70/34-45-55 TAPE-DISC OPERATING SYSTEM (TDOS) UTILITY ROUTINES

RCA
Information
Systems

SPECTRA 70

SYSTEMS 70/35-45-55

Tape-Disc Operating System (TDOS) Utility Routines

RCA
Information
Systems

SPECTRA 70

SYSTEMS 70/35-45-55

Tape-Disc Operating System (TDOS) Utility Routines

70-35-306
May 1969

The information contained herein is subject to change without notice. Revisions may be issued to advise of such changes or additions.

First Printing: April 1967
Edition: October 1967
Edition: June 1968
Edition: May 1969

The October 1967 edition includes revisions 1 and 2.

The June 1968 edition includes revisions 1 thru 4.

The May 1969 edition includes revisions 1 thru 6.

CONTENTS

1. INTRODUCTION

♦ The TDOS Utility System comprises an integrated set of generalized routines designed to relieve programming effort and to simplify testing and production operations for the TDOS installation. The components of this system include a variety of routines in the areas of data conversion, program testing, and maintenance of both RCA and installation library systems.

The routines supplied for this system are an adaptation of the routines provided for the Tape Operating System (TOS), and in most cases operate in the same manner. The basic difference between the two systems is that TDOS utility routines may be operated from a70/564 Disc Storage Unit or a 70/565 Drum Memory Unit, rather than from magnetic tape.

Two special library conversion routines are provided: the Program Library Transcriber, which is used to transcribe loadable TOS/TDOS and TDOS programs from magnetic tape to random access; and the Call Library Transcriber, which is used to transcribe a TDOS Call Library to random access.

Except for the Linkage Editor, the Test Data Generator, the Automatic Integrated Debugging System, and the Library Conversion routines, the following publications contain a complete description of the utility routines provided for the TDOS system:

TOS UTILITY ROUTINES, 70-35-302

TOS SORT/MERGE SYSTEM, 70-35-303

The reader is also advised to consult the Tape/Disc (TDOS) Operator's Guide, 70-35-404, which contains the operating procedures for these routines, and the Spectra 70 Systems Standards Reference Manual, 70-00-610, which describes system standards related to such areas as label conventions and record formats.

2. PERIPHERAL CONVERSION

◆ The Peripheral Conversion routines used with TDOS are the same as those used with TOS. These routines are listed below and are described in the TOS Utility Routines manual, 70-35-302.

TAPE VOLUME INITIALIZER (TPINIT)
CARD TO TAPE (CDTP)
SELECTIVE CARD TO PRINTER AND/OR PUNCH (CDPR)
TAPE TO TAPE (TPTP)
SELECTIVE TAPE TO PRINTER AND/OR PUNCH (TPPR)
TAPE DUPLICATE (DUP)

3. PERIPHERAL CONVERSION - RANDOM ACCESS

◆ The Random Access Peripheral Conversion routines used with TDOS are the same as those used with TOS with the exception of a minor restriction in the Random Access Volume Initializer routine. The routines that are the same in both systems are listed below and are described in the TOS Utility Routines manual, 70-35-302.

RANDOM ACCESS TO PRINTER AND/OR PUNCH (RAPR)

RANDOM ACCESS TO TAPE (RATP)

RANDOM ACCESS TO RANDOM ACCESS (RARA)

TAPE TO RANDOM ACCESS (TPRA)

CARD TO RANDOM ACCESS (CDRA)

RANDOM ACCESS TO RANDOM ACCESS/MASS STORAGE (RARAM)

TAPE TO RANDOM ACCESS/MASS STORAGE (TPRAM)

CARD TO RANDOM ACCESS/MASS STORAGE (CDRAM)

The Random Access Volume Initializer (RAINIT) restrictions are described below.

RANDOM ACCESS VOLUME INITIALIZER (RAINIT)

◆ The TOS Volume Initializer routine operates in the same manner under TDOS in preparing and formatting random access volumes (refer to TOS Utility Manual).

There is one restriction which applies only when initializing the disc pack or drum to be the resident device for the TDOS Executive. In this case, the location of the VTOC must be specified in the VOLIN parameter card, and it may not be assigned to track 0, cylinder 0.

4. DIAGNOSTICS

◆ The Diagnostic routines used with TDOS are the same as those used with TOS except for the Test Data Generator and the Automatic Integrated Debugging System* routines. The routines that are the same in both systems are listed below and are described in the TOS Utility Routines manual, 70-35-302.

EXECUTIVE DUMP PRINT (DUMPRT)
SELF-LOADING MEMORY PRINT
SNAPSHOT
SELF-LOADING TAPE EDIT
TAPE EDIT (TPEDIT)
TAPE COMPARE (TPCOMP)

The TDOS Test Data Generator routine (DIAGDG) is described below.

TEST DATA GENERATOR (DIAGDG)

General Description

◆ The Test Data Generator routine automatically prepares files of program test data generated onto punched cards, magnetic tapes, random access volumes, or paper tape. This routine can be used to produce single or multivolume files, or multifile volumes.

For output tapes, standard Spectra 70 labels can be automatically generated; the output tapes may be unlabeled; or the programmer may supply his own label set. For random access output, standard Spectra 70 user header and trailer labels may be supplied by the programmer.

Records generated for the output files can vary in length, contain up to 12 data fields, and can be blocked or unblocked. Records can also contain programmer-supplied data.

Preset Functions

This routine is not preset to perform any functions for a random access file. It is, however, preset to perform the following functions for a magnetic tape file mounted on logical TDG001:

Rewind the output tape to BOT.

Generate 1,000 unblocked, 80-byte records to the output tape. (The first field of each record is a 10-byte, zoned-decimal field, with each succeeding record incremented by 10. All other positions of the record contain the fill character X.)

Rewind the output tape to BOT and deallocate logical device TDG001.

*To be supplied.

General Description *(Cont'd)*

Note:

If the output tape contains VOL and HDR labels, a purge-date check is made to determine if the tape is releasable. If not, an error halt occurs.

The preset function of this routine creates an output tape in the following format:

TM data TM TM

Optional Functions

The following options may be selected by programmer-supplied parameter cards:

1. Designating a random access volume as the output file.
2. Designating up to 12 data fields per file, which may vary in size and format.
3. Designating the length (fixed or variable) of the test records to be generated; blocking of test records; specifying the number of blocks to be generated per file or per volume.
4. Specifying for magnetic tape a standard-labeled file, an unlabeled file, or providing programmer-prepared label sets for the output file.
5. Specifying random access test records with Keys.
6. Providing programmer-prepared user header and trailer label sets for the random access output file.
7. Limiting the number of test records generated on random access by extent, by right-hand end address, or by number of records desired.
8. Producing multifile volumes or multivolume files.
9. Deallocating the output device when multivolume output is desired.

Input

◆ No input is required when all preset functions are used; for optional functions and random access output, the programmer must supply the appropriate parameter cards.

Output

◆ This routine is preset to produce a single-volume unlabeled file, generated to a magnetic tape mounted on logical device TDG001.

When optional functions are elected, the output can be generated to a random access volume, magnetic tape, paper tape, or punched cards.

Equipment Configuration

Required

◆ Processor (65K).

Console typewriter.

Card reader, or Videoscan document reader with card read feature.

Disc storage unit or drum memory unit.

Optional

◆ An additional random access device (70/564, 70/565, 70/568), magnetic tape, card punch, or paper tape punch may be used as the output device.

A printer is required if a listing of routine parameters is desired at generation time.

Routine Parameters - General

◆ The Test Data Generator routine requires no parameters for its preset options, which apply to all output devices except random access devices. For other than preset options, and for random access output, the following parameters are entered from the card reader.

File Parameter

For magnetic tape, punched cards, or paper tape, this parameter describes the record length (fixed or variable); the number of blocks to be generated and the number of records per block (fixed or variable); provides for logging of routine parameters, label generation, and the fill character for unused positions of the test records.

Label Parameters for Magnetic Tape (Optional)

These parameters contain programmer-supplied output tape labels.

Record Parameter

For random access devices, this parameter describes the record length (fixed or variable), the number of blocks to be generated, and the number of records per block (fixed or variable); provides for logging of routine parameters and the fill character for unused positions of the test record; provides for generation of records with or without Keys; and provides three options to terminate test data generation:

1. Data will be generated for all extents for a file.
2. Data will be generated for the first extent through a given right-hand end address.
3. A specified number of records will be generated.

Label Parameters for Random Access Devices

These parameters contain file identification and programmer-supplied user header and trailer labels.

Data Parameter

This parameter describes the number and format of the data fields to be generated within each output record.

Routine Parameters - General *(Cont'd)*

Device Deallocation Parameter

This parameter permits the device assigned to TDG001 to be deallocated and another device assigned.

END Parameter

This parameter signifies the end of parameter information.

Routine Parameters - Detailed

FILE Parameter for Magnetic Tape, Punched Cards, or Paper Tape

◆ When other than preset functions are desired, a FILE parameter is mandatory for each file to be generated on magnetic tape, punched cards, or paper tape.

Format

ΔFILEnΔaaaa,bbbb,cccc,dddd,fpt,eeee

Card Columns	Content	Meaning
1		Not used, leave blank.
2-5	FILE	Parameter identifier.
6	n	File identifier (any alphanumeric character).
7		Not used, leave blank.
8-11	aaaa	Minimum record length (0012-9999). See note 3.
12-16	,bbbb	Maximum record length (0012-9999). See note 3.
17-21	,cccc	Minimum number of records for each output block (0001-9999).
22-26	,dddd	Maximum number of records for each output block (0001-9999).
27-28	,f	Fill character to appear within unused positions of output records: f = any alphanumeric character.
29	p	Display generation parameters to printer: p = 0 no. = 1 yes.
30	t	Generate standard labels on output file: t = 0 no. = 1 yes (destroying existing labels). = 2 yes (retaining any existing labels). See note 1.
31-35	,eeee	Number of blocks to be generated for output file (0001-9999). See note 2.

FILE Parameter for Magnetic Tape, Punched Cards, or Paper Tape (Cont'd)

Notes:

1. When automatic label generation is selected, the following labels are produced:

Label	Serial Number	Owner Name	File ID
VOL1	TDG001	TESTΔDATA	
HDR1	TDG001		*FILEx
EOV1/EOF1	TDG001		*FILEx

   When automatic label generation is not selected, the output tape will be unlabeled, or the programmer may provide the label set to be used immediately following the File Parameter. See examples on page 4-6.

2. If the columns following the "eeee" entry are blank, the output file will contain the number of blocks specified. However, if desired, the programmer can force an end-of-volume condition by specifying the number of blocks to be generated for each output volume up to eight volumes. In this case, the format of the File parameter is extended as follows:

Card Columns	Content	Meaning
31-35	,eeee	Number of blocks to be generated for first output volume (0001-9999).
36-40	,eeee	Number of blocks to be generated for second output volume (0001-9999).
41-45 . .	,eeee	Number of blocks to be generated for third output volume (0001-9999).
66-70	,eeee	Number of blocks to be generated for the eighth output volume (0001-9999).

   For example, to generate a file consisting of 3 output volumes, each of which contains 50 blocks:

Columns	
31-35	,0050
36-40	,0050
41-45	,0050

*x = character appearing in column 6 of File card.

FILE Parameter for Magnetic Tape, Punched Cards, or Paper Tape (Cont'd)

When this option is used, the Test Data Generator automatically deallocates the output device after each volume has been generated. The console operator must then reallocate the next device to be assigned for TDG001.

Examples:

```
ΔFILEAΔ0080,0080,0001,0001,A11,0200

ΔFILEBΔ0040,0222,0003,0008,A11,3333
```

3. This routine provides for a maximum output block of 2,000 bytes. If larger blocks are desired, additional memory can be allocated at load time, or this routine can be processed through the Linkage Editor.

Label Parameters for Magnetic Tape

◆ If the automatic generation of labels has not been requested, and the programmer wishes to supply his own label set, label parameter cards are prepared as follows. Note that these cards must follow the File parameter.

Format	Remarks
ΔVOL1.....text.....	
ΔHDR1.....text.....	
ΔUHL1.....text.....	optional
ΔUTL1.....text.....	optional
ΔEOV1.....text.....	optional
ΔEOF1.....text.....	

Notes:

1. Only one label of each type is permissible. If multiple labels are provided, only the last label of the set is accepted.

2. All label cards must be 80 characters and conform to Spectra 70 standards. Column 1 of each card must be blank; the information supplied in columns 2-80 will be placed in columns 1-79 of the output label.

Record Parameter for Random Access Devices

◆ A RECRD parameter is mandatory for each file to be generated on a random access device.

Format

$$\Delta\text{RECRD}\Delta\text{aaaa,bbbb,cccc,dddd,fpk}\left\{\begin{array}{l}\text{,mmcccch}\\\text{,rrrrr}\end{array}\right\}$$

Card Columns	Content	Meaning
1		Not used; leave blank.
2-6	RECRD	Parameter identifier.
7		Not used; leave blank.
8-11	aaaa	Minimum record length (0001-9999). See note 1.
12-16	,bbbb	Maximum record length (0001-9999). See note 1.
17-21	,cccc	Minimum number of records for each output block (0001-9999).
22-26	,dddd	Maximum number of records for each output block (0001-9999).
27-28	,f	Fill character to appear within unused positions of output records: f = any alphanumeric character.
29	p	Display generation parameters to printer: p = 0 no. = 1 yes.
30	k	Indicates if a Key field is to be generated preceding the data field of each output record. k = 0 no Key to be generated. = 1-C Indicates which data field, as defined in the DATA parameters, is to be duplicated and used as the Key. (If k = 3, the third data field defined will be used as the Key.)
31-38		Determines the termination of a run:
	blank	All extents of this file as defined in the VTOC shall be filled with data. Note: A VOL card must follow and contain the file identification.

(Cont'd)

Record Parameter for Random Access Devices (Cont'd)

Card Columns	Content	Meaning
31-38 (Cont'd)	,mmcccch	,mmcccch The address of the rightmost limit (right-hand end) of the file beyond which a record will not be generated, where: mm = 00 when output device is disc or drum. = 00-15 magazine number for mass storage unit. cccc = cylinder number (0000-4095). h = head number (0-9). Notes: 1. If a VOL card is supplied, all extents for this file preceding this address will be filled with data. 2. If a VOL card is <u>not</u> supplied the user may supply a starting address (the left-hand end) through the console typewriter. (See note 2.)
	,rrrrr	,rrrrr total number of records to be generated for this file. Data generation will begin at the left-hand end of the file as defined in the VTOC. Note: A VOL card must follow and contain the file identification.

Notes:

1. This routine provides for a maximum output block of 2,000 bytes. If larger blocks are desired, additional memory can be allocated at load time, or this routine can be processed through the Linkage Editor.

2. All random access volumes must be initialized by the TDOS Random Access Volume Initializer.

   It is also recommended that the volume be allocated by the TDOS Random Access Storage Allocator, as the Volume Table of Contents (VTOC) is accessed by this routine for file generation. However, the following special option exists for the non-VTOC user:

   When the routine senses that the file will be generated to random access but does not find a VOL card immediately following the RECRD parameter, it types out the following message:

   4620A NO VOL LABEL

Record Parameter for Random Access Devices (Cont'd)

The programmer can then enter the starting address (left-hand end) for the file through the console typewriter as follows:

Response	Meaning
C,mmcccch	mm = magazine number for mass storage. = 00 for disc or drum.
	cccc = cylinder number.
	h = head number.

Note:

The right-hand end of the file must have been supplied in the RECRD parameter card.

Examples:

1. To fill all extents of a file, RCA PAYROLL, as defined in the VTOC with unblocked records:

```
ΔRECRDΔ0100,0100,0001,0001,Δ10
ΔVOLΔRCAΔPAYROLL
```

2. To generate 2,000 unblocked records beginning at first extent of the file, RCA PAYROLL

```
ΔRECRDΔ0100,0100,0001,0001Δ10,02000
ΔVOLΔRCAΔPAYROLL
```

Label Parameters for Random Access Devices

◆ When the programmer wishes to generate data to areas defined in the Volume Table of Contents, a VOL parameter card must immediately follow the RECORD parameter.

Card Columns	Format
1-5	ΔVOLΔ
6-49	File Identification (as found in the Format 1 file label in the VTOC).

If the programmer wishes to supply user Header and Trailer labels, additional label parameter cards are prepared as follows:

Format

```
ΔUHL1....text....
ΔUTL0....text....
```

Notes:

1. User labels cannot be generated for mass storage.
2. Only one label of each type is permissible. If multiple labels are provided, only the last label of the set is accepted.
3. All user label cards must be 80 characters and conform to Spectra 70 standards. Column 1 of each card must be blank; the information supplied in columns 2-80 will be placed in columns 1-79 of the output label.

DATA Parameter

◆ One DATA parameter is required for each file described. The DATA parameter must immediately follow the FILE or RECRD parameter, or the final label card, for the file to which it refers.

Format

ΔDATAΔnnppppfsiiixxxx,..........,nnppppfsiiixxxx

Card Columns	Content	Meaning
1		Not used; leave blank.
2-5	DATA	Parameter identifier.
6		Not used; leave blank.
7-8	nn	Length of data field (01-15).
9-12	pppp	Position of the leftmost character of data field relative to the first character in the record. (The first character of the record is considered position 0000.)
13	f	Format of data field: f = 0 alphabetic. = 1 decimal, packed. = 2 decimal, zoned. = 3 binary. = 4 ASCII. = 5 Baudot (teletype). = 6 Baudot (dataspeed).
14	s	Sequence of data field: s = 0 sequential for all above formats. = 1 random for all above formats. = 2 sequential for all above formats in groups incremented by 1 after each group (see iii entry below). = 3 sequential by List entries. = 4 random by List entries.

DATA Parameter (Cont'd)

Card Columns	Content	Meaning
15-17	iii	Increment value for data field: a. If "s" entry is 0: iii = 001-999. Increment is converted to a hexadecimal or decimal value depending on format of data to be generated. b. If "s" entry is a 1 or 4: iii = 000. c. If "s" entry is a 2: iii = 001-999. In this case, iii specifies the number of consecutive records in the file that are to have the same data field values. Each succeeding group of records will have its value incremented by 1. d. If "s" entry is a 3: iii = 001-999. In this case, iii specifies the number of times the data field will be repeated before the next data field in the card is accessed.
18-21	xxxx	a. If sequential fields have been specified: xxxx = value of the data field in the first record. b. If random fields have been specified, this entry is left blank. c. If the "s" entry is a 3 or 4, this is the number of entries in the List.
22-37	,nnppppfsiiixxxx	Requirements for data field 2, in the same format described above. (See note 1.)
38-53	,nnppppfsiiixxxx	Data field 3 requirements.
54-69	,nnppppfsiiixxxx	Data field 4 requirements.

Notes:

1. If the "s" entry is a 3 or 4, columns 22-71 are considered to be the user's data for the List.
2. No more than four data fields may be indicated in any DATA parameter; however, a total of 12 fields will be accepted for each record.
3. Entries for data fields 2, 3, and 4 are optional.

Examples:

ΔDATAΔ090000200010001,050009200020300

ΔDATAΔ030000030020004RCATOSRCAPOS
(All data records: RCARCATOSTOSRCARCAPOSPOS)

Device Deallocation Parameter

◆ This parameter is used only to deallocate the device assigned to TDG001. It is required each time that a new file is to be generated starting on a new volume.

Format

ΔEOD

END Parameter

◆ This parameter signifies the end of parameter input and must be used when other than preset functions are desired.

Format

ΔEND

Considerations for Use

◆ 1. If multiple FILE or RECRD parameters are supplied, and the Device Deallocation parameter (EOD) is not used, a multifile output volume is produced. If the EOD parameter is supplied, the next file will begin on a new volume. See examples.

2. This routine will not generate overflow records or blocks to a random access volume.

Parameter Examples

◆ 1. Generation of a Single-Volume File

ΔRECRD
ΔVOL
(optional user label cards)
ΔDATA
ΔEND

2. Generation of a Multivolume File

ΔRECRD
ΔVOL
(optional label cards)
ΔDATA
ΔEND

3. Generation of a Multifile Volume

ΔRECRD
ΔVOL
(optional label cards)
ΔDATA
ΔRECRD
ΔVOL
(optional label cards)
ΔDATA
ΔRECRD
ΔVOL
(optional label cards)
ΔDATA
ΔEND

Parameter Examples *(Cont'd)*

4. Generation of Multifiles, with each File on a Separate Volume

```
ΔRECRD
ΔVOL
(optional label cards)
ΔDATA
ΔEOD
ΔRECRD
ΔVOL
(optional label cards)
ΔDATA
ΔEOD
ΔRECRD
ΔVOL
(optional label cards)
ΔDATA
ΔEND
```

5. To generate a single volume file of eight-character records, associating two numeric codes with each of three alphabetic fields:

```
ΔRECRDΔ0008,0008,0001,0001,A10
ΔVOLΔSTATIONΔFILE
ΔDATAΔ050000030020003WFILΔWCAUΔKYWΔΔ
ΔDATAΔ03000523001000211Δ15Δ
ΔEND
```

```
1st data record:WFILΔ11Δ
2nd data record:WFILΔ15Δ
3rd data record:WCAUΔ11Δ
4th data record:WCAUΔ15Δ
5th data record:KYWΔΔ11Δ
6th data record:KYWΔΔ15Δ
7th data record:WFILΔ11Δ
```

Note that the records will be repeated starting with the seventh record generated until all extents of this file are filled with data records.

Device Assignments

- Under Executive Control:

SDN	Device Type	Remarks
TDGRDR	Card reader.	Parameter input.
TDG001	Random access, magnetic tape, card punch, or paper tape punch.	Output device.
TDGLST	Printer.	To display parameters.

Device Assignments *(Cont'd)*

Under Monitor Control:

SDN	Device Type	Remarks
SYSIPT	Card reader.	Parameter input.
TDG001	Random access, magnetic tape, card punch, or paper tape punch.	Output device.
SYSLST	Printer.	To display parameters.

AUTOMATIC INTEGRATED DEBUGGING SYSTEM (TDSAID)

General Description

◆ The Automatic Integrated Debugging System (AIDS) provides either a console-controlled or an automatic method for testing TDOS programs. The console-controlled method is controlled by parameters entered from the console and the card reader while program testing is in progress. In this case the programmer controls the test session or he may direct the operator as to what parameters to use. The automatic method requires no operator intervention; parameters are entered automatically by means of the card reader or from a magnetic tape device.

Although AIDS runs under Executive control, the routine modifies Executive areas and, therefore, cannot be run in the multiprogramming mode.

It is not necessary to make any special changes to programs to be tested.

All programs run under AIDS are not altered in any way.

Automatic Testing

◆ All testing in the automatic system is controlled by parameters entered from the card reader. Each programmer can set up the tests for his program; then, all tests become part of the AIDS job stream. Each program is tested until all requests have been satisfied or an unrecoverable error halt occurs. AIDS then automatically proceeds to the next program, or terminates if there is no more input.

Preset Functions

None.

Optional Functions

1. Automatic assignment of work and output devices to the program to be tested.
2. Allocation of files on random access devices. These files may be deallocated or saved for use by subsequent programs in the AIDS job stream.
3. Generation of test data to tape or random access devices.
4. Use of run-time parameters by the program to be tested.
5. Selection of information to be displayed (memory prints, tape edits, random access edits) and the display medium (printer or tape).
6. Diagnostic functions (such as traces and snapshot prints) performed as specified by input parameters.
7. Patches applied to the program or stored in the AIDS program with linkage set up between the patch and the program.

Console-Controlled Testing

◆ Testing in the console system is accomplished by requests entered from the Console Typewriter. The request may be a test parameter or may cause AIDS to read a test parameter from the card reader. Control is returned to the console each time a test is performed. The programmer can then enter another test, return control to the program being tested, or terminate the program or AIDS. Every program to be tested in the console system must be loaded individually from the console.

Preset Functions

None.

Optional Functions

1. Printing of registers and selected parts of memory.
2. Displays to the console typewriter of registers and portions of memory.
3. Changing registers and portions of memory.
4. Inserting test points in the program to be executed a specified number of times before control is returned to the console.
5. Diagnostic functions (such as traces and snapshot prints) performed as specified by parameters entered from the console or the card reader.
6. Patches applied to the program or stored in the AIDS program with linkage set up between the patch and the program.

Input

◆ The input to this routine consists of (1) a program or series (batch) of programs to be tested, (2) the user's test data or test data generated from user requirements, and (3) routine parameters entered automatically or from the console.

Output

◆ Outputs from the AIDS routine are program diagnostic data which can be displayed on the typewriter, the printer, or written to magnetic tape.

Typewriter outputs consist of memory or register displays selected by console-controlled program testing.

Printer outputs consist of tape edits, random access edits, memory prints, traces, and snapshots resulting from automatic or console-controlled input parameters. Also, all typewriter messages and replys are listed on the printer.

Memory prints, tape edits, trace output, and snapshot prints can be written to magnetic tape instead of the printer for subsequent printing.

Equipment Configuration

Required

- Processor (65K).

  Console typewriter.

  Printer.

  Card reader, or Videoscan document reader with card read feature.

  Other devices required by the program to be tested.

Optional

- Magnetic tape devices may be substituted for the card reader and the printer.

Routine Parameters - General

- The parameters used for automatic and console-controlled testing are summarized in the tables below.

Table 4-1. Automatic Testing Parameters

Parameter	Function
Program ID	Gives the name of the program to be tested.
Device	Informs AIDS of devices needed by program under test.
Snapshot	Requests a snapshot of specified areas of memory.
Trace	Requests a trace of all or specified instruction areas in the program.
Patch	Adds or exchanges data in the program.
End Program	Defines the end of input parameters for the current program.
VOL	Identifies the File ID and serial number of a random access volume and defines editing options and the disposition of the file.
Limit	Defines the number and sizes of extents for random access files to be allocated by AIDS for programs under test.
RTP	Informs AIDS that run-time parameters follow.
END AIDS	Indicates the end of all AIDS input.

Routine Parameters - General *(Cont'd)*

Table 4-2. Console-Controlled Testing Parameters

Parameter	Function
Continue	Gives control to test program and returns control to the programmer at segment loads.
Proceed	Gives control to test program but does not return control to the programmer at segment loads.
Open Diagnostic Device	Opens AIDS output tape.
Close Diagnostic Device	Closes AIDS output tape.
Read Device	Reads parameters from card reader or magnetic tape.
Memory Print	Prints registers and selected parts of memory.
Display Memory	Displays a limited portion of memory.
Change Memory	Makes limited changes to memory.
Display Registers	Displays general purpose, floating-point, and status registers.
Change Registers	Changes general purpose or floating-point registers.
Address Stop	Inserts a test point in the program to be executed a specified number of times, then returns control to the programmer.
Snapshot	Requests a snapshot of specified areas of memory.
Trace	Requests a trace of all or specified instruction areas in the program.
Patch	Adds or exchanges data in the program.
Write Tape Mark	Write tape marks to the AIDS output tape.
End	Defines the end of input parameters, terminates a program under test, or terminates the AIDS routine.

