file must be changed accordingly. If the EVFU file is changed, all 585 definitions must precede 533/536 definitions. If there should be a mismatch between the EVFU file printer definition and the connection received from the network, the connection is placed in a HOLD state, and a K-display message provides the pertinent information. If the K-display is not assigned when this occurs, there will be a flashing B-display message requesting the K-display be assigned to PSU. No commands will be accepted for a connection in HOLD state; the only way to clear the condition is to break the connection external from PSU.

CHAPTER 6

Tape Management System-NOS Hooks

TMS provides NOS sites and end users with a method of accessing magnetic tape files in a manner similar to accessing permanent files. TMS has been implemented on NOS since 1983. Primary availability was through CYBERNET Services. In 1985, TMS was made available through Professional Services.

Only the operating system "hooks" for TMS have been incorporated into NOS 2.5.2. The stand-alone portions of TMS are not included with the standard NOS 2.5.2 release. If the stand-alone portion of TMS is desired, which is still required if you want to "run" TMS, then it must be acquired from Professional Services through your local sales office.

6.1 Operating System Changes

With this feature, code has been added to several decks in the standard NOS PL. Most of the code for the TMS standardization has been written against the decks 1MT, MAGNET, and RESEX. This code was needed so that the stand-alone portions of TMS can communicate with the system's tape processing routines.

With the "hooks" already incorporated into NOS, it is not be necessary to rebuild any standard NOS routines when installing TMS. Previously, when TMS was acquired, approximately 3,600 lines of code had to be incorporated into the NOS system and several routines had to be rebuilt. Now all that may be required is to install the stand-alone TMS routines into the site's system. This could easily be done with a SYSEDIT until it is convenient to build a new deadstart tape.

6.2 Brief Overview of TMS Processing

Each family that uses a TMS has a tape catalog file. This file is fast-attached and contains information about which users can access which tapes. The tape catalog file is organized by site defined VSNs and user created "tape files". The tape files can consist of one to 60 VSNs. A user can manage his or her tape files in much the same way as permanent files. For instance, the user can allow or prevent other users from reading or writing his or her tape files. A user can access a TMS tape file by either the site defined VSN or by the user defined tape file name. TMS also "remembers" physical information about the tape file such as density and conversion mode, so that the user does not have to include that information on a request for a reserved tape.

The tape catalog files are managed by the PP program TFM (Tape File Manager). TFM is called by RESEX to verify user tape requests and by MAGNET when end-of-reel conditions occur. TFM is also called by the stand-alone programs TFSP, LDISTAP, and TFILES.

TFSP (Tape File Supervisor) is used by operators and tape librarians to get information from and make changes to the tape catalog files. Properly validated users can use TFSP from non-system origin jobs.

LDISTAP contains three entry points. TMSDIS (TMS Display Utility) is used by operators and tape librarians to check the status of tape catalog files, tape drives, and jobs requesting

tapes. TMSDIS can be used as an L-Display utility or from an interactive terminal by validated users. TMSON and TMSOFF are used to make a tape catalog file fast-attached or not fast-attached. LDISTAP has a helper PP, TFU, that is used to communicate with MAGNET.

TFILES (Tape File Commands) contains entry points that roughly correspond to permanent file commands:

ADMIT (PERMIT)
AMEND (CHANGE)
AUDIT (CATLIST)
RELEASE (PURGE)
RESERVE (SAVE)

TMS also has procedure file commands that are used for building and rebuilding tape catalog files.

NOTE
TMS does NOT require a dedicated control point or a dedicated PP.

Use of TMS is described in the NOS Tape Management System Reference Manual (60463110) and the NOS Tape Management System Operations Manual (60463350).

6.3 New Parameters for LABEL Command

Several TMS parameters are available on the LABEL command. Most of the parameters correspond to permanent file parameters on ATTACH and DEFINE commands. For example, there is a UN parameter for an alternate user name and PW for password.

