

**Burroughs**

**B 1000 Systems**

**SYSTEM SOFTWARE  
OPERATION GUIDE**

**VOLUME 2**

**(RELATIVE TO B 1000 SYSTEMS SOFTWARE RELEASE – MARK 10.0)**

Copyright © 1982, Burroughs Corporation, Detroit, Michigan 48232

PRICED ITEM

Burroughs cannot accept any financial or other responsibilities that may be the result of your use of this information or software material, including direct, indirect, special or consequential damages. There are no warranties extended or granted by this document or software material.

You should be very careful to ensure that the use of this software material and/or information complies with the laws, rules, and regulations of the jurisdictions with respect to which it is used.

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

Correspondence regarding this publication should be forwarded using the Remarks form at the back of the manual, or may be addressed directly to TIO West Documentation, Burroughs Corporation, 1300 John Reed Court, City of Industry, California 91745, U.S.A.

## LIST OF EFFECTIVE PAGES

Page	Issue	Page	Issue
Title	Original	21-26	Blank
ii	Original	22-1 thru 22-9	Original
iii	Original	22-10	Blank
iv	Blank	23-1 thru 23-6	Original
v thru xvii	Original	24-1 thru 24-6	Original
xviii	Blank	25-1 thru 25-3	Original
1-1 thru 1-8	Original	25-4	Blank
2-1 thru 2-2	Original	26-1 thru 26-4	Original
3-1 thru 3-2	Original	27-1 thru 27-7	Original
4-1 thru 4-2	Original	27-8	Blank
5-1 thru 5-9	Original	28-1 thru 28-5	Original
5-10	Blank	28-6	Blank
6-1	Original	29-1 thru 29-8	Original
6-2	Blank	30-1 thru 30-26	Original
7-1 thru 7-4	Original	31-1 thru 31-4	Original
8-1 thru 8-3	Original	32-1 thru 32-9	Original
8-4	Blank	32-10	Blank
9-1 thru 9-3	Original	33-1 thru 33-4	Original
9-4	Blank	34-1 thru 34-5	Original
10-1 thru 10-5	Original	34-6	Blank
10-6	Blank	35-1 thru 35-2	Original
11-1 thru 11-3	Original	36-1 thru 36-8	Original
11-4	Blank	37-1 thru 37-2	Original
12-1 thru 12-3	Original	38-1 thru 38-2	Original
12-4	Blank	39-1	Original
13-1 thru 13-3	Original	39-2	Blank
13-4	Blank	40-1 thru 40-2	Original
14-1 thru 14-10	Original	41-1 thru 41-14	Original
15-1 thru 15-6	Original	42-1 thru 42-2	Original
16-1 thru 16-14	Original	43-1 thru 43-4	Original
17-1 thru 17-3	Original	44-1 thru 44-2	Original
17-4	Blank	45-1 thru 45-3	Original
18-1	Original	45-4	Blank
18-2	Blank	A-1 thru A-3	Original
19-1 thru 19-14	Original	A-4	Blank
20-1	Original	B-1 thru B-3	Original
20-2	Blank	B-4	Blank
21-1 thru 21-25	Original		





## TABLE OF CONTENTS

Section	Title	Page
	PREFACE . . . . .	xvii
1	INTRODUCTION . . . . .	1-1
	Classification of Utility Programs . . . . .	1-1
	General Description of Utility Programs . . . . .	1-3
	System Start-Up Programs . . . . .	1-3
	Copy/File Copy Programs . . . . .	1-3
	Disk File Copy . . . . .	1-3
	Tape File Copy . . . . .	1-4
	Printer Listings . . . . .	1-5
	Cassette File Copy . . . . .	1-5
	Card File Copy . . . . .	1-5
	Diskette File Copy . . . . .	1-6
	Disk Initializer Programs . . . . .	1-6
	General Maintenance Programs . . . . .	1-7
	Log Analysis Programs . . . . .	1-7
2	CART/INIT . . . . .	2-1
	Operating Instructions . . . . .	2-1
3	CASSETTE/MAKER . . . . .	3-1
	Operating Instructions . . . . .	3-1
	Input Specifications . . . . .	3-1
	Sample Execution Commands . . . . .	3-1
	Program Output . . . . .	3-1
	Files Created . . . . .	3-1
	Informative Messages . . . . .	3-2
	Error Messages . . . . .	3-2
4	CHECK/LOAD.DUMP . . . . .	4-1
	Operating Instructions . . . . .	4-1
	Program Switches . . . . .	4-1
	Run-Time Interrogation . . . . .	4-2
5	CLEAR/START and Memory Dump Procedure . . . . .	5-1
	CLEAR/START Functions . . . . .	5-1
	System Initializer Functions . . . . .	5-1
	MCPII Functions . . . . .	5-2
	CLEAR/START Operating Instructions . . . . .	5-2
	Description of the Name Table . . . . .	5-3
	Temporary Operating Environment Changes . . . . .	5-5
	General Specifications . . . . .	5-5
	System Disk Channel Selection Override . . . . .	5-6
	I/O Channel Deletion . . . . .	5-7
	Pseudo-Memory Size Specification . . . . .	5-7
	Temporary XM Specification . . . . .	5-8
	Supplementary Information Required With A Memory Dump . . . . .	5-8
6	CODE/ANALYZER . . . . .	6-1
	Operating Instructions . . . . .	6-1
	Program Switches . . . . .	6-1
7	COLDSTART/DISK . . . . .	7-1
	Operating Instructions . . . . .	7-1
	Error Messages . . . . .	7-2

## TABLE OF CONTENTS (Cont)

Section	Title	Page
8	COLDSTART/TAPE . . . . .	8-1
	Operating Instructions . . . . .	8-1
	Loading the Coldstart Program . . . . .	8-1
	Loading the System Software . . . . .	8-2
	Register Settings and Error Descriptions . . . . .	8-2
9	CONVERT/BACKUP . . . . .	9-1
	Operating Instructions . . . . .	9-1
	Error Messages . . . . .	9-3
10	CREATE/TABLE . . . . .	10-1
	Translation Table . . . . .	10-1
	Translate Header Format . . . . .	10-1
	Operating Instructions . . . . .	10-2
	Execution . . . . .	10-2
	Program Input . . . . .	10-2
	Option Summary . . . . .	10-3
	ASCII 8-bit Option . . . . .	10-4
	ASCII 7-bit Option . . . . .	10-4
	BCL Option . . . . .	10-4
	BCL 8-Bit Option . . . . .	10-4
	B500 Option . . . . .	10-4
	Identification Option . . . . .	10-4
	Soft Input Option . . . . .	10-4
	Error Messages . . . . .	10-4
	Internal Files . . . . .	10-5
11	DISK/ALLOCATOR . . . . .	11-1
	Operating Instructions . . . . .	11-1
	Execution . . . . .	11-1
	Command Specifications . . . . .	11-1
	Allocated File Structure . . . . .	11-2
	Sample Execution . . . . .	11-2
	Error Messages . . . . .	11-3
12	DISK/DUMP . . . . .	12-1
	Operating Instructions . . . . .	12-1
	Error Messages . . . . .	12-1
13	DISK PACK INTERCHANGE PROGRAMS . . . . .	13-1
	Operating Instructions . . . . .	13-1
	Error Messages . . . . .	13-1
14	DISKETTE/COPY . . . . .	14-1
	Operating Instructions . . . . .	14-1
	Program Switches . . . . .	14-1
	Input Specifications . . . . .	14-1
	AUTOLOAD Facility . . . . .	14-2
	COPY Command . . . . .	14-2
	Routine Type Specifier . . . . .	14-2
	Input File Specifications . . . . .	14-3
	Output File Specifications . . . . .	14-3
	BYPASS Option . . . . .	14-4

## TABLE OF CONTENTS (Cont)

Section	Title	Page
	PSEUDO Option . . . . .	14-4
	KA Command . . . . .	14-4
	PURGE Command . . . . .	14-6
	RELABEL Command . . . . .	14-6
	Sample Input Specifications . . . . .	14-6
	Error Messages . . . . .	14-7
15	DISKMAP/UTILITY . . . . .	15-1
	Operating Instructions . . . . .	15-1
	Program Switches . . . . .	15-2
	Phases of Execution and Program Output . . . . .	15-2
	Phase 1 . . . . .	15-2
	Phase 2 . . . . .	15-3
	Phase 3 . . . . .	15-4
16	DMPALL . . . . .	16-1
	Operating Instructions . . . . .	16-1
	Program Switches . . . . .	16-1
	Command Specifications . . . . .	16-2
	Input Specifications . . . . .	16-2
	Print Directory (PD) Specifications . . . . .	16-2
	LIST Specifications . . . . .	16-3
	List-specifier . . . . .	16-3
	File Specifications . . . . .	16-3
	TAG Option . . . . .	16-4
	Mode Option . . . . .	16-4
	Format Option . . . . .	16-4
	Hardware-Type Option . . . . .	16-4
	SKIP Option . . . . .	16-5
	INCLUDE Option . . . . .	16-5
	VARIABLE Option . . . . .	16-5
	Record Selection Option . . . . .	16-5
	SEARCH Record Selection Option . . . . .	16-5
	SELECT Record Selection Option . . . . .	16-5
	EXCLUDE Record Selection Option . . . . .	16-5
	KEY Record Selection Option . . . . .	16-6
	Required Parameters . . . . .	16-6
	Output Format . . . . .	16-7
	Reproducing (COPY, PFM) Specifications . . . . .	16-7
	Routine Type Specifier . . . . .	16-8
	Input File Specifications . . . . .	16-8
	Output File Specifications . . . . .	16-9
	Concatenation (CAT) Specifications . . . . .	16-9
	Routine Type Specifier . . . . .	16-10
	Concatenation File Specifications . . . . .	16-10
	Base File Specifications . . . . .	16-10
	Tape Parity Modifications . . . . .	16-11
	Examples . . . . .	16-11
	Sample Input Specifications . . . . .	16-11
	Sample Executions . . . . .	16-12

## TABLE OF CONTENTS (Cont)

Section	Title	Page
	Program Output . . . . .	16-12
	Output Files . . . . .	16-12
	Disk Files . . . . .	16-12
	Tape Files . . . . .	16-12
	Printer Listings . . . . .	16-12
	Card Files . . . . .	16-12
	Paper Tape Files . . . . .	16-13
	Error Messages . . . . .	16-13
17	FILE/LOADER . . . . .	17-1
	Operating Instructions . . . . .	17-1
	Specification Control Cards . . . . .	17-1
	Dollar Card . . . . .	17-1
	Dollar-Dollar Card (\$\$) . . . . .	17-1
	Asterisk Card . . . . .	17-2
	General Rules . . . . .	17-2
	Sample Execution Deck . . . . .	17-2
	Program Output . . . . .	17-3
	Files On Disk . . . . .	17-3
	Informative Messages . . . . .	17-3
	Error Messages . . . . .	17-3
18	FILE/PUNCHER . . . . .	18-1
	Operating Instructions . . . . .	18-1
	Program Output . . . . .	18-1
	Error Message . . . . .	18-1
19	FOREIGN/TAPECOPY . . . . .	19-1
	Operating Instructions . . . . .	19-1
	Input Specifications . . . . .	19-3
	Error Messages . . . . .	19-11
	Internal Files . . . . .	19-14
20	INITIALIZE/ANALYZER . . . . .	20-1
	Operating Instructions . . . . .	20-1
	Program Output . . . . .	20-1
	Error Messages . . . . .	20-1
21	LOGCONVERT . . . . .	21-1
	Operating Instructions . . . . .	21-1
	Output File Format . . . . .	21-1
	Record Types . . . . .	21-1
	General Information . . . . .	21-2
	Record Format Tables . . . . .	21-2
	Sample Program Record Declarations . . . . .	21-12
	COBOL Record Formats . . . . .	21-12
	RPGII Record Formats . . . . .	21-19
22	PACK/INIT . . . . .	22-1
	Operating Instructions . . . . .	22-1
	Console Keyboard Execution . . . . .	22-1
	Card Reader Execution . . . . .	22-2
	Run Time Options . . . . .	22-2

## TABLE OF CONTENTS (Cont)

Section	Title	Page
	Initialization Specifications . . . . .	22-3
	User Interface Information . . . . .	22-3
	DRIVE . . . . .	22-3
	ACTION . . . . .	22-3
	SERIAL NUMBER . . . . .	22-4
	PACK ID . . . . .	22-4
	PACK TYPE . . . . .	22-4
	DATE . . . . .	22-4
	OWNER'S NAME . . . . .	22-4
	OPTIONS . . . . .	22-4
	CYLINDER NUMBER . . . . .	22-5
	Initialization Options . . . . .	22-5
	REMOVE Option . . . . .	22-5
	LIMIT Option . . . . .	22-5
	Marginal Sectors Option . . . . .	22-6
	Dollar Sign Options . . . . .	22-6
	Informative Messages . . . . .	22-7
	Relocation Notification . . . . .	22-7
	Error Messages . . . . .	22-8
23	QWIKLOG . . . . .	23-1
	Operating Instructions . . . . .	23-1
	Program Switches . . . . .	23-1
	Work Files . . . . .	23-2
	Description of Output . . . . .	23-2
	Summary Listing . . . . .	23-2
	Mnemonics . . . . .	23-4
	Error Messages . . . . .	23-4
	Program Traps . . . . .	23-6
24	SQUASH/USER.DISK . . . . .	24-1
	Operating Instructions . . . . .	24-1
	Integrity Check . . . . .	24-2
	Run-Time Options . . . . .	24-2
	Trace Functions . . . . .	24-3
	Program Output . . . . .	24-4
	Summary Listing . . . . .	24-4
	Squashed Disk . . . . .	24-4
	Error Recovery . . . . .	24-4
	Error Messages . . . . .	24-5
	Nonfatal errors . . . . .	24-5
	Fatal Errors . . . . .	24-6
25	SSLOAD/MAKCAS . . . . .	25-1
	Operating Instructions . . . . .	25-1
	Program Switches . . . . .	25-1
	CLEAR/START Cassettes . . . . .	25-2
	Stand-alone Utility Cassettes (SDL Utility Programs) . . . . .	25-2
	Instructions . . . . .	25-2
	Stand-alone SDL Utility Programs . . . . .	25-2
	Sample Execution Code . . . . .	25-3

## TABLE OF CONTENTS (Cont)

Section	Title	Page
	Error Correction . . . . .	25-3
	Error Messages . . . . .	25-3
26	STANDALONE/DISK.DUMP . . . . .	26-1
	Operating Instructions . . . . .	26-1
	Error Messages . . . . .	26-1
27	SYSTEM/BACKUP . . . . .	27-1
	Operating Instructions . . . . .	27-1
	Program Switches . . . . .	27-1
	Run-time Control Options . . . . .	27-3
	Autoprint Facility . . . . .	27-4
	Error Messages . . . . .	27-4
	Printer/Punch Backup File Format . . . . .	27-4
	Control Information . . . . .	27-5
	FPB . . . . .	27-5
	Label . . . . .	27-5
	Data Record . . . . .	27-5
	Carriage Control Field . . . . .	27-6
28	SYSTEM/BUILDTRAIN . . . . .	28-1
	Translate Table Format . . . . .	28-1
	Operating Instructions . . . . .	28-1
	Program Switches . . . . .	28-2
	Input Record Format . . . . .	28-2
	Standard Translate Tables . . . . .	28-3
29	SYSTEM/COMPARE . . . . .	29-1
	Operating Instructions . . . . .	29-1
	Program Switches . . . . .	29-2
	Command Specifications . . . . .	29-2
	Sample Execution Strings . . . . .	29-3
	Run-Time Interrogation . . . . .	29-3
	Output Messages . . . . .	29-4
	Fatal Errors . . . . .	29-4
	Errors . . . . .	29-4
	Warning Messages . . . . .	29-6
	Informative Messages . . . . .	29-7
30	SYSTEM/COPY . . . . .	30-1
	Operating Instructions . . . . .	30-1
	Syntax of Command Specification . . . . .	30-1
	Syntax Overview . . . . .	30-1
	Action Specifier . . . . .	30-2
	ADD . . . . .	30-2
	COPY . . . . .	30-3
	Options List . . . . .	30-3
	COMPARE Option . . . . .	30-3
	DATE Option . . . . .	30-4
	MANDATORY Option . . . . .	30-4
	SUMMARY Option . . . . .	30-5
	File Specification Clause . . . . .	30-5
	< file-id > . . . . .	30-5

## TABLE OF CONTENTS (Cont)

Section	Title	Page
	AS . . . . .	30-5
	ONTO . . . . .	30-5
	FAMILYINDEX . . . . .	30-6
	SAVEFACTOR . . . . .	30-6
	FROM Clause . . . . .	30-6
	<family-name> . . . . .	30-7
	DISK and " " . . . . .	30-7
	SERIALNO Attribute . . . . .	30-7
	KIND Attribute . . . . .	30-7
	HOSTNAME Attribute . . . . .	30-8
	VOLUMEINDEX Attribute . . . . .	30-8
	TO Clause . . . . .	30-8
	<family-name> . . . . .	30-9
	DISK and " " . . . . .	30-9
	SERIALNO Attribute . . . . .	30-9
	KIND Attribute . . . . .	30-10
	HOSTNAME Attribute . . . . .	30-10
	DENSITY Attribute . . . . .	30-10
	DIR Command . . . . .	30-10
	<tapename> . . . . .	30-11
	VOLUMEINDEX Attribute . . . . .	30-11
	AX Commands . . . . .	30-11
	TEACH/HELP Command . . . . .	30-12
	STATUS Command . . . . .	30-12
	QUIT Command . . . . .	30-12
	Program Switches . . . . .	30-12
	Time Required for COPY . . . . .	30-14
	Sample Input Strings for Copying . . . . .	30-14
	User Disk to Tape . . . . .	30-14
	System Disk to Tape . . . . .	30-14
	Disk to Disk . . . . .	30-15
	Tape-to-Tape . . . . .	30-15
	System Copy Output . . . . .	30-15
	File Formats . . . . .	30-15
	Error Handling . . . . .	30-16
	I/O Errors . . . . .	30-16
	In-Use Files . . . . .	30-16
	Missing Files . . . . .	30-17
	Disk Input Units . . . . .	30-17
	Tape Input Units . . . . .	30-17
	Duplicate File Identifier Checking . . . . .	30-17
	Rules of Precedence . . . . .	30-18
	Error Messages Displayed on the ODT . . . . .	30-18
	Syntax and Semantics Error Messages (Fatal Errors) . . . . .	30-22
	Syntax and Semantics Advisory Messages . . . . .	30-23
	Library Maintenance Messages . . . . .	30-23
	Messages Denoting Actions . . . . .	30-23

## TABLE OF CONTENTS (Cont)

Section	Title	Page
	Explanatory Messages . . . . .	30-24
	COMPARE Function Error Line Printer Listing . . . . .	30-25
31	SYSTEM/DISK.DUMP . . . . .	31-1
	Operating Instructions . . . . .	31-1
	Dump Specifications . . . . .	31-1
	Program Output . . . . .	31-2
	Output Disk . . . . .	31-2
	Informative Messages . . . . .	31-2
	Error Messages . . . . .	31-2
32	SYSTEM/DISK.INIT . . . . .	32-1
	Operating Instructions . . . . .	32-1
	Console Keyboard Execution . . . . .	32-1
	Card Reader Execution . . . . .	32-2
	Run-Time Options . . . . .	32-2
	Initialization Specifications . . . . .	32-3
	User Interface Information . . . . .	32-5
	Initialization Integrity Protection . . . . .	32-5
	Initialization Options . . . . .	32-5
	REMOVE Option . . . . .	32-5
	LIMIT Option . . . . .	32-5
	Marginal Sectors Option . . . . .	32-6
	Dollar Sign Option . . . . .	32-6
	Program Output . . . . .	32-7
	Informative Messages . . . . .	32-7
	Summary Listing . . . . .	32-7
	Error Messages . . . . .	32-8
33	SYSTEM/ELOGOUT . . . . .	33-1
	Operating Instructions . . . . .	33-1
	Execution . . . . .	33-1
	Program Switches . . . . .	33-1
	Description of Output . . . . .	33-2
	System Information . . . . .	33-2
	Operator Messages . . . . .	33-2
	Memory Error Report . . . . .	33-2
	Console Cassette Error Report (B 1800 and B 1900 only) . . . . .	33-3
	Unit Error Report . . . . .	33-3
	Unit Error Summary . . . . .	33-4
34	SYSTEM/FILE.INIT . . . . .	34-1
	Relative Files . . . . .	34-1
	Initialization . . . . .	34-1
	Operating Instructions . . . . .	34-2
	Operator Execution . . . . .	34-2
	MCPII Internal Use . . . . .	34-3
	Switch Settings . . . . .	34-3
	Error Messages . . . . .	34-4
	Internal Files . . . . .	34-5



## TABLE OF CONTENTS (Cont)

Section	Title	Page
35	SYSTEM/ICMD.INIT . . . . .	35-1
	Operating Instructions . . . . .	35-1
	Error Messages . . . . .	35-2
36	SYSTEM/IS.MAINT . . . . .	36-1
	Operating Instructions . . . . .	36-1
	Execution . . . . .	36-1
	Input From a Data File . . . . .	36-1
	Input from the ODT . . . . .	36-1
	Command Syntax Overview . . . . .	36-3
	CHANGE Command . . . . .	36-3
	COPY . . . . .	36-3
	END, EOJ, or <job#>AX<null> . . . . .	36-4
	HELP . . . . .	36-4
	LIST . . . . .	36-4
	NEXT . . . . .	36-4
	REMOVE . . . . .	36-5
	TEACH . . . . .	36-5
	UPDATE . . . . .	36-5
	VERIFY . . . . .	36-6
	Error Messages . . . . .	36-6
	Internal Files . . . . .	36-8
37	SYSTEM/LDCONTRL . . . . .	37-1
	Operating Instructions . . . . .	37-1
	General Rules . . . . .	37-1
38	SYSTEM/LOAD.CAS . . . . .	38-1
	Error Messages . . . . .	38-1
39	SYSTEM/LOAD.DUMP . . . . .	39-1
	Operating Instructions . . . . .	39-1
	Program Switches . . . . .	39-1
	Control Specifications . . . . .	39-1
40	SYSTEM/LOGOUT . . . . .	40-1
	Operating Instructions . . . . .	40-1
	Program Output . . . . .	40-1
	CLEAR/START Records . . . . .	40-1
	Job Information Records . . . . .	40-1
	File Information records . . . . .	40-2
41	SYSTEM/MAKEUSER . . . . .	41-1
	Operating Instructions . . . . .	41-1
	File Security Attributes . . . . .	41-2
	Commands . . . . .	41-4
	ADD Command . . . . .	41-4
	CHANGE Command . . . . .	41-6
	COPY Command . . . . .	41-7
	CREATE Command . . . . .	41-8
	DEBUG Command . . . . .	41-9
	DELETE Command . . . . .	41-9
	DISPLAY Command . . . . .	41-10
	END/EOJ Command . . . . .	41-10

## TABLE OF CONTENTS (Cont)

Section	Title	Page
	LIST Command . . . . .	41-11
	PUNCH Command . . . . .	41-12
	The PURGE Command . . . . .	41-12
	Error Messages . . . . .	41-13
42	SYSTEM/MARK.SEGS . . . . .	42-1
	Operating Instructions . . . . .	42-1
	Input Specifications . . . . .	42-1
	Program Output . . . . .	42-2
43	SYSTEM/ODT . . . . .	43-1
	Operating Instructions . . . . .	43-1
	KB Instruction . . . . .	43-1
	Files . . . . .	43-3
	Memory Requirements . . . . .	43-3
44	SYSTEM/ODTLOGOUT . . . . .	44-1
	Operating Instructions . . . . .	44-1
	Switch Settings . . . . .	44-1
	Error Messages . . . . .	44-2
	Internal Files . . . . .	44-2
45	TAPECOPY . . . . .	45-1
	Operating Instructions . . . . .	45-1
	Input Specifications . . . . .	45-1
	Program Switches . . . . .	45-3
A	DISK DEVICE CHARACTERISTICS . . . . .	A-1
	Disk Cartridge Characteristics . . . . .	A-1
	Physical Characteristics . . . . .	A-1
	Initialization Error Limits . . . . .	A-1
	Initialization Characteristics . . . . .	A-1
	B 9484 (205 and 206) Disk Pack Characteristics . . . . .	A-1
	Physical Characteristics . . . . .	A-1
	Initialization Error Limits . . . . .	A-2
	Initialization Characteristics . . . . .	A-2
	B 9499 (215, 225 and 207) Disk Pack Characteristics . . . . .	A-2
	Physical Characteristics . . . . .	A-2
	Initialization Error Limits . . . . .	A-2
	Initialization Characteristics . . . . .	A-3
B	SYNTAX CONVENTIONS . . . . .	B-1
	Required Items . . . . .	B-2
	Optional Items . . . . .	B-2
	Loops . . . . .	B-2
	Bridges . . . . .	B-3