Routine Parameters - Automatic Testing - Detailed

◆ All automatic testing parameters are entered from the card reader or magnetic tape. The following discussions refer to theprogram to be tested by AIDS as the "test program."

Program ID Parameter

◆ This parameter identifies the name of the test program and must be the first parameter submitted for the program.

Format:

ΔPROGΔpppppp

Card Column	Entry	Meaning
1-6	ΔPROGΔ	Parameter identifier.
7-12	pppppp	Name of test program (one to six characters).

Device Parameter

◆ This parameter specifies the input, output, and work devices used by the test program. A card is submitted for each magnetic tape and random access device used. Cards are also submitted for card and paper tape readers and punches, and printers. If more than one reader, punch, or printer is used, only one card has to be submitted for each device type. The Executive will request assignment of any additional readers, punches, or printers. Device cards for magnetic tape and random access files specify whether they are used for input, work, or output. For input devices, this card indicates if the file is to receive test data generated by AIDS.

Device parameters must be entered before any test parameters.

Format (Magnetic Tape):

ΔDEVΔaaΔddddddΔFf,Oo,Cnnnnn,PnnnΔ...ΔbΔFf,Oo,Cnnnnn,PnnnΔ..

ΔeRnnn

Card Column	Entry	Meaning
1-5	ΔDEVΔ	Parameter identifier.
6-8	aaΔ	aa = OT output tape. = WT work tape. = TD input tape to receive test data generated by AIDS*. = IT input tape containing user-supplied test data.
9-15	ddddddΔ	Symbolic name used by test program for device.

*See page 4-41, Test Data section.

Device Parameter (Cont'd)

Card Column	Entry	Meaning
16-32 (See Notes 1 and 3)	Ff	Print format for tape prints: f = G EBCDIC graphics. = H hexadecimal. If blank, the format is hexadecimal with graphic equivalents.
	,Oo	Print option: o = 0 rewind to BOT; print to double tape mark. = 1 rewind to BOT; print x blocks. = 2 rewind to BOT; print x tape marks. = 3 rewind x blocks; print x blocks. = 4 rewind x tape marks; print x tape marks. = 5 print x blocks from current position. = 6 print x tape marks from current position. = 9 print to double tape mark from current position. If an invalid print option is given, the last block read or written is printed.
	,Cnnnnn	nnnnn = decimal count for print option (00000-99999) specified by O entry. If blank, a count of 00001 is assumed.
	,Pnnn	nnn = printer size (132 or 160). If blank, a printer size of 132 is assumed.
33-34		Not used; leave blank.
35-36	bΔ	Tape mark generation at normal termination: For work and output tapes (columns 6-7 = OT). b = 1 write double tape mark. = blank do not write tape mark. For all other tapes, leave blank.

Device Parameter (Cont'd)

Card Column	Entry	Meaning
37-53 (See Notes 2 and 3)	Same as cols. 16-32.	Same as columns 16-32.
54-55		Not used; leave blank.
56	e	Tape mark generation at abnormal termination: For output tapes (columns 6-7 = OT). e = 1 do not write tape mark. = blank write double tape mark. For work tapes (columns 6-7 = WT). e = 1 write double tape mark. = blank do not write tape mark. For all other tapes, leave blank.
57-60	Rnnn	nnn = size of records for fixed-length, blocked records (001-999) on a tape to be printed. Leave blank for variable-length or unblocked records.

Notes:

1. Columns 16-32 contain tape printing information for normal termination.
2. Columns 37-53 contain tape printing information for abnormal termination. If abnormal printing is to be the same as normal printing, place an S in column 37.
3. If this field is blank, no printing occurs. Items can appear in any order, but the first item must begin in column 16 (or 37). Commas must separate each item.

Examples:

```
ΔDEVΔITΔSOURCE
ΔDEVΔOTΔMASTERΔFG,O1,C00100Δ...Δ1ΔO0Δ....ΔR080
ΔDEVΔTDΔINPUT1ΔFH,O3,C01000Δ....ΔS
ΔDEVΔWTΔSTORESΔ....ΔFG,O0Δ..Δ1
```

Device Parameter (Cont'd)

Format (Card Reader, Punch, or Printer):

ΔDEVΔ aaΔdddddd

Card Column	Entry	Meaning
1-5	ΔDEVΔ	Parameter identifier.
6-8	aaΔ	Device type: aa = CR card reader. = PU card punch. = PR printer. = PT paper tape reader. = PP paper tape punch.
9-14	dddddd	Symbolic name used by test program for device.

Format (Random Access):

ΔDEVΔ aaΔ ttttbbΔdddddd, . . . , dddddd

Card Column	Entry	Meaning
1-5	ΔDEVΔ	Parameter identifier.
6-8	aaΔ	Device usage: aa = RD test program input device; AIDS is to allocate a file to this device and generate test data for it. = RA test program work or output device; AIDS is to allocate a file on this device. = IR test program input device which contains test program files.
9-12	tttt	Device type: tttt = DISK disc storage unit. DRUM drum storage unit. MASS mass storage unit.
13-15	bbΔ	Bin number (00-07) for mass storage. Blank if disc or drum.
16 . . .	dddddd	Symbolic name used to reference this device (one to six characters). If more than one name is used for this device, each name should be given and followed by a comma, except the last.

Examples:

Δ DEVΔ RD ΔMASS01
Δ DEVΔ IRΔ DISKΔ Δ ΔSYS001, SYSIN
Δ DEVΔ RAΔ DRUM

VOL Parameter

◆ *Format:*

ΔVOLΔfilenameΔssssssΔnad

Card Column	Entry	Meaning
1-5	ΔVOLΔ	Parameter identifier.
6-50	filenameΔ	File identification (1 to 44 characters). This name is placed in the Format 1 label of the VTOC when AIDS allocates the file.
51-57	ssssssΔ	Volume serial number for an input random access file volume. Leave blank for work or output files.
58	n	Normal termination edit format code: n = H hexadecimal. = G graphic. = C hexadecimal with graphic equivalents. = blank do not edit.
59	a	Abnormal termination edit format code: a = H hexadecimal. = G graphic. = C hexadecimal with graphic equivalents. = blank do not edit.
60	d	Disposition code: d = S save file for use by subsequent test program. = blank deallocate and purge file. Not applicable for user supplied files.

Examples:

ΔVOLΔTDOSΔAIDSΔTESTΔ...ΔCC
ΔVOLΔINPUTΔNOΔ1000555ΔCCS
ΔVOLΔFILEΔTHREE

LIMIT Parameter

◆ *Format:*

Δ LIMITΔnnnn,....,nnnn

Card Column	Entry	Meaning
1-7	ΔLIMITΔ	Parameter identifier.
8...	nnnn	Four-character decimal number of cylinders to be allocated for this extent. Up to nine extents may be specified for a file.

Examples:

ΔLIMITΔ0185
ΔLIMITΔ0020,0020,0040

RTP Parameter

◆ *Format:*
ΔRTP

Snapshot Parameter

◆ The Snapshot parameter requests a listing of portions of memory, the general purpose registers, and the floating-point registers. The programmer can specify an instruction and the number of times that this instruction is to be executed before the snapshot is taken. The snapshot is reapplied each time the segment named in the parameter is loaded until the total number of snapshots desired is obtained.

This parameter (along with the run-time parameters) should immediately follow the PROG parameter.

Format:

ΔSNAPSΔppppppΔssssssallllllΔrrrrrrΔfgΔxxxxxxΔn,s,t

Card Column	Entry	Meaning
1-7	ΔSNAPSΔ	Parameter identifier.
8-14	ppppppΔ	Name of test program.
15-20	ssssss	Name of segment containing area to be printed. If blank, the root segment is assumed.
21	a	a = P left- and right-hand addresses are program-relative. = blank left- and right-hand addresses are segment-relative.
22-28	llllllΔ	Left-hand end of memory area to be printed (program or segment-relative hexadecimal address).
29-35	rrrrrrΔ	Right-hand end of memory area to be printed (program or segment-relative hexadecimal address).
36-38	fgΔ	Format of output listings: f = H hexadecimal. = G EBCDIC graphics. = C hexadecimal with graphic equivalents. g = 1 one byte per print group. = 2 two bytes per print group. = 4 four bytes per print group.
39-45	xxxxxxΔ	Address of instruction to be used as test point (segment-relative hexadecimal).
46 . . .	n	Number of times test point is to be executed before first snapshot (0-99999).
	,s	Number of times test point is to be executed between additional prints (0-99999).
	,t	Total number of prints to be taken (0-99999).

Snapshot Parameter (Cont'd)

Note:

More than one snapshot can be taken using the same test point by punching different area and format information in columns 22 to 38 for each additional area to be printed.

Examples:

```
ΔSNAPSΔPAYROLΔSEGMT6Δ003100Δ003FAOΔC4Δ000100Δ1,1,2
ΔSNAPSΔSDUPΔΔΔΔΔΔΔΔΔP000050Δ0005EAΔH4Δ000050Δ50,10,5
```

TRACE Parameter

◆ This parameter provides a diagnostic listing of an instruction and its associated registers after the instruction has been executed. Every instruction in a program may be listed or a trace made only of a selected portion of the program. The programmer can specify an instruction in the program as a test point to be executed a certain number of times before the trace is made. The trace is reapplied each time the segment named in the parameter is loaded until the total number of traces required is obtained.

The use of the Trace parameter should be kept to a minimum.

Format:

```
ΔTRACEΔppppppΔssssssallllllΔrrrrrrΔxxxxxxΔn,s,t
```

Card Column	Entry	Meaning
1-7	ΔTRACEΔ	Parameter identifier.
8-14	ppppppΔ	Name of test program.
15-20	ssssss	Name of segment containing area to be traced. If blank, the root segment is assumed.
21	a	a = P left- and right-hand addresses are program-relative. = blank left- and right-hand addresses are segment-relative.
22-28	llllllΔ	Address of first instruction to be traced (program or segment-relative hexadecimal).
29-35	rrrrrrΔ	Address of last instruction to be traced (program or segment-relative hexadecimal).
36-42	xxxxxxΔ	Address of instruction to be used as test point (segment-relative hexadecimal).
43	n	Number of times the test point is to be executed before the area is traced the first time (0-99999).
	,s	Number of times the test point is to be executed between any additional traces (0-99999).
	,t	Total number of traces to be made (0-99999).

TRACE Parameter (Cont'd)

Note:

A complete trace can be requested by submitting the following parameter:

```
ΔTRACEΔpppppp
```

Examples:

```
ΔTRACEΔINVEN
ΔTRACEΔINVENΔΔ CORDERΔ 00A1F0Δ 00A510 Δ00A200 Δ0,0,1
ΔTRACEΔAIROPTΔSEG2Δ Δ Δ001000Δ 00100EΔ00750A Δ100,100,3
```

PATCH Parameter

◆ Two forms of the Patch parameter are available. One causes a branch to a patch and the other replaces the original data used by the program with new data.

The Add patch feature causes the test program to branch to instructions and constants stored by AIDS. These patches may be applied anywhere in the program and applied immediately or stored for future use. Registers to be used for referencing instructions and constants within the patch may be specified. There is no limit, other than storage area needed by AIDS, to the number or size of the patches to be added.

The Exchange patch feature replaces data in the test program. The data in the patch can be graphic or hexadecimal and replaces the program data on a byte-for-byte basis. No additional general purpose registers can be used with the Exchange patch.

Patches are reapplied each time the segment named in the parameter is loaded.

Format (Exchange Patch):

```
ΔPATCHppppppssssssEellllll Δ Δ Δxx...xxss
```

Card Column	Entry	Meaning
1-6	ΔPATCH	Parameter identifier.
7-12	pppppp	Name of test program.
13-18	ssssss	Name of segment to be patched. If blank, patch is applied to the root segment.

PATCH Parameter (Cont'd)

Card Column	Entry	Meaning
19-20	Ee	e = G patch information is graphic. = H patch information is hexadecimal.
21-26	llllll	Address of left-hand end of area to receive patch (program-relative hexadecimal.)
27-29		Not used; leave blank.
30-78	xx...xx	Patch information. Graphic: Up to 48 characters plus a termination indicator. If the field contains the end of the patch information, a logical NOT (11,8,7 punch) must immediately follow the last character. If the field is not the last of the information, column 78 must be blank. Hexadecimal: Up to 48 characters. Any commas used are ignored. Column 78 must be left blank whether there are additional characters or not.
78-80	ss	Sequence number (01-99) of patch card when more than one card contains information for the same patch. If blank, AIDS assumes all information is on one card.

Note:

Both graphic and hexadecimal patch cards can be used for the same patch. All cards that apply to the same patch must have the same information in columns 1 to 18 and 21 to 26.

Example:

∧PATCHTPPARM∧... ∧EH00070A∧∧∧FFF04A21,03,A47E

PATCH Parameter (Cont'd)

Format - (Add Patch):

ΔPATCHppppppssssssAallllllicΔxx...xxss

Card Column	Entry	Meaning
1-6	ΔPATCH	Parameter identifier.
7-12	pppppp	Name of test program.
13-18	ssssss	Name of segment to be patched. If blank, the patch is applied to the root segment.
19-20	Aa	a = I patch information is instructions, = G patch information is graphic constants. = H patch information is hexadecimal constants.
21-26	llllll	Address of last instruction to be executed before patch (segment-relative hexadecimal).
27	i	Number of general purpose register to be used with added instructions (0-F). If not needed, leave blank. (See notes 2 and 4.)
28-29	cΔ	Number of general purpose register to be used with added constants (0-F). If not needed, leave blank.´(See notes 3 and 4.)
30-78	xx...xx	Patch information. Graphic Constants: Up to 48 characters plus a termination indicator. If the field contains the end of the graphic constants, a logical NOT (11,8,7 punch) must immediately follow the last character. If the field is not the last of the constants, column 78 must be blank. Instructions and Hexadecimal Constants: Up to 48 constants and instructions punched in hexadecimal. Commas may be used in instructions for convenience but they are ignored. Column 78 must be left blank whether or not there are additional constants or instructions.
79-80	ss	Sequence number (01-99) of patch card when more than one card contains information for the same patch. If blank, AIDS assumes all information is on one card.

PATCH Parameter (Cont'd)

Notes:

1. Both graphic and hexadecimal patch cards can be used for the same patch. All cards that apply to the same patch must have the same information in columns 1 to 18 and 21 to 29. When both instructions and constants are added, instructions must be entered first. Constants may not be added without preceding instructions.
2. The contents of i are stored and the address of the left-hand end of the added instructions is placed in i when the patch is made.
3. The contents of c are stored and the address of the left-hand end of the added constants is placed in c when the patch is made.
4. The contents of the registers specified by i and c before the patch was applied are not restored until all of the added instructions have been executed.

Example:

```
ΔPATCHDESU01ΔΔΔΔΔΔG040EB6Δ3Δ00$CONEND*
```

* = 11,8,7 punch

END Program Parameter

◆ This parameter indicates the end of test parameters for the current program. The programmer can specify that a memory dump is to be taken upon normal termination and its format. The format of the abnormal termination memory dump also can be specified.

Format:

```
ΔENDΔPROGΔfgΔmn
```

Card Column	Entry	Meaning
1-10	ΔENDΔPROGΔ	Parameter identifier.
11-13	fgΔ	Format of memory print upon normal termination: f = H hexadecimal. = G EBCDIC graphics. = C hexadecimal with graphic equivalents. = F floating-point. = M mnemonic. g = 1 one byte per print group. = 2 two bytes per print group. = 4 four bytes per print group. If no memory print is desired, leave blank.
14-15	mn	Format of memory print upon abnormal termination. Same as f and g. If blank, AIDS prints memory in full-word hexadecimal with graphic equivalents.

END AIDS Parameter

◆ This parameter indicates the end of the AIDS job stream.

Format:

ΔENDΔAIDS

Routine Parameters For Console-Controlled Testing - Detailed

◆ Parameters used for console testing are divided into two groups: immediate and latent. Immediate functions are entered from the console typewriter and executed as soon as they are entered, except for Address Stop. Latent functions are stored by AIDS and executed only when certain conditions have been satisfied. Two latent functions, Trace and Snapshot, can be entered from the typewriter, card reader, or magnetic tape; the third, Patch, can only be entered from the card reader or magnetic tape.

Continue Parameter

◆ This parameter indicates that control is to be returned to the test program at the last point of interruption or to a specific address. The programmer is given control again with the next load of a segment of the test program.

Format:

pΔ@CONΔaΔxxxxxx

Type Position	Entry	Meaning
1-2	pΔ	Program number of test program (1-6).
3-7	@CONΔ	Parameter identifier.
8-15	aΔxxxxxx	a = A absolute address. = P program-relative address. xxxxxx = six-character hexadecimal address of instruction where control is to be transferred. If blank, control is transferred to the last point of interrupt (address contained in P counter).

Proceed Parameter

◆ This parameter indicates that control is to be returned to the test program at the last point of interruption or to a specific address. Format 1 returns control to the last point of interrupt; format 2 returns control to a specific address. The programmer does not regain control again unless an Address Stop parameter has been previously entered.

Proceed Parameter (Cont'd)

Format 1:

p@PROΔS

Type Position	Entry	Meaning
1-2	pΔ	Program number of test program (1-6).
3-7	@PROΔ	Parameter identifier.
8	S	Display each test program segment on the console typewriter as it is loaded. If blank, segment loads are not displayed.

Format 2:

pΔ@PROΔaΔxxxxxxΔS

Type Position	Entry	Meaning
1-2	pΔ	Program number of test program (1-6).
3-7	@PROΔ	Parameter identifier.
8-16	aΔxxxxxxΔ	a = A absolute address. = P program-relative address. xxxxxx = six-character hexadecimal address of instruction where control is to be transferred.
17	S	Display each test program segment on the console typewriter as it is loaded. If blank, segment loads are not displayed.

Open Diagnostic Device Parameter

♦ An output diagnostic tape is opened using this parameter. A standard header label is written and the tape is used for memory prints, snapshots, and trace outputs.

Format:

pΔ@OPD

Type Position	Entry	Meaning
1-2	pΔ	Program number of test program (1-6).
3-6	@OPD	Parameter identifier.

Close Diagnostic Device Parameter

◆ This parameter writes a double tape mark and deallocates the magnetic tape assigned by the Open Diagnostic Device parameter. The output tape is rewound.

Format:

pΔ@CLD

Type Position	Entry	Meaning
1-2	pΔ	Program number of test program (1-6).
3-6	@CLD	Parameter identifier.

Read Device Parameter

◆ The Read Device parameter causes AIDS to read parameters and/or associated data from the card reader or magnetic tape. The input is read until an END parameter is recognized.

Format:

pΔ@RDVΔdd

Type Position	Entry	Meaning
1-2	pΔ	Program number of test program (1-6).
3-7	@RDVΔ	Parameter identifier.

Memory Print Parameter

◆ This parameter causes a listing of selected portions of memory, the general registers, the floating-point registers, and the program status registers. This listing may be directed to the printer or to magnetic tape. If a tape device is used, it must have been assigned by an Open Diagnostic Device parameter.

Printer output is full-word hexadecimal with graphic equivalents, 48 bytes to the line. Duplicate lines are suppressed. Tape output is 133-character, unbatched records.

Format:

pΔ@DMPΔaΔllllllΔrrrrrrΔfΔn

Memory Print Parameter (Cont'd)

Type Position	Entry	Meaning
1-2	pΔ	Program number of test program (1-6).
3-7	@DMPΔ	Parameter identifier.
8-9	aΔ	a = A absolute addresses. = P program-relative addresses.
10-16	llllllΔ	Address of left-hand end of memory area to be printed (hexadecimal).
17-23	rrrrrrΔ	Address of right-hand end of memory area to be printed (hexadecimal).
24-25	fΛ	Format in which memory is to be printed: f = M mnemonic. = S short-precision floating-point. = F long-precision floating-point. = H hexadecimal. = G EBCDIC graphics. = C hexadecimal with graphic equivalents.
26	n	Printer grouping factor: n = 1 one byte per print group. = 2 two bytes per print group. = 4 four bytes per print group. Not used when f is equal to M, S, or F.

Note:

To print registers and all of memory assigned to the program under test, enter pΛ@DMP only.

Display Memory Parameter

◆ This parameter can be used to display a memory area up to 99 bytes to the console typewriter. Twenty-four bytes are displayed per line with each line preceded by the hexadecimal address of the leftmost character in the line.

Format:

pΛ@DMYΛaΛllllllΛnn

Type Position	Entry	Meaning
1-2	pΔ	Program number of test program (1-6).
3-7	@DMYΔ	Parameter identifier.
8-9	aΔ	a = A absolute address. = P program-relative address.
10-16	llllllΔ	Address of left-hand end of memory to be displayed (hexadecimal).
17-18	nn	Number of bytes to be displayed (1-99).

Change Memory Parameter

◆ This parameter allows limited changes to be made to memory during program testing. The changes are made on a byte-for-byte basis. The area to be changed is specified by giving an absolute or program-relative address of its left-hand end. A maximum of 52 bytes can be changed with each Change Memory parameter.

Format:

pΔ@CMYΔaΔllllllΔdΔxx...xx

Type Position	Entry	Meaning
1-2	pΔ	Program number of test program (1-6).
3-7	@CMYΔ	Parameter identifier.
8-9	aΔ	a = A absolute address. = P program-relative address.
10-16	llllllΔ	Address of left-hand end of memory to be changed (hexadecimal).
17-18	dΔ	d = H hexadecimal data. = G graphic data.
19-70	xx...xx	Hexadecimal or graphic data to be inserted. The last character of graphic data cannot be a space.

Display Registers Parameter

◆ One or more state 1 general purpose registers, the floating-point registers, or the program status registers are displayed on the typewriter by using this parameter. The general purpose registers are displayed in full-word hexadecimal, preceded by the register number (0-F); program status registers are printed in full-word hexadecimal, preceded by an identification tag.

Format:

pΔ@DRGΔrΔf

Type Position	Entry	Meaning
1-2	pΔ	Program number of test program (1-6).
3-7	@DRGΔ	Parameter identifier.
8-9	rΔ	Registers to be displayed: r = 0-F general purpose register. = G floating-point registers. = P program counter and IMR.
10	f	f = final general purpose register to be displayed (registers r through f will be displayed). If blank, only the register specified by r is displayed.

Change Registers Parameters

◆ This parameter allows the programmer to change the contents of the general purpose and the floating-point registers. The same data can be placed in more than one general purpose register by one parameter. A series of parameters is used to load the general purpose registers with different information.

Format (Change General Purpose Register):

pΔ@CRGΔrΔhhhhhhhh

Type Position	Entry	Meaning
1-2	pΔ	Program number of test program (1-6).
3-7	@CRGΔ	Parameter identifier.
8-9	rΔ	General purpose register to be changed (0-F).
10-17	hhhhhhhh	Data to be inserted into register (full-word hexadecimal).

Format (Change More than One General Purpose Register):

pΔ@CRGΔrΔfΔhhhhhhhh

Type Position	Entry	Meaning
1-2	pΔ	Program number of test program (1-6).
3-7	@CRGΔ	Parameter identifier.
8-9	rΔ	First general purpose register to be changed (0-E).
10-11	fΔ	Last general purpose register to be changed (1-F).
12-19	hhhhhhhh	Data to be inserted into each register (full-word hexadecimal).

Change Register Parameters (Cont'd)

Format (Change Floating-Point Register):

pΔ@CRGΔrs.hhh...hhhΔEsee

Type Position	Entry	Meaning
1-2	pΔ	Program number of test program (1-6).
3-7	@CRGΔ	Parameter identifier.
8	r	Floating-point register to be changed (0, 2, 4, or 6).
9	s	Sign of number to be inserted in register (+ or -).
10	.	Decimal point.
11-25	hhh...hhhΔ	Hexadecimal value to be inserted in register (zero-filled).

26-29	Esee	s = sign of exponent (+ or -).
		ee = exponent ($00_{(16)}$ $-65_{(16)}$).
		Value can range from $-65_{(16)}$ to $+64_{(16)}$.

Example:

5Δ@CRGΔ2+.000E074B604A00ΔE-08

Address Stop Parameter

◆ The Address Stop parameter specifies a test point (instruction address) in the test program where control is to be returned to the programmer. This parameter can cause the point to be executed a certain number of times before control is given to the programmer.

Up to three address stops can be stored by AIDS at one time. Address Stop parameters are not reapplied and must be entered each time they are to be used.

Format:

pΔ@STPΔ aΔxxxxxxΔ sΔnnnn

Address Stop Parameter (Cont'd)

Type Position	Entry	Meaning
1-2	pΔ	Program number of test program (1-6).
3-7	@STPΔ	Parameter identifier.
8-9	aΔ	a = A absolute address. = P program-relative address.
10-16	xxxxxxΔ	Address of instruction to be used as test point (hexadecimal).
17-18	sΔ	Address Stop identifier (1, 2, or 3). Identifies the test point as one of three which can be in AIDS at the same time. When a parameter is entered with the same identifier as an existing parameter, it replaces the existing parameter.
19-22	nnnn	Number of times the test point is to be executed before control is returned to the programmer (0-9999).

Snapshot Parameter

◆ This parameter provides a listing of specified portions of memory, the general purpose registers, and the floating-point registers. The programmer can specify an instruction address and the number of times the instruction is to be executed before the snapshot is taken. The Snapshot parameter is entered from the console typewriter, card reader, or magnetic tape device.

When the typewriter is used, the output is in full-word hexadecjmal with graphic equivalents. The snap is taken as soon as the test point is satisfied.

When the card reader or magnetic tape is used, the programmer can select the format and grouping of the output. He can also specify additional snapshots after the first.

Format - Card:

Same as described for automatic testing; see page 4-24.

Snapshot Parameter (Cont'd)

Format - Typewriter:

pΔ@SNPΔaΔllllllΔrrrrrrΔxxxxxxΔnnnn

Type Position	Entry	Meaning
1-2	pΔ	Program number of test program (1-6).
3-7	@SNPΔ	Parameter identifier.
8-9	aΔ	a = A absolute addresses. = P program-relative addresses.
10-16	llllllΔ	Address of left-hand end of memory area to be printed (hexadecimal).
17-23	rrrrrrΔ	Address of right-hand end of memory area to be printed (hexadecimal).
24-30	xxxxxxΔ	Address of instruction to be used as a test point (hexadecimal).
31-34	nnnn	Number of times the test point is to be executed before the area is printed (1-9999). If blank, 0 is assumed.

Trace Parameter

◆ This parameter provides a diagnostic listing of an instruction and its associated registers after the instruction has been executed. Every instruction in a program may be listed or a trace made only of a selected portion of the program. The programmer may specify a test point (instruction address) in the program that is to be executed a certain number of times before the trace is made.

The use of this parameter should be kept to a minimum and only when other AIDS functions cannot solve the problem.

This parameter can be entered from the console typewriter, card reader, or magnetic tape device. When a card reader or magnetic tape is used, additional traces after the first one can be specified.