The SDM= entry point has been added to RESEX. This is so that the values of the FA parameter and the new PW parameter do not appear in the dayfile.

In addition, TMS directives have been added to the RECLAIM command in order to access RECLAIM tapes of an alternate user.

6.4 User Impact

There is no impact on users of sites that do not run TMS. Also, there is no impact on users of the NOS 2.4.3 TMS.

6.5 Operational Impact

There is no impact on sites that do not run TMS. There are several minor changes for sites that previously ran SIS or older versions of TMS. These changes include enabling/disabling TMS through IPRDECK commands/entries, being able to remove a tape catalog file from fast-attached status even if the family is active, and the ability to initiate the TMS tape display utility with the TMSDIS (formerly LDISTAP) command.

CHAPTER 7

Disk Display File Utility (DDF)

The Disk Display File Utility (DDF) provides the capability to read, display, change, and print the contents of physical disk sectors. It is intended to be used by analysts as an aid in working on system problems that involve data stored on mass storage devices.

> WARNING
> Changing disk table contents should be done carefully and only by analysts who understand the effects of the changes. Careless use of DDF can lead to system hangs or loss of permanent files.

7.1 Bringing DDF Up

DDF can be brought up by entering X.DDF. from DSD. On a secured system, SECURITY-UNLOCKED status must be set before bringing up DDF.

7.2 Screen Contents

7.2.1 Left Screen

The left screen displays the following disk and control point information:

- Disk EST ordinal and equipment type.
- Current disk track and sector address.
- Central memory address and byte number of the TRT link byte for the current track. The link byte is intensified if it is not consistent with the sector linkage bytes.
- Reservation, interlock, and preserved file statuses for the TRT link byte.
- Family and pack names, control point number and JSN.
- TRT link bytes for the first track of a permit chain and the first track of an indirect access file data chain.
- First and second control point messages, if issued. The second control point message, if present, is usually a disk error and intensified.
- Contents of DDF scratch areas, labeled A through F (refer to the DDF STORE command).
- Track pointer and recovery word when pack recovery is in progress.
- Various statuses and errors may be displayed at the bottom of the screen.

7.2.2 Right Screen

The right screen displays the contents of the disk sector. The byte numbers displayed in parenthesis after the central memory address are the actual byte numbers for the physical disk sector. The byte numbers are used in the memory changing commands.

The right screen format is changed with the right blank key on the CC545 or the tab key on the CC634B console.

The right screen displays available are:

- Five Bytes Per Line. The contents of the sector are displayed 5 bytes per line. The CC545 displays one half sector per page and use the / to toggle between the first and second half of the sector. The CC634B displays one quarter of a sector per page, and the / increments the addresses.

- Ten Bytes Per Line. The contents of the sector are displayed 10B bytes per line with byte numbers. The CC634B console displays one half sector per page and uses the / to toggle.

- Text. The contents of the sector are displayed in text mode. The CC634B displays one half sector per page and the / is used to toggle between the first and second half of the sector.

- Permanent File Catalog Entry. The contents of one permanent file catalog entry are displayed. The / is used to advance to the next PFC entry in the sector.

- System Sector. The FNT and FST of the file are displayed along with the date and time when the sector was last updated.

- End Of Information Sector. This is a short format of the system sector display.

- Legal Commands. It displays a list of the DDF commands.

7.3 Keyboard operation

Most DDF commands do not cause a disk sector to actually be read and displayed. After entering commands that change the equipment, track, or sector, a space must be entered to read the sector. After the period (.), any alpha-numeric characters can be entered.

Above the command line, the following messages may appear:

FORMAT ERROR.	A format error was detected during translation of the entry.
INCORRECT ENTRY.	The command was not legal.
INCORRECT PARAMETER.	The parameter in the entry was invalid or too long.
REPEAT ENTRY.	The entry will not be cleared after execution.
SYSTEM BUSY.	DDF is waiting for the system to process a request.
BYTE NNNN.	The data specified is in byte NNNN.
LENGTH = NNNNNNNNB.	The SKIPEI command has skipped NNNNNNNN octal sectors.