## LIST OF ILLUSTRATIONS

Figure	Title	Page
10-1	Sample Card Deck Execution . . . . .	10-1
10-2	Sample Disk File Execution . . . . .	10-2
10-3	Sample Deck Using All Options . . . . .	10-3
21-1	Clear/Start . . . . .	21-19
21-2	Schedule . . . . .	21-19
21-3	Beginning-of-Job . . . . .	21-20
21-4	File Open . . . . .	21-21
21-5	File Close . . . . .	21-22
21-6	DMS Statistics . . . . .	21-23
21-7	DMS Open/Close . . . . .	21-24
21-8	Peripheral Assign/Release . . . . .	21-24
21-9	End-of-Job . . . . .	21-25
21-10	Log Transfer . . . . .	21-25
27-1	Format of the Carriage Control Field . . . . .	27-6

## LIST OF TABLES

Table	Title	Page
5-1	Name Table Format . . . . .	5-4
10-1	Character Sets and Record Sizes . . . . .	10-1
11-1	DISK/ALLOCATOR Attribute Keywords and Default Values . . . . .	11-1
15-1	DISKMAP/UTILITY Program Switches . . . . .	15-2
15-2	DISKMAP/UTILITY Area Address Records . . . . .	15-6
15-3	DISKMAP/UTILITY Numeric and Mnemonic Value . . . . .	15-6
16-1	DMPALL Program Switches . . . . .	16-1
16-2	DMPALL Hardware Device Notation-LIST Command . . . . .	16-4
16-3	DMPALL Record Selection Parameters . . . . .	16-6
16-4	DMPALL Hardware Device Notation . . . . .	16-8
19-1	FOREIGN/TAPECOPY Program Switches . . . . .	19-1
21-1	LOGCONVERT CLEAR/START Record Format . . . . .	21-3
21-2	LOGCONVERT SCHEDULE Record Format . . . . .	21-3
21-3	LOGCONVERT BOJ Record Format . . . . .	21-4
21-4	LOGCONVERT File Open Record Format . . . . .	21-4
21-5	LOGCONVERT File Close Record Format . . . . .	21-6
21-6	LOGCONVERT DMS Statistics Record Format . . . . .	21-8
21-7	LOGCONVERT DMS Open/Close Record Format . . . . .	21-8
21-8	LOGCONVERT Peripheral Assignment/Release . . . . .	21-9
21-9	LOGCONVERT EOJ Record Format . . . . .	21-9

## LIST OF TABLES (Cont)

Table	Title	Page
21-10	LOGCONVERT Comment Record Format . . . . .	21-10
21-11	LOGCONVERT Log Transfer Record Format . . . . .	21-10
21-12	LOGCONVERT Hardware Type Definitions . . . . .	21-10
21-12	LOGCONVERT Hardware Type Definitions . . . . .	21-11
21-13	LOGCONVERT File Types . . . . .	21-11
21-14	LOGCONVERT Access Modes . . . . .	21-12
21-15	LOGCONVERT Execute Types . . . . .	21-12
21-16	Clear/Start . . . . .	21-13
21-17	Schedule . . . . .	21-13
21-18	Beginning-of-Job . . . . .	21-14
21-19	File Open . . . . .	21-14
21-20	File Close . . . . .	21-15
21-21	DMS Statistics . . . . .	21-16
21-22	DMS Open/Close . . . . .	21-17
21-23	Peripheral Assignment/Release . . . . .	21-17
21-24	End-of-Job . . . . .	21-18
21-25	Log Transfer . . . . .	21-18
21-26	Comment . . . . .	21-18
25-1	SSLOAD/MAKCAS Program Switches . . . . .	25-1
27-1	SYSTEM/BACKUP Program Switches . . . . .	27-2
27-2	Control Information Record . . . . .	27-5
27-3	TYPE Field Definition . . . . .	27-6
27-4	Values for Printer Spacing/Skipping . . . . .	27-7
27-5	Values for Punch Stacker-Selection . . . . .	27-7
28-1	SYSTEM/BUILDTRAIN Program Switches . . . . .	28-2
28-2	1100/1500 LPM Train Printer . . . . .	28-4
28-3	400/750 LPM Train Printer . . . . .	28-5
30-1	SYSTEM/COPY Program Switches . . . . .	30-13
34-1	File Attributes . . . . .	34-2
41-1	SYSTEM/MAKEUSER Security Attributes . . . . .	41-2
45-1	TAPECOPY Program Switches . . . . .	45-3

## PREFACE

The System Software Operation Guide (SOG), in two volumes, describes the operator interface to Burroughs B 1000 computer systems.

Volume 1 contains a general description of the B 1000 system and documents the operator-interface commands that are recognized by the Master Control Program II (MCPII).

Volume 2 contains an overview of available system utility programs, as well as detailed descriptions and operating instructions for these programs. The overview is presented in section 1. The programs are described individually in sections 2 through 45. The sequence is alphabetical, by program name:

2	CART/INIT	24	SQUASH/USER.DISK
3	CASSETTE/MAKER	25	SSLOAD/MAKCAS
4	CHECK/LOAD.DUMP	26	STANDALONE/DISK.DUMP
5	CLEAR/START	27	SYSTEM/BACKUP
6	CODE/ANALYZER	28	SYSTEM/BUILDTRAIN
7	COLDSTART/DISK	29	SYSTEM/COMPARE
8	COLDSTART/TAPE	30	SYSTEM/COPY
9	CONVERT/BACKUP	31	SYSTEM/DISK.DUMP
10	CREATE/TABLE	32	SYSTEM/DISK.INIT
11	DISK/ALLOCATOR	33	SYSTEM/ELOGOUT
12	DISK/DUMP	34	SYSTEM/FILE.INIT
13	DISKPACK/INTERCHANG	35	SYSTEM/ICMD.INIT
14	DISKETTE/COPY	36	SYSTEM/IS.MAINT
15	DISKMAP/UTILITY	37	SYSTEM/LDCONTRL
16	DMPALL	38	SYSTEM/LOAD.CAS
17	FILE/LOADER	39	SYSTEM/LOAD.DUMP
18	FILE/PUNCHER	40	SYSTEM/LOGOUT
19	FOREIGN/TAPECOPY	41	SYSTEM/MAKEUSER
20	INITIALIZE/ANALYZER	42	SYSTEM/MARK.SEGS
21	LOGCONVERT	43	SYSTEM/ODT
22	PACK/INIT	44	SYSTEM/ODTLOGOUT
23	QWIKLOG	45	TAPECOPY

Volume 2 also includes appendixes. Appendix A, Disk Device Characteristics, provides physical characteristics and initialization information on disk cartridge and disk pack units. Appendix B, Notations and Syntax Conventions, provides a definitive description of the use of railroad syntax.



## SECTION 1 INTRODUCTION

The system utility programs for Burroughs B 1000 systems are designed for efficient and consistent operation of the computer systems. The system utility programs are classified and documented in this volume.

### CLASSIFICATION OF UTILITY PROGRAMS

All utility programs are either normal-state or stand-alone programs. Normal-state utility programs are executed under MCPII control. Stand-alone utility programs are loaded from a cassette and require the MCPII to be inoperative when they are executed.

All of the normal-state and stand-alone utility programs are further classified according to function and are described briefly in this introduction. These classifications allow the operator who knows what task needs to be accomplished to select the utility program that will best do the job. There is some overlap in these classifications because many of the utility programs are versatile and can accomplish many tasks. The major classifications are:

- Start-up Programs
- Copy/File Copy Programs
- Disk Initializer Programs
- General Maintenance Programs
- Log Analysis Programs

The normal-state and stand-alone utility programs are listed alphabetically in the following, and the major classification of each utility program is noted. These lists are followed by general descriptions of the functions of the utility programs included in each of the major classifications.

Normal-State Program Name	Classification
CASSETTE/LOADER (*)	Copy/File Copy
CASSETTE/MAKER	Copy/File Copy
CHECK/LOAD.DUMP	Copy/File Copy
CODE/ANALYZER	General Maintenance
CONVERT/BACKUP	Copy/File Copy
CREATE/TABLE	General Maintenance
DISK/ALLOCATOR	General Maintenance
DISKETTE/COPY	Copy/File Copy
DISKMAP/UTILITY	General Maintenance
DISKPACK/INTERCHANG	General Maintenance
DMPALL	Copy/File Copy
FILE/LOADER	Copy/File Copy
FILE/PUNCHER	Copy/File Copy
FOREIGN/TAPECOPY	Copy/File Copy
INITIALIZE/ANALYZER	General Maintenance

B 1000 Systems SSO, Volume 2  
Introduction

<b>Normal-State Program Name</b>	<b>Classification</b>
LOGCONVERT	Log Analysis
PACK/INIT	Disk Initializer
QWIKLOG	Log Analysis
SQUASH/USER.DISK	General Maintenance
SSLOAD/MAKCAS	Copy/File Copy
SYSTEM/BACKUP	Copy/File Copy
SYSTEM/BUILDTRAIN	Start-up
SYSTEM/COMPARE	General Maintenance
SYSTEM/COPY	Copy/File Copy
SYSTEM/DISK.DUMP	Copy/File Copy
SYSTEM/DISK.INIT	Disk Initializer
SYSTEM/ELOGOUT	Log Analysis
SYSTEM/ICMD.INIT	Disk Initializer
SYSTEM/IS.MAINT	General Maintenance
SYSTEM/LDCONTRL	Copy/File Copy
SYSTEM/LOAD.CAS	Copy/File Copy
SYSTEM/LOAD.DUMP	Copy/File Copy
SYSTEM/LOGOUT	Log Analysis
SYSTEM/MAKEUSER	Start-up
SYSTEM/MARK.SEGS	General Maintenance
SYSTEM/ODT	General Maintenance
SYSTEM/ODTLOGOUT	Log Analysis
TAPECOPY	Copy/File Copy

\* Program not documented because it is not explicitly executed.

<b>Stand-Alone Program Name</b>	<b>Classification</b>
CART/INIT	Disk Initializer
CLEAR/START	Start-up
COLDSTART/TAPE	Start-up
COLDSTART/DISK	Start-up
DISK/DUMP	Copy/File Copy
PACK/INIT	Disk Initializer
STANDALONE/DISK.DUMP	Copy/File Copy
STANDALONE/INTERCHANG (see DISKPACK/INTERCHANG)	General Maintenance



## GENERAL DESCRIPTION OF UTILITY PROGRAMS

### System Start-Up Programs

The System Start-up utility programs bring the system and the MCPII to an operable state. A brief description of each program follows.

#### CLEAR/START

The CLEAR/START program initiates the MCPII and gives the MCPII control of the system.

#### COLDSTART/DISK

The COLDSTART/DISK program initially loads system software and constructs required tables. Software source media is a disk pack.

#### COLDSTART/TAPE

The COLDSTART/TAPE program initially loads system software and constructs required tables. Software source media is a library (COPY) tape labeled SYSTEM.

#### SYSTEM/BUILDTRAIN

The SYSTEM/BUILDTRAIN program creates translate tables used by line printers. Refer to the LT system control instruction in volume 1, section 2 for additional information.

#### SYSTEM/MAKEUSER

The SYSTEM/MAKEUSER program supports the file (SYSTEM)/USERCODE which is required for file security. File security is optional.

#### NOTE

All disk packs or disk cartridges must be initialized prior to use. Refer to Disk Initializer Routines in this section for additional information.

### Copy/File Copy Programs

Copy/File Copy utility programs reproduce files. Files can be altered during the copy, or moved from one medium to another, or copied exactly.

#### Disk File Copy

A disk dump is a full copy of a disk in which all files on a disk are copied to another disk. Both normal-state and stand-alone utilities can dump a disk. A single-file copy reproduces only the files or family of files specified. All single file copy programs are normal-state utility programs. A brief description of each disk file copy program follows.

#### CHECK/LOAD.DUMP

The CHECK/LOAD.DUMP program compares the file(s) copied by SYSTEM/LOAD.DUMP to the original file(s) to ensure an accurate copy.

#### DISK/COPY

The DISK/COPY program copies disk files to a new disk pack or to a different location on the original disk.

#### DISK/DUMP

The DISK/DUMP program is a stand-alone disk dump program; performs a sector-by-sector copy.

#### DMPALL

The DMPALL program is a generalized media conversion program that lists disk file contents, copies data files between hardware devices, and also can move disk files from one pack to another.

#### STANDALONE/DISK.DUMP

The STANDALONE/DISK.DUMP program is a stand-alone disk dump program; performs a file-by-file copy, which results in a squashed output disk.

#### SYSTEM/COPY

The SYSTEM/COPY program is a generalized library maintenance program. It copies files between tape and disk, disk and disk, and tape and tape.

#### NOTE

The following programs duplicate SYSTEM/COPY functions and are being phased out of use as system utility programs.

#### SYSTEM/DISK.DUMP

The SYSTEM/DISK.DUMP program is a normal-state disk dump program; performs either a file-by-file or a sector-by-sector copy.

#### SYSTEM/LOAD.DUMP

The SYSTEM/LOAD.DUMP program copies files from a LOAD.DUMP tape to disk. Primary operation is through the AD and LOAD system control instructions. Refer to documentation on these commands in volume 1, section 2 for additional information.

#### Tape File Copy

The B 1000 computer system reads library (COPY) tapes, Load Dump tapes, and unlabeled tapes; both 9-track and 7-track tapes can be read. A brief description of each tape file copy program follows.

#### CHECK/LOAD.DUMP

The CHECK/LOAD.DUMP compares disk files with the original LOAD DUMP tape to ensure an accurate copy.

#### DMPALL

The DMPALL program is a generalized media conversion routine. It prints the directory of Load Dump library tapes, copies a file to or from tape, and handles 7-track tape conversions with even or odd parity. Handles translation to or from EBCDIC code recognized by the system.

#### FOREIGN/TAPECOPY

The FOREIGN/TAPECOPY program copies or lists any 7-track or 9-track magnetic tape to disk, tape or printer.

#### SYSTEM/COPY

The SYSTEM/COPY program is a generalized library maintenance program. It prints the directory of COPY library tapes, copies files to or from tape, and can handle 7-track tape conversions. Handles translation to or from the EBCDIC code recognized by the system.

#### NOTE

The following tape file copy programs duplicate SYSTEM/COPY functions and are being phased out of use.

#### SYSTEM/LOAD.DUMP

The SYSTEM/LOAD.DUMP program creates disk files from LOAD DUMP tapes.

#### TAPECOPY

The TAPECOPY program duplicates, merges, compares, and concatenates non-library multifile or single-file tapes.

#### Printer Listings

A line printer listing of the contents of any file can be generated. When a line printer is not available to a printer file, a printer backup file is created. The operator can request that a printer backup file be created for specific files. Line printer listings can be generated programmatically without using either the DMPALL or SYSTEM/BACKUP utility programs. The following utility programs produce printer listings of file contents.

#### DMPALL

The DMPALL program lists any file in alphanumeric, numeric, or hexadecimal format.

#### SYSTEM/BACKUP

The SYSTEM/BACKUP program lists printer backup files. Primary operation is through the PB input message. Refer to PB and AB input messages in volume 1, section 2 for additional details about printing backup files.

#### Cassette File Copy

The cassette is the system input medium for stand-alone operations. On some systems, cassettes can also be used as peripheral input/output devices. The programs used to create and to read cassettes are described in the following paragraphs.

#### CASSETTE/LOADER

The CASSETTE/LOADER program is a stand-alone program loaded onto a cassette by the SSLOAD/MAKCAS program. It is a bootstrap loader for the remainder of the cassette.

#### CASSETTE/MAKER

The CASSETTE/MAKER program writes files to a cassette. The cassette system must be in the input/output mode.

#### SSLOAD/MAKCAS

The SSLOAD/MAKCAS program creates stand-alone utility cassettes.

#### SYSTEM/LOAD.CAS

The SYSTEM/LOAD.CAS program loads files from a cassette to disk for access by the system.

#### Card File Copy

The punched card is both an input and an output medium for the B 1000 system. The following programs are used to read and punch card files.

#### DMPALL

The DMPALL program copies files to or from cards.

#### FILE/LOADER

The FILE/LOADER program loads a card deck created by the FILE/PUNCHER program onto disk. This provides for easy transportation of files.

#### FILE/PUNCHER

The FILE/PUNCHER program punches a file. The file must be loaded by the FILE/LOADER program.

#### SYSTEM/BACKUP

The SYSTEM/BACKUP program punches a card (punch) backup file. Primary operation is through the PB input message. System-named backup punch files have the name format %nnnn.

#### SYSTEM/LDCONTRL

The SYSTEM/LDCONTRL program loads a card deck into a special disk file called a pseudo file and named DECK/<#>. This pseudo file is read when a pseudo reader is activated with the RN input message. Primary operation is through the LD input message. Refer to the LD input message in volume 1, section 2 for additional information.

#### Diskette File Copy

The industry-compatible mini-disk (diskette or floppy-disk) is both an input and output medium for the B 1000 computer system. All access to data on mini-disks is through the utility program DISKETTE/COPY. Mini-disks are initialized with the utility program SYSTEM/ICMD.INIT.

#### DISKETTE/COPY

The DISKETTE/COPY program copies files to or from industry compatible mini-disks (diskettes). The PSR option provides automatic loading of pseudo reader files stored on the diskette. Purges diskettes. Generates a KA listing of files on a diskette. Relabels diskettes.

### Disk Initializer Programs

Disk packs and disk cartridges must be initialized prior to their use on the B 1000 computer system. During initialization, disks are assigned a pack or cartridge type, all sectors are verified to be readable, and the disk is written with an initialization pattern. Tapes are scratched or purged rather than initialized. Industry compatible mini-disks must be initialized with the utility program SYSTEM/ICMD.INIT. The following programs are used to initialize disk devices.

#### CART/INIT

The CART/INIT program is a stand-alone utility program which initializes and verifies disk cartridges. Allowable types are System, Unrestricted (User), and Restricted.

#### PACK/INIT

The PACK/INIT program is a stand-alone utility program which initializes, verifies, and reconfigures disk packs. Allowable types are System, Unrestricted (User), and Restricted.

#### SYSTEM/DISK.INIT

The SYSTEM/DISK.INIT program is a normal-state utility program which initializes, verifies, and reconfigures disk packs and disk cartridges. Allowable types are System, Unrestricted (User), and Restricted. Additionally, B 9499-7 disk packs can be initialized as Interchange. Interchange packs allow data on the pack to be exchanged between Burroughs small, medium, and large system computers.

#### SYSTEM/ICMD.INIT

The SYSTEM/ICMD.INIT program is a normal-state utility program which initializes industry compatible mini-disks (diskettes).

## General Maintenance Programs

The utility programs listed in this subsection provide access to disk file information and provide standard, useful system maintenance functions.

### CODE/ANALYZER

The CODE/ANALYZER program generates a line printer listing of program code segments, and internal file identifiers and attributes.

### DISK/ALLOCATOR

The DISK/ALLOCATOR program creates a file located at a specific disk address.

### DISKMAP/UTILITY

The DISKMAP/UTILITY program provides a listing of the status and the integrity of disk areas.

### DISKPACK/INTERCHANG

The DISKPACK/INTERCHANG program is a normal-state program that copies and translates data to or from an interchange pack. Interchange packs allow data to be exchanged between non-B1000 Burroughs systems.

### INITIALIZE/ANALYZER

The INITIALIZE/ANALYZER program analyzes an initialized disk and provides a printer listing of relocated and removed sectors and of all label information.

### SQUASH/USER.DISK

The SQUASH/USER.DISK program creates compact storage on a operator disk by moving files and consolidating available space.

### STANDALONE/INTERCHANG

The STANDALONE/INTERCHANG program is a stand-alone program that copies data to or from an interchange pack with translation. Interchange packs allow data to be exchanged between Burroughs small, medium, and large system computers.

### SYSTEM/COMPARE

The SYSTEM/COMPARE program compares files and flags the differences between the file. This program is useful in isolating hardware problems.

### SYSTEM/IS.MAINT

The SYSTEM/IS.MAINT program is a normal-state program that performs library maintenance functions on index-sequential files.

### SYSTEM/MARK.SEGS

The SYSTEM/MARK.SEGS program allows the operator to mark code segments as important for use by the System Memory Management algorithms.

### SYSTEM/ODT

The SYSTEM/ODT program handles all I/O operations on the Operator Display Terminal (ODT).

## Log Analysis Programs

The system optionally maintains three logfiles (ODTLOG, SYSTEM/LOG, and ELOG) which record all system activity. The following programs access these logfiles.

B 1000 Systems SSOG, Volume 2  
Introduction

**LOGCONVERT**

The LOGCONVERT program creates a copy of a system log file in a packed COBOL-readable and RPGII-readable format. Creates file NEW.LOG/<#>.

**QWIKLOG**

The QWIKLOG program provides a compact analysis of a system log file. Report can be restricted to specified job numbers. A graph and summary report can be produced.

**SYSTEM/ELOGOUT**

The SYSTEM/ELOGOUT program analyzes the error log, which contains hardware failure information. Primary operation is through the ET input message. Refer to the ET input message in volume 1, section 2 for additional information.

**SYSTEM/LOGOUT**

The SYSTEM/LOGOUT program analyzes the system log in complete detail. Primary operation is through the LG input message. Refer to the LG input message in volume 1, section 2 for additional information.

**SYSTEM/ODTLOGOUT**

The SYSTEM/ODTLOGOUT program transfers, formats, and prints a log of ODT (operator display terminal) messages. Primary operation is through LG ODT input message. Refer to the LG input message in volume 1, section 2 for additional information.

**TABS package**

The TABS package analyzes the log files and provides system use and billing statistics. Refer to the B 1000 Systems Time Analysis and Billing System Reference Manual, Form Number 1090669, for additional information.

## SECTION 2

### CART/INIT

The CART/INIT program is a stand-alone utility program that initializes disk cartridges. Disk cartridges are single platter disk units used as storage media on the B 1000 computer system. Every disk cartridge must be initialized before it can be used on the system. Disk initialization accomplishes three tasks. First, it assigns addresses to all segments on the disk. Second, it checks to see what segments, if any, are unusable (cannot be read from or written to). Any segment found to have errors causes the entire track in which it resides to be removed from the Master Available Table. If errors occur in track zero or one, the entire pack is considered faulty and cannot be used on the system. Third, skeleton table entries (the disk directory and available tables, for example) are built and the label is written in segment zero.

Refer to appendix A of this volume for disk cartridge characteristics.

### OPERATING INSTRUCTIONS

The CART/INIT program does not operate under the control of the MCPII. It must be loaded and executed through the cassette reader on the control panel, as follows:

#### NOTE

With dual processor systems, the operator must press the console interrupt switch before attempting to execute the CART/INIT program.

1. Place the disk cartridge initializer cassette in the cassette reader in the control panel. If the BOT light is not lit, depress the RWND (rewind) pushbutton and wait for the BOT light to be lit.
2. Place the console printer on-line, or place the console display in receive mode.
3. Set the system MODE pushbutton to the TAPE position and press the CLEAR and START pushbuttons respectively. This loads the bootstrap loader from the cassette tape and halts the processor. The L register must be equal to @AAAAAA@ at this time.
4. Set the system MODE pushbutton to the RUN position and press the START pushbutton (do not press the CLEAR pushbutton). This loads and executes the system initializer program.

When the disk cartridge initializer program begins execution, it displays the following message on the operator display terminal (ODT).

**DISK CARTRIDGE INITIALIZER - MARK <level-number> -**

The operator should verify that the current version of the disk cartridge initializer program is being used.

The following prompt messages are then displayed, each requiring a response:

**WHICH CARTRIDGE - DC <X> OR LEAVE BLANK TO TERMINATE**

Enter only the single letter needed to specify the cartridge. After initializing the specified cartridge, the program repeats this prompt sequence, and another disk cartridge can be initialized.

**VERIFICATION ONLY? - <YES OR NO>**

Verification ensures that all segments are usable. The disk cartridge is always verified. A NO response causes the program to initialize the disk cartridge as well as to verify it.

B 1000 Systems SSOG, Volume 2  
CART/INIT

ENTER 6 DIGIT SERIAL NUMBER

This entry should be a unique six-digit number.

ENTER PACK.ID

The pack-id is a 10-character alphanumeric field. It cannot contain embedded blanks.

ENTER CARTRIDGE TYPE - <U, S, OR R>

U = Unrestricted (User); S = System; R = Restricted.

ENTER JULIAN DATE - <YYDDD>

The DDD portion of the Julian date represents the number of the day, beginning with January 1 as 001, and December 31 as 365.

ENTER OWNERS NAME

This entry is a 10-character alphanumeric field. It cannot contain embedded blanks.

When initialization and/or verification is complete, the following messages are displayed.

```
ID= <PACK.ID> SER#= <integer><integer> BAD SECTORS  
INITIALIZATION COMPLETE DC <X>  
WHICH CARTRIDGE? -DC <X> OR LEAVE BLANK TO TERMINATE
```

At this time, an additional disk cartridge can be initialized or verified, or the program can be terminated. The prompt sequence is repeated. Entering a null response to the first prompt terminates the program.



## SECTION 3

### CASSETTE/MAKER

The CASSETTE/MAKER program is a normal-state utility program that writes files to a cassette in a format that can be read by the SYSTEM/LOAD.CAS utility program. Cassettes can be created only by a B 1800 system that has a magnetic tape cassette I/O control, or by a B 1700 system that has a magnetic tape cassette subsystem (not the console cassette drive).

### OPERATING INSTRUCTIONS

The CASSETTE/MAKER program is executed from either the card reader or from the console keyboard, but the program expects to receive input specifications from the card file labeled CARDS.