Format - Card:

Same as described for automatic testing; see page 4-25.

Trace Parameter (Cont'd)

Format - Typewriter:

pΔ@TRCΔaΔllllllΔrrrrrrΔxxxxxxΔnnnn

Type Position	Entry	Meaning
1-2	pΔ	Program number of test program (1-6).
3-7	@TRCΔ	Parameter identifier.
8-9	aΔ	a = A absolute addresses. = P program-relative addresses.
10-16	llllllΔ	Address of first instruction to be traced (hexadecimal).
17-23	rrrrrrΔ	Address of last instruction to be traced (hexadecimal).
24-30	xxxxxxΔ	Address of instruction to be used as a test point (hexadecimal).
31-34	nnnn	Number of times the test point is to be executed before the trace is made (1-9999). If blank, 0 is assumed.

Note:

A complete trace is made by submitting only pΔ@TRC in the parameter.

Patch Parameter

◆ The Patch parameter functions and format are identical to those of the automatic testing Patch parameter described on page 4-26.

Write Tape Mark Parameter

◆ This parameter causes a single or double tape mark to be written to a designated magnetic tape used by the test program. The tape may be rewound to BOT if desired.

Format:

pΔ@WTMΔddddddΔtΔr

Type Position	Entry	Meaning
1-2	pΔ	Program number of test program (1-6).
3-7	@WTMΔ	Parameter identifier.
8-14	ddddddΔ	Symbolic name used by test program for device.
15-16	tΔ	t = 0 do not write tape mark. = 1 write single tape mark. = 2 write double tape mark.
17	r	r = 1 rewind to BOT. = 2 do not rewind. = 3 rewind to tape mark. When t = 0, the tape is rewound to BOT and the r entry is ignored.

END Parameter

◆ Two End parameters are used. One is entered from the card reader or magnetic tape, the other from the typewriter.

Card or tape parameters are used to signify the end of Snapshot, Trace, and Patch input. An End card must follow each parameter or each set of parameters to be read following a Read Device statement.

An End parameter entered from the typewriter signifies that: (1) AIDS is to be terminated and control returned to the program under test, (2) AIDS and the program under test are to be terminated, or (3) the program under test is to be terminated and AIDS used to test another program.

Format - Card:

```
Δ@END
```

Format - Typewriter:

```
pΔ@ENDΔz
```

Type Position	Entry	Meaning
1-2	pΔ	Program number of test program.
3-7	@ENDΔ	Parameter identifier.
8	z	z = A terminate AIDS and return control to the test program. = B terminate AIDS and test program. = C terminate test program and continue AIDS routine.

Considerations for Use

General

◆ AIDS offers two methods of testing a program: automatic and console-controlled. The programmer, therefore, must select the method best suited to meet his needs and that will utilize both his and the processor's time effectively.

Automatic Testing

◆ In an automatic AIDS session, a minimum of operator intervention is required because all user data concerned with the test is on cards, tape or random access devices. The operator loads AIDS using the Executive and responds to AIDS messages to start the testing. The testing proceeds automatically until completed or an abnormal termination occurs. At this time, another program may be tested or the same program tested with different data.

Automatic Testing (Cont'd)

The considerations for using automatic AIDS testing are discussed below.

Device Assignments

The programmer must supply a Device parameter for each magnetic tape and random access device used by the test program. He must also supply a Device parameter for card readers, card punches, paper tape readers and punches, and printers. If a program uses more than one reader, punch, or printer, only one parameter is needed for each device type. The Executive will request assignment of any additional readers, punches, or printers.

Assignments are normally made by the Executive for the first program to be tested in a session. AIDS keeps these devices in a pool and automatically uses them for additional programs in the testing stream. Exceptions to this are that the assignment of an input device is always requested, and work and output device assignments are requested if they are not available in the device pool. Any new devices assigned are added to this pool.

If a device card is not supplied for a device used by the test program, the Executive asks for its assignment. When the program under test terminates, the device is deallocated and is not added to the AIDS device pool.

Test Data

Test data is available for program testing from the following sources:

1. User-supplied data recorded on cards, tape, or random access devices.
2. The TDOS Test Data Generator routine (part of AIDS) may be used to generate test data to a tape or random access device that is used as input to the test program. The data is prepared according to Test Data Generator parameters (refer to page 4-1, Test Data Generator routine). The parameters must follow the Device card that defines the tape or random access device that is to receive the test data.

Regardless of the source, all data must be labeled as required by the program under test. Different sets of test data may be used to test the same program by repeating the Program ID parameter in the AIDS input and using new test data.

Tape Printing

AIDS does not save work or output tapes. If the programmer wants a record of any of these tapes he must indicate in the Device card associated with the tape that it is to be printed. The tapes will be printed, or written to an output tape, depending on the assignment of AIDOPT. AIDS can print both normal and preedited tapes. Partial printing of tapes is possible by using the count option in the Device parameter.

Automatic Testing (Cont'd)

Random Access Files

Three types of random access files can be specified by the programmer. They are (1) work and output files, (2) input files to which the AIDS Test Data Generator is to generate test data, and (3) input files containing user data.

Work and output files are automatically allocated by AIDS. A Device card, VOL card, and Limit card must be supplied for each work or output file. The device must have been initialized by the Random Access Volume Initializer. If a file is to be used by a subsequent program, the Save option in the VOL card must be used. The last program in the job stream to use a particular random access file must not use the Save option. This is to insure that all files allocated by AIDS are deallocated and purged before the AIDS job stream is terminated.

The same considerations apply to AIDS Test Data Generator files as to work and output files. In addition, the test data generator parameters must follow the Device, VOL, and Limit cards.

When user input files are required, only a Device card and a VOL card are required.

Random Access File Printing

AIDS does not save work, output, or Test Data Generator input files. If the programmer wants an edit of these files, he must indicate this in the AIDS VOL parameter associated with the file. The file will be printed, or written to tape, depending on the assignment of AIDOPT.

Run-Time Parameters

When a test program requires run-time parameters (RTP), they must be supplied as part of the AIDS job stream. The RTP's are preceded by the AIDS parameter RTP. The following run-time parameters are valid input: // FILES, // VOL, // TPLAB, // VDC, // END.

The format of these parameters is the same as described in the TDOS Operators' Guide, except for the // VDC card. The volume serial number in the // VDC parameter for a file allocated by AIDS must be all zeros. For user supplied files the // VDC serial number must be the same as the Volume serial number.

Memory Dumps

A memory dump is always taken by AIDS upon an abnormal termination. The programmer can also specify that a memory dump be taken upon normal termination.

The memory dump routine is part of AIDS and cannot be called in under Executive control. The dump is made to AIDOPT, which may be assigned to the printer or magnetic tape. When an abnormal memory dump is taken, it is preceded by a description of the type of error causing the dump and the contents of the P1 counter at that time.

Automatic Testing (Cont'd)

AIDS Output

AIDS output is made to the printer or a magnetic tape depending on AIDOPT assignment when AIDS was initiated. When both AIDS output and the test program output are assigned to the same printer they will share it. The information and data contained in the AIDS output is listed below.

1. A listing of all AIDS parameters submitted and reasons for rejection if any were invalid.
2. Results of snapshots or traces that were requested.
3. The location and type of any interrupts that occurred.
4. All typewriter messages and replies.
5. All memory dumps.
6. Edited contents of all or parts of magnetic tapes and random access devices selected for printing.

Console-Controlled Testing

◆ After AIDS has been loaded, the message AUTO OR CONSOLE? is typed. The programmer then supplies the name of the program to be tested and the symbolic names used by the program for the card reader and/or printer if these devices are to be shared with AIDS.*

If the program to be tested is in memory, AIDS types the message AIDS REQUEST REQUIRED; the programmer then starts his program testing. If the program is not in memory, AIDS idles until the test program is loaded and device assignments are made in the normal TOS Executive manner before AIDS input is requested. (If the program to be tested was loaded before AIDS, the Executive Change Priority routine must be used to give AIDS a higher priority than that of the program to be tested.)

The programmer begins testing by submitting any of the console-controlled testing parameters. The operation given in the parameter is executed and control is returned to the programmer, with the exception of the Address Stop and the latent parameters. If one of the latter two parameters is submitted or no input is desired at this time, control must be given to the program under test. Control is given to the program only by a Continue or Proceed parameter or a reply after an unsolicited interrupt. The programmer regains control when (1) a segment is loaded after a Continue parameter had been entered, (2) an Address Stop is satisfied, (3) a test program generated interrupt occurs, (4) an End of Job SVC is executed by the program under test, or (5) an unsolicited interrupt is typed in. These conditions of program and user control are described below.

*The format for the reply to AUTO OR CONSOLE? message appears in the Operators' Guide.

Console-Controlled Testing (Cont'd)

Continue

When this parameter is entered, control is given to the program at (1) the address specified, (2) the start address of the program if it was just loaded, or (3) the address in the LPOV statement if a segment of the program was just loaded. The program then executes until another segment is loaded or an Address Stop is completed, at which time control is returned to the programmer.

Proceed

The Proceed parameter gives control to the program under test in the same manner as Continue except that control is returned only after an Address Stop has been completed.

Unsolicited Interrupt and Reply

The Programmer can regain control at any time by entering an Executive or test program interrupt (EΔHLTΔn). AIDS then allows the programmer to type in a message to the program's unsolicited type-in logic or submit additional testing parameters. Control is returned to the program under test or AIDS depending on the type-in.

Program Segment Loaded

When a segment of the program is loaded after a Continue parameter has been entered. a message is typed requesting input and giving the name of the segment just loaded. Any of the AIDS parameters can then be entered.

Address Stop

After the instruction specified as a test point in an Address Stop statement has been executed the required number of times, AIDS returns control to the programmer. AIDS types a message containing the address of the test point; the programmer can then request any of the AIDS functions.

Test Program Generated Interrupt

When the test program generates a program check. interval timer, or unrecoverable error interrupt, AIDS determines if the test program contains a contingency routine (STXIT macro) to handle the interrupt. If it does, AIDS returns control to the program. If it does not, AIDS types out the type of interrupt and where it occurred, the programmer can then enter any of the AIDS parameters.

End of Job SVC

The occurrence of an EOJ SVC in the program causes AIDS to return control to the programmer. AIDS types a message requesting input and giving the address of the supervisor call. Any of the AIDS functions can be entered at this time. If a Continue or Proceed parameter is entered. an address must be specified.

Restrictions

◆ 1. Test point addresses specified for Address Stop, Snapshot, and Trace parameters must be the leftmost byte of the instruction.

2. If the test program modifies a location used as an AIDS test point, the function associated with the test point will not be executed.

3. A test point address within a program segment can not be used for more than one AIDS function. Identical test points can be used if they are in different segments.

4. Snapshots, address stops, and add patches can not be specified within an area to be traced.

5. The location specified by the address in an Add Patch parameter can not contain a test program SVC.

Device Assignments

◆ Console-Controlled Testing

SDN	Device Type	Remarks
AIDRDV	Card Reader or magnetic tape.	Input device for Snapshot, Trace, and Patch parameters.
AIDOPT	Printer or magnetic tape.	Output device for memory print and snapshot and trace results.

Automatic Testing

SDN	Device Type	Remarks
AIDIPT	Card reader or magnetic tape.	Parameter input device.
AIDOPT	Printer or magnetic tape.	Output device.

Parameter Examples

Automatic Testing

◆ AIDIPT (Card Reader or Magnetic Tape):

Card	Parameter
A1	ΔPROGΔTPPR
A2	ΔDEVΔTDΔPRIPT1ΔFG,O1,C00100Δ....ΔΔΔFH,O0
A3	Test Data Generator parameters.
A4	ΔEND
A5	ΔDEVΔCRΔPRPRM
A6	ΔDEVΔPRΔPRLST
A7	Test parameters SNAPS, TRACE, PATCH.
A8	ΔENDΔPROG
B1	ΔPROGΔOMALL8
B2	ΔDEVΔCRΔSYSIPT
B3	ΔDEVΔWTΔWORK1ΔΔFG,O0Δ...ΔΔΔSΔ...Δ1
B4	ΔDEVΔRAΔMASS01
B5	ΔVOLΔMASSΔFILEΔ...ΔCCΔ
B6	ΔLIMITΔ0005
B7	ΔRTP
B8	//ΔVDCΔSTATS,,MASSΔFILE,000000
B9	//ΔEND
B10	Test parameters - SNAPS, TRACE, PATCH.
B11	ΔENDΔPROGΔΔΔΔH4
C1	ΔPROGΔY
C2	ΔDEVΔIRΔDISKΔΔΔDISKIN
C3	ΔVOLΔTDOSΔUSERΔDATAΔ...Δ000777
C4	ΔDEVΔRAΔDISK
C5	ΔVOLΔTDOSΔOUTPUTΔ...ΔCCΔ
C6	ΔLIMITΔ0010,0010
C7	ΔRTP
C8	//ΔVDCΔFILEA,,TDOSΔUSERΔDATA,000777
C9	//ΔVDCΔFILEB,,TDOSΔOUTPUT,000000
C10	//ΔEND
C11	Test parameters - SNAPS, TRACE, PATCH.
C12	ΔENDΔPROGΔG4ΔG4
C13	ΔENDΔAIDS

Automatic Testing (Cont'd)

The series of cards shown on the previous page would automatically test three programs named TPPR, OMALL8, and Y. An explanation of the function of each card follows:

Card A1 - Defines the first program to be tested as TPPR.

Card A2 - Informs AIDS that test data is to be generated on an input tape called PRIPT1. If a normal termination occurs, print the first 100 records on PRIPT1 in EBCDIC graphics. For an abnormal termination print the complete tape in hexadecimal.

Card A3 - Test Data Generator parameter cards for data to be generated onto PRIPT1.

Card A4 - The END card for the Test Data Generator parameters.

Card A5 - Informs AIDS that PRPRM, an input device for TPPR, is a card reader. Since AIDIPT is a card reader. AIDS will assume that the PRPRM device is the same card reader and no device assignment for PRPRM will be requested. If AIDIPT were magnetic tape, AIDS would request assignment of PRPRM.

When AIDS and program TPPR share the card reader, AIDS informs the operator to remove the AIDS control cards and insert the card input for the test program. When program TPPR testing has been completed, AIDS informs the operator to replace the remaining AIDS control cards.

Card A6 - Informs AIDS that PRLST (the TPPR output device) is the printer. If AIDOPT is also the printer, AIDS and TPPR will share it and PRLST will be assigned automatically.

Card A7 - The AIDS test parameters would appear at this point in the input.

Card A8 - Signifies that there are no more AIDS input cards for this program. Because the additional fields of the card are blank, no memory dump will be taken after a normal termination but a memory dump will be made upon abnormal termination.

Card B1 - Informs AIDS that OMALL8 is to be tested.

Card B2 - Informs AIDS that the test program is to use the card reader. The symbolic name for the reader is SYSIPT.

Card B3 - Device card for a work tape named WORK1. The entire tape will be printed upon either a normal or abnormal termination.

Automatic Testing (Cont'd)

Card B4 - Device card designating that magazine 1 of a mass storage unit is to be allocated by AIDS and used for a work or output file by the test program.

Card B5 - Contains the name to be given to the file and specifies that it is to be printed in hexadecimal with graphic equivalents upon a normal or abnormal termination. The file is to be deallocated and purged when the test program terminates.

Card B6 - Informs AIDS that it is to allocate one 5-cylinder extent for this file.

Card B7 - Indicates that test program run-time parameters follow.

Card B8 and B9 - Run-time parameters required by the test program. Since the file is allocated by AIDS, the volume serial number in the VDC card is all zeros.

Card B10 - AIDS test parameter cards.

Card B11 - Indicates that there are no more AIDS input cards for Program OMALL8. No memory dump will be made after a normal termination. Memory will be printed in four-byte hexadecimal groups if an abnormal termination occurs.

Card C1 - Defines program Y as the next program to be tested.

Card C2 - Device card designating that a disc file containing user input data is to be used by the test program. AIDS will ask for its assignment using the name DISKIN.

Card C3 - Contains the name of the user input file and the volume serial number of the input volume. No editing of this file is requested.

Card C4 - Device card designating that a work or output file on a disc storage unit is to be allocated by AIDS for the test program.

Card C5 - Contains the name to be given to the file and specifies that it is to be printed in hexadecimal with graphic equivalents upon a normal or abnormal termination. The file is to be deallocated and purged when the test program terminates.

Card C6 - Informs AIDS that it is to allocate two 10-cylinder extents for this file.

Card C7 - Indicates that test program run-time parameters follow.

Automatic Testing (Cont'd)

Card C8, C9, and C10 - Run-time parameters required by the test program.

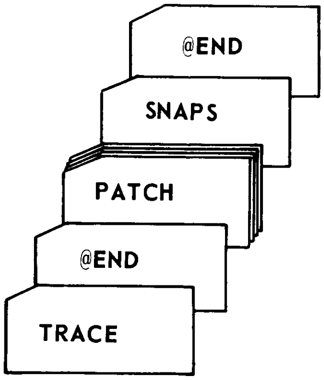
Card C11 - AIDS test parameters.

Card C12 - Indicates that there are no more AIDS input cards for program Y. A memory dump is to be taken upon a normal or abnormal termination. Its format will be graphic in two-byte groups.

Card C13 - Indicates the end of the AIDS session.

Console-Controlled Testing

◆ No specific examples of console-controlled testing input can be given because it is entirely up to the programmer which parameters are desired. The only input format that has to be followed is the use of the @END card when the TRACE, SNAPS, or PATCH parameters are entered from the card reader or magnetic tape.



AIDRDV

The first time a Read Device parameter is entered, AIDS will read the Trace parameter. The second time a Read Device is used, the Patch and Snaps parameters will be read.

Logging Messages

```
PROG JSMCD   INITIATED FOR TESTING WITH AIDS

DEV RD SYSIP

DEV 01 SYSLST 00                S                                    NO COUNT,ONE INSERTED IF REQUIRED  } Device Cards



                     AIDS REQUESTS FOR JSMCD

SNAPS JSMCD        000F34 000F68 C4 000F34 0,0,10

TRACE JSMCD        000F76 000F9A 000F72 0,0,6

PATCHJSMCD       EG00111E   EXCG PATCH                               NO END SENTINEL  } Test Parameters

PATCHJSMCD       FG00111E   EXCG PATCH-

END PROG C4

TEST PROGRAM I/W MESSAGE   } Console Typewriter Message and Response
   ARE CARDS SYSIPT AVAIL?
Y

        A DATA ERROR INTERRUPT  HAS OCCURRED AT PROG REL 000F3C ,P CTR 0057F4    AIDS Error Interrupt Message
 ¤¤



                      AIDS REQUESTS FOR JSMCK1

SNAPS JSMCK1         000A32 000AE9 C4 000B3C 0,1,10

TRACE JSMCK1         000B70 000BB8 000B70 0,0,6

PATCHJSMCK1      AI000C5234 D221202B202A,D211202A4000,98F12242,41E03018     01  } Test Parameters

PATCHJSMCK1      AI000C52   47FF004C,9240202A                               02

PATCHJSMCK1      AG000C52   EXECUTED ADD PATCH-                             03

PATCHJSMCK1      FG000AEE   EXCG PATCH-
```

Snapshot

```
                                                 AIDS SNAPSHOT AT  P 00B3C

                                               IMR FFF38E03     IFR 00000000

                            0          1          2          3          4          5          6          7
   P1 GEN REGS          00000000   00003BA8   4F0045D6   00000000   00000000   00000000   00000000   00000000

                            8          9          A          B          C          D          E          F
                        00000000   00000000   00000000   00000000   00000000   00000000   000046F8   00003D30

   FLT PT REGS     .0000000000000 E-40       .0000000000000 E-40       .0000000000000 E-40       .0000000000000 E-40

                B 0     0 3 5 1     9 0 0 0     0 5 4 3     0 0 4 5     6 7 8 0     0 8 7 2     3 1 0 1     3 2 3 6     6
 P 00A30    20CAC2F0    F0F3F5F1    F9F0F0F0    F0F5F4F3    F0F0F4F5    F6F7F8F0    F0F8F7F2    F3F1F0F1    F3F2F3F6    F6404040

            B 0 0 3     5 1 9 0     1 3 2 3     6 6
 P 00A58    C2F0F0F3    F5F1F9F0    F1F3F2F3    F6F64040    40404040    40404040    40404040    40404040    40404040    40404040

 P 00A80   *40404040    40404040    40404040    40404040    40404040    40404040    40404040    40404040    40404040    40404040

                                                B 0 0 3     5 1                        ? A B
 P 00AD0    40404040    40404040    40404040    C2F0F0F3    F5F10000    000C0132    366FC1C2
EXECUTED ADD PATCH
EXCG PATCH
```

} Test Program Output

Trace

```
   LC     OP LMGR B1  CB1    D1    1EA    X2   CX2   B2  CB2    D2   2EA   C        1                     2

*  0471E  47  D0                                      2  045D6  1DE  047B4 00
*  0471E  47  D0                                      2  045D6  1DE  047B4 02
*  04722  F2  36  2   045D6  0B4   0468A              2  045D6  012  045E8 02  0000750F              F0F0F0F0F7F5F0
*  04728  F2  36  2   045D6  0B8   0468E              2  045D6  019  045EF 02  0000750F              F0F0F0F0F7F5F0
*  0472E  FA  33  2   045D6  0B4   0468A              2  045D6  0B8  0468E 02  0001500C              0000750F
*  04734  F2  36  2   045D6  0B8   0468E              2  045D6  00B  045E1 02  0000700F              F0F0F0F0F7F0F0
*  0473A  FB  33  2   045D6  0B4   0468A              2  045D6  0B8  0468E 02  0000800C              0000700F
*  04740  F2  36  2   045D6  0B8   0468E              2  045D6  020  045F6 02  0000800F              F0F0F0F0F8F0F0
*  04746  FB  33  2   045D6  0B4   0468A              2  045D6  0B8  0468E 00  0000000C              0000800F
*  0474C  47  70                                      2  045D6  1DE  047B4 00
*  04750  D2  05  2   045D6  0AE   04684              2  045D6  004  045DA 00  C2F0F0F0F3F5          C2F0F0F0F3F5
*  04756  92  00  2   045D6  13F   04715                                   00  00
*  0475A  D2  06  2   045D6  02A   04600              2  045D6  004  045DA 00  C2F0F0F0F3F5F8        C2F0F0F0F3F5F8
*  04760  D2  06  2   045D6  031   04607              2  045D6  020  045F6 00  F0F0F0F0F8F0F0        F0F0F0F0F8F0F0

B0003580000800     Test Program Output

**
```

LC - instruction location (absolute)

OP - operation code

LMGR - instruction length, mask, or general register byte

B1 - first operand base register number

CB1 - contents of B1

D1 - first operand displacement

1EA - generated address for first operand

X2 - second operand index register number

CX2 - contents of X2

B2 - second operand base register number

CB2 - contents of B2

D2 - second operand displacement

2EA - generated address for second operand

C - condition code after instruction execution

1 - first ten bytes of data referenced by first operand

2 - first ten bytes of data referenced by second operand

Memory Print

```
                                        AIDS MEMORY PRINT

                                     IMR FFF3FE03     IFR 00000000

                         0         1         2         3         4         5         6         7
 P1 GEN REGS          00000000  000048B8  4F00577A  000048C0  00005F40  00006F40  8F006A52  00007184

                         8         9         A         B         C         D         E         F
                      00000000  00008BC8  00006F51  000050FA  AF005076  8F00980C  000057B8  00004AA0

 FLT PT REGS       .0000000000000 E-40     .0000000000000 E-40     .0000000000000 E-40     .0000000000000 E-40


                  ¤ ¤                                                  |    :                  -         ?          ^
A 04800        0010FFFF     0001000F     4001EFBA     00000000     000048B8     4F00577A     000048C0     00005F40     00006F40     8F006A52

                                              H           ?            &            &                                  ¢
A 04828        00007184     00000000     00008BC8     00006F51     000050FA     AF005076     8F00980C     000057B8     00004AA0     00000000

                                                                                                                     - ¤
A 04850        00000000     00000000     00000000     00000000     00000000     00000000     00000000     00005997     00005FFF     40000060

                 U K                          ?            &                                                             U
A 04878        0001E4D2     0F01EDBE     00006F51     000050FA     00000000     00000000     00000000     00000000     0001E4FA     00000000

                                              U                                              S Y S I      P T
A 048A0        00000000     00000000     0001E4B8     00000000     00000000     00000000     E2E8F2C9     D7E30510     000048E0     00003AA0

                    Y                                                                                        &
A 048C8        040048F8     00030000     05005920     00000048     00000080     00000000     050058D0     00000050     00000000     00000000

                0 C            S                                                F I L E      A
A 048F0        00F0C300     0000E283     00000000     00000000     000058D0     C6C9D3C5     C1404000     00040000     00000000     0000003C
```

STATISTICAL/ HISTORICAL INFORMATION ON RECOVERABLE AND UNRECOVERABLE ERRORS (SHIRUE)

General Description

♦ The TDOS Statistical and Historical Information on Recoverable and Unrecoverable Errors routine gathers and displays hardware error statistics for analysis of device and system efficiency by maintenance personnel. It also optionally permits the accumulation of the number of input/output operations performed. Statistics are maintained on magnetic tape and random access devices only.

When initially loaded, SHIRUE obtains control, modifies the Executive routine, and builds a statistics table from the Executive Device List table. The Executive modification establishes linkage between the Executive routine and SHIRUE.

Preset Functions

To furnish the number of errors that required error recovery, and the number of retries that occurred during error recovery, per device.

To furnish the number of Service Request Not Honored (SRNH) and the number of Transmission Parity Errors (TPE) that occurred, per device. These are types of errors of special interest. Their counters are included in the total error counters.

Optional Functions

To furnish the total input/output count, per device.

Input

♦ The input to this routine is a response to the typewriter message "nΔSHIRUEΔ0970AΔISΔI/0 COUNT DESIRED" and then a response to the typewriter message "nΔSHIRUEΔ0915AΔSHIRUEΔREADY" as indicated under Detailed Description.

Output

♦ The output to this routine consists of information that has been gathered and displayed to the typewriter or printer as indicated under Detailed Description.

Equipment Configuration

Required

♦ Processor (65K).

Console Typewriter.

Disc storage unit (system residency).

At least one Magnetic tape device or Disc storage unit or Drum storage unit or Mass storage unit.

Optional

Additional tape devices and random access units may be used in any mix up to 30 devices.

Printer.

Detailed Description

♦ Errors are accumulated by command within the device. The command categories are Write, Read, Read Reverse/Other and Write Control/Seek. Read Reverse and Write Control apply to magnetic tape only, while Other and Seek apply to random access only.

Error Types
(By Command Category)

♦ The types of error counters maintained are:

1. Number of errors.
2. Service request not honored.
3. Transmission parity error.
4. Retries.

Input
Descriptions

♦ When SHIRUE is initialed, the message "nΔSHIRUEΔ0970ΔISΔI/0ΔCOUNTΔDESIRED" is displayed on the typewriter and requires an input response to specify the desire to accumulate I/0 counts for the devices. This message must be replied to immediately. Operator response is in the format:

n=x

where:

n= is the one character program number assigned to SHIRUE by the executive system at load time.

x= Y yes; 1/0 count is desired.
 = N no; 1/0 count is not desired.