The keys listed below have special uses in DDF. This list provides a complete list of all of the special characters and the action they initiate.

Key	Action Initiated
+	Read the next sector. If at EOI or the end of the track, the current sector is reread.
-	Read the previous sector. If at the beginning of the track, the current sector is reread.
.	Advance to the track in the control bytes and sets the sector to 0. If the control bytes do not contain a track link, the current sector is reread.
,	Advance to the next track in the TRT chain. The sector number is not changed. If currently in the last track in the chain, the current sector is reread.
=	Read the next sector after EOI. If the EOI is at the end of an EOI track, the current sector is reread.
CR	Set REPEAT ENTRY.
SPACE	Read the current sector into the buffer.
*	Hold display.
8	Increment the track number by one and reads the sector.
9	Decrement the track number by one and reads the sector.
RIGHT BLANK	Change the right screen displays. This function is done by the tab key on the CC634B console.
/	Advance the address displays on the right screen.
BKSP	Delete the previous character typed (on a CC634B).
LEFT BLANK	Delete the current line being entered (back tab on a CC634B).

7.4 Commands

7.4.1 Basic Commands

These commands are used to position, display, list, and make disk changes.

AUTOREAD.NNNN.	Read the sector every NNNN seconds. If NNNN is not entered, 1 is used. AUTOREAD is terminated by clearing the command entry with the left blank key or back tab key.
BOT.NNNN.	Back up one track. Search the TRT starting at track NNNN for a track that points to the current track. If one is found, the current track is set to the value. If no track is found that points to the current track, "INCORRECT PARAMETER" is displayed. If NNNN is not entered, the search starts at track 0.
CTB.	Clear the track interlock bit for the current track and equipment. The keyboard must be unlocked to use this command.
DEP.	Disable error processing for calls to the mass storage driver to read a sector. (By default, error processing is disabled.)
DIS.	Drop display and call DIS to control point. If a channel is currently reserved, it will be dropped.

DROP.	Drop display and PP. If a channel is currently reserved, it will be dropped.
DTK.	Drop tracks to the end of the chain starting with the current track. The keyboard must be unlocked to use this command.
DTK.SSSS.	Drop tracks starting with the current track and set the EOI sector in the TRT to SSSS. The keyboard must be unlocked to use this command.
EEP.	Enable error processing for calls to the mass storage driver to read a sector. (By default, error processing is disabled.)
EJT.NNNN.	Enter disk address from EJT ordinal NNNN. (0 .LE. NNNN .LE. largest EJT ordinal.) The equipment, track, and sector are set to the beginning of the file. "INCORRECT PARAMETER". is displayed if the EJT entry is not used or the file does not have any tracks assigned.
EST.NN.	Enter EST ordinal. The equipment must be a mass storage device.
FAMILY.CCCCCCC.	Use permanent file family CCCCCCC. The family is set internally in DDF. PFCW in the control point area is not changed. The family is initially set to the family specified in PFCW.

The following FIND commands scan the catalog track starting at the current position searching for the specified catalog entry. The catalog track must be read via appropriate commands (FAMILY, PACKNAM, UI, etc.) before the FIND command is used.

FIND.CCCCCCC.NNNNNN.	Search for permanent file CCCCCCC. with user index NNNNNN.
FIND.CCCCCCC..	Search for permanent file with a zero user index (i.e., a hole).
FIND.CCCCCCC.	Search for permanent file CCCCCCC with any (non-zero) user index.
FIND..NNNNNN.	Search for a permanent file with user index NNNNNN. The file name is ignored.
FIND...	Search for a permanent file with a zero user index. The file name is ignored (i.e., search for any hole).
FINDISS.NNNNNN.	Starting at the current position, search for the next system sector with user index NNNNNN on the chain. If NNNNNN is not specified, the user index is not checked and the next system sector is displayed. This command is intended to be used on the indirect access permanent file chain.