### INPUT SPECIFICATIONS

The optional control keyword VERIFY requests verification of each cassette to be performed immediately after the cassette is written. If verification is desired, the first input specification must be a VERIFY control card. The following is the syntax of this control option where the dollar sign (\$) character must begin in position one of the record.

```
$VERIFY
```

The keyword VERIFY can be abbreviated V.

The specification strings in the input file CARDS contain the file-identifiers of the files to be written to the cassette, one per card, in columns 1 through 80.

### SAMPLE EXECUTION COMMANDS

The following set of sample execution commands creates a cassette containing the files RPGII, PAYROLL/APPLICATION, and USER/TEST/FILE. If necessary, the files are continued on additional cassettes, and each cassette is verified after it is written.

```
?EX CASSETTE/MAKER  
?DATA CARDS  
$V  
RPG  
PAYROLL/APPLICATION  
USER/TEST/FILE  
?END
```

### PROGRAM OUTPUT

#### Files Created

The utility program CASSETTE/MAKER writes files to a scratch cassette tape. Several files can be written to a single cassette. If necessary, additional scratch cassettes are used to hold continuation files.

B 1000 Systems SSOG, Volume 2  
CASSETTE/MAKER

### **Informative Messages**

After processing each cassette, the CASSETTE/MAKER program issues the following message on the Operator Display Terminal (ODT):

VERIFICATION COMPLETE ON REEL <n>  
CSx LOCKED [<serial number>]

### **Error Messages**

<file-identifier> NOT ON DISK

No files are written to the cassette.

DISK PARITY ERROR - UNABLE TO VERIFY

The temporary disk file used for verification has a parity error.

VERIFY ERROR REC# <record number>

An error was encountered during verification of the cassette.

WOULD YOU LIKE TO REMAKE THIS CASSETTE? <YES OR NO>

VERIFY ERROR message. A YES response to this inquiry causes the CASSETTE/MAKER program to remake the cassette in error. A NO response causes CASSETTE/MAKER to continue, ignoring the cassette in error.

INVALID REMAKE - DISK PARITY ERROR

The temporary disk file used to remake the cassette contains a parity error.

## SECTION 4

### CHECK/LOAD.DUMP

The CHECK/LOAD.DUMP program compares the files on a Load Dump tape (created by the SYSTEM/LOAD.DUMP program prior to the Mark 9.0 System Software release) to those files of the same name on disk. A line printer (or backup file) listing of the files on the tape is produced that indicates:

1. Relative file number
2. File-identifier
3. File type
4. Blocking factor
5. End-of-file pointer
6. Creation/compilation date

When comparison errors are detected during file comparison, the listing also contains a message indicating the type of comparison error.

In addition, CHECK/LOAD.DUMP provides the following capabilities:

1. Ability to limit comparison to certain specified files on the LOAD DUMP tape rather than the entire tape.
2. Ability to interrogate the current status of the program, including file-identifier, file-number, and cumulative error count.
3. Ability to request detailed analysis of errors on a separate line printer listing.

## OPERATING INSTRUCTIONS

To compare an entire Load Dump tape with CHECK/LOAD.DUMP, enter the following control message:

```
EXECUTE CHECK/LOAD.DUMP FILE T NAME= <Load Dump tape-id>;
```

If the FILE clause is not included in the EXECUTE string, the operator can identify the tape with an IL command when the input tape is requested by the program. If the files to be compared are located on a user disk rather than the system disk, include the following FILE clause in the EXECUTE string:

```
FILE D PACK.ID= <operator-pack-id> ;
```

## PROGRAM SWITCHES

The CHECK/LOAD.DUMP program is sensitive to the values of switches 0, 1, and 2. Switch 0 restricts comparison to requested files. Switch 1 requests detailed error analysis. Switch 2 suppresses printing of the standard output listing. The default value for each switch is zero (0), and a switch is considered set when it is set to any non-zero value. Details of the action of each switch follow:

### Switch 0

Requests that comparison be performed only on certain named files. When this switch is set, the program issues prompts at the operator display terminal which request the names of the files to be compared. The file-identifiers must be entered one name per accept (AX) message, and a blank

B 1000 Systems SSOG, Volume 2  
CHECK/LOAD.DUMP

message terminates the input. The file-identifier can have the format <family-id>/=, =/<file-id>, or =/=. The listing produced contains information only for the files specified in the accept messages.

Switch 1

Requests an additional listing which gives a detailed analysis of each library tape error. For "HEADER COMPARE" errors, the relevant fields from both the disk file header and the tape file header are printed. These fields include the file type, record size, records-per-block, and the end-of-file pointer. For FILE COMPARE errors, both the disk and the tape buffers are dumped in hexadecimal format, and each 4-bit character that does not compare is flagged with an asterisk (\*).

Switch 2

Suppresses printing of the standard output listing. The total number of comparison errors found on the tape is also displayed on the operator display terminal prior to EOJ of the program. This switch does not affect the detailed analysis listing.

The following example represents an execution of CHECK/LOAD.DUMP in which the operator specifies which files are to be compared (switch 0 = 1).

```
EXECUTE CHECK/LOAD.DUMP SWITCH 0=1 FILE T NAME=LIBRARY
CHECK/LOAD.DUMP =1 BOJ. ...
% CHECK/LOAD.DUMP =1 ENTER FILE NAMES
CHECK/LOAD.DUMP =1 ACCEPT.
1AXSDL/=
1AXCOBOL
1AX=/INTERP1M
1AX
```

The listing produced during the file comparison contains information describing only the files specified in the accept (AX) messages.

## RUN-TIME INTERROGATION

Once file comparison has started, the CHECK/LOAD.DUMP program can be interrogated for its current status by entering one of the following messages:

<job #> AX STATUS

<job #> AX ST

The response describes the file currently being compared and lists its file-identifier and its file-number on the Load Dump tape. Also, the cumulative error count for the entire tape is given.

## SECTION 5

### CLEAR/START and Memory Dump Procedure

The CLEAR/START program is a stand-alone utility procedure used to bring the system to an operable state. This procedure, which actually consists of two programmatic routines (the CLEAR/START and System Initializer programs), must be performed after any of the following situations occur:

1. A coldstart-tape or coldstart-disk operation.
2. System power-up. (For example, at the beginning of the day.)
3. A system failure which results in an irrecoverable system loop or system halt.
4. Execution of a stand-alone program, such as the PACK/INIT or DISK/DUMP programs.
5. The CM input message specifies a change in the system software or firmware.
6. The BRGR, FLMP, MPRI, LOG, RFAC, SQRM, THR, TOUT, VLCP, or VLIO options are set or reset.
7. The IC input message changes the size of the Interpreter Dictionary.

#### NOTE

Refer to volume 1 for documentation on all MCPPII commands referred to in this section.

### CLEAR/START FUNCTIONS

The CLEAR/START program, located on a cassette tape, performs the following functions:

1. When a memory dump is requested, dumps a copy of main memory to the file SYSTEM/DUMPFIL on disk.
2. Clears main memory, writing zeros and correct parity throughout.
3. Scans the I/O subsystem to locate the system disk.
4. Saves certain registers and toggles, that specify temporary environment changes for the System Initializer routine.
5. Loads the System Initializer program into memory from the system disk.
6. Turns control over to the System Initializer.

### SYSTEM INITIALIZER FUNCTIONS

After receiving control from the CLEAR/START routine, the System Initializer performs the following functions:

1. When a memory dump is requested, supplements the information in SYSTEM/DUMPFIL by copying into it certain MCPPII structures (such as the Name Table and the ODT Queue).
2. Allocates space for the initial structures in memory required for MCPPII operation, such as the Interpreter Dictionary and the MCPPII Stacks.
3. Loads the GISMO program into memory from the system disk, builds the Interpreter Dictionary entry for it, and discards all GISMO segments not required because of the system hardware configuration, or because certain MCPPII options (such as MPRI) are not set.
4. Loads the MICRO.MCP program into memory from the system disk, allocates the structures necessary for its operation, and builds the required Interpreter Dictionary entry.
5. Loads segment zero (non-overlayable) and segment one (overlayable initialization routines) of the MCPPII into memory from the system disk, and sets up the initial memory link structures required.

B 1000 Systems SSOG, Volume 2  
CLEAR/START and Memory Dump Procedure

6. Loads segment zero of the SDL Interpreter into memory from the system disk and builds the required Interpreter Dictionary entry.
7. Performs any XM operations required, building a special memory link around areas of memory that are removed.
8. Turns control over to the MCP.

## MCPII FUNCTIONS

When the MCPII receives control from the System Initializer, a number of operations are performed before the system is actually ready to begin program execution. The MCPII performs the following routines as part of system initialization:

1. Tests the I/O subsystem in order to determine the system configuration. The Input/Output Assignment Table (IOAT) is constructed from this information containing entries which describe the characteristics of all peripheral units present on the system.
2. Constructs the initial chain of DISK I/O descriptors, as well as TEST I/O descriptors for all other peripheral units present on the system.
3. Restores entries in the Temporary Disk Available Table to the Working Available Table.
4. Reads the disk directory and every disk file header (DFH) present on the system disk, clearing the user count in any DFH that was in use prior to the CLEAR/START. Disk files marked as TEMPORARY (such as compile and go code files) are removed from the disk directory.
5. Places entries identifying the CLEAR/START in the SYSTEM/LOG (if the LOG option is set) and the SYSTEM/ELOG.
6. Initiates the SYSTEM/ODT program. Also initiates the MCS program if the AMCS option is set.
7. Displays the CLEAR/START message on the operator display terminal (ODT), and then makes the system available for use.

## CLEAR/START OPERATING INSTRUCTIONS

The following procedure must be used to perform a CLEAR/START operation:

1. If the processor is not already halted (that is, the RUN light is out), bring the system to an orderly halt using either the INTRPT or the INTERRUPT pushbutton on the system console. If the INTERRUPT pushbutton fails to halt the processor (for example, due to an uninterruptible software loop), press the HALT pushbutton. If the HALT pushbutton also fails to halt the processor, press the HALT and CLEAR pushbuttons simultaneously.
2. Place the CLEAR/START cassette in the console cassette tape drive, and ensure that the cassette rewinds to Beginning-of-Tape (BOT).
3. Press the CLEAR pushbutton.
4. Set the MODE pushbutton to the TAPE position.
5. Press the START pushbutton.
6. The cassette reads the bootstrap loader, and then the processor halts with the RUN light out. The L register must contain @AAAAAA@ at this time; if not, the cassette must be rewound and the procedure restarted at step (3).
7. Any temporary environment changes to be made (such as a memory dump request) must be entered in the appropriate registers at this time. Refer to Temporary Operating Environment Changes within this section for details.
8. Set the MODE pushbutton to the RUN position.
9. Press the START pushbutton.
10. Reading of the cassette continues, loading the CLEAR/START program into memory. When the entire program has been loaded, control is given to the CLEAR/START program to begin the system initialization procedure. The cassette is then rewound by pressing the REWIND pushbutton. The rewind is done automatically on B 1900 systems.

11. If a second halt for a temporary XM was requested by setting bit 15 in the T register during step (7), the processor halts again when the System Initializer routine receives control. The L register must contain @000F28@ at this time; if not, the cassette must be rewound and the procedure restarted at step (3). Refer to Temporary XM Specification within this section for details on specifying the temporary XM address/length entry.

The cassette used in the CLEAR/START operation can be read on any B 1000 processor (except the B 1830/B 1825). Cassettes for the B 1830/B 1825 system require a different recording format. Thus, CLEAR/START cassettes are not interchangeable between the B 1830/B 1825 system and other B 1000 processors.

## DESCRIPTION OF THE NAME TABLE

The Name Table is a structure, initialized on systems disk by the COLDSTART/TAPE or COLDSTART/DISK programs, which contains entries identifying the names and disk addresses of system firmware and software available for use in the operational environment of the system.

Software and firmware routines are selected by the CLEAR/START and System Initializer programs from entries in the Name Table, based upon the system hardware configuration, as well as certain optional specifications that can be entered by the system operator. Under normal conditions, it is unnecessary for the system operator to enter any specifications for the CLEAR/START program; default software/firmware selections are made based upon the processor hardware type. Changes to the default selections are usually made only by Burroughs software development and support personnel in the course of system design and debug, and are not required during normal system operation.

The Name Table design allows certain system software or firmware routines to be identified as "experimental" or "trace" without affecting those routines identified as "standard". This permits a CLEAR/START to be performed with non-standard, untested, routines by Burroughs software development personnel without the danger of not being able to recover to the "standard" software in case of an irrecoverable failure in the experimental routines. Such a design has a limited application in a normal operating environment, where backup copies of system software or firmware can be created and identified as experimental in the Name Table. This often enables recovery in critical situations where the standard routines are corrupted.

The Name Table itself consists of 23 entries, each identified by an entry number used by the CLEAR/START and System Initializer programs and an alpha mnemonic used by the MCPPII and system operator. Each entry contains the file-identifier and disk address of the code file on systems disk. For the MCPPII entries (M, MX, MT), however, the disk address points to the MCPPII Parameter Block, an area of disk separate from the MCPPII code file constructed during the coldstart operation or by a CM input message. Each Name Table entry also has a specific defined function (for example, MCPPII or GISMO) and a condition (standard or experimental) for its selection by the CLEAR/START routine.

The structure of the Name Table is described in table 5-1.

B 1000 Systems SSOG, Volume 2  
CLEAR/START and Memory Dump Procedure

**Table 5-1. Name Table Format**

Entry	Mnemonic	Description
0	N	Standard System Initializer
1	NX	Experimental System Initializer
2	G	Standard GISMO
3	GT	Trace GISMO
4	GX	Experimental GISMO
5	I1	Standard SDL Interpreter (B 1830/B 1710)
6	I2	Standard SDL Interpreter (Other B 1000's)
7	I1T	Trace SDL Interpreter (B 1830/B 1710)
8	I2T	Trace SDL Interpreter (Other B 1000's)
9	IX	Experimental SDL Interpreter
10	M	Standard MCP II
11	MT	Trace MCP II
12	MX	Experimental MCP II
13	MM	Standard MICRO.MCP
14	MMX	Experimental MICRO.MCP
15	CX	Experimental Network Controller
16	MCS	Standard MCS program
17	MCX	Experimental MCS program
18	ODT	Standard SYSTEM/ODT program
19	ODX	Experimental SYSTEM/ODT program
20	CPY	SYSTEM/COPY program
21	C	NDL Network Controller
22	US	Usercode/Password File

Both the COLDSTART/TAPE and COLDSTART/DISK stand-alone utility programs construct the initial Name Table, and load and identify the following default files on the system disk:

Name Table Mnemonic	File Identifier
N	SYSTEM/INIT
G	GISMO
I1	SDL/INTERP1S
I2	SDL/INTERP1M
M	MCPII
MM	MCPII/MICRO.MCP
ODT	SYSTEM/ODT



B 1000 Systems SSOG, Volume 2  
CLEAR/START and Memory Dump Procedure

These defaults are selected from the Name Table when a standard CLEAR/START operation is performed. If temporary changes to the operating environment have been specified (for example, selecting the trace environment), the CLEAR/START program overrides the default Name Table selections with the entries required by the environment specified. For example, if the trace environment is selected, the following Name Table entries are used in place of their default counterparts:

<b>Name Table Mnemonic</b>	<b>File Identifier</b>
GT	Trace GISMO
I1T	Trace SDL Interpreter (B 1830/B 1710)
I2T	Trace SDL Interpreter (Other B 1000's)
MT	Trace MCP11

## TEMPORARY OPERATING ENVIRONMENT CHANGES

Several changes to the default selections made by the CLEAR/START routine can be specified by entering parameters in certain registers during the CLEAR/START process. The change requests are made by entering information through the hardware toggles and switches. The operator can make changes by doing the following:

1. Set the REGISTER SELECT and GROUP SELECT dials to point to the appropriate register.
2. Flip one or more of the 24 toggle switches upward to represent those bits that are to be set.
3. Press the LOAD button located on the front panel. This places the pattern of ones and zeroes represented by the switches into the specified register and causes the 24 console lamps to be illuminated in the same pattern.

These changes are described in the following subsections.

### General Specifications

The T register is used to specify changes to the system software and firmware selected by the CLEAR/START program, as well as to request a memory dump. Each of the changes requested is specified by setting one or more bits in the T register during step (7) of the CLEAR/START operating procedure, as follows:

<b>Bit</b>	<b>Function Requested</b>
0	Memory dump. Refer to the following subsection entitled Supplementary Information Required With A Memory Dump for additional information.
1	Retain all of the GISMO code segments. For use by Burroughs software development personnel only.
2	Not used.
3	Invoke the trace environment. For use by the Burroughs software development personnel only. CN or CT input message.
4	Select the experimental MCP11 (MX entry), rather than the default MCP11 (M entry).
5	Select the experimental System Initializer (NX entry), rather than the standard System Initializer (N entry).

B 1000 Systems SSOG, Volume 2  
CLEAR/START and Memory Dump Procedure

<b>Bit</b>	<b>Function Requested</b>
6	Select the experimental SDL Interpreter (IX entry), rather than the default SDL Interpreter (I1 and I2 entry).
7	Select the experimental GISMO (GX entry), rather than the default GISMO (G entry).
8	Select the experimental MICRO.MCP (MMX entry), rather than the standard MICRO.MCP (MM entry).
9	Select the experimental Controller (CX entry), rather than the standard NDL Controller (C entry).
10	Select the experimental MCS (MCX entry), rather than the standard MCS Program (MCS entry).
11	Select the experimental ODT (ODX entry), rather than the standard SYSTEM/ODT program (ODT entry).
12	Enable debug halts in System Initializer (for use by Burroughs software development personnel only).
13	Override the XM table (this can be necessary if it is impossible to complete the CLEAR/START operation due to problems in handling the XM requests by the System Initializer).
14	Not used.
15	Requests a second halt point so that additional temporary specifications (such as a temporary XM address/length pair) can be entered.
16	Initiates the MCS program if the AMCS system option is set. Resetting the AMCS system option does not initiate the MCS program.
17-23	Not used.

### **System Disk Channel Selection Override**

The X register can be used to override the normal selection by the CLEAR/START program of the systems disk channel. The CLEAR/START program normally selects Electronics Unit Zero (EU 0) or Drive Zero of the highest-speed disk device present as the base systems disk. The selection hierarchy is as follows (from highest to lowest):

- B 9470 Head-per-Track disk
- Head-per-Track disk (other than B 9470)
- Disk Pack
- Disk Cartridge

When more than one channel contains the same type of disk device, the CLEAR/START program selects the lowest-numbered channel as the systems disk channel.

B 1000 Systems SSOG, Volume 2  
CLEAR/START and Memory Dump Procedure

The default selection can be overridden by loading the address of the desired disk control in the X register during step (7) of the CLEAR/START procedure, as follows:

Bits	Contents
0-16	Not used
17-19	Port number (must be 7)
20-23	Channel number

The disk control address specified is used for the current CLEAR/START operation only. If the X register is zero, the default systems disk selection is performed.

### I/O Channel Deletion

The ability is provided through specifications in the FA register to delete I/O devices (that is, to make certain channels appear unassigned to the system software) during the CLEAR/START procedure. This feature is primarily of use to Burroughs field engineers when trying to isolate hardware malfunctions involving the I/O control subsystem.

The channels to be deleted are specified by setting one or more of bits 0 through 14 in the FA register during step (7) of the CLEAR/START operating procedure. For example, the following value in the FA register causes the CLEAR/START and System Initializer programs to delete channels 2, 5, 7, and 12 (port 7 is always implied):

@250800@ (0010 0101 0000 1000 0000 0000)

A channel deleted during the CLEAR/START operating procedure appears unassigned to the MCP. If the FA register contains all zeros, no channels are deleted.

### Pseudo-Memory Size Specification

The LR register can be used as a pseudo-MAXS register and (on B 1720 systems only) as a pseudo-MAXM register for specification of main and/or control memory sizes smaller than those physically present on the system. For this purpose, the LR register is functionally divided into two portions, as follows:

Bit	Function
0-11	Pseudo-MAXS register
20-23	Pseudo-MAXM register

The CLEAR/START program multiplies the value specified in the pseudo-MAXS register by 512 bytes to obtain the pseudo-main memory size. For example, the following values depict sample main memory sizes that might be specified.

Value	Main Memory Size
@080@	65,535 bytes
@200@	262,144 bytes
@0C8@	102,400 bytes
@400@	524,288 bytes

B 1000 Systems SSOG, Volume 2  
CLEAR/START and Memory Dump Procedure

The pseudo-MAXM register can contain any value between one and eight, inclusive, specifying the size of control memory in KB. The pseudo-MAXM register can only be specified on B 1720 systems.

Values larger than the physical size of main and/or control memory cannot be specified in these pseudo-registers. If a pseudo-register contains only zeros, the actual physical memory size is assigned by the CLEAR/START program.

### Temporary XM Specification

In addition to the memory areas specified in the XM Table (refer to the XM input message in volume 1, section 2), one area can be specified during the CLEAR/START operating procedure as a temporary XM location. This area of memory, specified as a beginning address and a length in bytes, is removed by the System Initializer during the current CLEAR/START operation only.

In order to enter this temporary XM specification, bit 15 in the T register must be set during step (7) of the CLEAR/START operating procedure. This requests the System Initializer to halt upon receiving control from the CLEAR/START program (the L register contains @000F29@ to identify this halt), allowing the XM specification to be entered.

The temporary XM address and length specification is entered using the X and Y registers; the beginning memory address to be temporarily deleted is entered in the X register, and the length (in bytes) to be temporarily deleted is entered in binary in the Y register. For example, the following values entered in these registers specify a temporary XM of 1000 bytes beginning at memory address @05C8FA@:

<b>X Register</b>	<b>Y Register</b>
@05C8FA@	@0003E8@

After the specifications have been entered in the X and Y registers, press the START button to continue loading the System Initializer and complete the CLEAR/START procedure.

## SUPPLEMENTARY INFORMATION REQUIRED WITH A MEMORY DUMP

If the machine comes to an orderly halt (RUN and ERROR lights are out), additional information is rarely needed. Any relevant facts about the halt must be documented and given to the Burroughs technical representative with the dump that is taken and printed. If the ERROR light is on, register settings must be recorded. The halt must be treated as if a machine hang had occurred and the Halt button were pressed, as described in the following.

Many system problems are defined to be hangs, meaning that the machine is running (RUN light is lit) but no work is completed. This situation is a loop; the machine is repetitively executing a sequence of instructions and not exiting the sequence. Before taking a memory dump, the operator must try two tests, document the results of these tests on the first page of the printed dump by circling YES or NO in answer to the test questions. The tests must be performed before a CLEAR/START and before the dump is printed.

1. The operator must determine whether or not console keyboard interaction is possible by attempting a simple system control instruction (such as MX or WT).
2. The INTERRUPT switch must be flipped.

B 1000 Systems SSOG, Volume 2  
CLEAR/START and Memory Dump Procedure

If the INTERRUPT switch halts the machine (that is, turns out the RUN light), the L register will display 10 in the last two positions. This is a well-defined halt, and no additional information need be taken.

If the INTERRUPT switch fails to halt the machine, it is extremely important that a dump not be taken until certain register values have been carefully recorded. If the dump is submitted with no register settings, or the erroneous information is given that all registers contain @FFFFFF@ (because the machine is still running), the dump is almost always useless and the problem cannot be determined or solved. Push the HALT pushbutton.

In the rare instance that the HALT pushbutton fails to stop the machine (that is, the RUN light remains lit), push the HALT and CLEAR pushbuttons simultaneously. This halts the machine. This situation reflects a processor problem and a Burroughs field engineer should be contacted. A memory dump is useless in this case and need not be taken.

In almost all cases, the HALT pushbutton stops the machine (turns out the RUN light). The operator must carefully record the following registers and attach the information securely to the front of the dump listing when it is printed.

L  
T  
X  
Y  
A  
FA  
FB  
LR  
CC  
CD  
TAS  
PERM  
PERP

At this point, the operator can CLEAR/START the machine, taking a memory dump as previously described. The dump should be printed (refer to the PM system control instruction in volume 1, section 2) and given with any other information to the Burroughs technical representative.

## SECTION 6

### CODE/ANALYZER

The CODE/ANALYZER program is a normal-state utility program that evaluates a code file to list information about its associated files, data and code segments and non-zero switch settings, and to calculate an estimate of memory required for the program to run. The code file and any intrinsic files used thereby must be present on disk.

#### NOTE

The memory estimate calculated by the CODE/ANALYZER program is a minimum memory requirement, not a working-set. The program can use more memory if it is available, or less if not all of the files are open at the same time.

### OPERATING INSTRUCTIONS

The CODE/ANALYZER program is executed from the console keyboard with an EXECUTE command. Upon execution, the CODE/ANALYZER program prompts the operator to enter the identifier of the object-code file that is to be analyzed. If the file is not a code file, or is not on disk, an error message is displayed on the operator display terminal (ODT). After the named file is analyzed, the prompt is repeated until the operator enters a blank input string.