The user has the option to accumulate I/0 counts by answering Y (yes) to the I/0 message (0970ΔISΔI/0ΔCOUNTΔDESIRED). A response of Y (yes) will cause I/0 counts to be accumulated for devices on the executive device list. The inclusion of devices to the SHIRUE statistical table is based on the following:

1. If the device type is less than hexadecimal 0A it is eliminated. (It is assumed that all tape, disc, drum, and mass storage devices have a device type value equal to or greater than hexadecimal 0A. The purpose of this test is to eliminate the printer, card reader, card punch, and the paper tape reader punch, etc.)

Input Descriptions (Cont'd)

2. If the device type is equal to hexadecimal 10, 11, 12, or OE, it is eliminated.

   It is assumed that device type

   10 is the data exchange control.
   11 is the communication control multichannel.
   12 is the switch controller.
   OE is reserved for future use.

Note:

When the count of the 1/0's fired is specified, it causes the execution of an additional instruction each time an 1/0 is fired. A minimum of 9 and a maximum of 192 instructions are required assuming 20 devices in the statistical table. The formula for maximum additional instruction is: 32+8(X)=maximum number of instructions, where (X) is the number of devices in statistical table.

After SHIRUE links itself to the Executive routine and constructs its program statistics table, it types a message "nΔSHIRUEΔ0915AΔSHIRUEΔREADY." SHIRUE continues to gather statistics on all system magnetic tape and random access devices until the Operator responds to this message.

Operator response is in the format: nΔxyΔcuu where:

n= is the one character program number assigned to SHIRUE by the executive system at load time.

Δ= space

x= C indicates display the statistical counters and clear the counters. (If cuu parameter is also specified only the counters for the one device will be displayed and cleared, otherwise all counters will be displayed and cleared).

= S indicates display the statistical counters and save their contents, do not clear. (If cuu parameter is also specified only the counters for the one device will be displayed otherwise all counters will be displayed.)

= D indicates display the statistical counters and terminate SHIRUE. (If the cuu parameter is also specified only the counters for the one device will be displayed otherwise all counters will be displayed.)

= T indicates terminate SHIRUE and do not display any statistics. (When T is specified the y and cuu parameters are ignored.)

= N indicates clear the statistical counters for all devices and do not display their contents. (When N is specified the y and cuu parameters are ignored.)

y= C indicates that the console is the display device and that the statistics are to be displayed in the compressed format.

= E indicates that the console is the display device and that the statistics are to be displayed in the expanded format.

= P indicates that the printer is the display device and that the statistics are to be displayed in the expanded format.

Notes:

1. The y parameter is meaningful only when the x parameter specified that statistics are to be displayed.
2. Display formats are explained under Output Descriptions section.

cuu is an optional three character parameter which specifies that the statistics for a single device are to be displayed and the contents of the counter cleared or saved, dependent on the x parameter, The cuu parameter is meaningful only when the x parameter is C, S, or D. When the cuu parameter is spaces or not present the statistical counters for all devices are displayed, and cleared or saved, dependent on the x parameter. The values for cuu must represent the physical channel and unit number as assigned at systems generation time.

c= a one character value for the channel number (0-9) representing the channel number in executive device list.

uu= a two character value for the actual device number. Each character may range from 0 to F to represent the hexadecimal value of the device address.

Example:

To display on the console in compressed format the accumulated statistics for one device which is on channel 1 and has the device number of $02_{(16)}$, and the counters are to be saved.

The reply to 0915A should be:

```
nΔSCΔ102
```

Output Description

The program upon Operator command, reports the information that has been gathered and displays it to the specified device. If no errors have been recorded at the time of a Display command, the message "NO ERROR TO REPORT" will be displayed.

When the Display option is exercised by the Operator, the following lines will be displayed on the output device. The Header line will be displayed once, and the Detail lines will be displayed only for those devices on which activity occurred.

Output Descriptions (Cont'd)

Header Line

Printer Header:

```
CCDDTP/ΔIOCTΔΔ/WRITE-ΔSRNH-ΔTPEΔRETRY/
ΔREAD-ΔSRNH-ΔMTAΔ-RETRY/RROT-ΔSRNH-ΔMTAΔ-RETRY/
ΔWCSK-ΔSRNH-ΔTPEΔ-RETRY/
```

where:

CC= Channel and co-channel respectively

DD= Device number

TP= Device type

IOCT= Input/Output count

WRITE= Write command errors

READ= Read command errors

RROT= Read reverse (tape only) and other (random access only) errors

WCSK= Write control (tape only) and Seek (random access only) command errors

SRNH= Service request not honored (within command category)

TPE= Transmission parity errors (within command category)

RETRY= Error recovery retries (within command category)

MTA= Magnetic tape alarm (in lieu of TPE for read and read reverse)

Command categories are Write, Read, Write Control/Seek, and Read Reverse/Other.

Console Header:

```
nΔSHIRUEΔ0950ΔΔCCDDTP/IOCT/WRT-S-P-R/RD-S-M-R/
RROT-S-M-R/WCSK-S-P-R/
```

where:

n= The one character program number assigned by executive at load time.

SHIRUE= The program name in the Executive message identifier.

0950= SHIRUE's message identifier

CC= Channel and co-channel respectively

Output Descriptions (Cont'd)

DD= Device number

TP= Device type:

1A- 7 Channel Magnetic Tape
0B- 9 Channel Magnetic Tape
0C- 70/564 Disc Storage Unit
0D- Drum Storage Unit 32 cylinders
1D- Drum Storage Unit 64 cylinders
2D- Drum Storage Unit 96 cylinders
3D- Drum Storage Unit 128 cylinders
4D- Drum Storage Unit 160 cylinders
5D- Drum Storage Unit 192 cylinders
6D- Drum Storage Unit 224 cylinders
7D- Drum Storage Unit 256 cylinders
0F- 70/568 Mass Storage Unit

IOCT= Input/output count

WRT= Write command errors

RD= Read command errors

RROT= Read reverse and other errors

WCSK= Write control and seek command errors

S= Service request not honored within command category

P= Represents transmission parity errors within command category

R= Error recovery retries within command category

M= Magnetic tape alarm in lieu of TPE for read and read reverse

Command categories are Write, Read, Write Control/Seek, and Read Reverse/Other.

Detail Line

Statistics may be displayed in expanded format on both the printer and the console typewriter. Statistics may be displayed in compressed format on the console typewriter only. The Header is the same for both formats. The Detail line in compressed format contains the same information as the expanded format with zero suppression except that the spaces are eliminated from the statistics.

An example of a minimum compressed message is:

```
nΔSHIRUEΔ0955ΔΔCCDDTP/1/- - -/- - -/- - -/- - -/
```

Output Descriptions (Cont'd)

where:

n= The one character program number assigned by executive at load time

SHIRUE= Program name

0955= SHIRUE message identifier

CC= Is the channel and co-channel respectively. If no co-channel exists the channel number is inserted into the co-channel, so that channel and co-channel are the same.

DD= Device number

TP= Device type

/1= Indicates that one I/0 command was fired to the device

/- - - = Indicates no write errors occurred

/- - - = Indicates no read errors occurred

/- - - = Indicates no read/reverse/other errors occurred

/- - - /= Indicates no write control/seek errors occurred

The expanded format to the console is distributed over two lines.

Notes:

1. The I/0 count field may range in value from 0 to 9,999,999. No provision has been made for overflow warning on this counter. If overflow occurs, the counter is returned to zero and accumulation continues.

2. The other counters may range in value from 0 to 65,535. An overflow warning is given when one of these counters reaches maximum value. Accumulation to the counter that has reached the maximum is discontinued until a clear of the counters for that device is requested. The overflow message will not appear until the user requests a display of the counters.

Considerations For Use

♦ This program will occupy a job slot and approximately 4096 hytes of HSM.

The priority assigned to this program has no bearing on the operation of this or any other user routine being processed.

SHIRUE will not run under the control of Monitor since it is required to accumulate statistics of programs which must run under Monitor, e.g. Language Translators.

If SHIRUE does not terminate normally (through its terminate option), the Executive must be reloaded. For this reason, the EΔHLT Executive console routine must not be used to terminate SHIRUE.

5. DIAGNOSTICS - RANDOM ACCESS

◆ The Random Access Diagnostic routines used with TDOS are the same as those used with TOS. These routines are listed below and are described in the TOS Utility Routines manual, 70-35-302.

SELF-LOADING RANDOM ACCESS EDIT

RANDOM ACCESS EDIT (RAEDIT)

70/568 CARD CHECK (CARDCK)

6. SYSTEM MAINTENANCE

◆ The System Maintenance routines used with TDOS are the same as those used with TOS except for minor differences in the Linkage Editor routine. The routines that are the same in both systems are listed below and are described in the TOS Utility Routines manual, 70-35-302.

OBJECT MODULE LIBRARY UPDATE (OMLU)

MACRO LIBRARY UPDATE (MLU)

LOAD LIBRARY UPDATE (LLU)

COBOL LIBRARY UPDATE (CLU)

SOURCE LIBRARY UPDATE (SLU)

TAPE FILE MAINTENANCE (TPMAIN)

The TDOS Linkage Editor routine (LNKEDT) differences are described below.

LINKAGE EDITOR (LNKEDT)

◆ The TDOS Linkage Editor operates in the same manner and performs the same functions as the TOS Linkage Editor (refer to TOS Utility Manual) with the following exceptions:

1. Object modules can be extracted from tape, disc, or drum and bound into output programs.

2. Only a Program Load Library can be produced by the TDOS Linkage Editor. The ACTION parameter is not used.

3. The TDOS Call Library (SYSLIB) may be on disc, drum, or magnetic tape.

The output of the TDOS Linkage Editor is still a magnetic tape (SYSUT2) and contains a Program Load Library. This library is made disc or drum resident by the Program Library Transcriber (refer to page 8-3).

7. SYSTEM MAINTENANCE - RANDOM ACCESS

◆ The Random Access System Maintenance routines used with TDOS are the same as those used with TOS, with restrictions. These restrictions apply to the Random Access Storage Allocator routine and are listed below.

The routines that are the same in both systems are listed below and are described in the TOS Utility Routines manual, 70-35-302.

RANDOM ACCESS INDEX EDIT (RAINDX)

RANDOM ACCESS STORAGE ALLOCATOR (RAALLR)

70/568 SERVICE PROGRAM (RAMSUP)

DISC/DRUM DUMP AND RELOAD (DDRL)

RANDOM ACCESS DUMP AND RELOAD (RADAR)

RANDOM ACCESS STORAGE ALLOCATOR (RAALLR)

◆ The TDOS Random Access Storage Allocator operates in the same manner and performs the same functions as the TOS version, with the following restrictions:

1. The fileid "EXCLIB" is reserved for the TDOS Executive.
2. The fileid "PGMLIB" is reserved for the TDOS Program Library.
3. The fileid "ASSEMBLYΔMACROS" is reserved for the TDOS Assembly Macro Library.
4. The fileid "COBOLΔSOURCEΔLIBR" is reserved for the TDOS COBOL Library.
5. The fileid "OBJECTΔMODULEΔLIB" is reserved for the TDOS OML.

The names listed above <u>must</u> be used when allocating for the associated file. They <u>cannot</u> be used for any other files.

LOAD LIBRARY TO TAPE (LLT)

General Description

◆ The Load Library to Tape routine provides a backup, on tape, for all or selected programs from a disc or drum program library. The tape is transcribed back to disc or drum using the Program Library Transcriber (PRGTRN) routine.

The output tape is in Load Library format and may be used in the system as a PLLT.

Preset Functions

None.

Optional Functions

All Load Library to Tape functions are controlled by parameters submitted at run time. The functions available are listed below.

1. Transcribe a disc or drum program library.
2. Transcribe all or selected programs to the output tape.

Input

◆ Input to this routine is a disc or drum containing a TDOS program library and parameter cards specifying the processing to be performed.

Output

◆ The routine's output is a tape containing programs which appeared in the program library on the disc or drum and a printer listing of the input parameters. Only programs listed in the program directory will be on the output. All programs in the directory will be written to tape unless specifically deleted.

Equipment Configuration

Required

◆ Processor (65K)

Console typewriter

Card reader

Disc storage unit or drum memory unit

Magnetic tape unit

Printer

Routine Parameters - General

◆ The following parameters are used with this routine.

Unscribe Parameter

This parameter defines the disc or drum containing the program library and the name of the program library.

Delete Parameter

This parameter names programs which are not to be transcribed to the output.

End Parameter

This parameter indicates the end of parameter input.

Routine Parameters - Detailed

Unscribe Parameter

◆ The UNS parameter must be used to describe the program library being transcribed.

Format

ΔUNSΔV=t,A=i,D=M

Entry	Meaning
ΔUNSΔ	Parameter identifier.
V=t	Volume serial number of the random access device containing the program library. t = six alphanumeric characters.
,A=i	File ID of the program library. i = PGMLIB
,D=M	Optional. When used, indicates that the random access device is a drum memory unit. If omitted, a disc is assumed.

Delete Parameter

◆ The DEL parameter is an optional parameter which specifies which programs contained in the program directory are not to be transcribed to the output.

Format

```
ΔDELΔV = t,A = i,N = (p1,p2 , . . . , pn)
```

Entry	Meaning
ΔDELΔ	Parameter identifier.
V = t	Volume serial number of the random access device containing the program library. t = six alphanumeric characters.
,A = i	File ID of program library. A = PGMLIB
,N = (p1,p2, . . . , pn)	Programs which are not to be transcribed to the output tape. p = program name; one to six alphanumeric characters. When more than one program name is used, each must be separated by a comma.

End Parameter

◆ This parameter signifies the end of parameter input and always must be used.

Format

```
ΔENDΔ
```

Considerations for Use

◆ The Load Library to Tape routine operates under TOS or TDOS.

Only those programs contained in the program library directory can be transcribed to the output tape.

The program library on the disc or drum is unaltered by this routine.

Parameter Examples

◆ 1. To transcribe a complete program library to tape.

```
//ΔASSGNΔSYSIPT,R1
//ΔASSGNΔSYSLST,L1
//ΔASSGNΔSYSUT2,01
//ΔASSGNΔRADEV,A0
//ΔEND
ΔUNSΔV = 000777,A = PGMLIB
ΔEND
```

Parameter Examples *(Cont'd)*

2. To transcribe all programs except COBOL and FORTRAN to tape.

```
//ΔASSGNΔSYSIPT,R1
//ΔASSGNΔSYSLST,L1
//ΔASSGNΔSYSUT2,02
//ΔASSGNΔRADEV,A2
//ΔEND
ΔUNSΔV=555555,A=PGMLIB
ΔDELΔV=555555,A=PGMLIB,N=(COBOL,FORTRN)
ΔEND
```

Device Assignments

◆ Under Executive or Monitor Control

SDN	Device Type	Remarks
SYSIPT	Card reader (see note).	Input parameters.
SYSLST	Printer (see note).	Output listing device.
SYSUT2	Magnetic tape.	Output load library tape.
RADEV	Disc or drum.	Contains input program library.

8. LIBRARY CONVERSION

◆ (Pages 8-1 through 8-2D were deleted by revision, December 1968.)

PROGRAM LIBRARY TRANSCRIBER (PRGTRN)

General Description

◆ The Program Library Transcriber transcribes loadable programs from a TOS System Load Library Tape (SLLT) or a Program Load Library Tape (PLLT) to a disc or drum, in a format acceptable to the TDOS Executive System.

The storage area on the disc or drum must first be initialized using the Random Access Volume Initializer, and have been allocated using the Random Access Storage Allocator.

Preset Functions

None.

Optional Functions

All program Library Transcriber processing is under the control of routine parameters submitted at run time. These parameters provide the facility to create a Program Load Library on disc or drum, or to modify existing programs stored thereon as follows:

1. Add all programs, or selected programs, from an input SLLT or PLLT.
2. Delete or nullify programs in a program storage area.
3. List the names of programs stored in a program storage area.
4. List the names and disc/drum locations of each load in a program.

Input

◆ Input to this routine consists of an SLLT or PLLT and parameters, entered through the card reader, that specify the functions to be performed.

Output

◆ The output of this routine consists of a disc or drum that contains the Program Load Library in TDOS format and an abstract listing of all the programs that were transcribed. An abstract of the programs already stored on the disc or drum may be obtained as an option.

Equipment Configuration

Required

◆ Processor (65K).
Console typewriter.
Magnetic tape device.
Card reader, or Videoscan document reader with card read feature.
Printer.
Disc storage unit or drum memory unit.
Random access controller with record overflow feature

Optional

◆ When running under monitor control, magnetic tape devices may be substituted for the printer and for the card reader.

Routine Parameters - General

◆ The following parameters are used with this routine:

Initialize Parameter

This parameter creates a program directory within a newly initialized and allocated program storage area in a random access volume.

Replace Parameter

This parameter clears an existing program storage area and creates a new program directory within the allocated area in a random access volume.

Add Parameter

This parameter transcribes all or selected programs from tape to disc or drum.

Test Parameter

This parameter transcribes programs from tape to disc or drum for testing purposes. Programs added in this manner must be deleted when the testing is completed.

Delete Parameter

This parameter removes program entry names from the program directory.

Abstract Parameter

This parameter requests a listing of the names of all programs stored on disc or drum.

End Parameter

This parameter denotes the end of parameter input.

Routine Parameters - Detailed

Initialize Parameter

◆ The INT parameter directs the Program Library Transcriber to establish a Program Directory within the allocated program storage area.

Format

```
ΔINTΔV=t,A=i,P=n,R=N
```

Initialize Parameter (Cont'd)

Entry	Meaning
ΔINTΔ	Parameter identifier.
V=t	Volume serial number of the random access device to be initialized. t=one to six alphanumeric characters.
,A=i,	File ID of the program storage area. i=PGMLIB.
P=n	Optional. Indicates the number of tracks to be allocated to the program directory. If omitted, only one track is reserved. Each track can contain up to 80 program directory entries. n = 0 to 9 and specifies address of last track in the first cylinder to be reserved for program directory.
,R=N	Optional. When used, indicates that no overflow records are to be written to the random access device. If omitted, overflow records are assumed.

Examples:

ΔINTΔV=21TMAN,A=PGMLIB,P=3

ΔINTΔV=ABLE26,A=PGMLIB,,R=N

ΔINTΔV=DEVIS9,A=PGMLIB,

ΔINTΔV=TAPLIB,A=PGMLIB,P=3,R=N

Note:

The field A=i is always followed by a comma whether or not the optional fields P=n or R=N are used.

Replace Parameter

◆ The Replace parameter directs the Program Library Transcriber to clear the allocated area and to establish a Program Directory within the allocated area.

Format

ΔRPDΔV=t,A=i,P=n,R=N

Replace Parameter (Cont'd)

Entry	Meaning
ΔRPDΔ	Parameter identifier.
V=t	Volume serial number of the random access device to be initialized. t=one to six alphanumeric characters.
,A=i,	File ID of the program storage area. i=PGMLIB.
P=n	Optional. Indicates the number of tracks to be allocated to the program directory. If omitted, only one track is reserved. Each track can contain up to 80 program directory entries. n = 0 to 9 and specifies address of last track in the first cylinder to be reserved for program directory.
,R=N	Optional. When used, indicates that no overflow records are to be written to the random access device. If omitted, overflow records are assumed.

Examples:

ΔRPDΔV=D00011,A=PGMLIB,,R=N

ΔRPDΔV=KOMALL,A=PGMLIB,

ΔRPDΔV=OPTI03,A=PGMLIB,P=4

Note:

The field A=i if always followed by a comma whether or not the optional fields P=n or R=N are used.

ADD Parameter

♦ The ADD Parameter causes specified programs to be transcribed from a System Load Library Tape or a Program Load Library Tape to disc or drum storage area.

Format

ΔADDΔI=s,V=t,A=i,N=(p1,p2....,pn).

Test Parameter

◆ The Test Parameter causes specified programs to be transcribed from a System Load Library Tape or a Program Load Library Tape to disc or drum. These programs are transcribed for testing purposes only; they must be deleted when the testing is completed and before any further transcription takes place.

Format:

```
ΔTSTΔI=s,V=t,A=i,N=(p1,p2,....,pn).
```

Entry	Meaning
ΔTSTΔ	Parameter identifier.
I=s	Optional. When used, indicates the symbolic name of the input tape. s=one to six alphanumeric characters. If this field is omitted, SYSUT2 is assumed.
,V=t	Volume serial number of the random access device to which programs are to be transcribed. t=one to six alphanumeric characters.
,A=i	File ID of the program storage area to which programs are to be transcribed. i=PGMLIB
N=(p1,p2,...,pn)	Optional. Programs to be added to the output library. p=program name; one to six alphanumeric characters. When more than one program is added, each name must be separated by a comma. When this field is not used, all programs on the input tape are transcribed.

Notes:

1. Multiple Test parameters may not be used.
2. Field A=i always must be followed by a comma.
3. Programs transcribed using the Test function must be removed by the Delete parameter before any further Add or Test functions are used. (See Delete parameter, note 3.)

Examples:

```
ΔTSTΔV=000555,A=PGMLIB,
ΔTSTΔI =SYS001,V=60015A,A=PGMLIB,N=(CLT,PUT)
```

ADD Parameter (Cont'd)

Entry	Meaning
ΔADDΔ	Parameter identifier.
I=s	Optional. When used, indicates the symbolic name of the input tape. s=one to six alphanumeric characters. If this field is omitted, SYSUT2 is assumed.
,V=t	Volume serial number of the random access device to which programs are to be transcribed. t=one to six alphanumeric characters.
,A=i,	File ID of the program storage area to which programs are to be transcribed. i=PGMLIB
N=(p1,p2,...,pn)	Optional. Programs to be added to the output library. p=program name; one to six alphanumeric characters. When more than one program is added, each name must be separated by a comma. When this field is not used, all programs on the input tape are transcribed.

Notes:

1. If the names of the programs exceed one parameter card, the last program name is followed by a comma and one or more blanks; a nonblank character is then placed in column 72. The following card must contain the name of the next program beginning in column 16; columns 1-15 must be blank.
2. Field A=i always must be followed by a comma.

Examples:

```
ΔADDΔI=TOSSLT,V=DISC01,A=PGMLIB,N=(JOE,MARY,WALT)
ΔADDΔI=TOSSLT,V=DISC03,A=PGMLIB,N=(FICA66)
ΔADDΔV=DISC06,A=PGMLIB,
```

Delete Parameter

◆ This parameter causes specified programs to be deleted from a program storage area.

Format

ΔDELΔV = t,A = i,N = (pl,p2 . . . pn)

Entry	Meaning
ΔDELΔ	Parameter identifier.
V = t	Volume serial number of the random access device containing programs to be deleted. t = one to six alphanumeric characters.
,A = i	File ID of the program storage area containing the program to be deleted. i = PGMLIB.
,N = (pl,p2,. . . , pn)	Programs to be deleted from the program directory. p = program name; one to six alphanumeric characters. When more than one program is. to be deleted, each name must be separated by a comma.

Notes:

1. If the names of the programs exceed one parameter card, the last program name is followed by a comma and one or more blanks; a nonblank character is then placed in column 72. The following card must contain the name of the next program beginning in column 16; columns 1-15 must be blank.

2. When this parameter is used to delete programs that were transcribed using the ADD function, the programs are deleted by removing the entry for the program from the program directory. It does not delete the program or the load directory for the program. The delete function does not apply to the input tape.

   Physically removing programs from the random access device requires retranscribing by using the Replace parameter and ADD parameters for those programs that are to be retained.

3. When this parameter is used to delete programs that were transcribed using the Test function, the area occupied by the test programs is made available for other programs.

Examples

ΔDELΔV = DEVISA, A = PGMLIB, N = (TITLES, PAYROL, LICENS)

ΔDELΔV = 000901, A = PGMLIB, N = (DAARR)

Abstract Parameter

◆ The Abstract Parameter permits a listing of the Program Directory and Load Directory entries to be printed.

Format

```
ΔABSΔV=t,A=i,L=PD
```

Entry	Meaning
ΔABSΔ	Parameter identifier.
V=t	Volume serial number of the random access device from which information is to be abstracted. t=one to six alphanumeric characters.
,A=i,	File ID of the program storage area from which information is to be abstracted. i=PGMLIB.
L=PD	Optional. If used, only a printed listing of program directory (PD) will be displayed. If omitted, both the Program Directory and Load Directory will be listed.

Note:

Field A=i always must be followed by a comma.

Examples:

```
ΔABSΔV=DEVIS9,A=PGMLIB,

ΔABSΔV=DEVIS9,A=PGMLIB,L=PD
```

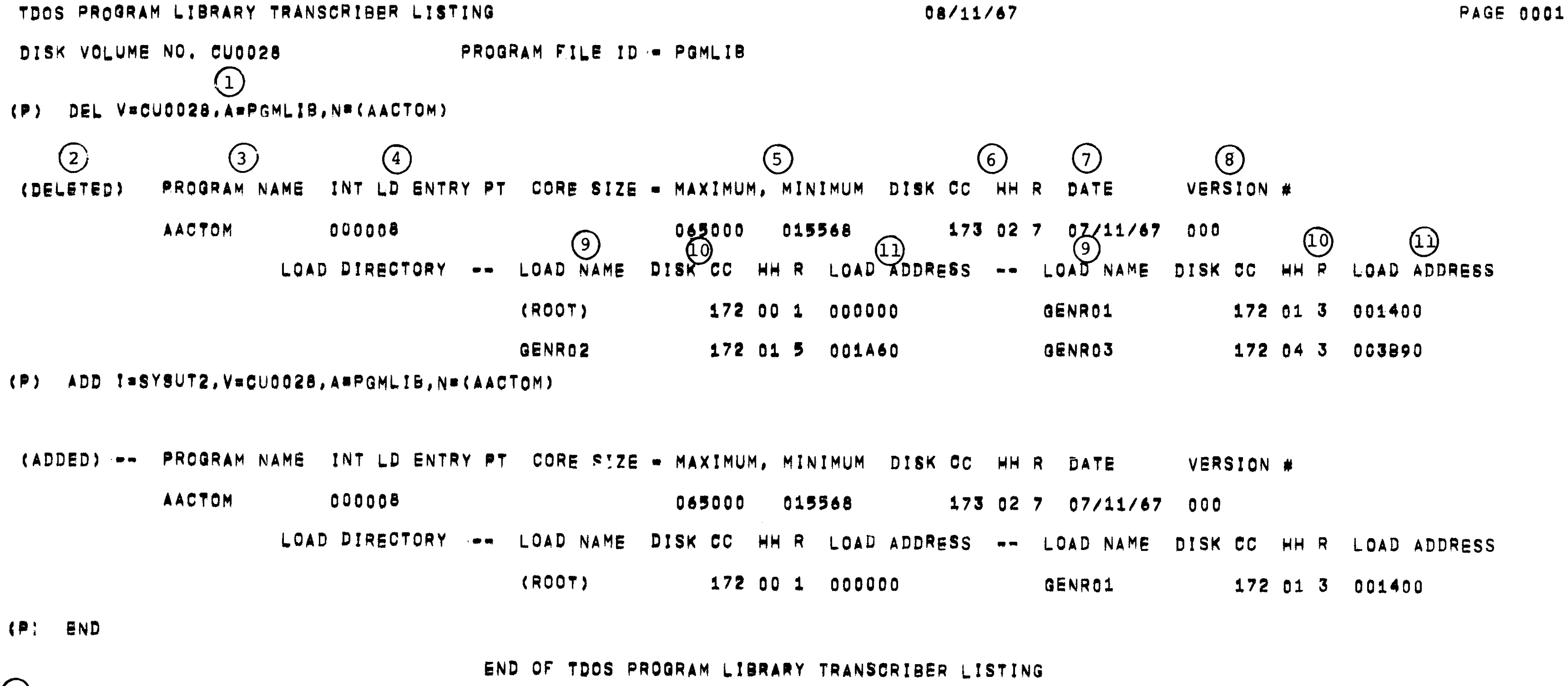
END Parameter

◆ This parameter signifies the end of parameter input and must always appear.