FINDO.NNNN NNNN NNNN NNNN NNNN.

Search from the current position for the octal number specified. The number can be 1 to 20 digits and is right justified in as many bytes as are necessary to hold the number. The search starts on a byte boundary and the byte number is displayed if the search is successful. If the first part of the number is found at the end of the buffer, the search is terminated even though the entire number was not found. If the carriage return is entered again, the search continues at the location of the first match.

FINDS.CCCCCCC.

Search from the current position for the string of characters CCCCCCC. If the first part of the string matches the end of the buffer, the search is terminated even though the entire string was not found. The byte number of the beginning of the string is displayed. If carriage return is entered, the search is begun again at that point.

FINDSS.NNNNNN.

Starting from the current track and searching to the end of the TRT, find and display the next system sector with user index NNNNNN. If NNNNNN is not specified, the user index is not checked.

FNT.NNNN.	Enter disk information from system FNT ordinal NNNN (0 .LE. NNNN .LE. largest FNT ordinal). The EST ordinal, track, and sector are set to the beginning of the file. "INCORRECT PARAMETER". is displayed if the FNT entry is not used or the file does not have any tracks assigned.
FNTL.NNNN.	Enter disk address from local FNT ordinal NNNN. (0 .LE. NNNN .LE. largest FNT ordinal). The equipment, track, and sector are set to the beginning of the file. "INCORRECT PARAMETER". is displayed if the FNT entry is not used or the file does not have any tracks assigned.
FNTLC.NNNN.	Enter disk address from FNT ordinal NNNN. (0 .LE. NNNN .LE. largest FNT ordinal). The equipment, track, and sector are set to the current position of the file. "INCORRECT PARAMETER". is displayed if the FNT entry is not used or the file does not have any tracks assigned.
GETTRT.NNNN.	This command is intended to be used on the label track. The equipment and track must be set for the label track before the command is used. GETTRT reads the sector that contains the checkpoint TRT information for track NNNN and displays the byte number within the sector for that track.

HOLD.	Drop display and wait for operator assignment.
LOAD.C.	Reload the EST ordinal, track, sector, and display selection from scratch area C. C is a single alphabetic character. Legal characters are displayed on the left screen. (Refer to the STORE command.)
PACKNAM.CCCCCCC.	Use permanent file pack CCCCCCC. (The pack name is set internally in DDF. PKNW in the control point area is not changed.) The pack name is initially set to the name in PKNW.
PREAD.	Read the current sector using the read protected This command is used instead of the space bar when reading protected sectors. The keyboard must be unlocked to use this command.
PTK.NNNN.	Enter protected track. This command works the same as the TK. command except the number entered is not checked and the keyboard must be unlocked.
PWRITE.	Write the current sector using the write protected sector function. This command should be used instead of the WRITE. command when writing protected sectors.
QFT.NNNN.	Enter disk address from QFT ordinal NNNN. (0 .LE. NNNN .LE. largest QFT ordinal). The equipment, track, sector are set to the beginning of the file. "INCORRECT PARAMETER". is displayed if the QFT entry is not used or the file does not have any tracks assigned.

RANDOM.NNNN.RRRR.	Set track and sector for random address RRRR using NNNN as the first track. The current track is used as the first track if NNNN is not specified. "INCORRECT PARAMETER". is displayed if the random address is not on the chain.
RANDOM..RRRR.	Set track and sector for random address RRRR using the current track as the first track. "INCORRECT PARAMETER". is displayed if the random address is not on the chain.
RANDOM.C.RRRR.	Set track and sector for random address RRRR using the track specified by scratch register C as the first track. "INCORRECT PARAMETER". is displayed if the random address is not on the chain.
SC.NNN.	Enter sector number. "INCORRECT PARAMETER". is displayed if the sector number is too large.
SC.*.	Enter the number of the last sector on the track into the sector number.
SCAN.	Scan from the current position until the end of information control bytes are encountered. The scan stops if, at some point, the control bytes are incorrect or the track is not reserved.