CODE/ANALYZER = <job #> ENTER PROGRAM NAME

A valid response has the following format:

<job #> AX <file-identifier>

### PROGRAM SWITCHES

The CODE/ANALYZER program is sensitive to the values of program switches 0, 1, and 9. The default value for each switch is 0, and a switch is set when its value is 1. Action when each switch is set is described in the following:

#### Switch 0

Causes the CODE/ANALYZER program to accept the file-identifier specifications from a card file labeled CARDS. File IDs are entered in free-form format in the first 80 columns of each card, one per card. When switch 0 is set, the ODT prompts are not issued.

#### Switch 1

Causes additional information contained in the File Parameter Blocks (FPB) to be printed.

#### Switch 9

Controls printer spacing and skipping. When switch 9 is set, the listing generated by CODE/ANALYZER is printed with single-spacing and no skips to heading.

## SECTION 7

### COLDSTART/DISK

The COLDSTART/DISK program is a stand-alone program designed to coldstart a disk using a second disk as the source of the required files. Its function is the same as the COLDSTART/TAPE program, except that a disk is used as the input source rather than a magnetic tape.

Only the first disk drive (DCA, DPA, or DKA) is allowed to be the output system disk. The input disk can be either a user disk/cartridge or a single system disk. Multiple system packs/cartridges can not be used as input to the COLDSTART/DISK program.

The COLDSTART/DISK program must be able to find the nine standard system software code files on the input disk. These files are labeled

MCPII  
SDL/INTERP1S  
SDL/INTERP1M  
GISMO  
SYSTEM/INIT  
MCPII/MICRO.MCP  
SYSTEM/ODT  
SYSTEM/COPY  
SDL.INTRIN/AGGREGATE

If these names cannot be located on the input disk, the COLDSTART/DISK program requests substitute names to be entered from the console keyboard. The code files named are then used in place of the standard system software. The headers on the system code files are modified during the COLDSTART operation in order to mark these files as protected. This is done so that certain MCPII commands, for example REMOVE, cannot affect them.

An option is also provided to copy the remaining files from the input disk to the newly-coldstarted system disk after the standard files have been copied. This option copies these files during execution of the COLDSTART/DISK program. A CLEAR/START is not required before the files are copied.

#### NOTE

Do not attempt to use the full copy option if there are more files on the input disk than can be contained on the output disk.

## OPERATING INSTRUCTIONS

The COLDSTART/DISK program is a stand-alone utility program and must be loaded through the cassette reader on the system console.

Follow the instructions for loading the coldstart program detailed in the section entitled COLDSTART/TAPE, except during step 2, mount the input disk (rather than the SYSTEM tape) on any available disk drive. The COLDSTART/DISK program locates the input disk by prompting the operator to specify the input drive.

#### NOTE

With dual processor systems, the operator must press the console interrupt switch before attempting to execute the COLDSTART/DISK program.

B 1000 Systems SSOG, Volume 2  
COLDSTART/DISK

All communication with the COLDSTART/DISK program is accomplished through the console keyboard. After the COLDSTART/DISK cassette has loaded successfully, the following prompt sequence appears on the console keyboard.

COLDSTART/DISK MARK <mark-level >

ENTER OUTPUT DRIVE - <DCA, DPA OR DKA >

A valid response causes the next message to be displayed.

ENTER INPUT DRIVE - <DC?, DP? OR DKA >

A valid response causes the next message to be displayed.

IS COMPLETE COPY DESIRED? <YES OR NO >

The complete copy option copies all files on the input disk to the newly-coldstarted system disk. These files are copied after the system software is loaded on the new disk.

IS DATA COMPARISON DESIRED? <YES OR NO >

A YES response causes the COLDSTART/DISK program to verify that all data is copied correctly to the newly-coldstarted system disk. An advisory message is printed for any file in which errors are found.

After the coldstart and optional complete copy (if requested) are complete, the following message is displayed.

COLDSTART COMPLETE - CLEAR/START REQUIRED

The following is an example of this sequence.

```
COLDSTART/DISK MARK X.0
ENTER OUTPUT DRIVE - <DCA, DPA OR DKA >
DKA
ENTER INPUT DRIVE - <DC?, DP? OR DKA >
DPB
IS COMPLETE COPY DESIRED? <YES OR NO >
YES
COLDSTART COMPLETE - CLEAR/START REQUIRED
```

The responses used in the example would cause the head-per-track (DKA) to be coldstarted using the disk pack on DPB as the input disk. The standard program names are used for the COLDSTART procedure, and the remaining files on DPB are copied to the new output disk.

## ERROR MESSAGES

DISK ERROR RESULT IN "T"

HIT START TO RETRY

The T REGISTER contains the error result descriptor. Push the START pushbutton to retry the I/O operation.

INVALID RESPONSE - TRY AGAIN

The operator attempted to respond with other than the requested information.



B 1000 Systems SSOG, Volume 2  
COLDSTART/DISK

**BAD FILE HEADER <file-id> ADDRESS <hex-address>**

The input file has a bad header or the header is located on another disk. The file cannot be copied to the output disk.

**INSUFFICIENT DISK SPACE FOR COLDSTART  
INVALID COLDSTART  
RESTART TO CONTINUE**

The necessary coldstart files require more disk space than is available on the output disk. Obtain another disk and rerun COLDSTART/DISK.

**NO DISK DEVICE ON SYSTEM  
RESTART TO CONTINUE**

The system configuration contains no disk packs or disk cartridges. Ensure that disk devices are on-line and connected properly. If there are no operator disks on the system, the COLDSTART/DISK program cannot be used.

**<family-id> INVALID SUB DIRECTORY**

The subdirectory for the specified <family-id> is bad and cannot be used. The COLDSTART/DISK program cannot access any file with the specified <family-id>.

**COMPARISON ERROR <file-id>**

Advisory message issued when data comparison is requested, and a comparison error is found. The file named should be manually verified.

**<file-id> NOT FOUND  
ENTER THE CORRECT <file-id> NAME**

The specified program cannot be located on the input disk. Another name can be entered; however, it is the responsibility of the operator to ensure that the file specified has the same function as the file requested by COLDSTART/DISK.

**<file-id> BAD AREA ADDRESS <hex-address>**

Retries could not correct bad read operations on the specified file. The file is copied to the output disk, possibly with errors (if it is not a system file), and should be verified manually.

**<file-id> IS A MULTIPACK FILE - CANNOT COPY**

Multipack files are not allowed on a system disk.

**DISK NOT READY <unit-mnemonic>**

Ready the disk, and push the START pushbutton to continue.

B 1000 Systems SSOG, Volume 2  
COLDSTART/DISK

DISK NOT PRESENT <unit-mnemonic>

Load and ready the disk, and push the START pushbutton to continue.

WRITE LOCKOUT <unit-mnemonic>

Remove the write lockout, and push the START pushbutton to continue.

INSUFFICIENT DISK SPACE FOR <file-id>  
INVALID COLDSTART

This message indicates that the complete copy option of the COLDSTART/DISK program was requested, and the space remaining on the output disk is not large enough to accommodate the specified file. Restart the COLDSTART/DISK program, and either do not request the complete copy option, or remove all unnecessary files from the input disk.

PACK LABEL BAD

The label on the input disk pack/cartridge is invalid. The COLDSTART/DISK program cannot use this disk.

## SECTION 8

### COLDSTART/TAPE

The COLDSTART/TAPE program is a stand-alone utility program that loads the basic system software and firmware from a library tape labeled SYSTEM to a disk whose pack type is S (System), creating a usable system disk. The COLDSTART/TAPE program is furnished on a cassette tape and loaded through the control panel cassette reader. The library tape labeled SYSTEM can be mounted on any tape drive; the coldstart program locates the tape automatically.

The system disk created by the COLDSTART/TAPE program is a single system pack configuration and does not contain any system logfiles except the ELOG. Once the system is running under MCP II control, the number of system drives can be increased using the SD input message, and the LOG option can be set with the SL input message.

Refer to volume 1 for documentation on all MCP II input messages referred to in this section.

The COLDSTART/TAPE program performs the following functions:

1. Constructs and initializes the disk directory and available tables on the system disk.
2. Loads the MCP II, SDL Interpreters, GISMO, the System Initializer (SYSTEM/INIT), SYSTEM/ODT, SYSTEM/COPY, SDL.INTRIN/AGGREGATE, and the MCP II/MICRO.MCP files from magnetic tape to disk.
3. Makes appropriate entries in the Name Table for all system software and firmware loaded.
4. Constructs the coldstart variables on system disk.
5. Displays a message on the operator display terminal (ODT) instructing the operator to perform a CLEAR/START.

#### NOTE

When a coldstart is performed on a system disk that was previously in operation, all the files previously entered in the disk directory are lost. This is because the COLDSTART/TAPE program initializes and clears the disk directory before loading the basic system software.

## OPERATING INSTRUCTIONS

Performing a coldstart on a system takes two steps: the operator must load the coldstart program (either COLDSTART/TAPE or COLDSTART/DISK) from a cassette, and then let the coldstart program load the system software.

### Loading the Coldstart Program

The following procedure is used to coldstart a system from tape:

#### NOTE

With dual processor systems, the operator must press the console interrupt switch before attempting to execute the COLDSTART/TAPE program.

1. Mount a system disk pack or cartridge on drive 0 (if not a head-per-track system). This disk will become the new system disk.
2. Mount the library tape (labeled SYSTEM) on any available tape drive.
3. Set MODE pushbutton on the front panel to TAPE.

B 1000 Systems SSOG, Volume 2  
COLDSTART/TAPE

4. Place the COLDSTART/TAPE cassette in the cassette reader. The cassette rewinds automatically.
5. Press the CLEAR, then START pushbuttons. The bootstrap loader is read from the cassette, and the system halts. The L register contains @AAAAAA@ at this time.
6. Set MODE switch to RUN, and press the START pushbutton. The cassette continues to read. If the system HALTS (HALT light on) with @000004@ in the L register, a hash total error was detected, and the cassette must be reloaded. When the cassette has finished loading, the STATE light comes on, and the COLDSTART/TAPE program begins execution.

### Loading the System Software

To coldstart the B 1000 system, the COLDSTART/TAPE program must find the following nine standard software files on the library tape labeled SYSTEM. These files can be contained on the tape in any order but must have the following file-identifiers:

MCPII  
SDL/INTERP1S  
SDL/INTERP1M  
GISMO  
SYSTEM/INIT  
MCPII/MICRO.MCP  
SYSTEM/ODT  
SYSTEM/COPY  
SDL.INTRIN/AGGREGATE

Files other than those shown in the preceding example can also be contained on the library tape labeled SYSTEM and can be loaded after completion of the coldstart from tape (and subsequent CLEAR/START) with the following control message:

ADD & COMPARE =/= FROM SYSTEM

### REGISTER SETTINGS AND ERROR DESCRIPTIONS

During the execution of the COLDSTART/TAPE program, halts and errors are indicated when the L register contains @000011@. The T register contains the specific error or halt identification, as follows:

@AAAAAA@

Normal End-of-Job. CLEAR/START required.

@0C0001@

Disk I/O error. Press START once to display the Result Descriptor in the T register.

@0C0002@

Tape I/O Error. Press START once to display the Result Descriptor in the T register.

@0C0003@

Unexpected Data or Result Descriptor from tape.

@0C0004@

No tape control on system.

@0C0005@

No disk control on system.

B 1000 Systems SSOG, Volume 2  
COLDSTART/TAPE

@0C0006@

Disk not initialized in the proper format.

@0C0007@

Attempted to COLDSTART a pack or cartridge not initialized as SYSTEM (S).

@0C0008@

Could not locate SYSTEM tape. Make tape ready and press START.

@0C0009@

One or more files are missing from the system tape. By pressing START repeatedly until @0C0009@ is again displayed, a list of numbers corresponding to the missing files is displayed in the low-order (right-most) 4 bits of the T register, as follows:

File Number	File-identifier
1	MCPII
2	SDL/INTERP1S
3	SDL/INTERP1M
4	GISMO
5	SYSTEM/INIT
6	MCPII/MICRO.MCP
7	SYSTEM/COPY
8	SDL.INTRIN/AGGREGATE
9	SYSTEM/ODT

@0C000A@

Missing device on I/O DISPATCH operation.

@0C000B@

Insufficient disk for COLDSTART.

@0C000C@

Read beyond end of file on tape.

@0C000D@

Missing tape mark.

## SECTION 9 CONVERT/BACKUP

The CONVERT/BACKUP program converts B 1000 printer backup files into a format acceptable to the B 9270 Page Printer. The files are created in the B 6000/B 7000 Systems printer backup format.

The program accepts B 1000 tape-or disk-labeled printer backup files as input. These files must have been created with system software release Mark 9.0 or later. Backup files created prior to the Mark 9.0 release are not accepted by the program.

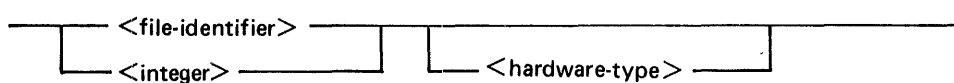
The CONVERT/BACKUP program outputs a single B 1000 labeled multifile tape named BACKUP which contains all of the input files converted for processing by the B 9270 Page Printer. Because of the labeling differences, the tape is not recognized by B 1000 Systems as a valid backup tape.

A B 1000 printer backup file with the internal file name B9270.NAMES is created which contains the names of all of the files converted during the program run.

### OPERATING INSTRUCTIONS

The CONVERT/BACKUP program, which is executed from the console keyboard, accepts input specifications entered through accept (AX) messages. The program reads and processes each input specification as it is received. A null input terminates the CONVERT/BACKUP program.

The input specifications are the input file identifier and an optional origin override. The syntax for the specification string follows:



The <file-identifier> syntax is used for specifying printer backup files by name, especially those with operator-assigned names. The format of the <file-identifier> is the same as that used in MCPPII control instructions and consists of one to three identifiers separated by slashes.

The <integer> syntax specifies by number the printer backup files that have MCPPII-generated names. In this case, the family name defaults to BACKUP.PRT for disk files and to BACKUP for tape files.

The optional <hardware-type> is the reserved word TAPE or MTP. If the <hardware-type> is not specified, the default is DISK.

B 1000 Systems SSOG, Volume 2  
CONVERT/BACKUP

A sample of the CONVERT/BACKUP program follows:

EXECUTE CONVERT/BACKUP

CONVERT/BACKUP =626 BOJ.  
CONVERT/BACKUP =626 "B9270.TAPE/DIRECTORY" = "BACKUP.PRT/419"  
% CONVERT/BACKUP =626 ENTER SPECS  
CONVERT/BACKUP =626 ACCEPT.

626 AX BACKUP.PRT/127                   % LOOK FOR BACKUP.PRT/127

% CONVERT/BACKUP =626 ENTER SPECS   % READY FOR NEXT FILE  
CONVERT/BACKUP =626 ACCEPT.

626 AX 418                           % LOOK FOR BACKUP.PRT/418

% CONVERT/BACKUP =626 ENTER SPECS  
CONVERT/BACKUP =626 ACCEPT.

626 AX 420 TAPE                   % LOOK FOR BACKUP/420 ON TAPE

% CONVERT/BACKUP =626 ENTER SPECS  
CONVERT/BACKUP =626 ACCEPT.

626 AX                           % TERMINATE THE RUN

CONVERT/BACKUP =626 "BACKUP.PRT/419" RELEASED  
CONVERT/BACKUP =626 EOJ.

## **ERROR MESSAGES**

### **BACKUP FILE MUST BE CREATED ON RELEASE 9.0 OR LATER**

The release level in the control record indicates that the backup file was created on an earlier release than 9.0. Program control returns to ENTER SPECS.

### **BACKUP FILE MUST BE LABELED**

The backup file must be created from a labeled printer file. Program control returns to ENTER SPECS.

### **FILE NOT BACKUP. RE-ENTER SPECS**

The input file is not a B 1000 backup file, so it cannot be converted. Enter the name of a B 1000 backup file or enter NULL to terminate the program.

### **FILE NOT PRESENT. RE-ENTER SPECS**

The input file could not be found. Re-enter the correct file name or enter NULL to terminate the program.

### **THERE ARE NO RECORDS TO PROCESS**

An EOF branch was encountered while looking for the control record in the backup file. Program control returns to ENTER SPECS.



## SECTION 10 CREATE/TABLE

The CREATE/TABLE program is a utility program that generates a translation table to be used by programs to translate from one character set to another. The translation table, in the form of a disk file, translates forward and backward from ASCII 8-bit, ASCII 7-bit, BCL, BCL 8-bit, and B500. All translation is to or from internal 8-bit EBCDIC. In addition, any other type of translation can be specified by the operator entering soft input.

### TRANSLATION TABLE

The translation table file contains three records: a translate header, a forward translation table, and a backward translation table. Record sizes vary according to the character set that generates the table. Table 10-1 lists the character sets and the record sizes for each.

**Table 10-1. Character Sets and Record Sizes**

Character Set	Default Bits Per Record
ASCII 8-bit	2048
ASCII 7-bit	1792
BCL	1536
BCL 8-bit	2048
B500	1536
EBCDIC	2048
Soft input	2048

### TRANSLATE HEADER FORMAT

The translate header describes the forward and backward translation tables. Eight fields of information are contained in the beginning of the header.

1. A 24-bit level identifier which changes for each system software release. It consists of the numbered day and year of the first release of that level, with the last two digits of the year listed first. For example, 79037 represents the year 1979 and the 37th day of the year.
2. An 8-bit forward translation entry size, giving the size of each entry of the second record.
3. An 8-bit backward translation entry size, giving the size of each entry of the third record.
4. A 24-bit forward translation table size for the second record.
5. A 24-bit backward translation table size for the third record.
6. A character table type containing TRAN.
7. A 24-bit field containing the filler character (in hexadecimal, right-justified) associated with the second record; @40@ = EBCDIC.
8. A 24-bit field containing the filler character (in hexadecimal, right-justified) associated with the third record; @10@ = BCL, @40@ = BCL 8-bit, @20@ = ASCII 7-bit, and @A0@ = ASCII 8-bit.

Hexadecimal Value	Character Set
@10@	BCL
@40@	BCL 8-bit
@20@	ASCII 7-bit
@A0@	ASCII 8-bit

## OPERATING INSTRUCTIONS

### Execution

Program execution can be achieved by entering card deck or disk file input. Figure 10-1 illustrates a sample card deck with control and data cards that produces an ASCII 8-bit translation table. Figure 10-2 shows disk file input including the necessary file-equate statement. All data must appear within the disk file.

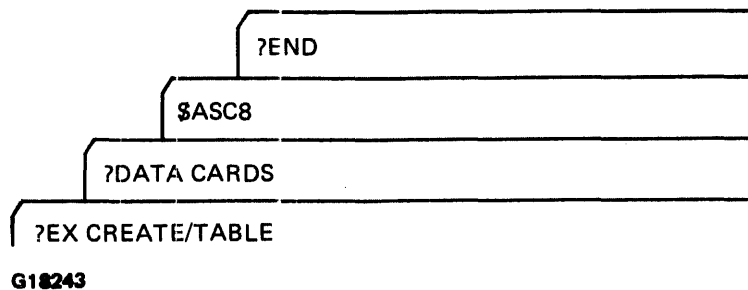


Figure 10-1. Sample Card Deck Execution

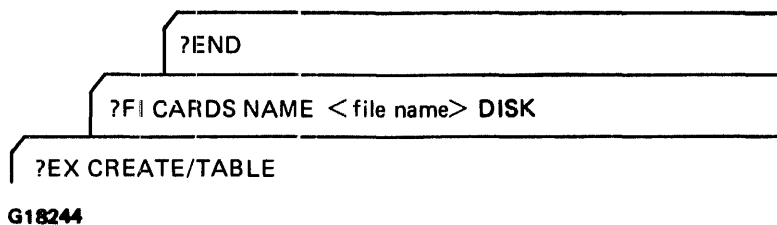


Figure 10-2. Sample Disk File Execution

### Program Input

The available options allow an operator to enter as little information as that required to indicate the desired translation table (see figure 10-1), or as much information as that called soft input, required to create an entirely new table. Figure 10-3 depicts a sample card deck utilizing all of the available options. The soft input cards provide the translation information for a table that translates from lower-case letters to upper-case letters. Note the use of the IDNT statement to name the table TRANSLATE/MYTEST.

Program input takes the form of statements entered in conjunction with the control cards. The statements are in free-form format with any columns after column 72 available for comments.

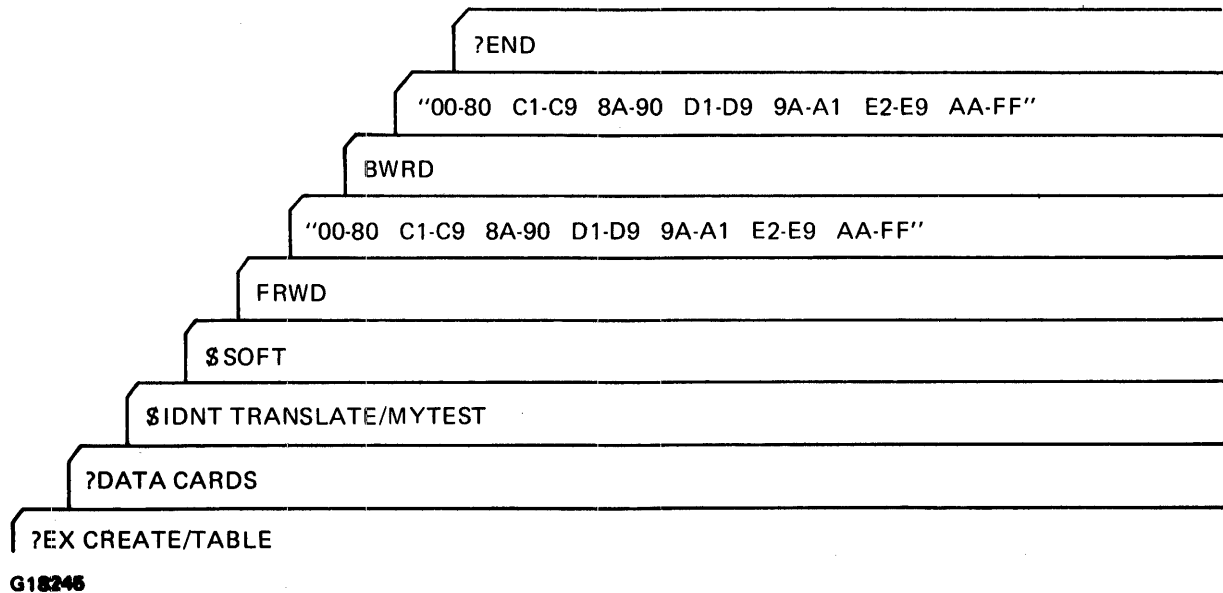


Figure 10-3. Sample Deck Using All Options

### Option Summary

The following overview summarizes the syntax for the program control statements.

#### Syntax Overview

```

[ $IDNT < file-name > ]

[ $ASC8 ]

[ $ASC7 ]

[ $BCL ]

[ $BCL8 ]

[ $B500 ]

[ $SOFT [FRWD " < constant > [ - < constant > ] . . . < constant > [ - < constant > ]" ]

[ BWRD " < constant > [ - < constant > ] . . . < constant > [ - < constant > ]" ]

```

### ASCII 8-bit Option

The \$ASC8 option produces a translation table for the ASCII 8-bit character set. If the IDNT statement is not used, the name of this table is TRANSLATE.

### ASCII 7-bit Option

The \$ASC7 option produces a translation table for the ASCII 7-bit character set. If the IDNT statement is not used, the name of this table is TRANSLATE.

### BCL Option

The \$BCL option produces a translation table for the BCL character sets. If the IDNT statement is not used, the name of this table is TRANSLATE.

### BCL 8-Bit Option

The \$BCL8 option produces a translation table for the BCL 8-bit character set. If the IDNT statement is not used, the name of this table is TRANSLATE.

### B500 Option

The \$B500 option produces a translation table for the B500 character set. If the IDNT statement is not used, the name of this table is TRANSLATE.

### Identification Option

The \$IDNT option allows the operator to name the translate table. The semantics for the file name are in accordance with the B 1000 file-naming conventions. If this statement is not used, the table is called TRANSLATE, and appears on the system disk as such. The MCP II assumes that translate files are on the system disk with the multiple file identification TRANSLATE.

### Soft Input Option

The \$SOFT option, invoked by a \$SOFT control card, indicates to the program that translation information follows. The direction forward (FRWD) or backward (BWRD) is specified and is followed by input enclosed in quotation marks ("). The input consists of pairs of hexadecimal constants, listed individually, or indicated as lower and upper limits connected with a hyphen. Figure 10-3 illustrates the use of this option.

## ERROR MESSAGES

### CONTROL INFORMATION NOT FULLY CONTAINED WITHIN COLUMN 72

Any information entered in column 72 and beyond was not recognized by the program.

### ERROR IN HEX-CONVERSION

The input was entered incorrectly, perhaps in decimal form, and could not be processed.

### INPUT FOR TRANSLATE TABLES NOT PROVIDED. CANNOT CONTINUE.

Either no input statement, such as \$ASC8, was provided or the information for the soft input translation table was omitted.

#### INVALID CONTROL STATEMENT

The control statement contains an invalid control word or incorrect syntax.

#### INVALID FILE NAME

Ensure that file name in the IDNT statement conforms with B 1000 file-naming conventions.

#### PARAMETER LIST MUST BE ENCLOSED IN QUOTE SIGNS

Quotation marks were omitted from hexadecimal input.