Format

```
ΔENDΔ
```

Program Addition or Deletion



```
TDOS PROGRAM LIBRARY TRANSCRIBER LISTING                                  08/11/67                              PAGE 0001

DISK VOLUME NO. CU0028               PROGRAM FILE ID = PGMLIB
                    ①
(P)  DEL V=CU0028,A=PGMLIB,N=(AACTOM)

   ②            ③         ④                             ⑤             ⑥     ⑦        ⑧
(DELETED)   PROGRAM NAME  INT LD ENTRY PT  CORE SIZE = MAXIMUM, MINIMUM  DISK CC  HH R  DATE      VERSION #

            AACTOM        000008                      065000   015568       173 02 7  07/11/67  000
                                            ⑨          ⑩          ⑪                   ⑨              ⑩       ⑪
                    LOAD DIRECTORY  --  LOAD NAME  DISK CC  HH R  LOAD ADDRESS  --  LOAD NAME  DISK CC  HH R  LOAD ADDRESS

                                        (ROOT)         172 00 1  000000             GENR01         172 01 3  001400

                                        GENR02         172 01 5  001A60             GENR03         172 04 3  003B90
(P)  ADD I=SYSUT2,V=CU0028,A=PGMLIB,N=(AACTOM)


(ADDED) --  PROGRAM NAME  INT LD ENTRY PT  CORE SIZE = MAXIMUM, MINIMUM  DISK CC  HH R  DATE      VERSION #

            AACTOM        000008                      065000   015568       173 02 7  07/11/67  000

                    LOAD DIRECTORY  --  LOAD NAME  DISK CC  HH R  LOAD ADDRESS  --  LOAD NAME  DISK CC  HH R  LOAD ADDRESS

                                        (ROOT)         172 00 1  000000             GENR01         172 01 3  001400

(P)  END

                           END OF TDOS PROGRAM LIBRARY TRANSCRIBER LISTING
```

① Input parameter

② Action taken

③ Program name

④ Initial load entry point (hexadecimal)

⑤ Maximum and minimum memory requirements (decimal)

⑥ Disc address of load directory (cylinder, head record)

⑦ Creation date

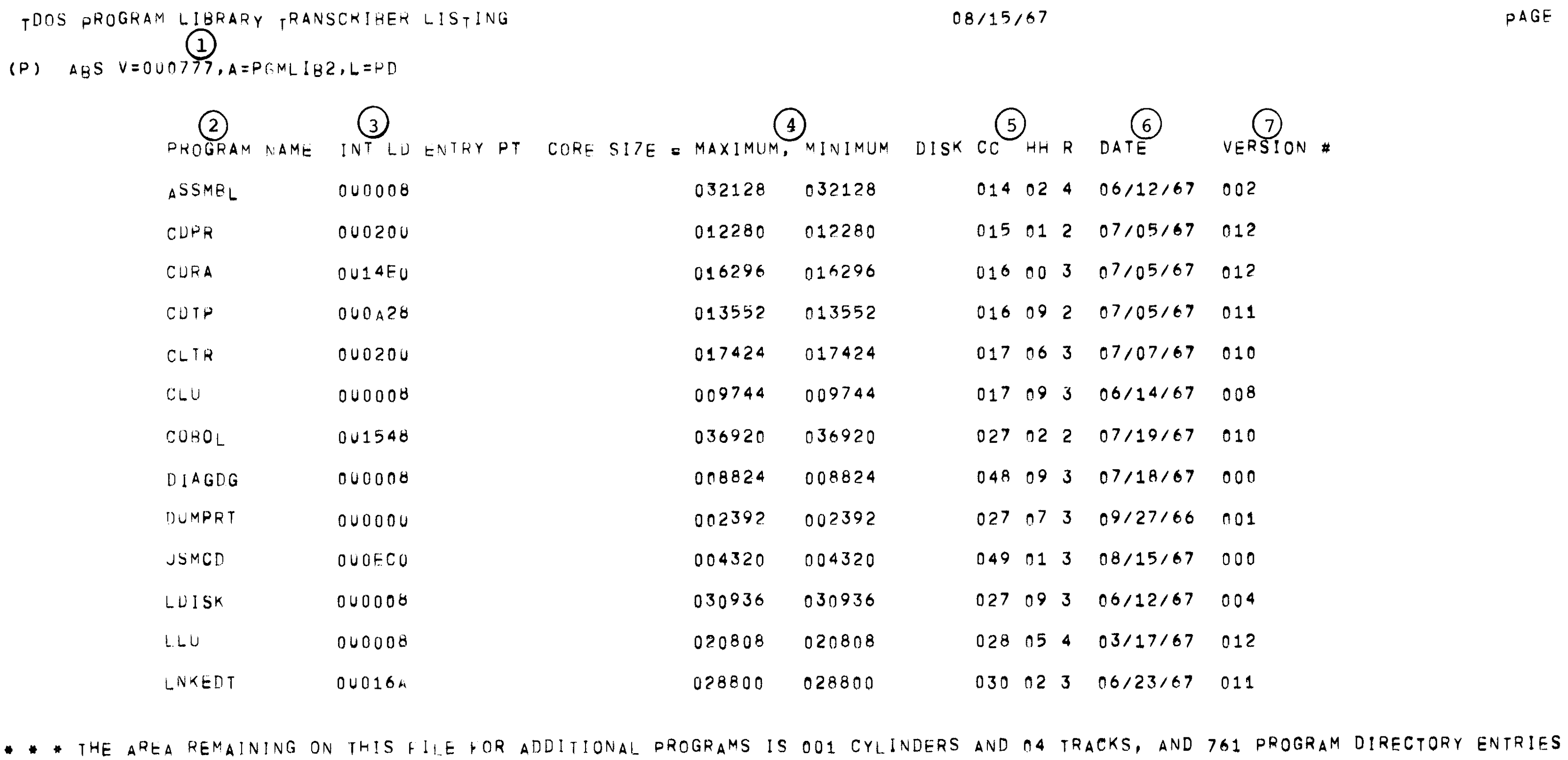
⑧ Version number

9 Load Name

10 Disc address of load (cylinder, head, record)

11 Program-relative load address (hexadecimal)

Abstract of Program Directory



```
TDOS PROGRAM LIBRARY TRANSCRIBER LISTING                              08/15/67                                         PAGE 0003

(P)  ABS V=000777,A=PGMLIB2,L=PD

          PROGRAM NAME   INT LD ENTRY PT  CORE SIZE = MAXIMUM, MINIMUM  DISK CC  HH R  DATE      VERSION #

          ASSMBL         000008                      032128   032128        014 02 4  06/12/67  002
          CDPR           000200                      012280   012280        015 01 2  07/05/67  012
          CDRA           0014E0                      016296   016296        016 00 3  07/05/67  012
          CDTP           000A28                      013552   013552        016 09 2  07/05/67  011
          CLTR           000200                      017424   017424        017 06 3  07/07/67  010
          CLU            000008                      009744   009744        017 09 3  06/14/67  008
          COBOL          001548                      036920   036920        027 02 2  07/19/67  010
          DIAGDG         000008                      008824   008824        048 09 3  07/18/67  000
          DUMPRT         000000                      002392   002392        027 07 3  09/27/66  001
          JSMCD          000EC0                      004320   004320        049 01 3  08/15/67  000
          LDISK          000008                      030936   030936        027 09 3  06/12/67  004
          LLU            000008                      020808   020808        028 05 4  03/17/67  012
          LNKEDT         00016A                      028800   028800        030 02 3  06/23/67  011

* * * THE AREA REMAINING ON THIS FILE FOR ADDITIONAL PROGRAMS IS 001 CYLINDERS AND 04 TRACKS, AND 761 PROGRAM DIRECTORY ENTRIES
```

(1) Input parameter	(6) Creation date
(2) Program name	(7) Version number
(3) Initial load entry point (hexadecimal)	
(4) Maximum and minimum memory requirements (hexadecimal)	
(5) Disc address of load directory (cylinder, head, record)	

Considerations For Use

◆ The Program Library Transcriber operates under TOS or TDOS.

The disc or drum output device must be initialized and allocated before the Program Library Transcriber can be run. The filename (entered in the VTOC) for the program storage area must be PGMLIB.

The maximum size of an individual load is 262K.

Parameter Examples

◆ 1. To create a program directory and transcribe an entire program library tape to a newly initialized disc:

```
//ΔASSGNΔSYSLST,L1
//ΔASSGNΔSYSUT1,01
//ΔASSGNΔ000001,A0
//ΔEND
ΔINTΔV = 000001,A = PGMLIB,P=5
ΔADDΔI = SYSUT1,V = 000001,A = PGMLIB,
ΔEND
```

2. To create a program directory and transcribe an entire program library tape to a disc with a previously existing program library. The existing library will be completely replaced by the new library.

```
//ΔASSGNΔSYSLST,L1
//ΔASSGNΔSYSUT1,01
//ΔASSGNΔ0000001,A0
//ΔEND
ΔRPDΔV = 000001,A = PGMLIB,P = 5
ΔADDΔI = SYSUT1,V = 000001,A = PGMLIB,
ΔEND
```

Device Assignments

◆ Under Executive or Monitor Control:

SDN	Device Type	Remarks
SYSIPT	Card Reader (see note).	Input parameters.
SYSLST	Printer or (see note).	Output Listing Device.
xxxxxx	Magnetic Tape.	xxxxxx = symbolic name assigned to the Load Library to be transcribed.
yyyyyy	Disc or Drum.	yyyyyy = volume number of the disc to which the Load Library is to be transcribed.

Note:

When running under Monitor, magnetic tape may be substituted for SYSIPT and SYSLST.

CALL LIBRARY TRANSCRIBER (CLTR)

General Description

◆ The Call Library Transcriber (CLTR) is used to transcribe call libraries from magnetic tape to disc or drum in a format suitable for the TDOS system. If desired, this routine may be run solely to compute the amount of storage area required for input libraries without physical transcription taking place.

The storage area on disc or drum must first be initialized using the Random Access Volume Initializer, and have been allocated using the Random Access Storage Allocator.

Preset Functions

This routine is preset to transcribe a complete TOS Call Library Tape mounted on SYSUT2 to a random access device with a volume serial number of 000000. In this case, no routine parameters are required.

Optional Functions

Routine parameters can be submitted at run-time for the following options:

1. To transcribe any or all of the call libraries listed below from tape to a random access device:

   Object Module Library
   Macro Library
   COBOL Library

2. To provide abstract listings of the directories for any or all libraries.

3. To compute the random access extent requirements of any or all libraries.

Input

◆ Input to this routine consists of a Call Library Tape and parameters that specify the functions to be performed. Parameters are entered through the card reader, paper tape reader, or magnetic tape.

Output

◆ 1. Any or all of the following call libraries:

   Object Module Library
   Macro Library
   COBOL Library

   transcribed to a random access device in a format suitable for processing.

2. Abstract listings of all or any of the specified directories of each library.

Equipment Configuration

Required

◆ Processor (65K).
Console typewriter.
Magnetic tape device.
Card reader, or Videoscan document reader with card read feature.
Printer.
Disc storage unit or drum memory unit.
Random access controller with record overflow feature.

Optional

◆ Magnetic tape devices may be substituted for the printer and the card reader.

Routine Parameters - General

◆ The following two parameters can be used with this routine when other than the preset functions are desired.

Transcribe Parameter

This parameter is used to transcribe specific call libraries from magnetic tape, to compute random access extent requirements, and to obtain an abstract listing of directories of call libraries.

END Parameter

This parameter signifies the end of input parameters.

Routine Parameters - (Detailed)

TRNS Parameter

◆ *Format*

$$\Delta\text{TRNS}\Delta\text{LIB}=\text{nnn,I}=\left\{\begin{matrix}\text{NONE}\\ \text{ssssss}\end{matrix}\right\}\text{,V}=\text{vvvvvv,}\left\{\begin{matrix}\text{ABS}\\ \text{CDS}\end{matrix}\right\}$$

Entry	Meaning
ΔTRNSΔ	Parameter identifier.
LIB=nnn	Library to be transcribed or abstracted: nnn = OML - Object Module Library = COB - COBOL Library = MAC - Macro Library = ALL - All libraries on tape. If this field is omitted, ALL is assumed.
,I=ssssss	Input device: ssssss = Symbolic name, one to six alphanumeric characters. If omitted, SYSUT2 is assumed. NONE = Indicates no transcription; abstract only.

(Cont'd)

TRNS Parameter (Cont'd)

Entry	Meaning
,V = vvvvvv	Volume serial number: vvvvvv = Serial number (six characters) of random access device to which library is to be transcribed, or from which an abstract is to be taken. If omitted, 000000 is assumed.
,ABS or ,CDS	Optional. ABS = Generate an abstract listing for libraries specified. CDS = List the number of cylinders and tracks which must be allocated for the libraries specified.

Notes:

1. More than one TRNS card may be submitted.

2. When the CDS option is used no transcription takes place, and the storage requirements only are computed.

Examples:

To transcribe all libraries on SYSUT2 to a random access device, volume serial number 001001, and provide an abstract listing:

```
ΔTRNSΔV=001001,ABS
```

To produce an abstract listing of the Macro Library stored on a random access device, volume serial number 001001:

```
ΔTRNSΔLIB=MAC,I=NONE,V=001001,ABS
```

To transcribe the COBOL Library on SYSUT3 to a random access device, volume serial number 500000:

```
ΔTRNSΔLIB=COB,I=SYSUT3,V=500000
```

To list the number of cylinders and tracks on device 001001 required for the transcription of the libraries on SYSLIB:

```
ΔTRNSΔI=SYSLIB,V=001001,CDS
```

To transcribe all libraries on SYSUT2 and provide an abstract listing:

```
ΔTRNSΔABS
```

END Parameter

◆ This parameter is only required when TRNS parameters have been supplied.

Format

ΔENDΔ

Considerations For Use

◆ Before a call library can be transcribed, the random access device must have been previously initialized by the RAINIT routine. In addition, the random access Storage Allocator routine must be used to allocate the file area in which the call library will be located. (If the size of the file area is not known, the CDS option of the TRNS parameter can be used.)

Libraries stored on a random access device cannot be updated or maintained by this routine. Should this be desired, the library must be updated on magnetic tape using the MLU, OMLU, or CLU routines. The updated library can then be retranscribed onto the random access device.

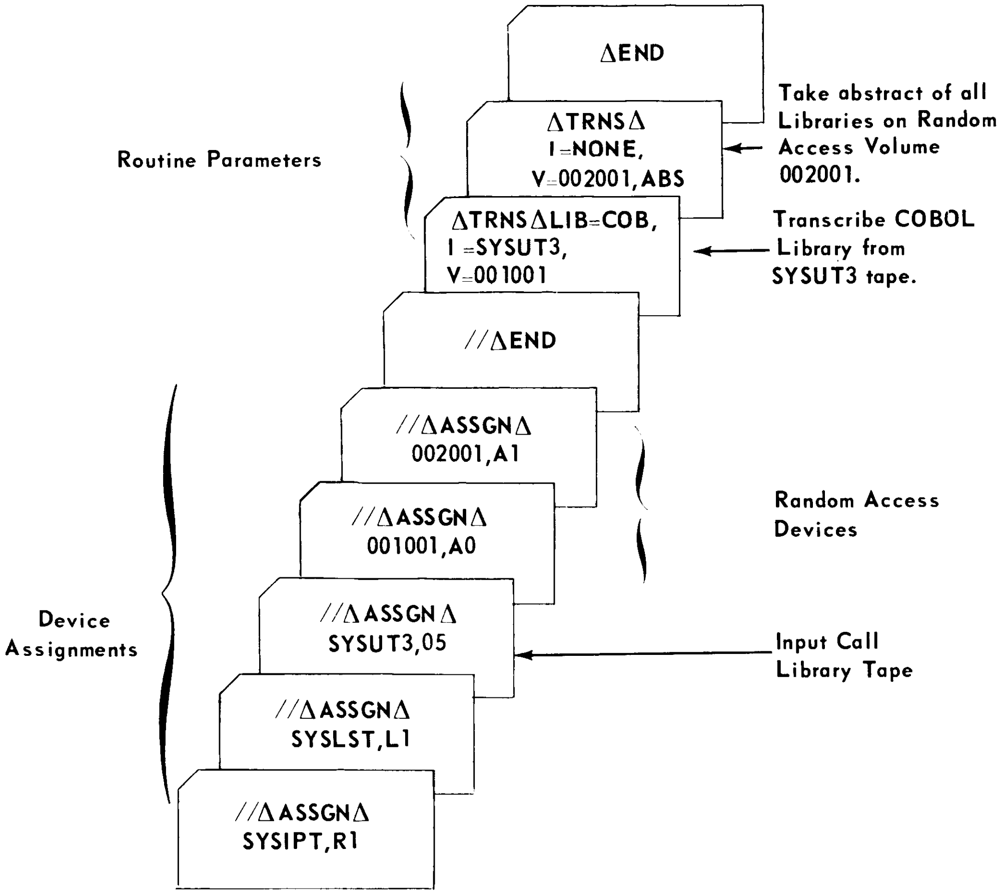
The label ID's for the libraries stored on a random access device are the same as those used for magnetic tape:

ASSEMBLYΔMACROSΔΔ
OBJECTΔMODULEΔLIB
COBOLΔSOURCEΔLIBR

When using the CDS option, the computed sizes of the input libraries are displayed by the following typeouts:

Typeout	Meaning
5991ΔΔttttttΔ cccΔttt	Total number of cylinders (ccc) and tracks (ttt) that must be allocated for the Macro Library on Volume tttttt.
5992ΔΔttttttΔ cccΔttt	Total number of cylinders (ccc) and tracks (ttt) that must be allocated for the COBOL Library on Volume tttttt.
5993ΔΔttttttΔ cccΔttt	Total number of cylinders (ccc) and tracks (ttt) that must be allocated for the Object Module Library on Volume tttttt.

Example of Job Stream Sequence



```
//ΔASSGNΔ
SYSIPT,R1

//ΔASSGNΔ
SYSLST,L1

//ΔASSGNΔ
SYSUT3,05

//ΔASSGNΔ
001001,A0

//ΔASSGNΔ
002001,A1

//ΔEND

ΔTRNSΔLIB=COB,
I =SYSUT3,
V=001001

ΔTRNSΔ
I=NONE,
V=002001,ABS

ΔEND
```

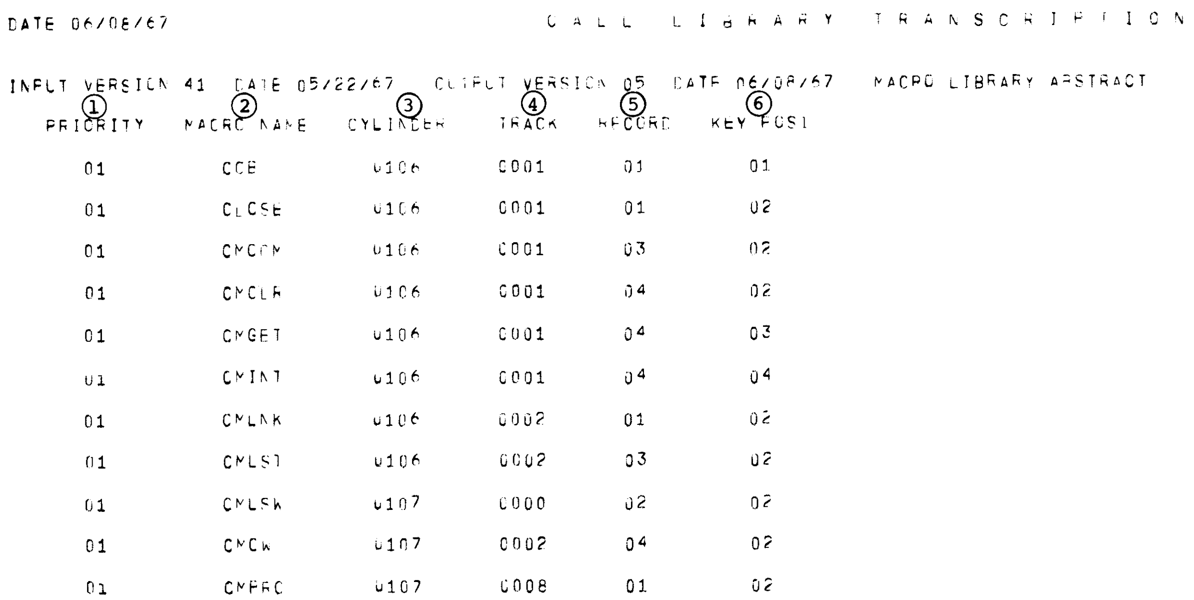
Routine Parameters
Take abstract of all Libraries on Random Access Volume 002001.
Transcribe COBOL Library from SYSUT3 tape.
Random Access Devices
Device Assignments
Input Call Library Tape

Device Assignments

◆ Under Executive or Monitor Control:

SDN	Device Type	Remarks
SYSIPT	Card reader or magnetic tape.	Input parameters.
SYSLST	Printer or magnetic tape.	Output abstract listing.
SYSUT2	Magnetic tape.	Input call library tape. (Required when preset transcription is specified.)
SYSnnn or ssssss	Magnetic tape.	Alternate input containing libraries to be transcribed.
vvvvvv (RA device volume serial number.)	Disc or drum.	Input and output. The volume serial number of the random access device to which the TOS call library is transcribed to, or from which an abstract is to be taken.

Macro Library Abstract



```
DATE 06/08/67                         C A L L   L I B R A R Y   T R A N S C R I P T I O N                       PAGE    1

INPUT VERSION 41  DATE 05/22/67   OUTPUT VERSION 05  DATE 06/08/67   MACRO LIBRARY ABSTRACT
        ①              ②              ③            ④           ⑤           ⑥
   PRIORITY     MACRO NAME     CYLINDER      TRACK      RECORD     KEY POSI

      01            CCB          0106         0001        01          01

      01            CLOSE        0106         0001        01          02

      01            CMCOM        0106         0001        03          02

      01            CMCLR        0106         0001        04          02

      01            CMGET        0106         0001        04          03

      01            CMINT        0106         0001        04          04

      01            CMLNK        0106         0002        01          02

      01            CMLST        0106         0002        03          02

      01            CMLSW        0107         0000        02          02

      01            CMCW         0107         0002        04          02

      01            CMPRC        0107         0008        01          02
```

① Macro priority section

② Macro name

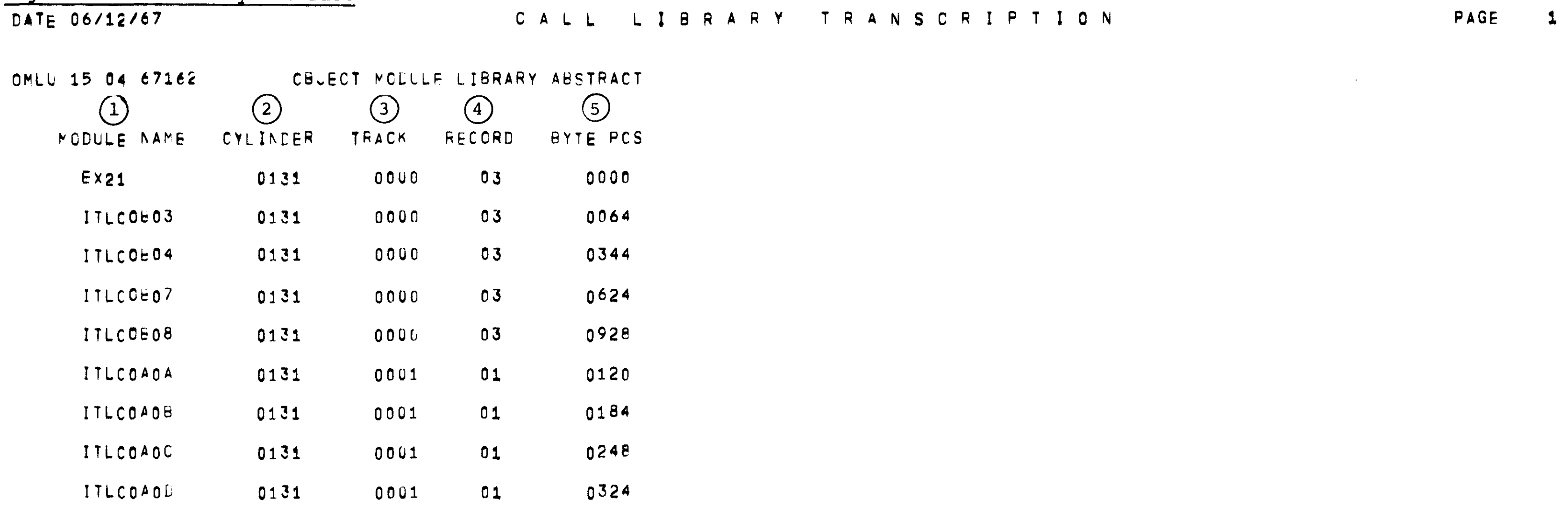
③ Beginning cylinder number

④ Beginning track number

⑤ Beginning record number

⑥ Key position within record that points to first byte of the macro

Object Module Library Abstract



```
DATE 06/12/67                        C A L L   L I B R A R Y   T R A N S C R I P T I O N                        PAGE    1

OMLU 15 04 67162        OBJECT MODULE LIBRARY ABSTRACT
        ①               ②           ③         ④          ⑤
    MODULE NAME     CYLINDER     TRACK     RECORD     BYTE PCS

      EX21            0131        0000       03         0000
      ITLC0B03        0131        0000       03         0064
      ITLC0B04        0131        0000       03         0344
      ITLC0B07        0131        0000       03         0624
      ITLC0B08        0131        0000       03         0928
      ITLC0A0A        0131        0001       01         0120
      ITLC0A0B        0131        0001       01         0184
      ITLC0A0C        0131        0001       01         0248
      ITLC0A0D        0131        0001       01         0324
```

① Module name

② Beginning cylinder number

③ Beginning track number

④ Beginning record number

⑤ Position within the record of the first byte of the module

Device Assignments *(Cont'd)*

Table 8-1. CLTR Routine Device Options

Options Devices	Transcribe	Abstract	Transcribe & Abstract	Compute Device Storage (CDS)
SYSIPT	X1	X	X	X
SYSLST	X	X	X	
SYSUT2 (Input)	X2		0	0
SYSnnn or ssssss (alternate input)	0		0	0
vvvvvv (Random access volume serial number.) Input and output	X	X	X	X

where:

X - Required.
X1 - Required unless preset options are specified.
X2 - Required when preset options are used.
0 - Optional.