SCAN.*.	Scan from the current position until the end of information indicated in the TRT is reached. This is intended to be used for the indirect access permanent file chain. NOTE - when PFM delinks a track in the middle of the chain, it does not update the track pointer in the preceding track. This will cause SCAN to stop at this point and display an error.
SHOWPF.	Display the permanent file whose catalog entry is currently displayed. The PFC display must be on the right screen to use this command. The device, track, and sector from the PFC currently displayed are used to display the file. If the device is not present, "INCORRECT PARAMETER". is displayed.
SKIPEI.	Set track and sector to EOI based on current position and information in the TRT. The number of sectors skipped is displayed on the left screen.
SKIPF.	Read the file starting at the current position until an EOF is encountered.
SKIPR.	Read the file starting at the current position until an EOR or EOF is encountered.
STB.	Set the track interlock bit for the current track. The keyboard must be unlocked to use this command.

STORE.C.COMMENT.	Store the current EST ordinal, track, sector, and display selection into scratch area C. C is a single alphabetic character (legal characters are displayed on the left screen). The comment is copied to the scratch area and displayed on the left screen. The comment is for convenience only and is truncated to 10 characters. The EST ordinal, track, sector, and display selection can be reloaded from the scratch area with the LOAD command.
TK.NNNN.	Enter track number. "INCORRECT PARAMETER". is displayed if the number is too large.
UI.NNNNNN.	Set EST ordinal and track for the catalog entries for user index NNNNNN (NNNNNN .LE. 377777). "INCORRECT PARAMETER". is displayed if the catalog is not found (this may be the result of entering the wrong family or pack name). If the family or pack name are changed, after entering the UI command, the UI command must be reentered.
WRITE.	Write the contents of the buffer to the sector currently indicated. If the keyboard is not unlocked, "INCORRECT ENTRY". is displayed.

The following commands are used to print the contents of the disk sector and manipulate the listing file.

LISTING.CCCCCCC.	Set the listing file name to CCCCCCC. The listing file is originally OUTPUT.

OUT.	Release the listing file to the output queue.
PRINT.	Print the current equipment, track, sector, TRT information, and the contents of the sector.
PRINT.NNNN.	Print the next NNNN (must be an octal number) sectors starting with the current sector. Printing will stop if EOI is encountered before NNNN sectors have been printed. One page of output is produced for each sector printed.
RETURN.	Return the listing file.
REWIND.	Rewind the listing file.
SETID.NN.	Set the ID for the output file to NN.
SKIPL.	Position to the end of the listing file.

The following commands are used to change the data in the current sector. If the comma (,) is replaced by a plus (+), the byte number is incremented after the entry is processed. The actual data on the disk is not changed until the write command is entered.

XXXX,YYYY.	Enter YYYY into byte XXXX.
XXXX,DCC.	Enter display code characters CC in byte XXXX.

7.4.2 Pack Recovery Commands

These commands are used to rebuild the label and TRT of a device that has lost them. Note that these commands do not necessarily recover the device correctly.

BEGINR.	Begin track recovery for the current selected equipment by reserving and clearing the CM recovery tables. The equipment to be recovered must be removable and unloaded. The MST in CMR for the selected equipment is used to initialize the MST for the label track built by the recovery commands. Be sure the equipment mnemonic in the E.M. display matches the pack to be recovered. If this command is used again, recovery is restarted for the current equipment.
BLDSL.	Build a label sector image in the data buffer from data in the CM recovery table. Once the other recovery commands have completed, the memory change can fill in any missing data. The buffer can then be written to disk with the write command. Note - If this command is used after the RECOVER command completes, the MST in the data buffer will show the pointers to the first track of the permit data and indirect catalog track chain. These pointers are helpful in locating the catalog track chain, which is allocated by IMS immediately preceding these two tracks. This only applies to the initial catalog tracks. Overflow tracks are linked through the last sector of one of the initial tracks.