#### SOFT STATEMENT DOES NOT CONTAIN ALL REQUIRED ENTRIES

Directional indicators FRWD or BWRD were omitted from the soft input, or hexadecimal sequences were missing.

#### VALUES ON BOTH SIDES OF HYPHEN ARE EQUAL AND DO NOT DEFINE A SERIES

No translation is requested.

#### INTERNAL FILES

The following is a list of the name and function of each file used by the CREATE/TABLE program.

<b>Internal File Name</b>	<b>Function</b>
REPORTFILE	Stores the printer file as it appears in the input for the FRWD translate table.
TRANFILE	A disk file that stores the translate tables.
CARDS	Stores the input file.

## SECTION 11

### DISK/ALLOCATOR

The DISK/ALLOCATOR program is a normal-state utility program used to create Installation Allocated Disk (IAD) files. IAD files are assigned to an absolute disk address and have operator-specified physical attributes. The capability of designating a file's physical location and its attributes is useful for recovering files after a coldstart operation, for allocating files to specific drives of multiple system disk configuration, and for aligning areas of a file on disk cylinder boundaries.

### OPERATING INSTRUCTIONS

The following paragraphs describe the operating instructions.

#### Execution

The DISK/ALLOCATOR program is executed either from the console keyboard or from cards. By default, the program expects to receive command specification input via accept (AX) instructions entered through the console keyboard. When switch 0 = 1, DISK/ALLOCATOR accepts all command specification input from a card file labeled CARDS.

#### Command Specifications

Command specifications describe the location and the attributes of the file being created. All command input specifications are entered in free-form format as keyword/value pairs. Multiple command specifications can be entered in a single input message or card with each keyword/value pair separated by at least one comma (,) or blank character.

If an error is detected in any input specification, the file is not allocated, and the DISK/ALLOCATOR program scans the input stream for another TITLE specifier. All keywords encountered before the next TITLE specifier are ignored.

The equal sign (=) character must be included between each keyword and its associated value. Numeric values can be specified in either decimal or hexadecimal notation. Hexadecimal values must be enclosed in at sign (@) characters and must not include the hexadecimal mode indicator (H) preceding the number.

Valid attributes and their meanings are described in table 11-1.

**Table 11-1. DISK/ALLOCATOR Attribute Keywords and Default Values**

Keyword	Description	Default
TITLE	File-identifier	--
RECSIZE	Record size	180
BLOCKSIZE	Records-per-block	1
AREASIZE	Blocks-per-area	100
AREAS	Number of areas	25
LASTRECORD	EOF pointer	0
FAMILYINDEX	Disk drive or EU	0
AREANBR	Area number	--
AREAADDRESS	Absolute disk address	--

## B 1000 Systems SSOG, Volume 2 DISK/ALLOCATOR

The TITLE attribute must be specified first for each file being allocated. The attribute keywords RESIZE, AREASIZE, AREAS, and LASTRECORD are optional and can follow the TITLE keyword in any order, but must precede the attribute keywords FAMILYINDEX, AREANBR, and AREAADDRESS. When the optional keywords are not specified, their default values are assumed.

Once the general attributes of the file are established, each area and its absolute disk address are then specified through the keywords FAMILYINDEX, AREANBR, and AREAADDRESS. It is not necessary to allocate all areas of the file at once.

FAMILYINDEX specifies the electronic unit (EU) number for head-per-track disks, or the disk drive for multiple system packs/cartridges. The default is DRIVE/EU 0. Once FAMILYINDEX is specified, it must be specified for all of the following pairs of AREANBR and AREAADDRESS.

AREANBR and AREAADDRESS must be specified in pairs, with AREANBR appearing first. AREANBR specifies the file area number and can have a value of 1 to 105, inclusive. AREAADDRESS is the absolute disk address of the area. The AREAADDRESS is encoded as the rightmost 24 bits of a standard address field. The remaining high-order bits locate the disk and are not included in the AREAADDRESS used by the DISK/ALLOCATOR program.

The DISK/ALLOCATOR program does not actually allocate any areas of a file until it receives another TITLE specifier (which indicates that another file is to be allocated) or until it receives a blank accept (AX) message (or EOF on the card file). When the described disk file is allocated, the following message is displayed on the operator display terminal (ODT):

```
<title> ALLOCATED
```

### ALLOCATED FILE STRUCTURE

When a file is allocated, DISK/ALLOCATOR constructs a disk file header. Constructing this header requires one to three contiguous disk sectors, depending on the number of areas specified for the file (1 to 25 areas require one sector, 26 to 65 areas require two sectors, and 66 to 105 areas require three sectors). The DFH is allocated automatically by the MCPPII at the lowest available disk address. Thus, the location of the disk file header is not determined by the DISK/ALLOCATOR program.

All file attributes are determined by the keyword attribute specifications processed by DISK/ALLOCATOR.

### SAMPLE EXECUTION

```
EX DISK/ALLOCATOR
DISK/ALLOCATOR = 1234 BOJ...
% DISK/ALLOCATOR = 1234 ENTER SPECS
1234AXTITLE = A/B/C
%DISK ALLOCATOR = 1234 ENTER SPECS
1234AXRECSIZE = 90,BLOCKSIZE = 2,AREASIZE = 100,AREAS = 25
% DISK/ALLOCATOR = 1234 ENTER SPECS
1234AXLASTRECORD = 250
% DISK/ALLOCATOR = 1234 ENTER SPECS
1234AXAREANBR = 1,AREAADDRESS = @003FC0@
% DISK/ALLOCATOR = 1234 ENTER SPECS
1234AXAREANBR = 2,AREAADDRESS = @DD0C@
% DISK/ALLOCATOR = 1234 ENTER SPECS
1234AX
% DISK/ALLOCATOR = 1234 A/B/C ALLOCATED
DISK/ALLOCATOR = 1234 EOJ....
```

## ERROR MESSAGES

INVALID <keyword> -- <value>

The specified <value> is invalid for the <keyword>.

INVALID TOKEN -- <string>

The specified <string> is not a valid <keyword>.

EQUAL SIGN EXPECTED

A <keyword> was not followed by an equal sign.

VALUE EXPECTED AFTER =

No value was specified following the <keyword> and equal sign.

UNEXPECTED TOKEN -- <keyword> -- TITLE EXPECTED

The keyword TITLE was expected, but some other token was encountered. This message can occur if an error is detected in an input message that causes the program to scan for another TITLE keyword to continue processing.

CANNOT SET FAMILYINDEX TO <integer>

The FAMILYINDEX cannot be set to the specified <value>.

CANNOT ALLOCATE AREA -- NEED AREANBBR

An AREAADDRESS was specified without first specifying the AREANBR.

CANNOT ALLOCATE AREA <integer>

The requested disk area could not be allocated. An associated MCPPII error message describes the reason for the failure.

LASTRECORD OF <integer> IS NOT WITHIN ASSIGNED FILE AREA

The value specified for LASTRECORD is beyond the maximum file size.

CANNOT SET LASTRECORD TO <integer>

The value specified for LASTRECORD is invalid.



## SECTION 12

### DISK/DUMP

The DISK/DUMP program is a stand-alone utility program that provides the capability of copying data from disk packs to disk packs, from disk cartridges to disk packs, and from disk cartridges to disk cartridges (except 200 TPI to 100 TPI cartridges). During the dump, the label is first checked for validity, and then all data is copied on a sector-for-sector basis. For system disks and operator disks located on drive zero, termination of the dump occurs immediately beyond the end of valid data, thereby reducing run time in many cases. All input and output specifications are entered from the console keyboard. When the number of input errors exceeds 10, the operator can specify the number of retries desired on an as occurs basis. If more than 10 output errors occur on a given area, the DISK/DUMP program terminates.

### OPERATING INSTRUCTIONS

The DISK/DUMP program does not operate under MCP II control and must be loaded from the cassette reader of the system console. Execute the DISK/DUMP program in the following manner:

#### NOTE

With dual processor systems, the operator must press the console interrupt switch before attempting to execute the DISK/DUMP program.

1. Place the DISK/DUMP cassette in the cassette reader. The BOT light must be lit at this time.
2. Place the console printer on-line.
3. Set the system MODE pushbutton to the TAPE position. Press the CLEAR pushbutton, then the START pushbutton. This procedure loads the bootstrap loader from the cassette tape and halts the processor. The L register must be equal to @AAAAAA@ at this time.
4. Set the MODE pushbutton to the RUN position; press the START pushbutton. Do not press the CLEAR pushbutton. This loads the DISK/DUMP program.

When the cassette tape has been read, the DISK/DUMP program begins operation, and the following message is displayed on the operator display terminal (ODT):

```
DISK DUMP MARK <level-number>  
ENTER INPUT DRIVE - <DC? or DP?>
```

After a valid response, the following messages are displayed in sequence:

```
ENTER OUTPUT DRIVE - <DC? or DP?>  
  
IS VERIFICATION DESIRED? - <YES or NO>
```

After valid responses, the data on the disk is copied and, if requested, compared. Upon completion, without errors, the following message is displayed:

```
DUMP COMPLETE FROM <input drive mnemonic> TO <output drive mnemonic>  
ENTER INPUT DRIVE -<DC? or DP?> OR BLANK TO TERMINATE
```

B 1000 Systems SSOG, Volume 2  
DISK/DUMP

If END OF MESSAGE is signaled (a null input string is entered) at this time, the DISK/DUMP program terminates and the following message is displayed:

END DISK DUMP

## ERROR MESSAGES

DISK ERROR - RESULT IN "T"

Observe the T register to determine type of error. Press the START pushbutton for retry.

DISK NOT READY <input or output drive mnemonic>

Make the disk ready, and then press the START pushbutton.

PARITY ERR ON <input drive mnemonic > ENTER DESIRED NUMBER OF RETRIES OR BLANK TO RESTART

TIMEOUT ON <input drive mnemonic> ENTER DESIRED NUMBER OF RETRIES OR BLANK TO RESTART

INVALID RESPONSE -TRY AGAIN

I/O ERROR <input or output drive mnemonic> <disk address>  
RESULT = <result>

<integer> RETRIES ON PARITY

<integer> RETRIES ON TIMEOUT

TEMP TABLE FILLED <input drive mnemonic>  
CLEAR/START this disk and try again.

COMPARE ERROR <disk address> HIT START TO RETRY

INPUT SIZE LARGER THAN OUTPUT

NO CART OR PACKS ON SYSTEM

The program must be re-executed. Ensure that a disk device is on-line.

DISK NOT PRESENT

Load the disk and press the START pushbutton.

B 1000 Systems SSOG, Volume 2  
DISK/DUMP

WRITE LOCKOUT

Remove the write lockout, and then press the START pushbutton.

REMOVED SECTORS ON <drive>

Program terminated. Copying to or from a cartridge or pack with bad sectors produces an invalid dump.

PACK LABEL BAD <drive>

The disk specified cannot be used.

## SECTION 13

### DISK PACK INTERCHANGE PROGRAMS

One Burroughs standard structure of interchange media, as defined for disk packs, is compatible across several of the Burroughs product lines. The B 1000 systems interface to this disk format through either of the following two utility programs:

DISKPACK/INTERCHANG, a normal-state utility program  
STANDALONE/INTERCHANG, a stand-alone utility program

The operation of these two programs is virtually identical; hence, they are referred to collectively as the INTERCHANGE PROGRAM.

The INTERCHANGE PROGRAM has two basic modes of operation:

The forward mode, which creates a disk pack in standard interchange format from a B 1000 operator disk pack.

The reverse mode, which creates a B 1000 user disk pack.

### OPERATING INSTRUCTIONS

If the forward mode is desired, mount the input B 1000 user disk pack in one disk pack drive, and mount a disk pack initialized as INTERCHANGE in another disk pack drive.

If the reverse mode is desired, mount the previously written interchange disk pack in one disk pack drive, and mount a newly initialized or purged B 1000 user disk in another disk pack drive.

Execute the INTERCHANGE PROGRAM in the appropriate manner (refer to DISK/DUMP and SYSTEM/DUMP.DUMP for instructions on execution of STANDALONE/INTERCHANG and DISKPACK/INTERCHANG, respectively). Messages from the INTERCHANGE PROGRAM are displayed on the operator display terminal (ODT). Responses are also entered through the ODT (with (AX) messages for DISKPACK/INTERCHANG). The messages displayed and the responses expected are as follows:

**WHICH PACK (SOURCE OR DESTINATION) IS B1700 FORMAT?**

An S response causes the forward mode to be selected which creates an interchange disk pack.

A D response causes the reverse mode to be selected which creates a B 1000 user disk.

**ENTER SOURCE PACK DRIVE (A,B,C,ETC.)**

Enter the unit-mnemonic of the input disk drive.

**ENTER DESTINATION PACK DRIVE (A,B,C,ETC.)**

Enter the unit-mnemonic of the output disk drive.

Neither drive specified can be a system drive, nor can the source and destination unit-mnemonics specify the same drive.

## ERROR MESSAGES

Due to differences between the B 1000 user disk format and the standard interchange format, it is not possible to copy certain files. As these incompatibilities are encountered during conversion, they are indicated by the error message listed in the following. The INTERCHANGE PROGRAM continues to copy the remaining valid files.

### FILE <file-id> CONTAINS MULTI-PACK LINKS

Files containing multi-pack areas are not copied.

### FILE <file-id> EXCEEDS 105 AREAS

Files with more than 105 disk areas are not copied (reverse mode only).

### FILE <file-id> EXCEEDS 6 CHARACTERS

Files with file-ids longer than six characters are not copied (forward mode only).

### FILE <file-id> IS MULTI-FILE NAME

Files with family identifiers are not copied (forward mode only).

The following messages can appear due to disk I/O or format errors:

### DISK ERROR AT <sector-address> IN FILE <file-name>

An irrecoverable I/O error occurred on the input disk pack. The file containing the error is not copied.

### TOO MANY ERRORS AT SECTOR <sector-address>

An irrecoverable I/O error occurred on the output pack. The sector in error is removed from the Master Available Table on the output pack, and processing continues.

### DISK ERROR AT <sector-address> ON INPUT DRIVE

An irrecoverable I/O error occurred in the directory of the specified pack. The INTERCHANGE PROGRAM terminates immediately.

### COULD NOT FIND SOURCE PACK

Indicates that the specified pack is (1) not on the proper drive, (2) not ready, or (3) in-use.

### COULD NOT FIND DESTINATION PACK

Indicates that the specified pack is (1) not on the proper drive, (2) not ready, or (3) in-use.

### DESTINATION PACK NOT AN INITIALIZED PACK

The destination pack is not a scratch pack (reverse mode only) or has not been properly initialized.

B 1000 Systems SSOG, Volume 2  
Disk Pack Interchange Programs

**ERROR IN PACK LABEL, CAN'T CONTINUE**

Indicates that one of the following has occurred:

The source and destination packs are reversed. (The copy attempt is in the wrong direction.)

The source pack is not a B 1000 user disk or a disk written in the standard interchange format.

## SECTION 14

### DISKETTE/COPY

The DISKETTE/COPY program is a normal-state utility program that copies data files to or from an industry-compatible mini-disk (diskette or floppy disk). Copying files between two diskettes is not allowed.

All communication with a diskette is accomplished through the DISKETTE/COPY utility program.

Diskettes can contain up to 19 files and can hold a maximum of 246,272 bytes of information. Each diskette has 74 addressable tracks, numbered 0 through 73. Each track contains 26 sectors, numbered 1 through 26, and each sector can hold a maximum of 128 bytes of information. Diskette track number zero is reserved for the volume identifier and the file directory. A diskette address consists of five digits: the first two digits are the track number, the third digit is always zero, and the last two digits are the sector number of the record addressed.

### OPERATING INSTRUCTIONS

The DISKETTE/COPY program can be executed either from the console keyboard or from a card reader. Program switch settings determine the source of input specifications.

### PROGRAM SWITCHES

The DISKETTE/COPY program recognizes program switches 0 and 2. When switch 0 = 1, DISKETTE/COPY reads its input specifications from the card file labeled SPEC; otherwise, input specifications are entered through the console keyboard. When switch 2 = 1, the DISKETTE/COPY program stops processing when a data error is encountered and requests that the operator enter an OK command by way of an accept (AX) input message and then continues. Otherwise, the program deletes the sector in error, displays an error message and continues processing.

### INPUT SPECIFICATIONS

DISKETTE/COPY reads its input specifications from accept (AX) messages entered through the console keyboard, or from the file labeled SPEC when program switch 0 is set.

The DISKETTE/COPY program can perform five distinct actions using diskettes. The input commands can request each of these actions, except AUTOLOAD. The actions are summarized in the following list.

#### AUTOLOAD

Automatically load all PSR pseudo reader files on a diskette to the system disk when the diskette is readied. This task cannot be requested with input commands.

#### COPY

Copies files to or from diskettes.

#### KA

Generates a KA listing of all files on a diskette. This listing is in the standard KA format.

#### PURGE

Purges all files on a diskette, converting it to a scratch diskette.

**RELABEL**

Relabels the diskette (change its volume identifier).

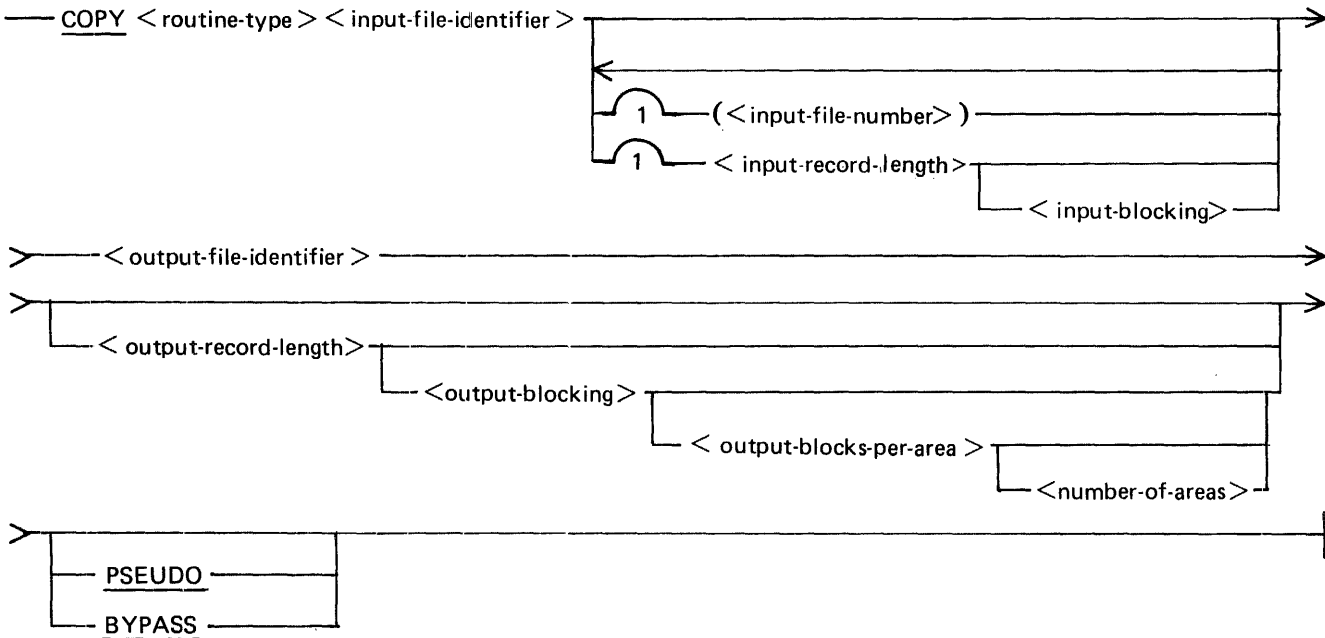
Detailed descriptions of the commands and the AUTOLOAD facility follow.

**AUTOLOAD Facility**

The DISKETTE/COPY program automatically loads all PSR pseudo-reader files contained on a diskette to the system disk when the diskette is placed on-line. PSR pseudo-reader files have the letters PSR as the first three characters of their eight-character file-id. When the diskette is readied, the MCP II searches the diskette directory for PSR file-ids. If one or more PSR files are found, then DISKETTE/COPY is executed by the MCP, which then automatically copies the PSR files to the system disk as pseudo-reader decks, without operator intervention.

**COPY Command**

A description of the COPY command syntax follows.



**Routine Type Specifier**

The `<routine-type>` is specified in the same manner as the shorthand notation for `<routine-type>` in the DMPALL utility program and must designate a diskette unit as one (and only one) of the two units. Abbreviations for the allowable devices follow:

Hardware Device	Abbreviation
Card	CRD
96-col. card	C96
Magnetic tape	MTP
Paper tape	PPT
Disk	DSK
Diskette	FDx (x indicates the drive)



B 1000 Systems SSOG, Volume 2  
DISKETTE/COPY

For example, to copy from cards to diskette unit 0 (FDA), the <routine-type> is CRDFDA.

### Input File Specifications

The format of <input-file-id> is the same as that used in MCPPII control instructions. A diskette <file-identifier> can be up to eight characters in length and must be specified as either a single-name identifier or a two-name identifier with the following format:

<volume-id>/<file-id>

The <volume-id> can be up to six characters in length, and must agree with the volume-id of the designated diskette. If the <volume-id> is omitted, IBMIRD or a blank <volume-id> is assumed, depending on the actual volume-id of the designated diskette. If <volume-id> is omitted and the volume-id on the diskette is neither blank nor IBMIRD, then an error message is displayed. A diskette with a volume-id that matches the volume-id specified in the COPY command must be loaded, or processing of the specification must be terminated. When a null accept message is entered to the DISKETTE/COPY program, processing of the current specification stops and the following prompt is issued:

PLEASE ENTER COPY PARAMETERS

The <input-file-number> is an integer (enclosed in parentheses) used when there is more than one input file on the diskette with the same name. In such a case, the <input-file-number> specifies which of these files to use as input (the first of such duplicate file names is considered number one). Specifying the <input-file-number> causes DISKETTE/COPY to ignore the BYPASS flags.

The <input-record-length> specifies the input file record size in bytes. If omitted, a record length of 80 is assumed except for disk files, where the actual record length of the file is used. The maximum record size permitted on a diskette input file is 128 bytes.

The <input-blocking-factor> is the second numeric entry, and specifies the number of logical records in a block (physical record). If omitted, a blocking factor of one is assumed, except for disk files, where the actual blocking factor is used. The maximum blocking factor permitted on a diskette input file is one.

### Output File Specifications

The format of the <output-file-identifier> is the same as that of the <input-file-identifier>.

The <output-record-length> specifies the output file record size in bytes. If omitted, a record length of 80 bytes is assumed, unless either the input or output file is a disk file (FDxDSK or DSKFDx). When a disk file is used, the default <output-record-length> is the same as the <input-record-length>. The maximum record size permitted on a diskette output file is 128 bytes, and the input records being written to the diskette file are truncated if they are longer than 128 bytes.

The <output-blocking-factor> is the second numeric entry following <output-file-identifier>. If omitted, a blocking factor of one is assumed. The maximum blocking factor permitted on a diskette output file is one.

The <output-blocks-per-area> value is the third numeric entry following the <output-file-id>. This entry is only applicable to non-diskette disk files. If omitted, 100 blocks per area are assumed by default.

B 1000 Systems SSOG, Volume 2  
DISKETTE/COPY

The <output-areas> value specifies the maximum number of disk areas allowed. This entry is applicable only to non-diskette disk files. If omitted, 25 areas are assumed by default.

#### BYPASS Option

The BYPASS option creates files on the diskette with duplicate identifiers and sets the BYPASS flag for the file. If there are duplicate diskette file identifiers, only one of these files can be NOT BYPASS.

#### PSEUDO Option

The PSEUDO option can be used following the <output-file-identifier> and in place of all other output file specifications. When used, the input file from the diskette is copied to system disk as a pseudo-reader file. The <output-file-id> is required for the syntax, but is ignored when the pseudo-reader file is created. The file is assigned a disk-file-id of DECK/<#>. Truncation or padding of the input records to 96 bytes is performed where necessary.

### KA Command

The DISKETTE/COPY program can provide a line printer listing of the file-ids and file characteristics of all files on a designated diskette. A description of the syntax of the KA command follows.

—— KA <unit-mnemonic> —————|

The KA listing includes the diskette addresses of each file's extent and lists the following information about the diskette:

#### DATE/TIME

The DATE/TIME fields represent the current system date/time.

#### V.IDENTIFIER

The V.IDENTIFIER field represents the volume identifier.

#### ACCESSIBILITY

The ACCESSIBILITY field must be blank if the DISKETTE/COPY program is to access the diskette. A non-blank field means that the diskette is inaccessible.

#### RESERVED

The RESERVED field is to be blank (non-reserved), and is ignored by the DISKETTE/COPY program.

#### STANDARD.LABEL

The STANDARD.LABEL field must be a W. This indicates a standard label format. If this field contains any other value, the DISKETTE/COPY program cannot read the diskette.

The KA command also lists the following information about each file (referred to as a "data set") on the diskette:

#### FILE.ID

The FILE.ID field is the diskette file identifier.

#### BLOCK.SIZE

The BLOCK.SIZE field is the record size of file.

#### H.BOE

The H.BOE field is the beginning-of-extent. This is the address of the first sector of the file.