Notes:

1. The volume serial number of the random access device must be the same as the serial number used in the TRNS parameter.
2. For seven-level tapes, the user must insure that the pack/unpack mode is on.

9. COMMUNICATIONS ROUTINES

CARD CONVERT (CDCONV)

General Description

◆ This routine is used in conjunction with the Communications Test Package (TSTCUP).

The Card Convert (CDCONV) provides the user the ability to keypunch a card in a graphic (EBCDIC) representation of hexadecimal values, introduce it to this program, and obtain the data punched in hexadecimal representation. This facility allows easier TSTCUP parameter and data card preparation. In addition, it will float certain addresses specified in the parameter cards by a predefined value.

Preset Functions

The standard function of this program is to accept input, which contains graphic representation of TSTCUP parameter cards (pseudo-parameter), and produce the parameter cards punched in hexadecimal. This provides easier preparation of parameter cards using graphics rather than the many multiple-punch combinations required for most hexadecimal values.

Optional Functions

None.

Input

◆ Input to this routine consists of the following:

1. CDCONV parameter cards.
2. TSTCUP pseudo-parameter cards, and
3. TSTCUP pseudo-data cards.

Output

◆ This routine produces output punched in hexadecimal format. A TSTCUP 'END DATA' parameter is generated each time the DATA function is terminated. A TSTCUP 'TERM' parameter is generated upon recognition of the END parameter.

Equipment Configuration

Required

◆ 70/35, -45, or -55 Processor.

Model 70/97 Console Typewriter.

Model 70/237 Card Reader.

Model 70/234 or 70/236 Card Punch.

Optional

◆ Magnetic Tape Unit(s), Models 70/432, 70/442, or 70/445 may be substituted for input (card reader) and/or output (card punch).

Routine Parameters - General

◆ The following parameters are recognized by Card Convert.

FLOAT Parameter

This parameter contains an address which represents the memory location into which TSTCUP would be loaded.

CONTROL Parameter

This parameter signals the start of paired input TSTCUP parameter cards to be converted.

DATA Parameter

This parameter signals the start of paired input data cards to be converted.

END Parameter

This is the final input parameter to Card Convert.

Routine Parameters - Detailed

FLOAT Parameter

◆ This parameter specifies the TSTCUP float factor to be applied to certain fields of the input data (in the pseudo-parameter), which are punched into the output (TSTCUP parameters) in hexadecimal format. This parameter must be the first card in the input deck.

The float factor is applied to the following fields in the appropriate GET and PUT TSTCUP parameter cards:

1. User Storage Area Address.
    a. in the GET User Interface Area
    b. in the PUT User Interface Area
2. Routing information storage area address in the PUT User Interface Area.

In addition, the addresses in the TSTCUP's SNAPSHOT and PATCH parameters are also floated.

This allows the user to code these addresses relative to the beginning of TSTCUP, disregarding where CUP will be loaded. Once the load address is determined or if it is changed, this float factor can be (re) applied by passing the input (original pseudo-parameters) through the routine, noting the load address for CUP as the float factor in this parameter. The method for specifying these addresses is explained under Consideration for Use.

FLOAT Parameter (Cont'd)

Column	Operand
1	/ (Slash)
2	/ (Slash)
3	(Space)
4-8	FLOAT
9	(Space)
10-15	(Float Factor)

CONTROL Parameter

◆ This parameter initiates the conversion of paired input cards (pseudo-parameters) into hexadecimal TSTCUP parameter cards. This parameter must precede the set of cards to be converted and may be used any number of times. It may be interspersed with DATA parameters. Conversion continues until another parameter card is encountered. The format of the pseudo-parameter cards is described under Considerations for Use.

Column	Operand
1	/ (Slash)
2	/ (Slash)
3	(Space)
4-10	CONTROL

DATA Parameter

◆ This parameter initiates the conversion of all subsequent paired input cards (pseudo-data) into hexadecimal TSTCUP data. The parameter must precede the set of cards to be converted and may be used any number of times. It may be interspersed with CONTROL parameters. Conversion continues until another parameter card is encountered. A TSTCUP 'END DATA' parameter is generated each time the DATA function terminates. The format of pseudo-data cards is described under Consideration for Use.

Column	Operand
1	/ (Slash)
2	/ (Slash)
3	(Space)
4-7	DATA

END Parameter

◆ This parameter must always appear as the last card in the input deck. When encountered, the routine will generate the TSTCUP 'TERM' parameter.

Column	Operand
1	/ (Slash)
2	/ (Slash)
3	(Space)
4-6	END

Considerations for Use

TSTCUP Pseudo-Parameter Cards

◆ All operands for TSTCUP parameters must be punched in hexadecimal format. Because many hexadecimal values consist of multiple-punch combinations, this routine allows the user to prepare the operands in a graphic representation of the hexadecimal value.

The value is represented on two punch cards in the applicable column. The input cards are always paired. For example, to create a BRANCH parameter whose card code (column 1) is $(00)_{16}$, punch an EBCDIC 0 in two successive cards in column 1. The program will produce an output card containing card punches 12,0,9,8,1 in column 1. The value $(4B)_{16}$, would be created by punching a 4 (single EBCDIC punch) in the first card of the pair, and a B (EBCDIC punch 12,2) in the second card. This would produce the card punches 12,8,3 (a period) in the appropriate column of the punched card.

Column 1 of each pair must contain a valid TSTCUP card code. Each set of cards (any number of paired cards) to be converted must be preceded by a CONTROL parameter. The logical number of the card (1 or 2) pair must appear in column 76. Also, the two cards of a pair must be sequential. For example:

Card Column	1	2	3	4	5	◄──────►	76	◄──────►
Card #1	0		4	2	F	....	1	....
Card #2	0		B	C	E	....	2	....

The only valid input characters are:

0, 1, 2, 3, 4, 5, 6, 7, 8, 9, A, B, C, D, E, F

As was discussed, a float factor may be applied to certain address fields, if specified. The User Storage Area Address (in both GUI and PUI), the Routing information storage address (PUI), the PATCH operand and the SNAPSHOT operands will be affected depending on the following rules:

1. If the operand is blank, $(40)_{16}$ will be generated.
2. If the operand contains 0's, $(00)_{16}$ will be generated.
3. If the operand contains F's (EBCDIC), the float factor will be generated.
4. If the operand is any other valid character, the value plus the float factor will be generated.

TSTCUP Pseudo-Data Cards

◆ This routine will also convert pseudo-data cards to hexadecimal format. The cards are prepared in the same manner as the pseudo-parameters except columns 1 and 76 are not checked for card code and logical number, respectively. Each set of cards (any number of paired cards) must be preceded by a DATA parameter. Float Factors are not applied to data cards.

Example:

	Input Cards			Equals	Output Cards		
Card 1	Δ	Δ	Δ		40	40	40
Card 2	Δ	Δ	Δ				
Card 1	0	0	0		00	00	00
Card 2	0	0	0				
Card 1	F	F	F		ff	ff	ff
Card 2	F	F	F				
Card 1	g	g	g		hh	hh	hh
Card 2	g	g	g				

Where:

Δ = Space.

0 = EBCDIC Zero.

F = EBCDIC F

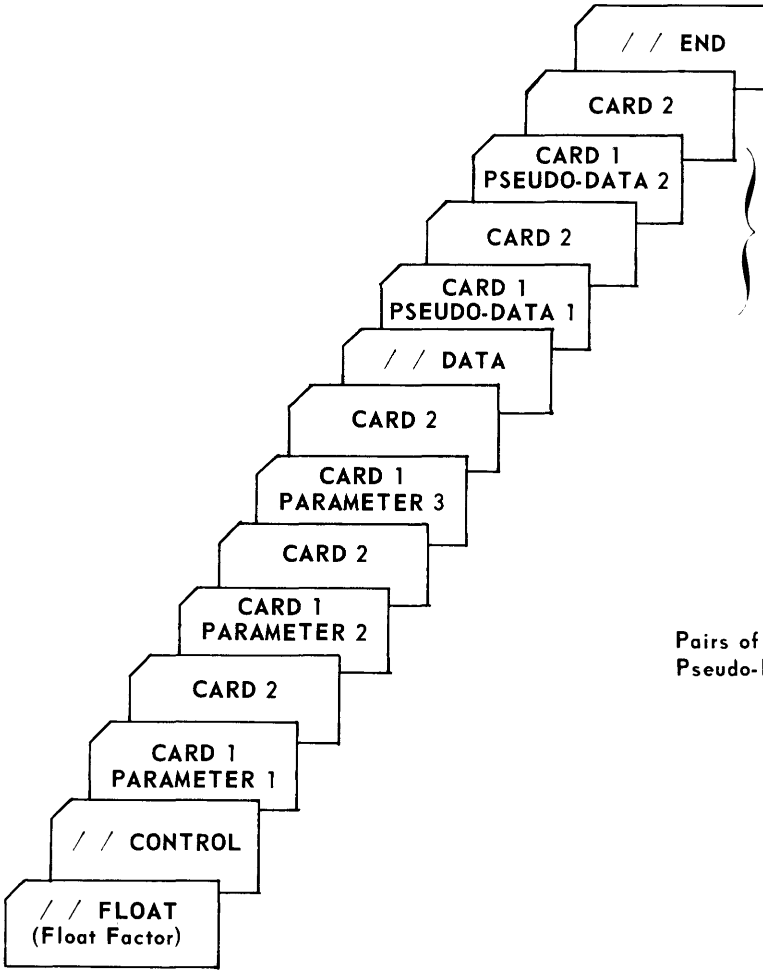
f = Float Factor.

g = Relative Value.

h = Relative Value Plus Float Factor.

Parameter Example

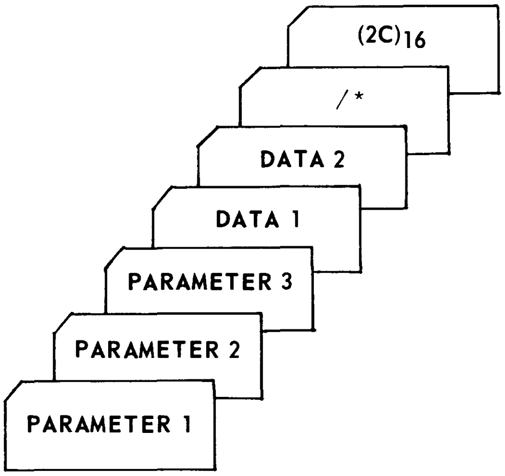
Card Convert Input



Pairs of TSTCUP Pseudo-Data

Pairs of TSTCUP Pseudo-Parameters

Card Convert Output



TERM Parameter (generated by CDCONV)

END DATA Parameter (generated by CDCONV)

Data in Hexadecimal Format

Parameters in Hexadecimal Format

Device Assignments

◆ Under Executive Control.

SDN	Device Type	Remarks
SYSRDR	Input Device.	Parameter Input.
SYSOPT	Output Device.	Parameter Output.

Operating Instructions

◆ This routine will execute under either the TOS or TDOS Executive. The symbolic device names are SYSRDR and SYSOPT. To initiate under Executive Control, the operator types in the proper load request. Refer to either TOS or TDOS Operators' Guide for a description of program initiation.

Message Typeouts

Printout	Explanation
7151 NO FLT CD	Float parameter is missing. Program terminates.
7152 INVAL PAR	Invalid parameter. Program terminates.
7153 CTL SEQ ERR	Pseudo-parameter cards are not in sequence (card #1 followed by a card #2). The card currently being processed is ignored. Cards are passed until a '1' card or a new parameter card is encountered. Normal processing is resumed.
7154 DAT SEQ ERR	An odd number of pseudo-data cards has been processed. The last card of the series is ignored and normal processing is resumed.

Minimum Memory Requirements

◆ The program requires approximately 1,830 bytes.

COMMUNICATIONS TEST PACKAGE (TSTCUP)

General Description

◆ The Communication User Test Program is a Communication User Program (CUP) which interacts with a Multichannel Communication Program (MCP) creating a communications environment.

The TSTCUP is used to simulate a user communications environment by providing the following functions:

System Initialization

Message Transmission

Exception Processing

Object Time Patching

Own Coding

Through the use of parameters, data cards, and own-coded modules, any or all of the above functions can be accomplished. These capabilities can also be employed to test Spectra 70/35-45-55 equipment configurations with data communications networks.

This program is designed so that user own-code modules and overlay segments (for CCM Memory loads) can be easily incorporated. All CMGETS and CMPUTS should be primed through use of TSTCUP parameters rather than own-coded. This will allow TSTCUP to maintain proper control of the user communication environment.

The CDCONV routine, page 9-1, will permit coding and punching of parameters using the graphic representation of EBCDIC values and will float addresses to suit the run time requirements of TSTCUP.

Preset Functions

None.

Optional Functions

All parameters are optional and need only be supplied for the various functions as desired. All operands are expressed in hexadecimal values, unless otherwise noted.

Input

◆ Input, in addition to the parameters and data cards, is communication environment dependent. Through the use of the Get function, TSTCUP provides complete message reception handling.

Output

◆ Output is communication environment dependent. Through the use of the Put function, TSTCUP provides complete message transmission. Through the verification and snapshot facility, complete system monitoring is possible.

Equipment Configuration

Required

◆ The minimum equipment required by the TDOS/MCS system is as follows:

Processor (Model E) 70/35, 45, or 55.

Console Typewriter, Model 70/97.

Random Access Controller, Model 70/551 with Record Overflow Feature 5512.

Input/Output Attachment, Feature 5501-1.

Disc Storage Unit, Model 70/564.

Magnetic Tape Controller, Model 70/472 or 473.

Magnetic Tapes, Model 70/432, 442, or 445 (three required; two must be nine-level).

Card Reader, Model 70/237*.

Printer, Model 70/242, 243, or 248**.

Communication Controller Multichannel, Model 70/668-11.

Buffer, Model 70/710 or 720.

Elapsed Time Clock, Feature 5002-35, 45 or 55.

Routine Parameters - General

◆ The following parameters are recognized by this routine:

BRANCH Function

This parameter permits the user to branch to a specified location within TSTCUP.

FILL DATA BANK Function

This parameter directs TSTCUP to read subsequent cards into consecutive memory in the predefined data bank.

*Magnetic tape may be substituted.

**The printer, or a magnetic tape substitute is not required if the SNAPSHOT option is not utilized.

Routine Parameters - General *(Cont'd)*

PLUG GUI Function

This parameter defines the values to be placed into the Get User Interface Area (in TSTCUP) and, optionally, the "branch to" point.

PLUG PUI Function

This parameter defines the values to be placed into the Put User Interface Area (in TSTCUP) and, optionally, the "branch to" point.

PLUG GUI and GET Function

This parameter defines the values to be placed into the Get User Interface Area (in TSTCUP); directs the program to issue a CMGET; and optionally, after the Get verifies the GUI.

PLUG PUI and PUT Function

This parameter defines the values to be placed into the Put User Interface Area (in TSTCUP); directs the program to issue a CMPUT; and optionally, after the Put verifies the PUI.

COMMENT/DECISION Function

This parameter allows comments to be typed or either of the following to be verified: (1) The System Notice Key and the contents of the System Notice, or (2) the Notification Word. Dependent upon the result of the verification, a conditional action is taken.

LOAD CCM Function

This parameter defines the (overlay) name of the CCM-Memory segment to be loaded and the CCM device number.

CONTROL Function

This parameter requests and controls snapshots of:

1. The GUI after each CMGET.
2. The GUI after a CMGET.
3. The PUI after each CMPUT.
4. The PUI after a CMPUT.
5. Any or all of the above conditions.

WAIT Function

This parameter directs TSTCUP to return control to the Executive.

Routine Parameters - General *(Cont'd)*

SNAPSHOT Function

This parameter defines a memory area to be "snapped."

TERMINATE Function

This parameter indicates the end of input parameters, and the program terminates.

PATCH Function

This parameter provides an object time "memory patch" function.

END DATA Function

This parameter indicates the end of a set of data cards.

Routine Parameters - Detailed

PATCH Parameter

◆ This parameter directs the program to move the character string starting in card column 6 and terminated by the character pair /* to the memory locations starting at the absolute location specified.

Column	Operand
1	30
3-5	Absolute address of patch.
6-	Patch characters . . . /*

BRANCH Parameter

◆ This parameter directs the program to branch to the relative address specified. The address specified is added to the float factor for TSTCUP and a branch is executed to that calculated address.

The branch address may be to any point within TSTCUP, be it either TSTCUP or own-coding (through the use of a program code). Certain predefined points within TSTCUP and their purpose are described under Program Considerations as well as the linkage points for own-code modules. Program codes are also discussed in the same section.

Column	Operand
1	00
77	Program Code
78-80	Relative Branch Address

FILL DATA BANK Parameter

◆ This parameter directs the program to read data cards from READER into a common storage area in TSTCUP. The program will continue to read cards until the END DATA parameter card is encountered. The program then reads the next parameter card.

The data bank, whose tag is IEDBANK, is the first 3,200 bytes in the program. The relative address is $(000000)_{16}$ to $(000C7F)_{16}$.

Cards will be stored by one of two methods. If the float field is undefined (blanks), cards will be stored in consecutive ascending memory positions starting with $(000000)_{16}$. Each card read will fill 80 memory positions regardless of the amount of data punched in the card.

If the float field is defined (must not exceed $(000C30)_{16}$ or $(3120)_{10}$), the card(s) will be stored starting at the defined relative position within the data bank. Data must not be read beyond the data bank.

Care should be taken that succeeding data loads do not overlay previously loaded data if that data is to be preserved.

Column	Operand
1	04
3-6	Float Field - blank or relative data bank address.

Examples:

1. Load data cards starting at the beginning of the data bank.

Column	Hex. Value	Card Punch
1	04	12,9,4

2. Load data cards starting in the 80th position in the data bank.

Column	Hex. Value	Card Punch
1	04	12,9,4
3	00	12,0,9,8,1
4	00	12,0,9,8,1
5	00	12,0,9,8,1
6	50	12,

END DATA Parameter

◆ This parameter denotes the end of a set of data cards when using the FILL DATA BANK function. This parameter terminates the FILL DATA BANK function. The program reads the next parameter.

Column	Operand
1	/ (Slash)
2	* (Asterisk)

PLUG GUI Parameter

◆ This parameter directs the program to insert into the GUI the values defined in the applicable fields in this card. Only nonblank characters are placed into the GUI. A branch is then taken if defined by the program code and the branch address fields.

The Get User Interface Area is initialized to blanks by TSTCUP. The GUI Area is not reinitialized by TSTCUP during object execution.

Column	Operand
1	08
10	User Request Code
11-13	User Storage Area Address
14-15	User Storage Area Size
77	Program Code
78-80	Branch Address

Note:

If the program code is omitted, TSTCUP will read the next card. If the program code is 0 (hexadecimal), and the branch address is omitted, TSTCUP will read the next card. If the program code is specified (other than 0) and the branch address is omitted, TSTCUP will branch to the beginning of the own-code module.

Example:

Set-up the GUI for the Get of an unsolicited type-in into the data bank, assuming TSTCUP is loaded at 50B8, then branch to read the next card.

Column	Hex. Value	Card Punch
1	08	12,9,8
10	03	12,9,3
11	00	12,0,9,8,1
12	50	12
13	B8	12,11,0,8
14	00	12,0,9,8,1
15	50	12

PLUG PUI Parameter

◆ This parameter directs the program to insert into the PUI the values defined in the applicable fields in this card. Only nonblank characters are placed into the PUI. A branch is then taken if defined by the program code and the branch address fields.

The Put User Interface Area is initialized to blanks by TSTCUP. The PUI Area is not reinitialized by TSTCUP during object execution. Special attention should be given to those items in the PUI which must be zeroed if not specified (that is, delete character byte count).

Column	Operand
1	0C
14	Put key.
15-17	User storage area address.
18	Translation control indicator.
19	Block count.
20-21	User storage area size or block size.
22	Action entry subfunction key.
23	Line number.
24-25	Routing - Single addressee or polled station. TSC or tape device installation mnemonic.
26	Queuing priority.
27-29	Routing information storage area address.
30-31	Routing information storage area size.
32-33	Delete character byte count.
77	Program code.
78-80	Branch address.

Note:

If the program code is omitted, TSTCUP will read the next card. If the program code is 0 (hexadecimal), and the branch address is omitted, TSTCUP will read the next card. If the program code is specified (other than 0) and the branch address is omitted, TSTCUP will branch to the beginning of the own-code module.

PLUG PUI Parameter (Cont'd)

Examples:

1. Set-up the PUI to activate a single CCM $(15)_{10}$, then branch to a PR (Wait With Nothing to do) at $(000CF4)_{16}$ in TSTCUP.

Column	Hex. Value	Card Punch
1	0C	12,9,8,4
14	FF	12,11,0,9,8,7
22	14	11,9,4
23	0F	12,9,8,7
77	00	12,0,9,8,1
78	00	12,0,9,8,1
79	0C	12,9,8,4
80	F4	4

2. Set-up the PUI to initiate a transmission of a dedicated message to a single station video device on line $(22)_{10}$, then read the next card. The message resides in the data bank and is 500 bytes long.

Column	Hex. Value	Card Punch
1	0C	12,9,8,4
14	01	12,9,1
15	00	12,0,9,8,1
16	50	12
17	B8	12,11,9,8
18	08	12,9,8
20	01	12,9,1
21	F4	4
23	16	12,11,9,8,1

PLUG GUI and GET Parameter

♦ This parameter directs the program to insert into the GUI the values defined in the applicable fields (columns 10-15) in this card, then issue a CMGET. Only nonblank characters are placed in GUI. If columns 40-49 are not blank cols. 40-59 are compared to the GUI after the GET and, if unequal, a snapshot of the GUI is printed. A branch is then taken if defined by the program code and the branch address fields.

PLUG GUI and GET Parameter (Cont'd)

Column	Operand
1	10
10	User request code.
11-13	User storage area address.
14-15	User storage area size.
40	User request code.
41-43	User storage area address.
44-45	User storage area size.
46-47	User storage area index.
48	Status flags.
49	System notice key.
50	Message indicators.
51	Control indicators.
52	Line number.
53	Language type.
54-55	Transmitter start code.
56-59	System notice.
60-63	Time of get.
64-67	Time of arrival.
77	Program code.
78-80	Branch address.

Note:

If the program code is omitted, TSTCUP will read the next card. If the program code is 0 (hexadecimal), and the branch address is omitted, TSTCUP will read the next card. If the program code is specified (other than 0) and the branch address is omitted, TSTCUP will branch to the beginning of the own-code module.

Example:

Set-up the GUI and issue a Get Entire-Dynamic into the first byte of the data bank. Do not verify the results and then branch to read the next card.

Column	Hex. Value	Card Punch
1	10	12,9,8,2
10	07	12,9,7
11	00	12,0,9,8,1
12	50	12
13	B8	12,11,0,8
14	03	12,9,3
15	E8	0,8

PLUG PUI and PUT Parameter

♦ This parameter directs the program to insert into the PUI the values defined in the applicable fields (columns 14-33) in this card and then issue a CMPUT. Only nonblank characters are placed in the PUI. If columns 42-49 are not blank, the PUI is compared after the Put to the information defined in columns 40-63 and, if unequal, a snapshot of the PUI is printed. A branch is then taken if defined by the program code and the branch address fields.

Column	Operand
1	14
14	Put key.
15-17	User storage area address.
18	Translation control indicator.
19	Block count.
20-21	User storage area size or block size.
22	Action entry subfunction key.
23	Line number.
24-25	Routing - Single addressee or polled station TSC or tape device installation mnemonic.
26	Queueing priority.
27-29	Routing information storage area address.
30-31	Routing information storage area size.
32-33	Delete character byte count.
40-43	Notification word.
44	Put key.
45-47	User storage area address.
48	Translation control indicator.
49	Block count.
50-51	User storage area size or block size.
52	Action entry subfunction key.
53	Line number.
54-55	Routing - Single address or polled station.TSC or tape device installation mnemonic.
56	Queueing priority.
57-59	Routing information storage area address.
60-61	Routing information storage area size.
62-63	Delete character byte count.
77	Program code.
78-80	Branch address.

Note:

If the program code is omitted, TSTCUP will read the next card. If the program code is 0 (hexadecimal), and the branch address is omitted, TSTCUP will read the next card. If the program code is specified (other than 0) and the branch address is omitted, TSTCUP will branch to the beginning of the own-code module.

COMMENT/ DECISION Parameter

◆ This parameter will permit the following:

1. Any comment to be typed out upon processing of the parameter, then branching to the address specified, or;

2. Compare the system notice key and system notice. The compare can be for either equality or inequality. If the compare is true, a branch is taken to the address specified. If the compare is false, the program will terminate, or;

3. Compare the notification word. The compare can be for either equality or inequality. If the compare is true, a branch is taken to the address specified. If the compare is false, the program is terminated.

Column	Operand
1	18
2	Nonblank character to indicate this card contains a comment to be typed.
3	Nonblank character to indicate this card contains a value to be compared for equality.
4	Nonblank character to indicate this card contains a value to be compared for inequality.
5-39	Up to a 35 character message to be typed. Column 2 also must be specified.
40-71	System notice or notification word values to be compared. Each card column represents a bit position of the value. Thus, the acceptable characters for defining the status of either item is 0 or 1. Either column 3 or 4 also must be specified.
73	System notice key. Also indicates that a system notice is to be verified.
77	Program code.
78-80	Branch address.

Note:

If the program code is omitted, TSTCUP will read the next card. If the program code is 0 (hexadecimal), and the branch address is omitted, TSTCUP will read the next card. If the program code is specified (other than 0) and the branch address is omitted, TSTCUP will branch to the beginning of the own-code module.

Column 40 is bit 2^0 for the notification word. Columns 40-47 represent byte 0 of the system notice.

LOAD CCM Parameter

◆ This parameter specifies the load name of the CCM memory overlay and the device address of the CCM to be loaded.

The segment defined in this parameter will then be called into memory by TSTCUP (using the Load Program Overlay macro) then loaded (by the CMCCM macro) into the device specified. If the load is not successful, the message "CCM XX DID NOT LOAD" (XX = CCM device address) will be typed, and TSTCUP will branch to IEREAD (read a card). If the load is successful, the message "CCM XX LOADED O.K." (XX = CCM device address) will be typed and then the program will branch to IEREAD (read a card). There is no restriction on the number of times this parameter can be used (dependent upon the number of CCM's to be loaded) nor when the parameter may be used.