BLDSTRT.	Build next sector of the TRT data for label track. After the last sector is built, the pointers are reset for the first sector again. Track and sector are set for WRITE command.
BLDSTRT.N.	Build sector N of the TRT data for label track. (N .GE. 1). Note - It is necessary to manually generate the EOI sector for the label track using the BLDEOI.
NEXTAT.	Locate the next available track on the current equipment and read the first sector of the track. The search begins at the displayed track pointer + 1. By resetting the track pointer to track zero and using this command, unreserved tracks can be found and checked for being part of the catalog track chain or indirect file chain.
NEXTSS.	Locate the next track in the recovery table that begins with a system sector. The search begins at the displayed track pointer + 1.
RECOVER.	Recover tracks starting at the first track of the device. If this command is used more than once after a BEGINR command, it continues recovery by finding the next system sector track.

RECOVER.C.	(C = any character.) Continue recovery by scanning from current position to EOI. Recovery data is added to the table by searching for tracks that begin with a system sector and following the linkage to EOI. Errors and anomalies are flagged. The indirect chain is not traced. The display is dropped until an error occurs, or until the search completes. It is then requested again as if a HOLD command had been used.
SETRW.B.NNNN.	Change byte B of current track recovery word to NNNN.
SETTP.NNNN.	Set recovery track pointer for RECOVER, NEXTAT, NEXTSS, and SETRW commands to NNNN.

7.5 DDF Examples

7.5.1 Example 1

Read the system sector for permanent file ABC on user index 1234 on family XYZ.

X.DDF.	Bring up DDF.
FAMILY.XYZ.	
UI.1234.	Set EST ordinal, track, and sector of the catalog track.
Space bar	Read the first catalog sector.
FIND.ABC.1234.	Search the catalog track for the PFC entry.
SHOWPF.	Set the EST ordinal, track, and sector of the file.
Space bar	Read the system sector of the file.

7.5.2 Example 2

Look at the PFC entries for user index 1234 on family XYZ.

X.DDF.	Bring up DDF.
FAMILY.XYZ.	
UI.1234.	Set EST ordinal, track, and sector of the catalog entry.
Space bar	Read the first catalog sector.
FIND..1234.	Search for the first file for the user index 1234.

7.5.3 Example 3

While using DIS, look at the contents of a local file. Be sure to remember the FNT ordinal (NN) of the file.

DDF.	Switch from DIS to DDF.
FNTL.NN.	Set the EST ordinal, track, and sector to the system sector of the file.
or	
FNTLC.NN.	Set the EST ordinal, track, and sector to the current position of the file.
Use +,-, and /	Look at different parts of the file.
DIS.	Switch from DDF to DIS.

7.5.4 Example 4

Display a system file (e.g. the VALIDUS file).

Look at the DSD H display to determine the FNT ordinal(NN).

X.DDF.	Bring up DDF.
FNT.NN.	Set the EST ordinal, track, and sector to the system sector of the file.

7.5.5 Example 5

Determine the length of a file and display the EOI sector.

Display the system sector of the file (see previous examples).

SKIPEI.	Set the track and sector to the file EOI and display the number of sectors skipped above the command.
Left blank	Clear the message and command.
Space bar	Read the EOI sector.

7.5.6 Example 6

Verify that a file can be read and that the linkage bytes are correct.

Display the first sector of the file.

SCAN.	Read every sector from the current position in the file to the EOI. If any errors are encountered, SCAN stops and displays a message. SCAN verifies that the linkage bytes match the TRT.