#### H.EOE

The H.EOE field is the end-of-extent. This is the address of the last sector reserved for the file and indicates the end of the disk area allocated to the file.

#### H.EOD

The H.EOD field is the end-of-data. This indicates where the data ends, yet actually points to the next available sector (EOF + 1). For a file that has filled its extent, this is equal to H.EOE + 1. If H.EOD is equal to H.BOE, then the file is actually an available area define. On a diskette that contains less than 19 files and has available space, there is always an extra file present which describes the available space. This dummy file-id is DATA if it is the only file present (scratch disk), or DATA <nn> where <nn> is the sector number of the track 0 sector in which the file header is located.

#### WRITE.PROTECT

A letter P in this field protects the file from being overwritten. The DISKETTE/COPY program only creates non-protected output files, and the WRITE.PROTECT field contains a blank.

#### INTERCHANGE

The INTERCHANGE field must be blank, indicating that the file can be used for data interchange.

#### MULTI.VOL

A blank character in the MULTI.VOL field indicates that the file is fully contained on the diskette; a letter C indicates the file continues on another diskette; a letter L indicates that this is the last diskette on which a continued file resides.

#### MV.SEQ#

The MV.SEQ# field an optional multi-volume sequence number which specifies the sequence of diskettes in a multi-volume file. If used, sequence numbers must be consecutive starting at 01 (to a maximum of 99).

#### CREATED

The CREATED field is the date the file was created, in YYMMDD format.

#### EXPIRES

The EXPIRES field is the date the file expires, in YYMMDD format. The DISKETTE/COPY program assigns a value of the creation date plus seven days. Files are never automatically purged, regardless of the value in the EXPIRES field.

#### BYPASS

A blank character in the BYPASS field indicates that the file is accessible to the DISKETTE/COPY program without using the <input-file-number>; The letter B in this field specifies that the <input-file-number> must be used to access the file.

## PURGE Command

The DISKETTE/COPY program can purge all files contained on a diskette, converting it to a scratch diskette. A description of the syntax of the PURGE command follows.

———— PURGE <unit-innemonic> —————|

Only the file that describes the available space on the diskette remains. Its name is DATA, its extent is 01001 to 73026, and its end-of-data pointer is 01001.

## RELABEL Command

The DISKETTE/COPY program can change the <volume-id> of a diskette by using the RELABEL command. A description of the syntax of the RELABEL command follows.

———— RL FDx <volume-id> —————|

The <volume-id> can have a maximum length of six characters.

The data files contained on the diskette are not affected by the RELABEL command.

## SAMPLE INPUT SPECIFICATIONS

Sample input specifications with a description of the action performed for each are described in the following paragraphs.

### COPY CRDFDA CARDS <volume-id>/CRDIMG

The card file named CARDS is copied to the diskette on drive A. The <volume-id> of the diskette must match the specified <volume-id>.

### COPY FDADSK INPUT (2) 80 1 USER/DISKETTE/INPUT 80 9 100 25

The second file with the file-id INPUT on the diskette is copied to the disk USER and is named DISKETTE/INPUT. The diskette file record size is 80. When the file is copied to disk, it is blocked 9 records to a block, and the blocks-per-area and number-of-areas for the output file are specified.

### COPY MTPFDB TAPEFILE 180 10 FDFILE 128 1 BYPASS

The BYPASS keyword allows the file-id FDFILE to be assigned, even when other files with that name exist on the diskette.

### COPY FDBDSK PSRDECK DUMMY PSEUDO

The diskette file named PSRDECK is copied to disk as a pseudo deck. A dummy output-file-id is required, even though the system assigns a pseudo-reader file-id to the file. Because the diskette file-id begins with PSR, the file was also automatically copied as a pseudo-deck when the diskette was readied.

### KA VOLID1

The diskette with the volume-id VOLID1 is analyzed, and a KA listing is generated.

### PURGE FDA

Creates a scratch diskette on drive A.

### RL FDB VOLID2

Changes the label of the diskette to VOLID2. Data on the diskette is not affected.

## ERROR MESSAGES

### NO INPUT FILE SPECIFIED

The <input-file-identifier> is missing from the COPY specifications.

### NO OUTPUT FILE SPECIFIED

The <output-file-identifier> is missing from the COPY specifications.

### INVALID COPY FILE HARDWARE TYPE <hardware-type>

The <hardware-type> is not a valid device abbreviation.

### FLEXIDISK MUST BE SPECIFIED AS ONE OF THE COPY MEDIA

### INVALID FILE NAME

### INVALID PARAMETER <parameter>

### MISSING COPY STATEMENT

The input request was not recognized.

### MISSING FILE SPECIFICATION

No file identifiers were entered in the COPY statement.

### MEMORY PARITY ERROR ON DISKETTE I/O

System had a memory parity error while doing the diskette I/O.

### UNEXPECTED NOT READY CONDITION ON <diskette-unit-id>

The diskette went not ready.

### <error-type> ERROR ON <unit> TRACK = <track> SECTOR = <sector> SECTOR BYPASSED.

An error (designated by <error-type>) was encountered on the diskette at the specified sector. If input, the copy continues, but the data in the sector is lost. If output, the sector is marked as deleted (if possible).

### TIME OUT ON <unit mnemonic>

The diskette unit had a timeout error. The copy is terminated.

### TRACK SEEK ERROR ON <unit-mnemonic>

The diskette had a track seek error. The copy is terminated.

UNCORRECTABLE WRITE FAULT ON <unit-mnemonic>

The diskette had a write fault error. The copy is terminated.

UNDIAGNOSED ERROR ON <unit-mnemonic>

The diskette had an unknown error. The copy is terminated.

YOUR SUPPLIED RECORD SIZE DOES NOT MATCH LABEL RECORD SIZE

The record size on the diskette is not the same as specified.

DISKETTE BLOCKING FACTOR MUST BE 1

INVALID BEGINNING OF EXTENT ON <unit> <h.boe>

The beginning of extent is not valid.

INVALID END OF DATA ON <unit> <h.eod>

The end-of-data is not valid.

BEGINNING OF EXTENT <h.boe> EXCEEDS END OF DATA <h.eod> ON <unit>

END OF DATA <h.eod> EXCEEDS END OF EXTENT <h.eoe> ON <unit>

INVALID HARDWARE TYPE FOR PSEUDO DECK

PSEUDO TYPE PERMITTED ON OUTPUT DISK FILE ONLY

The output <hardware-type> must be disk (DSK) if specifying PSEUDO.

MAXIMUM DISKETTE RECORD SIZE EXCEEDED

The maximum record size for diskette files is 128 characters.

WARNING: DISKETTE DEFAULTS ON OUTPUT ARE RECORD SIZE OF  
128 CHARACTERS BLOCKED 1

This is a warning message only. The record size of the input file is greater than 128 bytes. Records are truncated to 128 bytes when copied to the diskette.

WARNING: INVALID HEADER ON <unit-mnemonic>

This is a warning message only. One of the files on the diskette has a bad header.

UNABLE TO READ VOLUME LABEL ON <unit-mnemonic>

UNABLE TO READ HEADER IN SECTOR <sector> OF <unit-mnemonic>

The file whose header is in the specified sector is inaccessible.

B 1000 Systems SSOG, Volume 2  
DISKETTE/COPY

UNABLE TO WRITE ENDING HEADER IN SECTOR <sector> of <unit>

The header could not be written, so the copy is terminated.

DUPLICATE FILE FOUND ON UNIT <unit-mnemonic> -- NO COPY PERFORMED

The diskette had duplicate file identifiers with the BYPASS flag not set on the duplicate files.

NO FILE <file-id> FOUND ON FLEXIDISK <volume-id>

The file requested is not present on the diskette.

OVERLAPPING HEADER (BOE/EOE BOUNDARIES) ON FLEXIDISK LABELED

<volume-id>

VOLUME ACCESSIBILITY ON <unit-mnemonic> IS PROHIBITED

The ACCESSIBILITY flag in the volume header is not blank.

VOLUME LABEL ON <unit-mnemonic> IS NOT STANDARD

The STANDARD.LABEL field does not contain a W.

VOLUME SEQUENCE NUMBER MISMATCH, FOUND <sequence> LOOKING FOR

<sequence>

For multi-volume diskette files, the MV.SEQ# must be sequential.

NO SPACE AVAILABLE ON <unit-mnemonic> FOR FILE LABELED <file-id>

There is insufficient space on the diskette for the file.

NO DIRECTORY SPACE AVAILABLE FOR FILE <file-id> ON UNIT <unit>

The diskette already has 19 (the maximum allowed) files on it.

INVALID FD SPECIFICATION <parameter> ON <type> REQUEST

The request identified by <type> contains an invalid FD specification.

MISSING FD SPECIFICATION ON <type> REQUEST

The request identified by <type> does not contain an FD specification.

UNABLE TO WRITE VOLUME LABEL ON <unit-mnemonic>

The RELABEL procedure cannot write the new volume label.

MISSING RIGHT PAREN IN DUPLICATE FILE SPECIFICATION

The <input-file-number> must be enclosed in parentheses.

B 1000 Systems SSOG, Volume 2  
DISKETTE/COPY

DUPLICATE FILE NUMBER REQUESTED NOT FOUND ON UNIT <unit mnemonic>

The file requested is not present on the diskette.

DISKETTE <volume-id> FOR FILE LABELED <file-id> NOT FOUND. LOAD

AND <MIX> AX <UNIT MNEMONIC>

The requested diskette cannot be found. After readying the diskette, enter <job #> AX <unit-mnemonic> to resume the copy. Entering a null accept message causes the DISKETTE/COPY program to quit trying to process the current input specification. A new COPY PARAMETER can request a different diskette.



## SECTION 15

### DISKMAP/UTILITY

The DISKMAP/UTILITY program is a normal-state utility program that accesses the MCP II disk directory and available tables, and provides a number of optional forms of output to describe the information in the directory and the tables. A description of each option follows.

#### CHECK

Sorts disk information by address; checks for disk integrity errors.

#### MAP

Generates a listing during disk integrity check. Sets CHECK option by default.

#### ALPHA

Sorts disk information alphabetically by file ID, and prints in a format similar to a KA listing.

#### SAVE

Creates a file on disk containing the data printed as a result of the ALPHA option, in a format accessible by UPL and COBOL programs.

#### DECK

Punches a deck of cards containing the file identifiers of all files on disk, one to a card.

#### DEBUG

Prints a debug listing during phase 1 of the disk analysis. This lists in hexadecimal each structure read from the disk being analyzed.

#### DSKAVL

Creates a listing of all available areas of disk, as well as the total number of areas and segments and the size of the largest available area.

The DISKMAP/UTILITY program requires approximately 3000 segments of system disk for work file space and sorting if either the MAP or CHECK options are specified. Disk requirements for the files used by the other options vary, depending upon the number of files contained on the disk being accessed.

## OPERATING INSTRUCTIONS

An accept (AX) input message is generated at program BOJ. The message requests the unit-mnemonic of the disk to be mapped and any options desired for the run. Entering the unit-mnemonic of any system disk in a multiple systems disk configuration causes all system disks to be accessed. All options desired must be entered in a single accept message. Instructions are entered as shown in the following example:

```
DISKMAP/UTILITY =1 BOJ.  
% DISKMAP/UTILITY =1 ENTER <UNIT-MNEMONIC> AND OPTION(S).  
DISKMAP/UTILITY =1 ACCEPT.
```

```
1AXDPA MAP ALPHA DECK
```

The accept (AX) input message is not displayed if the unit-mnemonics (and options) are specified using the early AX feature. If no options are entered, MAP, CHECK, and ALPHA are set by default.

## PROGRAM SWITCHES

The DISKMAP/UTILITY program recognizes the program switches 0 through 6. These switches are used to specify the program options at the same time that the EXECUTE command is entered and thus avoid the accept (AX) messages. The meanings of the switch values are described in table 15-1.

When a switch is set, its corresponding prompt is not issued. For example, if switch 0 = 1, then the first accept message, which inquires which disk is to be mapped, is not issued, and the system disk is assumed as the disk device to be mapped. A user disk cannot be specified with a switch setting. To access a user disk, switch 0 must equal zero and the unit mnemonic on which the disk to be mapped is located must be specified in an accept (AX) input message.

Switches 1 through 6 control the program options; if any of these switches is non-zero, the ENTER OPTIONS prompt is not issued, and the options are set as described in table 15-1.

**Table 15-1. DISKMAP/UTILITY Program Switches**

Switch	Value	Meaning
0	0	Specify disk to be mapped using an accept (AX) message.
0	1	Map the system disk.
1	1	CHECK option is set.
1	2-15	MAP, CHECK options are set.
2	1	ALPHA option is set.
2	2-15	ALPHA and DSLAVL options are set.
3	1-15	SAVE option is set.
4	1-15	DECK option is set.
5	1-15	DEBUG option is set.
6	1-15	MAP system disk when programs (as well as DISKMAP/UTILITY) are executing.

## PHASES OF EXECUTION AND PROGRAM OUTPUT

The DISKMAP/UTILITY program executes in three phases, described in the following subsections.

### Phase 1

During phase 1, the DISKMAP/UTILITY program accesses the directory, available tables, and headers on the specified disk, and builds temporary work files for use in the other two phases.

If MAP or CHECK is specified, the selected disk must not be disturbed during phase 1, or the output produced can be in error. For the system disk, this means that library maintenance instructions (for example, REMOVE or CHANGE) and any commands that affect the disk directory (for example, XC, XD, TL), must not be performed during phase 1. Also, no attempt can be made to schedule or execute any other programs during phase 1.

B 1000 Systems SSOG, Volume 2  
DISKMAP/UTILITY

If MAP or CHECK has been specified on a operator disk, the MCPII saves the disk for the exclusive use of the DISKMAP/UTILITY program throughout the duration of phase 1. The disk cannot be saved while any programs are accessing the user disk, or if the disk has been specified as the default backup designation (refer to the BD input message in volume 1, section 2).

Errors detected during phase 1 are listed on the line printer. Each error specifies the disk address read by DISKMAP/UTILITY, the item being read (for example, DIRECTORY, DISK FILE HEADER), the name of the field in error (for example, AVL.SELF, DFH.AREA.ADDRESS), and the value contained by the field.

If DEBUG is specified, a listing is printed in hexadecimal that describes every structure read from the disk being analyzed. These structures include directory segments, available table segments, and file headers. The address and alphabetic identification of each structure is printed in addition to the hexadecimal representation of the data read.

## Phase 2

Phase 2 is invoked only if the MAP or CHECK option is requested. During phase 2, the disk information obtained during phase 1 is sorted by address and then checked for integrity errors. If MAP is specified, a listing of the disk areas sorted by address is generated.

The line printer listing produced by the MAP option contains one line for each logical portion of disk space, giving the starting and ending addresses, the size in disk segments, and a description of the chunk, plus the area number and file-id for all files. Any errors detected are identified by one of the following error messages.

### MISSING DISK AREA

The specified chunk of disk is unaccounted for in all MCPII directories and available tables.

### OVERLAPPING DISK AREAS

The specified chunk of disk is described by more than one header, directory, or available table entry.

### ADDRESS BEYOND DISK CAPACITY

The specified chunk of disk begins at a disk address greater than any shown by the Master Available Table.

### INVALID DFH.SELF

The DFH.SELF address field in the disk file header is not the same as the address from which the header was actually read.

### DFH.AREAS.RQST > 105

The header requested more than 105 areas, the maximum allowed.

### DFH.AREAS.CTR > DFH.AREAS.RQST

The count of areas in use in the disk file header is greater than the maximum number of areas it requested to use.

### INVALID DFH.HDR.SIZE

The DFH.HDR.SIZE field in the disk file is incorrect.

### INVALID DISK.SELF

The DISK.SELF field in a directory segment is not the same as the address from which the directory segment was actually read.

B 1000 Systems SSOG, Volume 2  
DISKMAP/UTILITY

**INVALID BPS NUMBER**

The base pack serial number contained in this continuation disk's label does not match that in the DISK.ADDRESS field.

Areas of disk described as TEMPORARY appear on the listing when mapping systems disk; these are work areas in-use by the DISKMAP/UTILITY program and are returned at program EOJ.

**Phase 3**

Phase 3 is invoked if the ALPHA, SAVE, or DECK option is requested. During phase 3, an output file is created after sorting the disk file header information by file-identifier. Any one or all of the three types of output (ALPHA, SAVE, and DECK) can be produced during a single execution of the DISKMAP/UTILITY program.

The listing generated by the ALPHA option can contain asterisks to the right of some of the SAVE FACTORS printed. Asterisks signify an expired file, which is a file whose LAST ACCESS DATE plus the SAVE FACTOR is less than the current date maintained by the MCP. This indicates that the file either has not been accessed for an extended period of time, or does not have any SAVE FACTOR specified in the DISK FILE HEADER (refer to the SAVE attribute of the FILE statement for syntax to specify the SAVE FACTOR). Such files should be examined as possible candidates for removal from disk; however, the MCPII does not automatically remove an expired file.

The file produced by the SAVE option has the following attributes:

internal-file-id:	"USER"
external-file-id:	"USER"
record size:	180 bytes
records per block:	5
areas:	25
blocks per area:	100

The first record of the file contains global information about the disk being analyzed. The format (in COBOL notation) of this record follows:

```
01  HEADER-RECORD.  
   02  PACK-NAME           PC X(10).  
   02  PACK-SERIAL-NUMBER  PC 9(6).  
   02  PACK-HARDWARE-TYPE  PC 9(1).
```

**NOTE**

PACK-HARDWARE-TYPE must be 0 for a head-per-track disk device, 1 for a disk pack, or 2 for a disk cartridge.

B 1000 Systems SSOG, Volume 2  
DISKMAP/UTILITY

The information for each file on the disk occupies from one to four records in the file created by the SAVE option, based upon the number of disk areas in use by the file. The record format (in COBOL notation) is as follows:

```
01  DATA-RECORD.
    02  FILE-INDENT                PC X(20).
    02  RECORD-COUNT               PC 9      CMP.
    02  HEADER-DISK-ADDRESS        PC 9(12)  CMP.
    02  RECORD-SIZE               PC 9(4)   CMP.
    02  RECORDS-PER-BLOCK         PC 9(4)   CMP.
    02  BLOCKS-PER-AREA           PC 9(6)   CMP.
    02  SEGMENTS-PER-AREA         PC 9(6)   CMP.
    02  AREAS-DECLARED            PC 999   CMP.
    02  AREAS-IN-USE              PC 999   CMP.
    02  EOF-POINTER              PC 9(7)   CMP.
    02  FILE-TYPE                 PC 999   CMP.
    02  CREATION-DATE             PC 9(5)   CMP.
    02  LAST-ACCESS-DATE          PC 9(5)   CMP.
    02  SAVE-FACTOR               PC 9(4)   CMP.
    02  CREATE-TIME               PC 9(7)   CMP.
    02  UPDATE-DATE              PC 9(5)   CMP.
01  AREA-ADDRESSES REDEFINES DATA-RECORD.
    02  AREA-ADDRESS OCCURS 30 PC 9(12)  CMP.
```

NOTE

Expired files on the B 1000 system are not removed. They are marked with an asterisk to indicate that the last access date was prior to the save factor specified when the file was created. Expired files can be copied to other systems. When expired disk files are copied to other systems, the disk file header that contains the save factor field is replaced by a new disk file header created by the receiving system.

The RECORD-COUNT field contains a count of how many records are required for file description. For example, if a file has less than 16 areas in use (AREAS-IN-USE), it only requires one record; for 16 through 45 areas in use, it requires two records; for 46 through 75 areas in use, it requires three records; and for 76 through 105 areas in use, it requires four records.

Disk addresses are stored in the following format:

```
01  DISK-ADDRESS-FORMAT.
    02  SERIAL-NUMBER-FLAG        PC 9      CMP.
    02  PORT-NUMBER               PC 9      CMP.
    02  CHANNEL-NUMBER           PC 99     CMP.
    02  UNIT-NUMBER              PC 99     CMP.
    02  DISK-ADDRESS             PC 9(6)   CMP.
```

B 1000 Systems SSOG, Volume 2  
DISKMAP/UTILITY

If the SERIAL-NUMBER-FLAG field is 1, the PORT-NUMBER, CHANNEL-NUMBER, and UNIT-NUMBER fields are not used; the DISK-ADDRESS field contains a disk cartridge/pack serial number (for multipack files). If the SERIAL-NUMBER-FLAG field is zero (0), the other fields contain decimal values describing the disk address.

The entire AREA-ADDRESS field is zero for an unallocated area.

The AREA-ADDRESS fields are stored using the DISK-ADDRESS-FORMAT field, with 30 addresses on each 180-byte record (except for the first record). The first 15 AREA-ADDRESS fields are stored in the second half of the first record, as shown in table 15-2, from which it can be seen that the address for area number 3 is located in the AREA-ADDRESS(18) field of record 1; the address for area number 30 is located in AREA-ADDRESS(15) field of record 2; and so forth.

The date fields CREATION-DATE and LAST-ACCESS-DATE use the Julian format (YYDDD) to represent the date.

**Table 15-2. DISKMAP/UTILITY Area Address Records**

Area Number	Record Number	Subscripts
1-15	1	16-30
16-45	2	1-30
46-75	3	1-30
76-105	4	1-30

The headings for all listings produced by the DISKMAP/UTILITY program include the disk name and serial number of the disk being mapped. If the user disk being mapped is a continuation disk, the base pack serial number is printed. Program listings contain a 3-or 4-character mnemonic for file types rather than a number. The mnemonics and numbers for file types on the B 1000 systems are shown in table 15-3.

**Table 15-3. DISKMAP/UTILITY Numeric and Mnemonic Value**

File Description	Number	Mnemonic
Log file	001	LOG
Directory file	002	DIR
Pseudo reader file	003	DECK
Backup print file	004	PRT
Backup punch file	005	PCH
Dump file	006	DUMP
Interpreter file	007	INTP
Code file	008	CODE
Data file	009	DATA
Variable length data file	011	VAR
Intrinsic file	012	INSC
DMS audit file	015	AUDT
Usercode file	016	USER
Relative file (MCP)	017	REL
Index seq. global file (MCP)	018	IS.G
Index seq. data set file (MCP)	019	IS.D
Index seq. index file (MCP)	020	IS.I
Indexed tag file (old style)	021	TAGS
Indexed data file (old style)	022	INXD
MCPII temporary file	023	TEMP

## SECTION 16

### DMPALL

The DMPALL program is a generalized media-conversion utility program that operates under MCPPII control. The DMPALL program has four distinct functions.

1. Printing/punching the directory of Load Dump tapes.
2. Printing the contents of files.
3. Reproducing data from one hardware device to another.
4. Concatenating files to produce a single disk file.

### OPERATING INSTRUCTIONS

The DMPALL program is executed from the console keyboard or from the card reader. By default, the DMPALL program accepts its input specifications from the console keyboard, entered through accept (AX) input messages. When program switch 0 is set, the DMPALL program accepts its input specifications from a card file-labeled SPEC. The DMPALL program processes each input specification as it is received; after completing the requested action, the DMPALL program reads and processes the next input specification.

### PROGRAM SWITCHES

Setting the program switches changes the manner in which the DMPALL program executes. A switch is considered set when it is equal to 1. The switches used and their functions are described in table 16-1.

**Table 16-1. DMPALL Program Switches**

Switch	Value	Meaning
0	0	Specifications accepted only from the ODT.
	1	Specifications accepted only from cards.
1	0	Continue process until the end.
	1	Terminate processing the current specification and proceed to the next specification.
2	0	Number of syntax errors encountered are not displayed at any time.
	1	Number of syntax errors are displayed at EOJ.
7	0	Continue with next specification.
	1	Terminate DMPALL after completion of the next executed specification.
8	0	Interpret card decks.
	1	Do not interpret cards.
9	0	Terminate DMPALL if an error is encountered in input files.
	1	Ignore input file errors (for example, parity errors).

When DMPALL detects that switch 1 is set, it terminates the specification being processed and then resets switch 1 by zipping a <job #> SWITCH 1=0 message; thus, processing of only the current specification is terminated.

## COMMAND SPECIFICATIONS

DMPALL, by default, issues prompts asking for input specifications. The responses are entered through the console keyboard.

Specifications can, as an option, be entered from a file labeled SPEC if DMPALL is executed with switch 0 set to 1. By default, this file is defined as a card file. The DMPALL card control deck has the following format:

```
? EXECUTE DMPALL SWITCH 0:= 1
? DATA SPEC
  <specification cards>
? END
```

Changing the external file identifier of the file SPEC with a FILE statement, as shown in the following, has the same effect as setting switch 0.

```
? EXECUTE DMPALL
? FILE SPEC NAME= SPECS;
? DATA SPECS
  <specification cards>
? END
```

There can be more than one record in the specification file; however, a single specification string cannot extend beyond one record. The file containing the specifications is loaded to disk by DMPALL (to free the reader for other programs); each specification is then processed from disk.

All specifications are entered in free-form format and can be up to 96 characters in length. Options and keywords within any one specification are separated by either a space or a comma, or any combination thereof. A semicolon terminates any specification string, after which comments can be entered.

## INPUT SPECIFICATIONS

The input specifications must request one of the DMPALL program's four functions: PD, LIST, COPY (PERFORM), or CAT (concatenation). These commands and their functions are described in detail in the following pages.