Column	Operand
1	1C
3-8	Load name.
9	Device address.

CONTROL Parameter

◆ This parameter defines the conditions under which, after an I/O operation, the program should or should not snapshot the User Interface Area within TSTCUP. The verification function of the PLUG GUI AND GET and PLUG PUI AND PUT parameters are only checked if the control function for this I/O operation is not requested.

Any of the following functions can be indicated by any nonblank character in the appropriate column:

1. Snapshot of the GUI after every CMGET (column 4), or
2. Snapshot of the PUI after every CMPUT (column 5), or
3. Both of the above (columns 4, 5).

If a GET or PUT parameter contains one of the five program codes, TSTCUP will first check to determine if the control parameter requested snapshots for this specified program code before branching to the own-code module requested by the program code.

CONTROL Parameter (Cont'd)

Column	Operand
1	20
4	Snap GUI after every CMGET.
5	Snap PUI after every CMPUT.
11	Section 1 - Program Code 04.
12	- Snap GUI.
13	- Snap PUI.
19	Section 2 - Program Code 08.
20	- Snap GUI.
21	- Snap PUI.
27	Section 3 - Program Code 0C.
28	- Snap GUI.
29	- Snap PUI.
35	Section 4 - Program Code 10.
36	- Snap GUI.
37	- Snap PUI.
43	Section 5 - Program Code 14.
44	- Snap GUI.
45	- Snap PUI.

Examples:

1. To snapshot the respective User Interface Areas with TSTCUP after all CMGETS and CMPUTS:

Column	Hex. Value	Card Punch
1	20	11,0,9,8,1
4	00	12,0,9,8,1
5	00	12,0,9,8,1

2. To snapshot the following:

   a. The GUI after each Get with a Program Code - 04;

   b. The PUI after each Put with a Program Code - 10;

   c. Both GUI and PUI after a Get or Put, respectively, with a Program Code - 0C:

Column	Hex. Value	Card Punch
1	20	11,0,9,8,1
11	00	12,0,9,8,1
12	00	12,0,9,8,1
27	00	12,0,9,8,1
28	00	12,0,9,8,1
29	00	12,0,9,8,1
35	00	12,0,9,8,1
37	00	12,0,9,8,1

SNAPSHOT Parameter

◆ This parameter defines the memory limits to be snapped. A snapshot is effected each time this parameter is encountered. After the snapshot is printed, the program reads the next card.

Column	Operand
1	28
3-5	Absolute address of the left-hand-end of the area to be snapped.
6-8	Absolute address of the right-hand-end of the area to be snapped.

Example:

After issuing a Get Status to obtain the threshold status information, snap the 18-byte area that starts at $(5100)_{16}$.

Column	Hex. Value	Card Punch
1	28	0,9,8
3	00	12,0,9,8,1
4	51	12,11,9,1
5	00	12,0,9,8,1
6	00	12,0,9,8,1
7	51	12,11,9,1
8	11	11,9,1

For a complete description on the use of the SNAPSHOT function, refer to the TDOS 70/35-45-55 Operators' Guide, Section 7, Snapshot.

TERMINATE Parameter

◆ This parameter indicates the end of input parameters. The program is terminated immediately (by way of the TERM macro).

Column	Operand
1	2C

WAIT Parameter

◆ This parameter causes TSTCUP to issue a WAIT SVC (PR).

Column	Operand
1	24

Considerations for Use

Structure and Organization

◆ This program is a single program load. Included in this load may be user own-code modules bound into the root segment at Linkage Editor time. TSTCUP requires a minimum of 9,100 bytes which include the data bank and Snapshot. This memory is in addition to the memory required for MCP.

If this routine is to load a CCM(s), the CCM memory segment(s) must be linked as an overlay module(s) following the TSTCUP segment. Also, the overlay(s) must be in the same region.

Example of a Monitor job stream to assemble a CCM memory load and link it with the TSTCUP modules on SYSLIB.

```
//_STARTM
...Monitor assignments...
//_JOB
//_ASSMBL
...CCM Memory...
//_LNKEDT
_PROG_TSTCUP
_NOCTL
_NCAL
_LET
_INCLUDE_(ITCDCUP)
_OVERLAY_N1
_INCLUDE_(IEOVLY)
_OVERLAY_N1
_INCLUDE_SYSUT1
//_ENDMON
```

The node point name (N1) is arbitrary but must be identical for all overlays included in TSTCUP.

If own-code modules are to be used in TSTCUP, they must be included between the module ITCDCUP and the module IEOVLY.

Program Considerations

Own-Coding

◆ The program is designed to allow the user to incorporate own-coding with a minimum of effort. Up to five separate own-code modules can be bound with TSTCUP. Each module must contain the tag of one of the defined externs (EXTRN) in TSTCUP which is unique and is synonymous with a program code. Therefore, to transfer control to that module the applicable program code and branch address is used in the:

1. BRANCH parameter.
2. PLUG GUI parameter.
3. PLUG PUT parameter.
4. PLUG GUI AND GET parameter.
5. PLUG PUI AND PUT parameter, or
6. COMMENT/DECISION parameter.

The following list defines the own-code EXTRN names and their applicable program codes.

TESTCUP EXTRN	Program Code
IE0004	04
IE0008	08
IE000C	0C
IE0010	10
IE0014	14

For example, if a user own-code module is tagged IE0004, control can be transferred to that module using any one of the above named parameter cards. The use of a BRANCH parameter with a program code of 04 would cause TSTCUP to branch to the beginning of the own-code module. If a branch address is specified, TSTCUP would transfer control to that relative address within the module.

The following list defines TSTCUP Entries, their relative address and purpose.

TSTCUP Entry	Relative Address	Purpose
IEDBANK	000000	Data Bank.
IEREAD	000CCC	Read the next parameter card.
IEGUI	000C90	Get User Interface Area.
IEPUI	000CB0	Put User Interface Area.
IEWAIT1	000CF4	Wait with Nothing to do.

The relative addresses of IEREAD and IEWAIT1 can be used as branch addresses with a program code of 00.

Own-Coding (Cont'd)

Control can be transferred from an own-code module to either of the appropriate TSTCUP entries (that is, IEREAD or IEWAIT1). For example, control could have been transferred to a module after a Get. The user could process the message and prepare a response. At this time, the user is ready for a Put which is the next parameter. Thus, the module would branch to IEREAD and TSTCUP would read the next card.

Registers 1 and 2 may be used exclusively for own-coding. Registers 8, 9, and 11 are also available; however, these registers are used by TSTCUP as work registers.

The Put User Interface Area (IEPUI) includes an eight-byte (IEPUI+24) for use by MCS SNAPSHOT containing User Station Sequence Number Table Address and Alternate area. The PATCH function can be used to change these eight-bytes.

Operating Instructions

Program Initiation

◆ For a complete explanation of program initiation, refer to the TDOS 70/35-45-55 Operators' Guide, Section 2, Program Initiation. Once loaded, TSTCUP will read the first card in the parameter device.

Message Typeouts

At Load CCM Time

Printout	Explanation
7105 CCMXX LOADED O.K.	CCM memory specified in the LOAD CCM parameter has been successfully loaded into the CCM designated.
7104 CCMXX DID NOT LOAD	The program is unable to load the CCM memory module specified in the LOAD CCM parameter into the designated CCM.

where XX - Common Device Address.

At Object Patch Time

Printout	Explanation
7102 NO_ TERMINAL_ /* _ ON_ PATCH_ CARD	The two-character sequence cannot be found on a PATCH card (code 30).

Other Typeout

Printout	Explanation
7103 1ST COLUMN OF PREV. CD. GREATER THAN X30 OR NOT MULT4.	The parameter read is not valid. The card code (column 1) must be between hexadecimal values 00 and 30 and a multiple of 4.

Device Assignments

◆ Under Executive Control.

SDN	Device Type	Remarks
READER	Card reader or Magnetic tape substitute.	Parameter input.
SNAPOP	Printer or magnetic tape substitute.	SNAPSHOT output.

MULTICHANNEL COMMUNICATIONS DISC FORMATTING ROUTINE (MCDF)

General Description

◆ The Multichannel Communications Disc Formatting (MCDF) routine pre-formats the area of the 70/564 Disc that is used by the Multichannel Communications Program (MCP) for dynamic buffering (dynamic core storage used for the intermediate storage of user message segments). Execution of the MCDF routine is not required if the user has selected the direct access option of MCP, or if dedicated core storage only is used within the communications environment.

The MCDF routine operates under control of the Tape Disc Operating System (TDOS) Control System. The output formats of MCDF are used by MCP and conform to the system standards for an unlabeled data file.

Prior to execution to MCDF, the Random Access Volume Initializer (RAINIT) must be executed to prepare and format the 70/564 Disc Unit(s) used by MCP. (See Section 3.) The Random Access Storage Allocator (RAALLR) must be executed to reserve the disc storage to be used by MCP. (See Section 7.)

Detailed Description

◆ The Random Access Volume Initializer formats the random access volume in the following manner:

1. A service analysis is performed by writing to.and reading from each track. If a defective track is detected, an alternate track in the volume is assigned.
2. A Home Address record and a Track Descriptor record are created and written at the beginning of each track.
3. Records 1 and 2 of track 0, volume 0 are reserved for the system.
4. A Standard Volume label is created and written as record 3 of track 0, volume 0.
4. A dummy volume Table of Contents (VTOC) is created for the volume. This table is subsequently used to contain a directory of all files stored in the volume, the boundaries of each file, and the available alternate track areas within the volume.

Detailed Description *(Cont'd)*

The Random Access Storage Allocator reserves storage for a file on a random access volume by entering the name and limits of the file in the VTOC. The file name entered in the DLAB parameter must be COMDISC and the entire file must be entered as one contiguous area (that is, one extent).

After the volume has been initialized by way of RAINIT and the file COMDISC has been allocated by way of RAALLR, the Multichannel Communications Disc Formatting routine is loaded into main memory with the following console typein:

```
E LOD MCDF,,,,,xxxxx
```

After MCDF has been loaded, the assignment of the 70/564 Disc Storage Unit to be used for the Multichannel Communications Program (MCP) is requested with the console typewriter. After device assignment is completed MCDF performs the following:

1. The VTOC of the assigned disc is searched until the file COMDISC is located or until the entire VTOC has been searched.

   a. If the file COMDISC is found, its extent is stored in main memory.

   b. If the file COMDISC is *not* found, the message FILE NAME COMDISC NOT FOUND is typed out and the program is terminated.

2. The message CELL SIZE is typed out and the program awaits a response. The response is XXXX, where XXXX is from one- to four-decimal digits representing the dynamic cell size (less header) that is to be used by MCP; XXXX must be the same value as the second operand of the CMBUF macro (see TDOS MCS Reference Manual, 70-00-612).

   The dynamic cell size, XXXX, may not be less than 4 or greater than 3534. If an error is found in the response typein, the message INVALID PARAMETER, RETYPE is typed out and the program again awaits a response.

3. Based on the cell size, the maximum number of records that can be written to a track is computed using the following formula:

$$Q + R = \frac{3617 + 0.049(DL)}{62 + 1.049(DL)}$$

   where:

   Q = maximum number of records per track.

   R = residue of track after last record.

   3617 = remainder of a track after the track descriptor record.

   DL = cell size + 16.

Detailed Description *(Cont'd)*

4. A matrix (203x10) is constructed in main memory to designate each track that has been assigned to an alternate track.

5. The program loops through the allocated area updating the Track Descriptor Record (R0) and writing Q foundation records to each track except the first track of the allocated area. Each foundation record has a key length field set equal to zero (KL = 0) and a data length field set equal to the cell size + 16 (DL = XXXX + 16) in binary.

6. The last record of the last track is an EOF record (DL = 0).

7. If an alternate track is detected, a bit is set in the alternate track matrix. This matrix is used by MCP at object time to bypass defective tracks.

8. The MCDF-MCP coordination information (including the alternate track matrix) is written to the first allocated track and the program terminates. If the first track is defective, an alternate track is used.

This program may be rerun to change cell size without rerunning the Random Access Volume Initializer or Storage Allocator.

Console Typewriter Messages

◆ Listed below are the messages initiated by the MCDF routine and the required responses:

Typeout	Response
CELL SIZE	XXXX
INVALID PARAMETER RETYPE	XXXX
FILE NAME COMDISC NOT FOUND	No response required - program is terminated.

First Track Format

Record Number 1

Byte No.	Meaning
0,1	Number of allocated tracks minus one.
2,3	Number of defective tracks.
4,5	Number of records per track (Q).
6,7	Cell size (XXXX + 16).

Record Number 2

Byte No.	Meaning
0-253	Alternate track matrix.

MULTICHANNEL COMMUNICATIONS DISC SNAPSHOT FORMATTER (MCDSF)

General Description

◆ The Multichannel Communications Disc Snapshot Formatter (MCDSF) utility routine pre-formats the area of the 70/564 Disc; that is, the area used by the Multichannel Communications Program (MCP) for the Snapshot to Backup option. The COMSNAP file area on the disc must be formatted for storage of the Snapshot information.

The MCDSF routine operates under control of the Tape Operating System (TOS) or Tape Disc Operating System (TDOS) Control System. The output formats of MCDSF are used by MCP and conform to the systems standards for an unlabeled data file.

Prior to the execution of MCDSF, the Random Access Volume Initializer (RAINIT) must be executed to prepare and format the 70/564 Disc Unit(s) used by MCP. The Random Access Storage Allocator (RAALLR) must be executed to reserve the disc storage to be used by MCP. (See Sections 3 and 7 and the TOS Utilities Routines Manual, No. 70-35-302, for a description of RAINIT and RAALLR routines.)

The Random Access Volume Initializer formats the random access volume and is described in the Multichannel Communications Disc Formatting Routine (MCDF).

Detailed Description

◆ The user determines the disc area for utilization by the Snapshot Program and runs the Random Access Volume Initializer Program against this area. The Random Access Storage Allocator Program must be run to register the file in the Volume Table of Contents and reserve storage area on the disc. COMSNAP must be assigned as the file name when running this program.

The MCDSF program writes one record consisting of a 4-byte key of zeros and a 2,800-byte data field of zeros to each nondefective track assigned to the Snapshot function.

This program constructs a matrix in main memory of two bytes per cylinder to reflect the status of each track within each cylinder. This matrix is written to the first allocated track as record number 2. The eight bits of the first byte correspond to the first eight tracks (0-7) of the cylinder. The first two bits of the second byte correspond to tracks 8 and 9 of the cylinder. The status of a track is either defective or nondefective. A 1 in the track bit within the matrix denotes a defective track; a 0 in the track bit within the matrix denotes a nondefective track. The matrix is read from the disc when the Snapshot function is initialized by MCP. The matrix is used in assigning tracks for writing the Snapshot information.

Detailed Description *(Cont'd)*

If the file parameters remain unchanged, this MCDSF routine does not need to be rerun unless the user wishes to change the location or size of the disc area assigned to the Snapshot function.

After the volume has been initialized by the RAINIT and the file COMSNAP has been allocated by the RAALLR, the Multichannel Communications Disc Snapshot Formatting routine is loaded into main memory by the following console typein:

```
E LOD MCDSF
```

After MCDSF has been loaded, the assignment of the 70/564 Disc Storage Unit to be used for the Multichannel Communications Program (MCP) is requested by the console typewriter. This utility routine operates with or without Monitor control. After device assignment is completed MCDSF performs the following:

1. The VTOC of the assigned disc is searched until the file COMSNAP is located or until the entire VTOC has been searched.

   a. If the file COMSNAP is found, the cylinder assignment is stored in main memory.

   b. If the file COMSNAP is not found, the message FILE NAME COMSNAP NOT FOUND is typed out and the program is terminated.

2. The program loops through the allocated area updating the Track Descriptor Record (R0) and writing a key and record to each nondefective track within the cylinder assignment.

3. If an alternate track is assigned, a bit is set in the track matrix indicating the original or home track as defective. A defective track count is maintained for control of the Snapshot routine.

4. After writing a key and record to each nondefective track within the cylinder assignment, the track control counts and track matrix are written to the first track of the allocated area as record number 2.

First Track Format

Record Number 1

Field	No. of Bytes	Contents
Key Field	4	Zeros.
Data Field	2800	Zeros.

Record Number 2

Field	No. of Bytes	Contents
Key Field	4	Byte 0-1- contains the total number of tracks assigned. Byte 2-3- contains the total number of defective tracks.
Data Field	2800	Contains the track matrix block indicating the defective and nondefective tracks on the cylinders allocated (two bytes per cylinder.)

All Other Allocated Tracks

Field	No. of Bytes	Contents
Key Field	4	Zeros.
Data Field	2800	Zeros.

Example of track matrix block for one cylinder

Byte 1								Byte 2							
T 0	T 1	T 2	T 3	T 4	T 5	T 6	T 7	T 8	T 9	1	1	1	1	1	1

Each bit represents a track within the cylinder. A 1 bit is set in the bit position relating to the track number (0-9) of that cylinder that is defective. A 0 is in the bit position to indicate a nondefective track.

Always 1.

MCS OFF-LINE RECOVERY PROGRAM (MCSREC)

General Description

◆ The Multichannel Communications System Off-Line Recovery Program routine (MCSREC) facilitates the MCP restart following a system failure or an emergency shutdown. This routine utilizes information contained in the COMDISC file (temporary storage area for message cells) and the COMSNAP file (storage area for snapshot information) to create a tape that can be used to rebuild message queues.

The following information is written to and is uniquely identified on the tape:

1. The User Station Sequence Number Table (if included in the snapshot).
2. The Common Data Area (if included in the snapshot).
3. Each cell of each message that was active in the system at the time of snapshot.

The tape created by this routine can be used as an input to a sort and/or a restart program.

Only those MCP's that utilize dynamic buffering can include the snapshot facility on which the MCS Off-Line Recovery Program is based. Those using Direct Access can recover output messages only.

Input

◆ The input to this routine consists of the COMSNAP and COMDISC files and the input parameters.

Output

◆ The output of this routine is on a single magnetic tape. Multivolume output is not supported.

Epuipment Configuration

Required

◆ Processor 70/35-45-55 (65K)

Console typewriter
Magnetic tape device
Disc storage unit

Optional

◆ An additional disc storage unit and a card reader or one additional magnetic tape device.

Detailed Description

◆ The VTOC(s) and the DISC(s) are searched to locate the two required input files: COMSNAP - the snapshot area and COMDISC - the dynamic buffer storage area. The files may reside on the same or separate disc

Detailed Description *(Cont'd)*

storage units. The input parameters are then verified. Parameters may be submitted in one of three available options. If running under Monitor, parameters must be present on SYSIPT (card reader or magnetic tape). Missing or invalid parameters from SYSIPT cause termination of the job. If running under the Executive, parameters can be submitted from either the console typewriter or the card reader. Missing or invalid parameters from the card reader cause termination of the job. Missing or invalid parameters from the console typewriter can be resubmitted.

Two options are available to search the disc for a snapshot. The first is a search for the last complete snapshot in the file. This option is preset when neither option is specified. The second option is a search for a specific snapshot by snapshot number. If the search cannot be satisified (specified snapshot not found or not complete) the program is terminated.

If a User Station Sequence Number Table and/or Common Data Area are included in the snapshot, this information is written to tape. The Message Table entries in the snapshot are then examined sequentially. If a Message Table entry was active at the time the snapshot was taken, the entry is processed according to its state. For each active entry, at least one record is written to the output tape. The number and format of the records for each state are covered under Format of Data Records.

Routine Parameters Detailed

◆ The input parameters are positional, and the intervening commas are required.

Format:

ΔMSGΔnnnn,[dddd][uuuuu] ,cccc,xxxx

Entry	Meaning
ΔMSGΔ	Parameter identifier.
nnnn,	Decimal number of Message Table entries (same as specified in the CMMSG macro). This is a required entry.
[dddd],	Decimal number of bytes of the Common Data Area snapped (same as specified in the CMMSG macro). This entry is required if the Common Data Area is included in the snapshot.
[uuuuu],	Decimal number of bytes in the User Station Sequence Number (same as specified in the CMMSG macro). This entry is required if the User Station Sequence Number Table is included in the snapshot.
cccc,	Decimal number of bytes in each disc cell (same as specified in CMBUF macro). This entry is required.

Routine Parameters Detailed *(Cont'd)*

Entry	Meaning
xxxx	Snapshot option - One of the following must be specified. LAST - indicates that the last complete snapshot in the COMSNAP file is to be processed. A one to four-character hexadecimal representation of the snapshot number for the snapshot to be processed.

Considerations for Use

◆ 1. In using the option of recovering from a snapshot selected by number, the possible effect of the elapsed time between that snapshot and the latest complete snapshot should be considered. In a high volume system, disc wrap-around of cells and/or a long interval of time between successive snapshots could lead to an unrecoverable situation.

2. The MCS Off-Line Recovery Program can be utilized in conjunction with an MCP that includes the Snapshot capability; however, individual system considerations must be considered for each MCP. For example, responses to inquiries can be retrieved from the COMDISC file if they were queued for output; however, these messages could not be transmitted.

3. The output tape, MCSTAP, produced by this routine can be sorted on fields 2, 3, and 4 to bring the records type (as described in Format of Data Records) together.

Parameter Example

◆ MSG 150,2000,750,120,4F

150 Message Table Entries.
2000 Bytes of Common Data Area.
750 Byte User Station Sequence Number Table.
120 Byte Cells.
Snapshot $(004F)_{16}$ is to be processed.

Tape Format

◆ The tape will have standard labels and variable-length, unblocked data records. The filename is MCSTAP. The block count is included in the EOF label. Since the PURGE macro is used, the assigned tape, if not previously labeled, will have a dummy VOL written to it.

Format of Data Records

User Station Sequence Number Table Record

Field No.	1	2	3	4	5
	LENGTH	I. D.	ZERO	SEQ. NO.	DATA (UP TO 2800 BYTES)

Field No.	Bytes	Meaning and Contents
1	0-7	Four bytes - Binary Block and Record Length (data Length 14).
2	8,9	Two bytes - $(0001)_{16}$ - Identifies this record as the User Station Sequence Number Table.
3	10,11	Two bytes - $(0000)_{16}$.
4	12,13	Two bytes - Binary sequence number used to order the records containing the User Station Sequence Number Table. The maximum number of data bytes per record is 2,800. Example: An 8,000 - byte User Station Sequence Number Table would require 3 records as follows: First Record - Contains 2,814 bytes and the sequence number (bytes 12-13) is $(0001)_{16}$. Second Record - Contains 2,814 bytes and the sequence number (bytes 12-13) is $(0002)_{16}$. Third Record - Contains 2,414 bytes and the sequence number (bytes 12-13) is $(0003)_{16}$.
5	14-2814	Up to 2,800 bytes maximum - User Station Sequence Number Table Data.

Common Data Area Record

Field	1	2	3	4	5
	LENGTH	I. D.	ZERO	SEQ. NO.	DATA (UP TO 2800 BYTES)

Field No.	Bytes	Meaning and Contents
1	0-7	Four bytes - Binary Block and Record Length (Data Length + 14).
2	8-9	Two bytes - $(0002)_{16}$ identifies this record as Common Data Area.
3	10-11	Two bytes - $(0000)_{16}$.
4	12-13	Two bytes - Binary Sequence number used to order the records containing the Common Data Area. The maximum number of data bytes per record is 2,800. Example: An 8,000-byte Common Data Area would require 3 records as follows: First Record - Contains 2,814 bytes and the sequence number (bytes 12-13) is $(0001)_{16}$. Second Record - Contains 2,814 bytes and the sequence number (bytes 12-13) is $(0002)_{16}$. Third Record - Contains 2,414 bytes and the sequence number (bytes 12-13) is $(0003)_{16}$.
5	14-2814	Up to 2,800 bytes maximum - Common Data Area Data.

Dynamic Input Queue Record

Field No.	1	2	3	4	5	6	7
	LENGTH	I. D.	M. T. E. No.	SEQ. No.	ACTIVE M. T. E. OR BINARY ZEROS	DYNAMIC CELL HEADER	MESSAGE CELL (UP TO 3534 BYTES)

Field No.	Bytes	Meaning and Contents
1	0-7	Four bytes - Binary Block and Record Length (Data Length + 46).
2	8-9	Two bytes - $(0003)_{16}$ identifies this record as being on the dynamic input queue at the time of snapshot.
3	10-11	Two bytes - Binary Message Table Entry Number used to order the records by entry. For example, 100 Message Table entries, 15 of which are active, would result in numbers $(0001)_{16}$ to $(000F)_{16}$ being assigned in this field.
4	12-13	Two bytes - Binary sequence number used to order the records within Message Table entry number. One record is written for each cell of the message.
5	14-29	Sixteen bytes - The active Message Table entry itself appears in record sequence number $(0001)_{16}$ only (in subsequent records this field contains binary zeros).
6	30-45	Sixteen bytes - The dynamic cell header for this cell.
7	46-3579	Up to 3,534 bytes maximum - Only one message cell per record allows the largest cell size available in MCS to be contained in one tape record.

Intercept Input Queue Record

Field No.	1	2	3	4	5	6	7
	LENGTH	I. D.	M. T. E. NO.	SEQ. NO.	ACTIVE M. T. E. OR BINARY ZEROS	DYNAMIC CELL HEADER	MESSAGE CELL (UP TO 3534 BYTES)

Field No.	Bytes	Meaning and Contents
1	0-7	Four bytes - Binary Block and Record Length (Data Length 46).
2	8-9	Two bytes - $(0004)_{16}$ identifies this record as being on the intercept input queue at the time of the snapshot.
3	10-11	Two bytes - Binary Message Table entry Number used to order the records by entry. For example, 100 Message Table entries, 15 of which are active, would result in numbers $(0001)_{16}$ to $(000F)_{16}$ being assigned in this field.
4	12-13	Two bytes - Binary sequence number used to order the records within Message Table entry number. One record is written for each cell of the message.
5	14-29	Sixteen bytes - The active Message Table entry itself appears in record sequence number $(0001)_{16}$ only (in subsequent records this field contains binary zeros).
6	30-45	Sixteen bytes - The dynamic cell header for this cell.
7	46-3579	Up to 3,534 bytes maximum - Only one message cell per record allows the largest cell size available in MCS to be contained in one tape record.