7.5.7 Example 7

Print the contents of sectors. First display the first sector.

PRINT.N.	N=octal number of sectors to print. Default = 1.
OUT.	Put the output in the output queue to be printed.
DIS.	File OUTPUT contains the output.

7.5.8 Example 8

Often it is necessary to look at several different places on the disk. DDF has a facility to remember disk locations.

X.DDF.	Bring up DDF.
FAMILY	
UI	Set EST ordinal, track, and sector.
Space bar	Read PFC sector.
FIND	Examine file.
STORE.A.PFC	Save the current EST ordinal, track, and sector in scratch area A. Save PFC as a comment. This information is displayed on the left screen.
SHOWPF.	
Space bar	
STORE.B.SYSTEM SC	
SKIPEI.	
Left blank	
STORE.C.EOI	
LOAD.A.	Reset to display the PFC entry.
LOAD.B.	Reset to display the system sector.
LOAD.C.	Reset to display the EOI.

7.5.9 Example 9

Changing data on the disk.

> WARNING
> Be very careful and be sure you know what you are doing before changing data on the disk.

Before proceeding, do the following:

- Display the sector to be changed.
- Determine from the display the byte number of the byte to change.
- Be sure another program does not change the sector between the time DDF reads and writes it otherwise changes may be lost. Depending on what you are changing, it may be best to only make changes when the system is idle.

XXXX,YYYY.	Set byte XXXX to value YYYY. If a mistake is made, press space bar to reread the sector and start over.
*	Return to DSD.
UNLOCK.	Allow privileged commands to be entered.
*	Return to DDF.
WRITE.	Write what is displayed to the disk.

7.5.10 Example 10

Display the disk flaw map. First, determine (from the hardware manual) the logical track and sector of the flaw map.

X.DDF.	Bring up DDF.
EST.NN.	Set EST ordinal.
PTK.NNNN.	Set the track number. DDF does not validate that the track number is valid.
SC.NN.	Set the sector number.
PREAD.	Read the sector. NOTE - reading the sector with space bar will cause a disk error because it is a protected sector.

(This page left intentionally blank.)

CHAPTER 8

CCL Features

NOS now has the ability to reference and manipulate long strings in CCL expressions. Four new functions and a new CCL directive have been added to aid in the use of long strings.

8.1 New Functions

8.1.1 LEN

The LEN function returns the length of a string. The format of the function is:

```
LEN(string)
```

string A string of any length up to the maximum length of a CCL line, which defaults to 150. If the string is a literal, the length returned is the length of the string after the removal of the surrounding $ and any other CCL editing characters.

8.1.2 STR

The STR function returns a left justified string of any length up to the maximum length of a CCL line. The format of the STR function is:

```
STR(strexp,lc,rc)
```

strexp Any legal CCL expression, usually, but not limited to a literal. Literals are left justified and may be any length up to the maximum CCL line length, less the length of the rest of the line.

lc Leftmost character. This parameter must be an expression producing a positive or negative numeric result. This value indicates which character of strexp is to be the leftmost character of the result. If lc is positive, then the character position is found by counting from the left of strexp, if negative, from the right. A value of 0 or a negative value exceeding the length of strexp is equivalent to 1. A positive value greater than the length of strexp produces a null string.

rc Rightmost character. This parameter must be an expression producing a positive or negative numeric result. The value indicates the rightmost character of the result. If rc is positive, then the position is found by counting from the left, if negative, from the right. A value greater than the length of strexp is considered equal to it. A value of 0 or a negative value greater than the length of strexp produces a null result.

The lc and rc parameters may be omitted if you want all of strexp returned. If lc is omitted then rc must also be omitted. The rc value may be omitted if the rightmost character is the rightmost character of strexp.