### Print Directory (PD) Specifications

The directory of a library tape created by the SYSTEM/LOAD.DUMP program can be listed on the line printer or punched into cards using the PD (Print/Punch Directory) specification.

A description of the syntax of the PD specification follows.

```
_____ PD [PUNCH] < tape-identifier > _____|
```

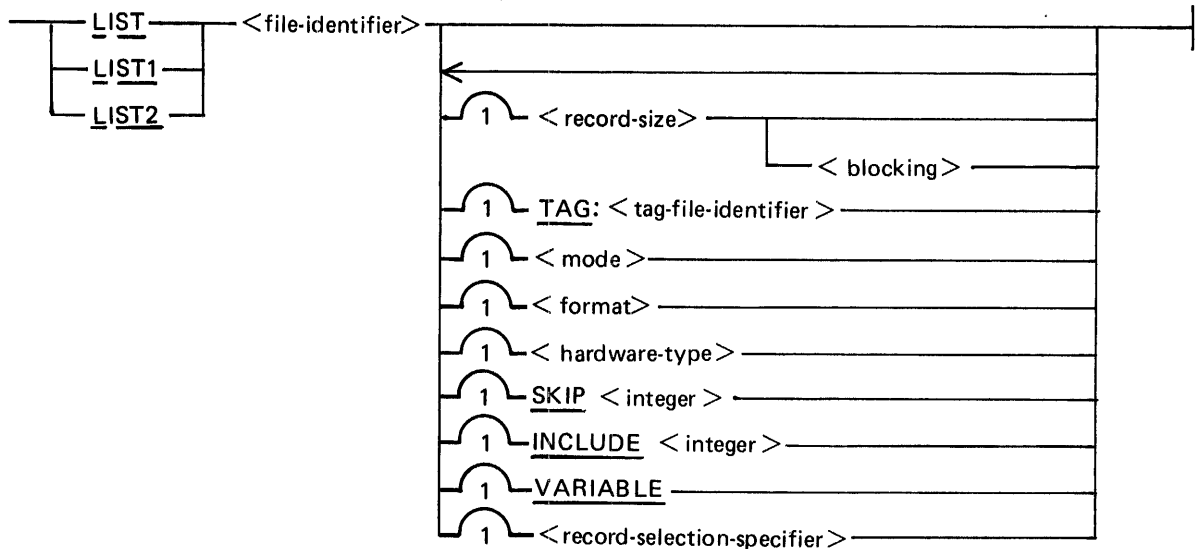
When PUNCH is specified, the tape directory is punched onto cards (one file-identifier per card). If the PUNCH option is omitted, the tape directory is listed on the line printer.



## LIST Specifications

A line printer listing of files in hexadecimal, alphanumeric, or combined form can be generated from card, magnetic tape, cassette, paper tape, or disk files. Options allow the selective inclusion or exclusion of records in the listing, based upon operator-specified conditions.

A description of the specification string used to list a file follows.



### List-specifier

The <list-specifier> must be one of the following:

Single-spacing: LIST, LST, LIST1, OR LST1  
Double-spacing: LIST2 or LST2

### File Specifications

The <file-identifier> entry must follow the <list-specifier>. The format of the <file-identifier> is the same as that used in MCPPII control instructions and consists of one to three identifiers separated by slashes. Any identifier that is entirely numeric or contains special characters must be enclosed within quotation marks. Unlabeled tape files are specified by a <file-identifier> of "NONE" (the quotation marks are required).

The <record-size> option, if used, must be the first numeric entry following the <file-identifier> and is specified in bytes. If omitted, a <record-size> of 80 is assumed, except for disk and labeled B 1000/B 6000/B 7000 tape files, where the <record-size> with which the file was created is assumed.

The <blocking> option, if used, must be the second numeric entry following the <file-identifier>. If omitted, a <blocking> of one is assumed. For disk and labeled B 1000/B 6000/B 7000 tape files, when both the <record-size> and <blocking> are omitted, the blocking with which the file was created is used.

### TAG Option

The TAG option can be used to specify a tag file for accessing a COBOL or RPGII index-sequential file. The tag file must be created with the COBOL or RPGII naming conventions. The tag file specified is read sequentially, and the key field of each tag record is used to access the associated indexed file record.

### Mode Option

The <mode> option can specify one of the following (the quotation marks are required):

“ASCII”  
“BCL”  
“BINARY”  
“EBCDIC”

If the mode option is omitted, EBCDIC is assumed as the default, except for labeled B 1000/B 6000/B 7000 tape files, where the mode with which the file was created is assumed.

### Format Option

The <format> option, if used, must specify one of the formats described in table 16-3. When a format is not specified, the file is listed in both alphanumeric and hexadecimal format.

Alphanumeric: ALFA or A

Numeric/hexadecimal: N,NUM,H, or HEX

### Hardware-Type Option

Table 16-2 shows which hardware devices can be specified by the <hardware-type> option. Either the long or the short form can be used.

**Table 16-2. DMPALL Hardware Device Notation-LIST Command**

Hardware Device	< hardware-type >	
	Long Form	Short Form
Card files	CARD	CRD
96-column card files	CARD96	C96
Binary card files	BINARY	BIN
Magnetic tape files	TAPE	MTP
7-track tape files	TAPE7	MT7
9-track tape files	TAPE9	MT9
Cassette tape files	CASS	CAS
Paper tape files	PAPER	PPT
Disk files	DISK	DSK
Multi-pack disk files	MULTI	MPF

If the <hardware-type> option is omitted, DISK is assumed by default. The <hardware-type> BINARY is applicable only to 80-column card files. If BINARY is specified, a card with END-OF-DECK punched in columns 1 through 11 must be the last card in the input deck.

### SKIP Option

The **SKIP** option is used to begin printing at the record specified by the <integer>. The first record in the input file is considered record number one.

### INCLUDE Option

The **INCLUDE** option is used to specify the number of records to be printed. The default is to print until end of file is reached. **INCLUDE** is abbreviated **INCL**.

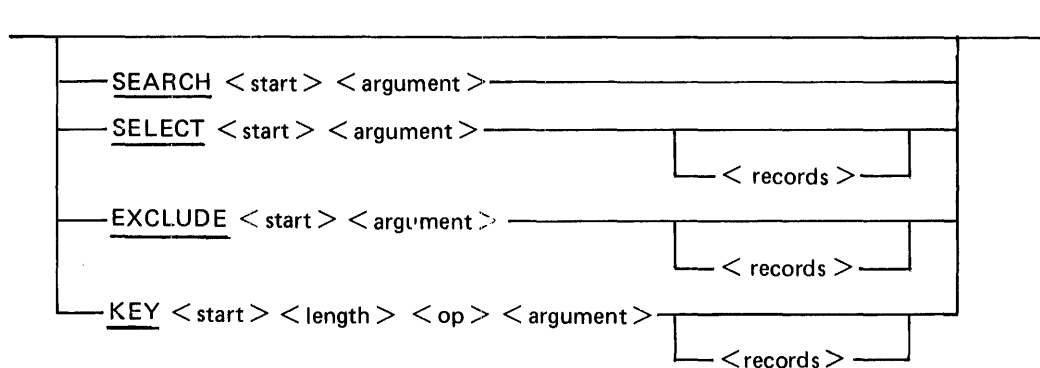
### VARIABLE Option

The **VARIABLE** option is used to identify tape or disk files having variable-length records. If the input file is a disk file, the **DMPALL** program checks the disk file header to verify that the file consists of variable-length records; if not, the **VARIABLE** option is ignored. **VARIABLE** is abbreviated **VARY**.

### Record Selection Option

The <record-selection-specifier> option allows parameters to be specified to control which records from the file are to be printed or ignored. Detailed <record-selection-specifications> can be used in conjunction with the selection capabilities offered by the **SKIP** and **INCLUDE** options.

The syntax of the <record-selection-specifier> is described as follows.



#### **SEARCH Record Selection Option**

The **SEARCH** option is used to specify that printing is to begin with the first record that satisfies the entered parameters (within the range specified by the **SKIP** and **INCLUDE** options). **SEARCH** is abbreviated **SEA**.

#### **SELECT Record Selection Option**

The **SELECT** option is used to specify that only records that satisfy the entered parameters (within the range specified by the **SKIP** and **INCLUDE** options) be printed. **SELECT** is abbreviated **SEL**.

#### **EXCLUDE Record Selection Option**

The **EXCLUDE** option is used to specify that only records that do not satisfy the entered parameters (within the range specified by the **SKIP** and **INCLUDE** options) be printed. **EXCLUDE** is abbreviated **EXC**.

**KEY Record Selection Option**

The KEY option functions in a manner similar to the SELECT option; however, the parameters can specify more detailed selection criteria.

**Required Parameters**

The <record-selection-specifier> options require the following parameters:

<start>

Specifies the starting position of the field to be compared in the input record. The first position in the record is relative position one. The numeric <start> entry specifies either a byte, digit, or bit starting position, depending upon the <argument> type.

<argument>

The data string against which the specified field in each record is compared. The field type is determined by the <argument> type, as shown in table 16-3.

Alphanumeric data is delimited by either blanks or quotation marks. Hexadecimal data is delimited by @ (at) signs. Digit data and binary data are also delimited by @ signs but with mode indicators in parentheses after the first @. The mode indicator for digit data is 4; for binary data, it is 1. Refer to table 16-3.

<records>

Specifies the maximum number of records satisfying the comparison criteria to be printed.

<length>

Specifies the size of the compare field in the record for use with the KEY option. The maximum <length> that can be specified is 20. The units (bytes, digits, or bits) associated with the <length> are determined by the <argument> type.

<op>

Specifies the type of comparison to be performed between the compare field in the record and the <argument> for use with the KEY option. The <op> field can be one of the following:

<op>	<Meaning>
GTR or >	Greater than
GEQ or >=	Greater than or equal
EQL or =	Equal
LEQ or <=	Less than or equal
LSS or <	Less than
NEQ	Not equal

**Table 16-3. DMPALL Record Selection Parameters**

<argument>	type	Field type	Sample <argument>
Alphanumeric		Byte	"ABCDEF 1234"
Hexadecimal		Byte	@0123ABCDEF@
Digit		Digit	@(4)01@
Binary		Binary	@(1)11001@

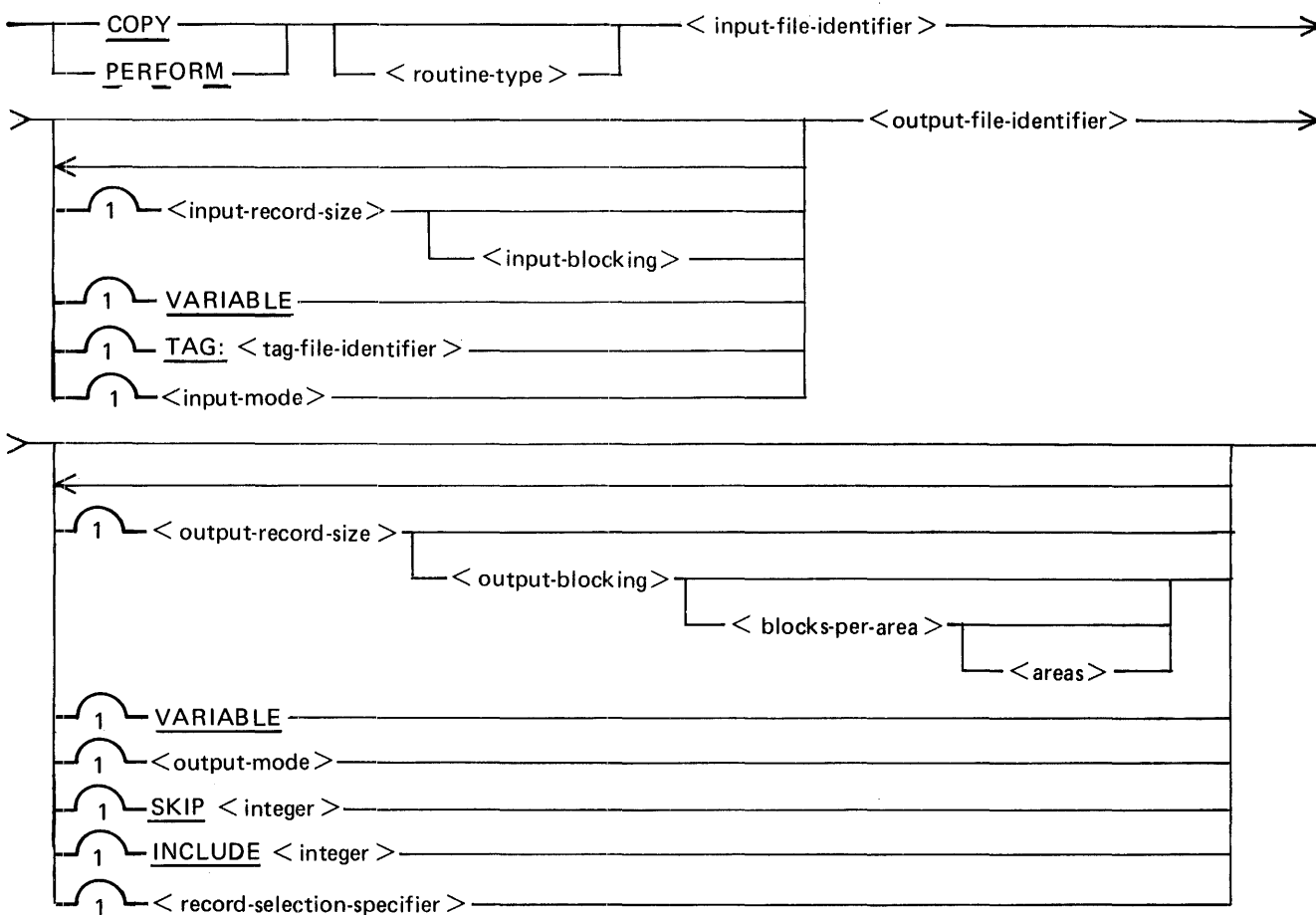
## OUTPUT FORMAT

The output listing contains the file-identifier, record size, blocking, and the current date and time. In addition, a listing of a disk file also contains the end-of-file pointer value in the heading. For each record printed, the current record count is printed in the left-hand margin.

## REPRODUCING (COPY, PFM) SPECIFICATIONS

Records can be copied from any card, magnetic tape, paper tape, or disk file to any output file desired. Various file attributes, including record sizes, blocking factors, and file-identifiers can be changed during the copy process. It is also possible to select certain records from the input file to be copied based upon specified conditions.

A description of the specification string used to copy a file follows.



## Routine Type Specifier

The <routine-type> specifies the input and output file <hardware-types>, which can include any two of the hardware devices listed in table 16-4.

**Table 16-4. DMPALL Hardware Device Notation**

Hardware Device	<hardware-type>	
	Long Form	Short Form
Card files	CARD	CRD
96-column card files	CARD96	C96
Binary card files	BINARY	BIN
Magnetic tape files	TAPE	MTP
7-track tape files	TAPE7	MT7
9-track tape files	TAPE9	MT9
Cassette tape files	CASS	CAS
Paper tape files	PAPER	PPT
Disk files	DISK	DSK
Multipack disk files	MULTI	MPF
Printer output files	LIST	LST

The <routine-type> can be specified using either a longhand or shorthand form. The longhand form is specified using two long form <hardware-type> entries, separated by spaces or the optional word TO. The shorthand form is specified using two concatenated short form <hardware-type> entries (not separated by any spaces). If the <routine-type> option is omitted, DISK TO DISK is assumed by default. For example, the following specifications all request that a card file be copied to magnetic tape:

```
CARD TO TAPE
CARD TAPE
CRDMTP
```

A <hardware-type> of BINARY is applicable only to 80-column card files. If BINARY is specified for the input file, a card with END-OF-DECK punched in columns 1 through 11 must be the last card in the input deck.

A <hardware-type> of LIST can only be specified for an output file and causes the file to be listed on the line printer (without headings or record numbers), using the specified output file parameters.

## Input File Specifications

The format of the <input-file-identifier> is the same as that used for <file-identifier> in the list specifications.

The <input-record-size> and <input-blocking> options are specified in the same manner as the <record-size> and <blocking> option in the list specifications.

The VARIABLE option is used to specify that the input tape or disk file has variable-length records. If the input file is a disk file, the DMPALL program checks the disk file header to verify that the file consists of variable-length records; if not, the VARIABLE option is ignored. VARIABLE can be abbreviated VARY.

The TAG option is specified in the same manner and has the same effect as described for the TAG option in the list specifications.

An optional <input-mode> can be specified in the same manner as the <mode> in the list specifications.

## Output File Specifications

The format of the <output-file-identifier> is the same as that of the <input-file-identifier>.

The first numeric entry following the <output-file-identifier> must be the <output-record-size>, specified in bytes. If omitted, an <output-record-size> of 80 is assumed, unless both the input and output files are either disk files or labeled B 1000/B 6000/B 7000 tape files, when the <output-record-size> is assumed to be the same as the <input-record-size>.

The second numeric entry following the <output-file-identifier> is the <output-blocking>. If omitted, a blocking factor of one is assumed. If both the input and output files are either disk files or labeled B 1000/B 6000/B 7000 and the <output-record-size> is omitted, the <output-blocking> is assumed to be the same as the <input-blocking>.

The third numeric entry following the <output-file-identifier> is the <blocks-per-area> and is only applicable to disk files. If omitted, 100 blocks per area is assumed. If both the input and output files are disk files and both the <output-record-size> and <output-blocking> are omitted, the <blocks-per-area> is assumed to be the same as the blocks per area of the input file.

The fourth numeric entry following the <output-file-identifier> is the <areas>, which specifies the number of disk areas to assign to the output file (only applicable to disk files). If omitted, 25 areas are assumed. If both the input and output files are disk files and the <output-record-size>, <output-blocking>, and <blocks-per-area> are omitted, the <areas> is assumed to be the same as the number of areas assigned to the input file.

An optional <output-mode> can be specified in the same manner as the <input-mode>.

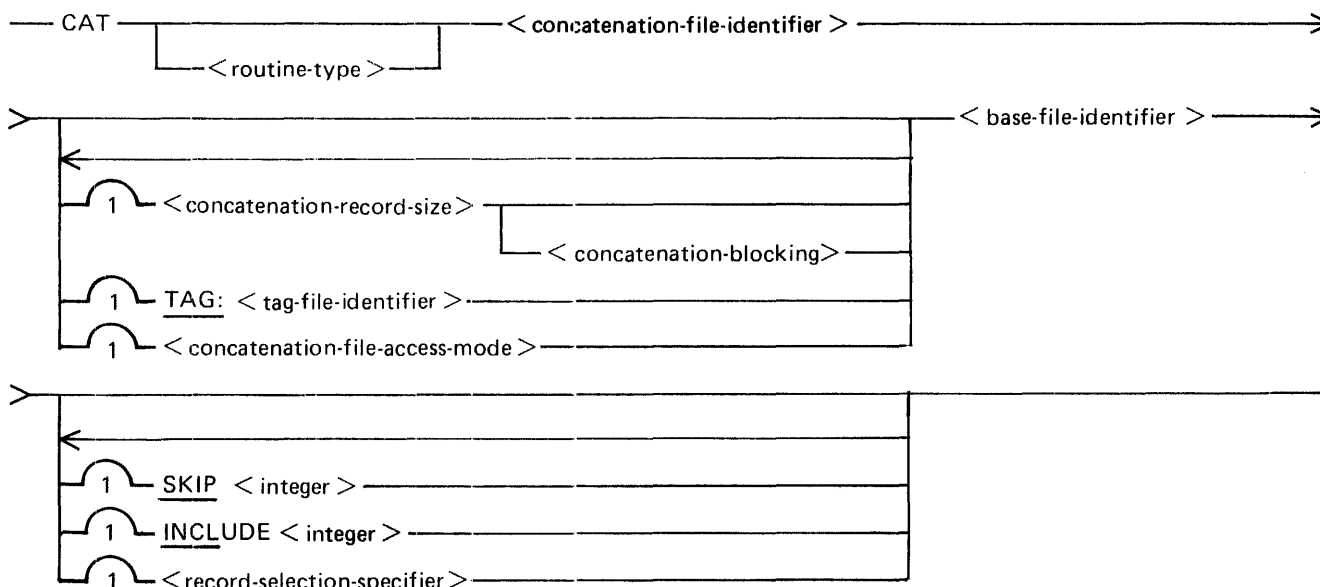
If VARIABLE is specified following the output file specifications, both the input and output files are assumed to have variable-length records. If the input file is a disk file, the DMPALL program checks the disk file header to verify that the file consists of variable-length records; if not, the VARIABLE option is ignored. VARIABLE can be abbreviated as VARY.

The <record-selection-specifier> option is used to specify the conditions under which records are selected from the input file for copying to the output file. The syntax is the same as is shown for the list specifications.

## CONCATENATION (CAT) SPECIFICATIONS

Records from any card, magnetic tape, paper tape, or disk file can be copied onto the end of a specified disk file, called the base file. The base file is opened, and the specified concatenation file records are added to the base file; when the base file is closed, the EOF pointer is updated to include the new records. If the concatenated file size would exceed the size limit of the base file, and if both the concatenation and the base files are disk files, then the DMPALL program copies the base file, enlarging it, and then adds the concatenation file. Likewise, if the base file is crunched, the DMPALL program copies the base file before performing the concatenation. If both files are not disk files, the operator must ensure that the concatenated file will not exceed the file size limit of the base file.

The specification string used to concatenate files has the following format.



### Routine Type Specifier

The <routine-type> specifies the concatenation and base file <hardware-types>. Refer to the subsection entitled Reproducing (COPY, PFM) Specifications, Routine Type Specifier for a description of the syntax of <hardware-type> specifications. When concatenating files, the base file must be a disk file. The concatenation file can have any input hardware device; LIST (printer) is not a valid input device. Either the longhand or shorthand form of the <routine-type> can be specified; DSKDSK is assumed as default when <routine-type> is omitted.

### Concatenation File Specifications

The format of the <concatenation-file-identifier> is the same as that used for <file-identifier> in the list specifications.

The <concatenation-record-size> and <concatenation-blocking> options are specified in the same manner as the <record-size> and <blocking> options in the list specifications. If the record-size of the concatenation file is larger than the record-size of the base file, then the concatenation file records are truncated when they are added to the base file.

The TAG option is specified in the same manner and has the same effect as described in the list specifications.

The optional <concatenation-file-access-mode> is specified in the same manner as described in the List Specifications.

### Base File Specifications

The format of the <base-file-identifier> is the same as the <concatenation-file-identifier>. The <base-file-identifier> must name an existing disk file.



A record-size and blocking factor cannot be specified for the base file. These values are taken from the disk file header of the base file.

The optional <base-file-access-mode> is specified in the same manner as the <concatenation-file-access-mode>.

The SKIP, INCLUDE, and <record-selection-specifier> options are used to specify the conditions under which records are selected from the concatenation file for addition to the base file. The syntax for these options is the same as described in the subsection entitled List Specifications.

## TAPE PARITY MODIFICATIONS

To change the parity for an input or output tape file from ODD to EVEN, use one of the following FILE statements with the EXECUTE, MODIFY, or DYNAMIC MCPH control instructions.

Input files: FILE INP.FILE EVEN;  
Output files: FILE OUTP.FILE EVEN;

## EXAMPLES

Examples of input specification strings and of the format of a DMPALL execution control deck format are described in the following paragraphs.

### Sample Input Specifications

The following specification string is used to list a disk file on pack USERA in alphanumeric format, beginning with the fiftieth record:

```
LIST USERA/PAYROLL/A SKIP 50
```

The following specification string is used to start printing a tape file with the first record that has the letters CUBE beginning in byte 37 of the record (a byte contains one character):

```
LIST ABC MTP SEARCH 37 "CUBE"
```

The following specification string is used to print in hexadecimal format only the first record of records 10 through 20 that has a hexadecimal @FF@ beginning in byte 1.

```
LIST ABC H SKIP 10 INCL 10 SELECT 1 @FF@
```

To print (double-spaced) only the records whose second byte is greater than hexadecimal @C0@, any of the following three specifications can be used (note the different <argument> types and their effect on the <start> and <length> specifications):

```
LIST2 ABC KEY 2 1 GTR @C0@  
LIST2 ABC KEY 3 2 GTR @(4)C0@  
LST2 ABC KEY 9 8 > @(1)11000000@
```

The following specification string is used to copy a card file labeled CARDS to a disk file labeled SOURCE having 80-byte records blocked 9:

```
COPY CRDDSK CARDS SOURCE 80 9
```

The following specification string is used to copy an ASCII tape file to an EBCDIC disk file:

```
PFM TAPE TO DISK TAPEFILE "ASCII" DISKFILE
```

The following specification string is used to copy a disk file, exclude all records that have CUBE beginning in byte 37, and change the blocking factor:

```
PERFORM DSKDSK ABC 80 1 DEF 80 9 EXC 37 "CUBE"
```

## Sample Executions

To copy a card file labeled XXX to a disk file labeled DSKFILE, and then to list the resulting disk file, the following execution deck can be used:

```
?EXECUTE DMPALL SWITCH 0=1  
?DATA SPEC  
COPY CRDDSK XXX 801 DSKFILE 80 1  
LIST DSKFILE A  
?DATA XXX  
  <data cards>  
?END
```

## PROGRAM OUTPUT

The DMPALL program generates two types of output: output files produced by successful completion of the input specifications and error messages produced when DMPALL cannot process an input specification string.