Output Queue or During Transmission Record

Field No.	1	2	3	4	5	6	7
	LENGTH	I. D.	M. T. E. NO.	SEQ. NO.	ACTIVE M. T. E. OR BINARY ZEROS	DYNAMIC CELL HEADER	MESSAGE CELL (UP TO 3534 BYTES)

Field No.	Bytes	Meaning and Contents
1	0-7	Four bytes - Binary Block and Record Length (Data Length + 46).
2	8-9	Two bytes - $(0005)_{16}$ identifies this record as being on the output queue at the time of snapshot.
3	10-11	Two bytes - Binary Message Table entry number used to order the records by entry. For example, 100 Message Table entries, 15 of which are active, would result in numbers $(0001)_{16}$ to $(000F)_{16}$ being assigned in this field.
4	12-13	Two bytes - Binary sequence number used to order the records within Message Table entry number. One record is written for each cell of the message.
5	14-29	Sixteen bytes - The active Message Table entry itself appears in record sequence number $(0001)_{16}$ only (in subsequent records this field contains binary zeros).
6	30-45	Sixteen bytes - The dynamic cell header for this cell.
7	46-3579	Up to 3,534 bytes maximum - Only one message cell per record allows the largest cell size available in MCS to be contained in one tape record.

Being Queued for Output Record

Field No.	1	2	3	4	5	6	7
	LENGTH	I. D.	M. T. E. NO.	SEQ. NO.	ACTIVE M. T. E. OR BINARY ZEROS	DYNAMIC CELL HEADER	MESSAGE CELL (UP TO 3534 BYTES)

Field No.	Bytes	Meaning and Contents
1	0-7	Four bytes - Binary Block and Record Length (Data Length + 46).
2	8-9	Two bytes - $(0006)_{16}$ identifies this record as being queued for output at the time of snapshot.
3	10-11	Two bytes - Binary Message Table entry number used to order the records by entry. For example, 100 Message Table entries, 15 of which are active, would result in numbers $(0001)_{16}$ to $(000F)_{16}$ being assigned in this field.
4	12-13	Two bytes - $(0001)_{16}$ - Sequence number (only the first cell of the message is written to tape).
5	14-29	Sixteen bytes - The active Message Table entry itself appears in this record.
6	30-45	Sixteen bytes - The dynamic cell header for this cell.
7	46-3579	Up to 3,534 bytes maximum - Only the first cell of the message is written to tape.

During Message Reception (Cells on Disc) Record

Field No.	1	2	3	4	5	6	7
	LENGTH	I. D.	M. T. E. NO.	SEQ. NO.	ACTIVE M. T. E. OR BINARY ZEROS	DYNAMIC CELL HEADER	MESSAGE CELL (UP TO 3534 BYTES)

Field No.	Bytes	Meaning and Contents
1	0-7	Four bytes - Binary Block and Record Length (Data Length + 46).
2	8-9	Two bytes - $(0007)_{16}$ identifies this record as in the process of being received and one or more cells have been written to disc.
3	10-11	Two bytes - Binary Message Table entry number used to order the records by entry. For example, 100 Message Table entries, 15 of which are active, would result in numbers $(0001)_{16}$ to $(000F)_{16}$ being assigned in this field.
4	12-13	Two bytes - $(0001)_{16}$ - Sequence number (only the first cell of the message is written to tape).
5	14-29	Sixteen bytes - The active Message Table entry itself appears in this record.
6	30-45	Sixteen bytes - The dynamic cell header for this cell.
7	46-3579	Up to 3,534 bytes maximum - Only the first cell of the message is written to tape.

During Message Reception (No Cells on Disc) Record

Field No.	1	2	3	4	5	6
	LENGTH	I.D.	M.T.E. NO.	SEQ. NO.	ACTIVE M.T.E. OR BINARY ZEROS	DYNAMIC CELL HEADER

Field No.	Bytes	Meaning and Contents
1	0-7	Four bytes - Binary Block and Record Length (46 Bytes).
2	8-9	Two bytes - $(0008)_{16}$ identifies this record as in the process of being received and no cells have been written to disc.
3	10-11	Two bytes - Binary Message Table entry number used to order the records by entry. For example, 100 Message Table entries, 15 of which are active, would result in numbers $(0001)_{16}$ to $(000F)_{16}$ being assigned in this field.
4	12-13	Two bytes - $(0001)_{16}$ - Sequence number (only the first cell of the message is written to tape).
5	14-29	Sixteen bytes - The active Message Table entry itself appears in this record.
6	30-45	Sixteen bytes - The dynamic cell header for this cell.

Device Assignments

◆ Under Executive or Monitor Control:

SDN	Device Type	Remarks
CMSNAP	Disc Storage Unit.	Contains COMSNAP File.
CMDISC	Disc Storage Unit.	Contains COMDISC File (may be same as CMSNAP).
MCSTAP	Magnetic Tape.	Output Device
MCSRDR	Card Reader.	Optional.

APPENDIX A

TDOS LIBRARY FORMATS

EXECUTIVE LOAD LIBRARY

Library Format

◆ The disc or drum containing the TDOS Executive consists of the following components:

1. Track Descriptor Record
2. Bootstrap
3. Initial Program Loader (IPL)
4. Standard Volume Label (SVL)
5. Volume Table of Contents (VTOC)
6. Program Directory
7. Load Directory for Resident Executive and Executive and Error Recovery overlays
8. Load Directory for FCP and Monitor
9. Load Directory for Monitor overlays
10. Resident Executive text and Executive and Error Recovery overlays
11. FCP and Monitor text and modifier blocks
12. Monitor overlays

Record Formats

Track Descriptor Record

◆ The Track Descriptor record appears as the first record (R0) on every track of the random access device. Refer to the TOS Utility Routines manual, Random Access Volume Initializer routine for a description of this record.

Bootstrap and Initial Program Loader

◆ The Bootstrap and Initial Program Loader (IPL) are records one (R1) and two (R2), respectively on cylinder 0, track 0 of the system resident device. These records are read when the system is loaded and contain the coding used to load the Resident Executive and enter the Executive Initializer.

Standard Volume Label

◆ The Standard Volume Label (SVL) is record three (R3) on cylinder 0, track 0 of the random access device. It contains the address of the Volume Table of Contents. Refer to the TOS Utility Routines manual, Appendix C for a description of this record.

Volume Table of Contents

◆ The Volume Table of Contents (VTOC) contains a description of the contents of the random access volume. The entry EXCLIB gives the address of the Executive Library. The location of the VTOC depends upon parameters supplied to the Random Access Volume Initializer routine when the volume was initialized. Refer to the TOS Utility Routines manual, Appendix C for a description of the VTOC.

Program Directory

◆ The Program Directory is the first block in the area allocated for the Executive Library. It consists of three 30-byte records followed by six hexadecimal FF's.

Format:

Bytes	Contents

0-5	Name of Executive part.
	TDOSRE = Resident Executive and Executive and Error Recovery overlays.
	FCPOVL = FCP and Monitor.
	MON = Monitor overlays

6-8	Initial load entry point (hexadecimal).
9-11	Minimum memory requirement (hexadecimal).
12-14	Maximum memory requirement (hexadecimal).
15-19	Disc or drum address of load directory for part (CCHHR).
20-25	Creation date (ddmmyy).
26-28	Version number (vvv).
29	Reserved.

Load Directory for Resident Executive and Executive and Error Recovery Overlays

◆ The Executive load directory consists of 32 entries in the CCHHR format indicating the disc or drum address of the load.

The entries are in order by the load number $(00)_{16}$ to $(1F)_{16}$. $(00)_{16}$ is the Resident Executive; $(01)_{16}$ to $(1F)_{16}$ are the Executive and Error Recovery overlays. If a particular overlay is not used in the system, its entry is all zeros.

Load Directory for FCP, Monitor, and Monitor Overlays

◆ Entries in the load directories for FCP and Monitor and Monitor overlays are 14 bytes long, blocked to a maximum of 280 bytes (20 entries). The end of each load directory is indicated by eight hexadecimal FF's.

Format:

Bytes	Contents
0-5	Name of load.
6-10	Disc or drum address of load (CCHHR).
11-13	Program-relative load address of first text byte (hexadecimal).

Text and Modifier Blocks

◆ *Executive*

Executive text is written as one overflow record with no key and no modifiers.

Executive and Error Recovery Overlays

Each overlay is written as a single record with a maximum size of 1,024 bytes. There are no keys or modifier blocks.

FCP

FCP text is written as single records with a maximum size of 2,000 bytes. Each text record is preceded by a 6-byte key field. Modifier blocks are written after the text. The modifier blocks are a maximum of 500 bytes long and are preceded by a 6-byte key field.

Monitor

Main Monitor and Monitor Job Control text are each written as one overflow record with a maximum size of 4,096 bytes and are preceded by a 6-byte key field. Monitor Snap text is 1 record with a maximum size of 2,048 bytes and is preceded by a 6-byte key field. Modifier blocks are written after the text. The blocks are a maximum size of 500 bytes and are preceded by a 6-byte key field.

Monitor Overlays

Monitor overlays are written as single records with a maximum size of 1,024 bytes. There are no keys or modifier blocks.

Key Format:

Bytes	Contents
0-4	Disc or drum address of next block (CCHHR).

5	Type of block that follows:
	T = text.
	M = modifier.
	L = this is the last block for this partor overlay.

Text Format:

Text blocks consist of coding only, for the particular part or overlay to which they apply.

Text and Modifier Blocks (Cont'd)

Modifier Format:

Bytes	Bit	Contents
0-3		Program-relative float factor, in binary, by which the address constant is modified.
4	0	End of block indicator: 0 = more modifiers in this block 1 = last modifier in this block
	1-3	Zeros
	4-5	Length of address Constant: 00 = one byte 01 = two bytes 10 = three bytes 11 = four bytes
	6	Action flag 0 = add float factor to address constant 1 = subtract float factor from address constant
	7	Length of next modifier: 0 = eight bytes; modifier has new float factor 1 = four bytes; next modifier has same float factor Note: The length of the first modifier in a block is always eight bytes.
5-7		Program-relative location, in binary, of the address constant.

PROGRAM LOAD LIBRARY

Library Format

◆ A disc or drum containing a TDOS Program Library consists of the following components in the order given:

1. Program Directory
2. Text and Modifier blocks for first program
3. Load Directory for first program
4. Text and Modifier blocks for second program
5. Load Directory for second program
6. Text and Modifier blocks for nth program
7. Load Directory for nth program
8. Work tracks (3)

Record Format

Program Directory

◆ The program directory is located at the beginning of the area allocated for the program library. It occupies from 1 to 10 tracks of 1 cylinder.

The program directory blocks consist of ten 30-byte records. The entries in the directory are in sequence by program name. Each block is preceded by a 6-byte key containing the name of the last program in that block. The key for the last block in the directory that contains program entries consists of hexadecimal FF's.

The first record in the program directory contains information used by the Program Library Transcriber routine.

Key Format:

Bytes	Contents
0-5	Name of the last program in the directory block. Note: The key for the last block in the directory containing program entries and any additional blocks consists of hexadecimal FF's.

Record Format:

Bytes	Contents
0-5	Program name. Contains hexadecimal FF's if this is a dummy record.
6-8	Initial load entry point (hexadecimal).
9-11	Minimum memory requirement (hexadecimal).
12-14	Maximum memory requirement (hexadecimal).
15-19	Disc or drum address of load directory (CCHHR).
20-25	Creation date (ddmmyy).
26-28	Version number (vvv).
29	Reserved.

Text and Modifiers

◆ Program text normally consists of one overflow record for each program load. If the text is broken by bad tracks or cylinders, the text is continued on the tracks following the bad area.

Modifiers may follow text. Modifiers are 8- or 4-bytes long and are blocked to a maximum size of 500 bytes.

Program text and modifier blocks are preceded by a six-byte key.

Key Format:

Bytes	Contents
0-4	Disc or drum address of next block (CCHHR).

5	Type of block that follows:
	T = text
	M = modifier
	L = this is the last block for this load.

Text Format

Text blocks consist of coding only, for the particular load to which they apply.

Modifier Format:

Bytes	Bit	Contents
0-3		Program-relative float factor, in binary, by which the address constant is modified.
4	0	End of block indicator: 0 = more modifiers in this block 1 = last modifier in this block
	1-3	Zeros.
	4-5	Length of address constant: 00 = one byte 01 = two bytes 10 = three bytes 11 = four bytes
	6	Action flag 0 = add float factor to address constant 1 = subtract float factor from address constant
	7	Length of next modifier: 0 = eight bytes; next modifier has new float factor. 1 = four bytes; next modifier has same float factor. Note: The length of the first modifier in a block is always eight bytes.
5-7		Program-relative location, in binary, of the address constant.

Load Directory

♦ The load directory for a program follows the last load for that program. Load directory entries are 14 bytes long. A load directory block is a maximum of 280 bytes.

Format:

Bytes	Contents
0-5	Name of load.
6-10	Disc or drum address of load (CCHHR).
11-13	Program-relative load address of first text byte (hexadecimal).

Work Tracks

♦ The last three tracks in the area allocated for the Program Library are reserved as work tracks for the Program Library Transcriber routine.

APPENDIX B

RANDOM ACCESS AND MEMORY REQUIREMENTS

GENERAL

◆ The tables in this appendix list utility routine random access storage requirements, memory sizes, I/O block sizes (where applicable), and how each routine will use additional memory.

The random access storage requirements indicate the number of tracks needed by each routine for disc and drum storage.

The routine memory size is what is allocated to the routine by the Executive when the routine is loaded normally. This memory size may be changed by specifying more memory in the E LOD message or by processing the routine through the Linkage Editor and changing the memory requirements by using the PROG parameter.

The block sizes given indicate the block length that will be processed by the routine with the normal memory size. The routine will double-buffer any block up to the size given and single-buffer any block between the double buffer size and single buffer size.

The remarks column describes how the routine will use additional memory. The memory required for a particular application may be calculated by using the information in the remarks column or the formulas given at the end of this appendix.

Peripheral Conversion Routines *(Cont'd)*

Routine	Segments	Required Tracks		Required Memory (bytes)	Input/Output Area		Remarks
		Disc	Drum		Max. Block Single Buffer (bytes)	Max. Block Double Buffered (bytes)	
TPTP	(ROOT) LOAD1 LOAD2 LOAD3 LOAD4 ITUTPB	10	12	14,728	In: 1,000 Out: 1,000	500 500	Additional memory will be used first for Field Select coding, then for input/output area. (See note.)

Note:

The Peripheral Conversion routines generate Field-Select coding according to the options given in the FS parameter(s). A 100-byte area is allocated for this coding. If more area is required, additional memory must be given to the routine when it is loaded. The additional Field-Select area needed can be calculated as follows:

FS- 6(M-P-U-H) - 100

TRACK AND MEMORY TABLES

Peripheral Conversion Routines

Routine	Segments	Required Tracks		Required Memory (bytes)	Input/Output Area		Remarks
		Disc	Drum		Max. Block Single Buffer (bytes)	Max. Block Double Buffered (bytes)	
CDPR	(ROOT) LOAD1 LOAD2 LOAD3 LOAD4 ITUTPB	9	12	12,504	In: NA Out: NA	80 132/160	Additional memory will be used for Field Select coding. (See note.)
CDTP	(ROOT) LOAD1 LOAD2 LOAD3 LOAD4 ITUTPB	9	12	13,600	In: NA Out: 1,000	80 500	Additional memory will be used first for Field Select coding, then for output area. (See note.)
DUP	(ROOT)	3	*	10,040	In: 1,044 Out: 1,044	NA NA	Additional memory is used for input/output area. Three bytes of memory are required for each byte in the input block over 1,044 bytes.
TPINIT	(ROOT)	2	2	4,784	NA	NA	None.
TPPR	(ROOT) LOAD1 LOAD2 LOAD3 LOAD4 ITUTPB	10	13	16,200	In: 1,000 Out: NA	500 132/160	Additional memory will be used first for Field Select coding, then for input area. (See note.)

*To be supplied.

Peripheral Conversion Routines *(Cont'd)*

where:

$M = \frac{s}{256}\uparrow$ for each field to be moved (s = size of field.

$P = \frac{n}{8}\uparrow + 2$ for each field to be packed (n = size of field).

$U = \frac{n}{8}\uparrow + 2$ for each field to be unpacked (n = size of field).

$H = \frac{n}{8}\uparrow + 3$ for each field to be converted to hexadecimal (n = size of field).

If an option is not used, its value is 0.

Example:

$$\begin{array}{ccccc} M & U_1 & U_2 & H_1 & H_2 \end{array}$$

```
FS 5,50,1/115(U,8,16)51/123(U,25,50)67/150(X,2)117/152(X,2)121
```

$$FS = 6(M + P + U_1 + U_2 + H_1 + H_2) - 100$$

$$\begin{array}{lllll} M = \frac{s}{256}\uparrow & U_1 = \frac{n}{8}\uparrow + 2 & U_2 = \frac{n}{8}\uparrow + 2 & H_1 = \frac{n}{8}\uparrow + 3 & H_2 = \frac{n}{8}\uparrow + 3 \end{array}$$

$$\begin{array}{lllll} M = \frac{100}{256} & U_1 = \frac{8}{8} + 2 & U_2 = \frac{25}{8} + 2 & H_1 = \frac{2}{8} + 3 & H_2 = \frac{2}{8} + 3 \end{array}$$

$$\begin{array}{lllll} M = 1 & U_1 = 3 & U_2 = 6 & H_1 = 4 & H_2 = 4 \end{array}$$

$$FS = 6\ (1 + 3 + 6 + 4 + 4) - 100$$

FS = 8 bytes.

Peripheral Conversion-Random Access

Routine	Segments	Required Tracks		Required Memory (bytes)	Input/Output Area		Remarks
		Disc	Drum		Max. Block Single Buffer (bytes)	Max. Block Double Buffered (bytes)	
CDRA	(ROOT) LOAD1 LOAD2 LOAD3 LOAD4 ITUTPB	9	12	16,576	NA Out: 1,000	80 500	Additional memory will be used first for Field Select coding, then for output area. (See note, page B-2A.)
CDRAM	(ROOT) LOAD1 LOAD3 LOAD4 ITUTPB	10	*	19,800	In: NA Out: 1,000	80 500	Additional memory will be used first for Field Select coding, then for output area. (See note, page B-2A.)
RAINIT	(ROOT)	4	5	11,152	NA	NA	None.
RAPR	(ROOT) LOAD1 LOAD2 LOAD3 LOAD4 ITUTPB	10	13	16,240	In: 1,000 Out: NA	500 132/160	Additional memory will be used first for Field Select coding, then for input area. (See note, page B-2A.)
RARA	(ROOT) LOAD1 LOAD2 LOAD3 LOAD4 ITUTPB	9	12	17,224	In: 1,000 Out: 1,000	500 500	Additional memory will be used first for Field Select coding, then for input/output area. (See note, page B-2A.)
RARAM	(ROOT) LOAD1 LOAD3 LOAD4 ITUTPB	10	*	20,456	In: 1,000 Out: 1,000	500 500	Additional memory will be used first for Field Select coding, then for input/output area. (Seenote, page B-2A.)
RATP	(ROOT) LOAD1 LOAD2 LOAD3 LOAD4 ITUTPB	10	*	16,632	In: 1,000 Out: 1,000	500 500	Additional memory will be used first for Field Select coding, then for input/output area. (See note, page B-2A.)

*To be supplied.

Diagnostics *(Cont'd)*

Routine	Segments	Required Tracks		Required Memory (bytes)	Input/Output Area		Remarks
		Disc	Drum		Max. Block Single Buffer (bytes)	Max. Block Double Buffered (bytes)	
TDSAID (Cont'd)	CONFOR CONFIV CONSIX CONSEV CONATE CONNIN CONELV AUTONE AUTTWO AUTTHR AUTFOR AUTATE AUTFIV AUTNIN AUTTTW AUTTTH AUTTFO AUTTWL AUTELV AUTTEN AUTTIR AUTFRT AUTFVT AUTSVT AUTATT AUTTON AUTSTX AUTTNT						
TPCOMP	(ROOT)	3	4	8,928	500	250	Additional memory is used for input area.
TPEDIT	(ROOT)	2	3	6,316	500	250	Additional memory is used for input area.

System Maintenance Routines

Routine	Segments	Required Tracks		Required Memory (bytes)	Input/Output Area		Remarks
		Disc	Drum		Max. Block Single Buffer (bytes)	Max. Block Double Buffered (bytes)	
CLU	(ROOT)	4	*	10,096	NA	NA	None.
DDRL	(ROOT) ITURPM ITURDD ITURWR	5	7	13,448	NA	NA	None.
LLT	(ROOT)	6	8	18,408	NA	NA	None.
LLU	(ROOT)	6	9	21,440	NA	NA	Additional memory is used for processing tables.
LNKEDT	(ROOT) LINK1 LINK2 LINK3 LINK4 LINKX	17	22	32,768	NA	NA	Additional memory is used for processing tables (module, entry, extrn, load and V-type items).
MLU	(ROOT)	6	*	18,504	NA	NA	None.
OMLU	(ROOT)	8	*	26,592	NA	NA	The OMLU contains a table which can contain 100 entries. One entry is made for each module to be merged, extracted, or added. If more than 100 entries are expected, add 12 bytes to the memory size for each additional entry.
RAALLR	(ROOT)	5	*	17,024	NA	NA	Additional memory is used for internal processing storage.
RAINDX	(ROOT)	2	2	4,112	NA	NA	None.

Peripheral Conversion-Random Access *(Cont'd)*

Routine	Segments	Required Tracks		Required Memory (bytes)	Input/Output Area		Remarks
		Disc	Drum		Max. Block Single Buffer (bytes)	Max. Block Double Buffered (bytes)	
TPRA	(ROOT) LOAD1 LOAD2 LOAD3 LOAD4 ITUTPB	10	13	17,080	In: 1,000 Out: 1,000	500 500	Additional memory will be used first for Field Select coding, then for input/output area. (See note, page B-2A.)
TPRAM	(ROOT) LOAD1 LOAD3 LOAD4 ITUTPB	10	*	20,312	In: 1,000 Out: 1,000	500 500	Additional memory will be used first for Field Select coding, then for input/output area. (See note, page B-2A.)

Diagnostics

Routine	Segments	Required Tracks		Required Memory (bytes)	Input/Output Area		Remarks
		Disc	Drum		Max. Block Single Buffer (bytes)	Max. Block Double Buffered (bytes)	
CARDCK	(ROOT)	1	1	376	NA	NA	None.
DIAGDG	(ROOT)	4	5	8,872	2,000	NA	Additional memory will allow a greater output block size.
DUMPRT	(ROOT)	1	2	2,424	NA	NA	None.
RAEDIT	(ROOT)	2	3	9,704	NA	NA	None.
TDSAID	(ROOT) CONRES AIDINT CONTEN CONONE CONTWO CONTHR	32	39	9,920	NA	NA	Additional memory is used for working and parameter storage.

*To be supplied.

System Maintenance Routines *(Cont'd)*

Routine	Segments	Required Tracks		Required Memory (bytes)	Input/Output Area		Remarks
		Disc	Drum		Max. Block Single Buffer (bytes)	Max. Block Double Buffered (bytes)	
RAMSUP	(ROOT) ITURPV ITURCU ITURSC ITURCP ITURRR ITUREO ITURCI ITURTR ITURSO	12	*	7,800	NA	NA	Additional memory must be allocated when using TANK function. See TOS Utility Manual, page 7-26.
SLU	(ROOT)	9	11	30,688	NA	NA	The SLU contains 1,000 bytes for storage of Level 1 action and reorder entries. Action entries use 20 bytes for each program named in a RENAME DELETE, EXTRACT, OUTPUT, PRINT, or PUNCH card. Reorder entries use two bytes for each device named in a REORDER card. If the total number of entries require more than 1,000 bytes, allocate additonal memory as required.
TPMAIN	(ROOT)	7	*	24,296	4,000 (combined input/ output).	NA	Additional memory is used for input/ output area.

*To be supplied.

Library Conversion Routines

Routine	Segments	Required Tracks		Required Memory (bytes)	Input/Output Area		Remarks
		Disc	Drum		Max. Block Single Buffer (bytes)	Max. Block Double Buffered (bytes)	
CLTR	(ROOT) ITUCL2 ITUCL3 ITUCL4 ITUCL5 ITUCL6 ITUCL7	7	8	17,456	NA	NA	None.
PRGTRN	(ROOT)	12	*	37,400	NA	NA	None.

*To be supplied.

MEMORY FORMULAS

Peripheral Conversion Routines

◆ $MR = S + \left[n_i\ (BS_i - B_i)\right] + \left[n_o\ (BS_o - B_o)\right] + FS$

where:

MR = memory size requirement.

S = memory size of routine.

n_i = 1 for single buffer input or 2 for double buffer input.

BS_i = maximum input block size (cannot be > 4095).

B_i = input buffer size (when $n_i = 1$, B_i must equal single buffer size; when $n_i = 2$, B_i must equal double buffer size).

n_o = same as n_i except for output.

BS_o = same as BS_i except for output.

B_o = same as B_i except for output.

FS = additional Field-Select coding area (see note following Peripheral Conversion Routines table).

Example

◆ The programmer wishes to run the TPTP routine; single buffering 1,500-byte input blocks and double buffering 3,000-byte output blocks. No additional Field-Select coding area is needed.

$MR = S + \left[n_i\ (BS_i - B_i)\right] + \left[n_o\ (BS_o - B_o)\right] + FS$

$MR = 14,728 + 1\ (1500-1000) + 2\ (3000-400) + 0$

$MR = 20228$ bytes.

DIAGDG

◆ $MR = S + (BS_o - B_i)$

Example

◆ The programmer wants to generate variable-length blocks between 1,500 and 3,000 bytes.

$MR = S + (BS_o - B_i)$

$MR = 8872 + (3000 - 1000)$

$MR = 10872$ bytes.

TPCOMP

◆ $MR = S + \left[m_i\ (BS_i - B_i)\right]$

where:

$m_i = 2$ for single-buffer input or 4 for double-buffer input.

Example

◆ The programmer wishes to double-buffer 1,000 byte input blocks.

$MR = S + m_i\ (BS_i - B_i)$

$MR = 8928 + 4(1000 - 250)$

$MR = 11928$ bytes.

LNKEDT

◆ $MR = S + 28L + 20M + 14\ (E + V + U) - 4792$

where:

L = number of loads in the program.
M = number of modules in the program.
E = number of entries in the program.
V = number of VCONS in the program.
U = number of <u>unsatisfied</u> extrns in the program.

Example

◆ The programmer wishes to bind a program consisting of the following items:

48 loads
60 modules
400 entries
50 VCONS
37 unsatisfied extrns

$MR = S + 28L + 20M + 14\ (E+V+U) - 4792$

$MR = 32768 + 28(48) + 20(60) + 14\ (400+50+37) - 4792$

$MR = 37338$ bytes.

TPMAIN

◆ $MR = S + \left[(I + 0_1 + 0_2 + 0_3) - 4000\right]$

where:

I = maximum input block size.

0_1 = maximum first output block size.

0_2 = maximum second output block size.

0_3 = maximum third output block size.

Example

◆ The programmer wishes to copy an input tape with 2,000-byte blocks to an output tape, reblocking to 3,000 bytes, also listing the input on a 132-character printer.

$MR = S + (I + 0_1 + 0_2 + 0_3) - 4000$

$MR = 24296 + (2000 + 3000 + 132 + 0) - 4000$

$MR = 25428$