8.1.2.1 Example

The following examples all produce the string 'ABET'.

```
STR($ALPHABETAG$,5,-3)
STR($ALPHABETAG$,-6,8)
STR($ALPHABET$,5)
```

8.1.3 STRB

This function returns a string of 1 to 17 characters representing the octal value of an expression. The format of this function is:

```
STRB(numexp,lc,rc)
```

numexp Any legal CCL expression. If numexp is or contains a literal, then the literal can contain no more than 10 characters and will be right justified and binary zero filled before evaluation.

lc See lc parameter for STR function.

rc See rc parameter for STR function.

8.1.3.1 Example

Each of these examples will produce the string '102'.

```
STRB($ABC$,-5,-3)
STRB(10203B,-5,-3)
```

8.1.4 STRD

This function return a string of 1 to 16 characters representing the decimal value of an expression. The format of this function is:

```
STRD(numexp,lc,rc)
```

numexp Any legal CCL expression. If numexp is or contains a literal, then the lireral can contain no more than 10 characters and will be right justified and binary zero filled before evaluation.

lc See lc parameter for STR function.

rc See rc parameter for STR function.

8.1.4.1 Example

The following example produces the string '0012'.

```
STRD(10000+36/3,-4)
```

8.2 New Operator

8.2.1 Concatenation

A new CCL operator for concatenation has been defined. The operator // or .CAT. may be used to create strings from existing strings or substrings (created by STR, STRB and STRD functions).

8.2.1.1 Example

Given the following CCL procedure:

```
.PROC,CATTER*I,P1=(*A),P2=(*S3/D).
.IF,STR($P1$,1,4)//STRD(P2,-1) .EQ.  $GOOD1$,GO.
  NOTE./  THE PARAMETERS WORK.
.ELSE,GO.
  REVERT,EX.DISPLAY(STR($THE PARAMETERS DO NOT WORK.$))
.ENDIF,GO.
```

This procedure accepts two parameters, P1 and P2. If the first 4 characters of P1 are 'GOOD' and the last character of P2 is '1', then the .IF evaluates as true and the NOTE statement is executed, otherwise the DISPLAY statement executes.

8.3 New CCL directive

8.3.1 .SET

The .SET directive allows you to create new parameters using strings and substrings. The .SET directive is a CCL directive and not a NOS command. It functions only during BEGIN processing while the procedure is being expanded.

The format of the .SET directive is:

```
.SET,keywd1=strexp1,...,keywdi=strexpi.
```

keywdi This defines a new keyword or references an existing keyword from the .PROC header or from an earlier .SET directive. This keyword can then be used for substitution in subsequent statements as if it had been declared on the header statement.

strexpi Any valid CCL expression. If the result of strexpi is not a string, it is left justified and treated as a string.

8.3.1.1 Example 1

```
.PROC,CATTER*I,P1=(*A),P2=(*S3/D).
.SET,NEWPARA=STR($P1$,1,4)//STRD(P2,-1).
.IF,$NEWPARA$ .EQ.  $GOOD1$,GO.
  NOTE./  THE PARAMETERS WORK.
.ELSE,GO.
  REVERT,EX.DISPLAY(STR($THE PARAMETERS DO NOT WORK.$))
.ENDIF,GO.
```

This is a re-write of the last example using .SET to create a new keyword 'NEWPARA' that can be replaced by the string resulting from the concatenation of substrings from P1 and P2.

8.3.1.2 Example 2

```
.PROC,VALUES*I,P1=(*S/D),P2=(*S/D).
.SET,V1=STRD(P1+P2),V2=STRD(P1*P2),V3=STRD(P1**P2).
.IF,V1=V2=V3.NOTE./SUM, PRODUCT AND POWER ARE EQUAL.
NOTE./  THE SUM OF P1 AND P2 IS V1.
NOTE./  THE PRODUCT OF P1 AND P2 IS V2.
NOTE./  V1 TO THE V2 POWER IS V3.
```

This example also shows how the .SET directive can be used to create keywords that can be used as strings in other commands or directives.