### Output Files

#### Disk Files

The DMPALL program alters copied disk files as and when requested in the input specification. If there are no alterations requested, the disk file header retains the original creation date and file attributes.

#### Tape Files

The DMPALL program can write one file to a tape. This file is written to the tape in the format of a data file, and the tape is not a library tape.

#### Printer Listings

The LIST command includes heading information describing the file being printed in addition to the data. A COPY command, which specifies LIST as the output device, generates a listing that contains only data (no heading information).

#### Card Files.

The DMPALL program punches card files when CARD or C96 is specified as the output media during a COPY command. Only data is punched. Records longer than 80 bytes are truncated when punched to an 80-column CARD, and records longer than 96 bytes are truncated when punched to a 96-column card (C96).

## Paper Tape Files

Only data (no headers) is punched to paper tape files.

## **ERROR MESSAGES**

The following are the error messages generated by the DMPALL program.

<input-file-identifier> NOT ON DISK

The file is not in the disk directory.

MF.ID OR PACK.ID NUMERIC

Files identifiers that are numeric must be enclosed in quotation marks.

INVALID HARDWARE TYPE OR OUTPUT FORMAT

NUMERIC VALUE MUST FOLLOW <option>

SKIP, INCLUDE, SEARCH, SELECT, EXCLUDE, and KEY must be followed by parameters.

<input> ITEM SHOULD NOT APPEAR BEFORE LIST OR PERFORM

DMPALL input must start with either PD, PERFORM, PFM, COPY, LIST, LST, LIST1, LST1, LIST2, or LST2.

<areas> AREAS EXCEEDS 105 MAXIMUM

105 is the maximum number of output areas that can be specified.

SEARCH FIELD OUTSIDE RECORD

<start> + <length> is greater than <record-size>.

INVALID SECOND HDW-NAME <name>

MISSING SECOND HDW-NAME

NO SPECS

A blank card was read from the specification deck.

<input> IS NOT VALID AFTER SECOND FILE-NAME

<input-file-identifier> IS A SYSTEM LOG AND CANNOT BE COPIED OR LISTED

INVALID RECORD LENGTH <record-length>

The maximum <record-size> allowed is 8000 bytes (4000 bytes if either the TAG option or ASCII conversion is specified).

B 1000 Systems SSOG, Volume 2  
DMPALL

NO RECORDS PROCESSED - END OF FILE REACHED

The input file is empty, has been skipped past EOF, or no records satisfying the parameters specified for the <record-selection-specifier> were found. If a copy operation was specified, the output file is empty (EOF.POINTER==0).

EQUAL SIGN IS NOT A VALID FILE-ID

Files must be specified to DMPALL one at a time.

<argument> IS NOT A VALID KEY ARGUMENT

<argument> had invalid data for that type of argument.

PARITY ERROR - REC.NUM: <record-number>

Advisory message; processing continues.

<tag-file-identifier> I/O ERROR <record-number>

Advisory message; processing continues.

INVALID KEY LENGTH

Key <length> must be between 1 and 20, inclusive.

INVALID KEY OPERATOR <op>

WARNING - <output-file-identifier> IS NOT AN IAD FILE

This is a warning message indicating that the input file to a copy operation was an Installation Allocated Disk (IAD) file. The output file created is not an IAD file.

INPUT ERROR: BASE FILE IS NOT A DISK FILE

When using the CAT command to concatenate files, the BASE file (the second file named) must be a disk file.

INVALID PARAMETERS SPECIFIED FOR BASE FILE

When using the CAT command, the only valid parameter for the BASE (output) file is <access-mode>.

CONCATENATION ALLOWED ONLY ON DATA FILES

<base-file-id> IS NOT A DATA FILE.

The base file of a CAT command must be a data file. The concatenation file can be a non-data file; however, the resultant file is a data file.

## SECTION 17

### FILE/LOADER

The FILE/LOADER program is a normal-state utility program whose purpose is to load to disk the card decks punched by the utility program FILE/PUNCHER.

### OPERATING INSTRUCTIONS

The utility program FILE/LOADER is executed from either the console keyboard or from a card reader; however, the input specifications must be entered through an input card file labeled CARDS. More than one card deck can be loaded during a single execution of the utility program.

The FILE/LOADER execution deck consists of the standard EXECUTE control card, the dollar and asterisk specification control cards and the data cards for each file to be loaded, and an END card.

### SPECIFICATION CONTROL CARDS

The following paragraphs describe the control card specifications.

#### Dollar Card

The dollar card specifies the file-identifier of the card deck. The dollar card is punched by FILE/PUNCHER and identifies the file. The file-identifier can be changed by modifying the dollar card.

The syntax of the FILE/LOADER dollar card is:

\$ < file-identifier >

The dollar sign (\$) character must be in column one with the file-identifier entered in free-form format in columns 2 through 80.

#### Dollar-Dollar Card (\$\$)

Files produced by the MIL compiler (Micro Implementation Language) must be loaded using the dollar-dollar (\$\$) card to distinguish them from card files created by FILE/PUNCHER. The asterisk (\*) card must not be used when using the dollar-dollar (\$\$) card.

A description of the card format produced by the MIL compiler follows. Six MIL-code cards are necessary to contain the data for a disk segment.

Column	Description
1-6	Load address (Relative)
7	Blank
8-9	Number of bits on card
10	Blank
11-70	Data in hexadecimal format (30 Bytes)
71-72	Blank
73-80	Card sequence number

B 1000 Systems SSOG, Volume 2  
FILE/LOADER

The format of the FILE/LOADER dollar dollar card is

\$\$ < file-identifier >

### Asterisk Card

The asterisk (\*) specification control card provides header information that describes the file that is being loaded to disk. This card is produced by the FILE/PUNCHER program and must not be changed prior to input. When the asterisk card is missing, the card file is assumed to be a code file. The asterisk card must not be used when the first card of the file is a dollar-dollar card.

The format of the FILE/LOADER asterisk card follows.

Column	Description
1	* (asterisk)
3	File type: 1 LOG 3 Control Deck 4 Backup Punch 5 Backup Print 6 Dump 7 Interpreter 8 Code 9 Data
5-10	EOF pointer
12-17	Record size in bits, right justified
19-20	Records per block, leading zeros
22-24	Areas (optional)
26-31	Segments per area

### General Rules

1. If a code file is being loaded, the asterisk card is optional and default values are assumed.
2. If a code or interpreter file is designated on the asterisk card, only the EOF pointer is read from the card, and all other fields are ignored. If the EOF pointer field is blank, a default size of 100 segments for interpreter files or 500 segments for code files is assigned to the file being loaded.
3. All code and interpreter files are closed with CRUNCH, which frees the area not being used for the file.

### SAMPLE EXECUTION DECK

```
? EXECUTE FILE/LOADER DATA CARDS
$ file-identifier
* ... (optional)
  data deck
[$ file-identifier]
[* -----]
[ data deck ]

? END
```

## PROGRAM OUTPUT

The FILE/LOADER program produces three types of output: files on disk, informative messages, and error messages.

### Files On Disk

The FILE/LOADER program builds a file on disk according to the specifications on the asterisk control card. The file identifier is added to the disk directory and references a file that contains identical information as the source file on punched cards.

### Informative Messages

The FILE/LOADER program displays the following message on the operator display terminal (ODT) after loading each punched card deck.

< file-identifier > LOADED

### Error Messages

MISSING "\$" IN COLUMN ONE

The first card of the input deck does not have a dollar sign (\$) character in column one.

MISSING file-identifier

The first card of the input deck has a dollar sign (\$) character in column one, but is otherwise blank.

SEQUENCE ERROR FOLLOWING <nnnn> - <file-identifier> NOT LOADED

The card following the card number specified is out of sequence.

RECORD.SIZE SPECIFIED <nnnn> - <file-identifier> NOT LOADED

AREAS SPECIFIED = 0 -<file-identifier> NOT LOADED

RECORDS.BLOCK SPECIFIED = 0 - <file-identifier> NOT LOADED

SEGMENTS.AREA SPECIFIED = 0 - <file-identifier> NOT LOADED

EOF.POINTER SPECIFIED = 0 - <file-identifier> NOT LOADED

INVALID FILE TYPE SPECIFIED - <file-identifier> NOT LOADED

BLOCK SIZE 56 - <file-identifier> NOT LOADED

EMPTY DECK - <file-identifier> NOT LOADED

There are no cards following the specification card(s).

"" CARD INVALID - <file-identifier> NOT LOADED

An asterisk (\*) card following a dollar-dollar (\$\$) card is invalid.

## SECTION 18

### FILE/PUNCHER

The FILE/PUNCHER program, a normal-state utility program, reproduces a copy of a disk file by punching it to cards in a hexadecimal format acceptable as input to the FILE/LOADER program. The dollar card and the asterisk card used by the FILE/LOADER program are generated by the FILE/PUNCHER program.

### OPERATING INSTRUCTIONS

The file-identifier to be punched is supplied to the program from the ODT with an accept (AX) input message as follows:

```
EXECUTE FILE/PUNCHER;
```

```
FILE/PUNCHER = <job #> ENTER FILE IDENTIFIER
```

```
FILE/PUNCHER = <job #> ACCEPT
```

```
<job #> AX <file-identifier>
```

After punching the named disk file, the FILE/PUNCHER program repeats the prompt sequence and waits for another file-identifier to be entered. A blank accept response causes the program to go to EOJ.

### PROGRAM OUTPUT

The following card format is produced by the FILE/PUNCHER program. Five cards are required to contain the data for a disk segment.

Column	Description
1-72	Data in hexadecimal format (36 bytes)
73-80	Card sequence number

The following is an example of a punched code file:

Card #	Card Contents
1	?DATA CARDS .CARDS <date and time punch file created>
2	\$<file-identifier>
3	* 8 000009 (DO NOT MODIFY THIS CARD)
4-n	data in hexadecimal format
END	END CARDS .CARDS <date and time punch file created>

### ERROR MESSAGE

<file-identifier> NOT ON DISK

The MCPH cannot find the indicated file on disk. The punch request is ignored.



## SECTION 19

### FOREIGN/TAPECOPY

The FOREIGN/TAPECOPY program is designed to copy or list any 7-or 9-track magnetic tape. The program can copy the input tape to disk, line printer, or tape. Disk and line printer output files are in the standard B 1000 format and tape files are an exact copy of the input file. Data on the tape that is bounded by the specified delimiters is copied to a separate file. The operator is allowed to specify the tape format by using mnemonics to designate tape marks, header information, data, and labels. Input specifications can be entered from an ODT, remote terminal, or a card file.

### OPERATING INSTRUCTIONS

The FOREIGN/TAPECOPY program uses dynamic memory for both input and output tape buffers. This allows the program to use only as much memory as required. When executing the FOREIGN/TAPECOPY program, enough memory for an input buffer plus an output buffer must be specified. Refer to the MEMORY instruction in volume 1 of the B 1000 Systems System Software Operation Guide. When specifying output blocking, add the size of one record. The input sizes are determined by the size of the first block on the input tape. If the amount of memory specified is not large enough, the program terminates with an appropriate message.

Table 19-1 lists the program switch settings and their meanings.

**Table 19-1. FOREIGN/TAPECOPY Program Switches**

Switch	Value	Meaning
0	0	Input from ODT
	1	Input from cards
7	0	No action
	1-15	Message displayed for each file copied
8	0	No action
	1-15	Program dump at abnormal terminate

Executing the FOREIGN/TAPECOPY program with switch 0=1 directs the program to a card file (CARDS) which must contain the input specifications. See Input Specifications in this section. A maximum of 20 input cards is allowed.

Executing the FOREIGN/TAPECOPY program with switch 0=0 results in the display of the following message:

ENTER INPUT SPECIFICATIONS OR "HELP"

B 1000 Systems SSOG, Volume 2  
FOREIGN/TAPECOPY

The operator should enter a string of information representing the input specifications or the HELP command. A maximum of 1600 characters is allowed. Entering the HELP command causes the following messages to be displayed. The operator should respond with the appropriate information. Refer to Input Specifications in this section for details concerning input parameters. A blank or null response terminates the program at this point.

INPUT TAPE MNEMONIC? <MTx:>

OUTPUT DEVICE? <DISK, PRINTER, TAPE>

DEFAULT SPECIFICATIONS? <YES, NO>

If the operator wishes to override the default values, the following input messages are displayed.

ENTER TAPE FORMAT? V <D, EOF, H, L, TM, V>

INPUT TRANSLATION? <ASCII8, ASCII7, EBCDIC, BCL, file-name>

VARIABLE LENGTH RECORDS? <YES, NO>

OUTPUT BLOCKING?

BLOCKING FOR VARIABLE RECORDS: <MAX BLOCK LENGTH >

INPUT-OUTPUT BLOCKING: <characters per record> <records per block>

ENTER OPTIONS?

<PURGE <format list>,  
SKIP(number of files),  
INC(number of files),  
@xx@,  
FIRST, COPY,  
COPYALL>

NOTE

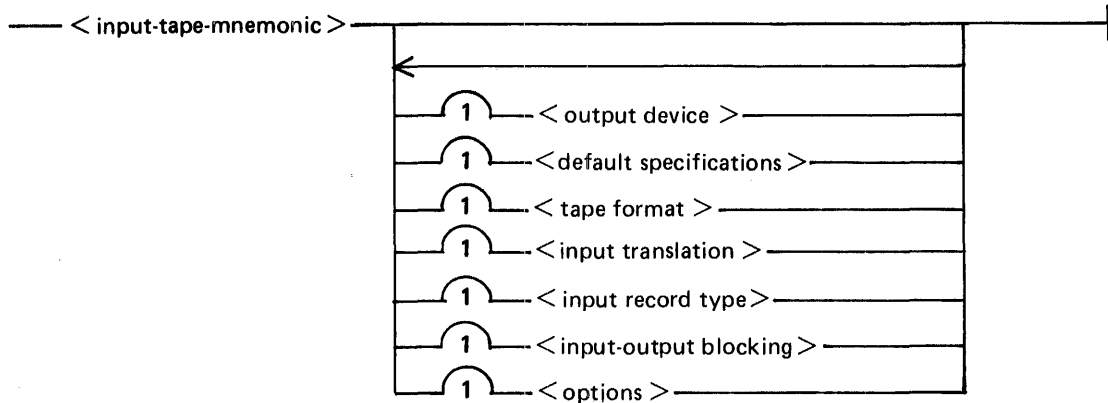
A blank or null response, in most cases, is equivalent to a no answer or no action answer.

If the output device is tape, only the relevant specifications are displayed.

## INPUT SPECIFICATIONS

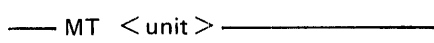
Syntax:

<input specifications>



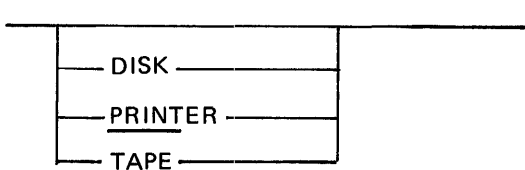
The input tape mnemonic is the only required specification. The remaining specifications are optional but, if entered, must appear in order and be separated by a blank. Only columns 1 through 72 of a card file are scanned for specifications. Comments or sequence numbers can be entered in columns 73 through 80.

<input tape mnemonic>



The input device must be a system-defined unit mnemonic for any 7-or 9-track tape, (for example, MTA). If the input device name is entered incorrectly, the operator can re-enter the appropriate device name. If the input device name is a valid device name but not present, the program hangs and waits for available hardware. The programmatic check of that device is not made until the copying actually begins. There is no default value for the input device name.

<output device>



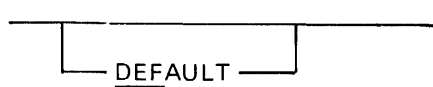
The output can be directed to disk or line printer using the words DISK, PRINTER, or PRINT . The default output device is DISK. All output files have a default multifile-id of OUTPUT. The filenames for disk files begin with FILE0001 and are numbered sequentially.

B 1000 Systems SSOG, Volume 2  
FOREIGN/TAPECOPY

Each file written to disk becomes a separate file. For printers, one backup print file is created with BOF and EOF messages separating the individual files. The print files contain a header consisting of blocking factor, filename, and input tape mnemonic for each file copied. The filename for a print file is FILES.

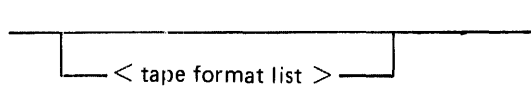
The output can also be directed to tape to create an exact copy of the input tape. Tape output is only allowed with the COPY or COPYALL options.

<default specification>

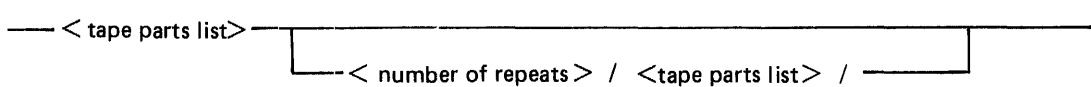


The word DEF or DEFAULT is used to specify the default specification. This enables the operator to copy an EBCDIC tape with fixed-length records to the output device. The output records are unblocked with a new file being created for data that exists between single tapemarks. The FOREIGN/TAPECOPY program stops copying when it encounters a double tapemark. If the <default specification> is used, no other input specifications are required. If TAPE is specified as the output device along with DEFAULT, the COPY option is assumed with default blocking. PRINT and DISK output devices assume no options.

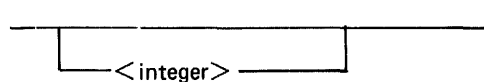
<tape format>



<tape format list>



<number of repeats>



The FOREIGN/TAPECOPY program assumes that all files are separated by single tapemarks and the end-of-file marker is a double tapemark. Since all tapes do not follow this form, the operator can specify the format of the tape to be copied. The following mnemonics are used to explain the tape format:

Mnemonic	Meaning
D	Data portion of the tape
EOF	End-of-file information
H	Header information
L	Label
TM	Tape mark
V	Volume label

B 1000 Systems SSOG, Volume 2  
FOREIGN/TAPECOPY

Any mnemonic can be optionally followed by <number of blocks> to specify the number of blocks to be included in that portion of the format. A zero block count is not valid. The FOREIGN/TAPECOPY program must rely on tapemarks to delimit information. The alternative to this is to specify the number of blocks for each portion of the file (specified by a mnemonic) between tapemarks. This causes the number of blocks specified to be placed in a separate file.

The FOREIGN/TAPECOPY program stops processing when it detects a double tapemark or a single tapemark immediately following an end-of-tape mark. If a tape does not conform to this standard, it is up to the operator to specify the physical limit of the input tape. This can be accomplished by specifying <number of repeats>. If <number of repeats> is specified, double tapemarks will not terminate the copy. If <number of repeats> is too large, the integrity of the data copied is unreliable; it is possible that the tape could run past the double tapemark and off the reel. It is suggested that the operator try his format with the FIRST option to a line printer to check for the correct operation. For example, a tape with one file would have a tape format as follows.

V H H TM D TM EOF EOF TM TM

Such a format results in the following files.

```
FILE0001 (V H H)
FILE0002 (D)
FILE0003 (EOF EOF)
FILE0004 (H H)
FILE0005 (D)
.
.
.
FILEEnnnn (EOF EOF)
```

If the <number of repeats> was specified for the preceding format, the result would be:

V 5/H H TM D TM EOF EOF TM

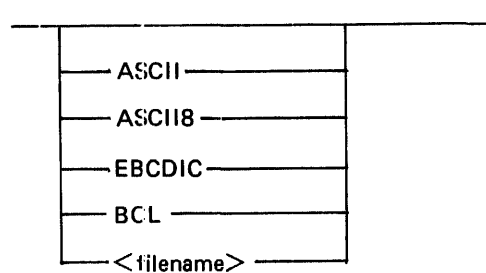
```
FILE0001 (V H H)
FILE0002 (D)
FILE0003 (EOF EOF)
FILE0004 (H H)
FILE0005 (D)
FILE0006 (EOF EOF)
FILE0007 (H H)
FILE0008 (D)
.
.
.
FILE00014 (D)
```

B 1000 Systems SSOG, Volume 2  
FOREIGN/TAPECOPY

A maximum of 40 different tape format tokens can be used with any one run.

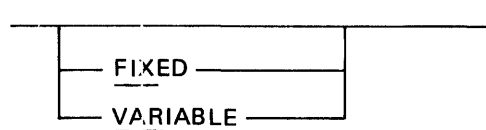
FILE00015 (EOF EOF)

<input translation>



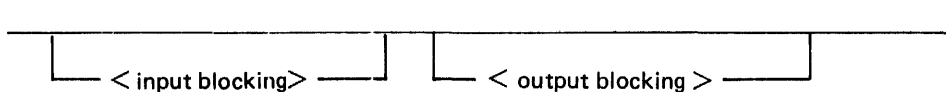
The input tape can be translated from BCL, ASCII-7, or ASCII-8 format to EBCDIC format. The default is EBCDIC, but the operator can specify translation from ASCII-7 and ASCII-8 for 9-track tapes, BCL for 7-track tape, or provide a unique translate table. The user-supplied translate table must appear as a system filename. All translate tables are to be created with CREATE/TABLE. The FOREIGN/TAPECOPY program looks for the translate table named TRANSLATE/EBCDIC, TRANSLATE/ASCII7, TRANSLATE/ASCII8, TRANSLATE/BCL, or <translate filename>. The second record of that file is used for translation.

<input record type>

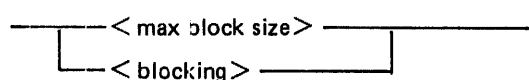


An input file can have fixed or variable length records. This can be specified by FIXED, FIX, VARIABLE, or VAR. The default is fixed length records for input. Output files are type FIXED only. The input record type is used for all files on the tape.

<input-output blocking>



<input blocking>

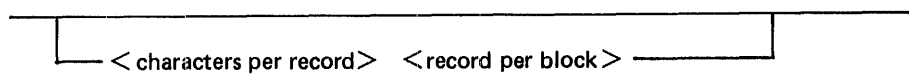


<output blocking>

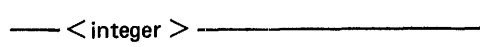


B 1000 Systems SSOG, Volume 2  
FOREIGN/TAPECOPY

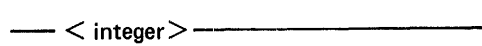
<blocking>



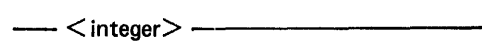
<max block size>



<characters per record>



<records per block>



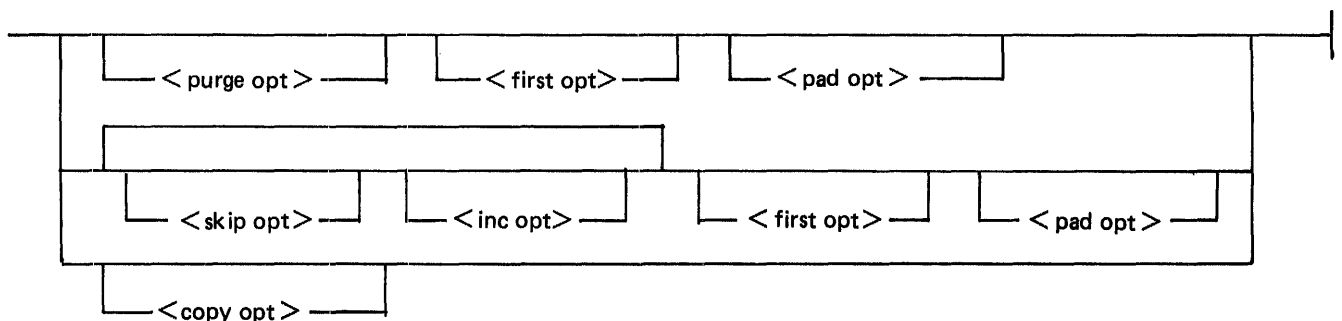
The blocking factors are used to determine block and record sizes on the input tape and output file. If no blocking specifications are provided, the default is unblocked, fixed length records. The first block of each file determines the length of the records. Blocking can be specified in characters per record and records per block. For instance, 80 2 is interpreted as 80 characters per record and 2 records per block. If a blocking factor is specified, it is applied to all files copied from that tape. If only one blocking specification is entered, it is applied to both the input and output files.

A maximum block size is required for variable length input blocks. The blocks, which can contain several variable or fixed-length records, is written to the output device as fixed-length records with the record size as the maximum block size specified. The records are to be padded with either zeros or the pad character specified as an option.

If the blocksize is too large or too small for a block of data, the appropriate error message is printed. Refer to Error Messages in this section.

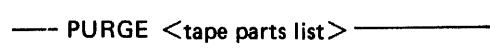
When using the COPY or COPYALL option, a maximum block size is used to determine the input buffer size only. If a block is shorter, it is written using its actual length.

<options>

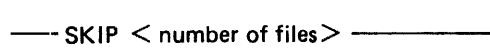


B 1000 Systems SSOG, Volume 2  
FOREIGN/TAPECOPY

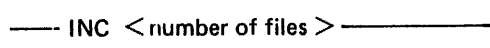
<purge opt>



<skip opt>



<inc opt>



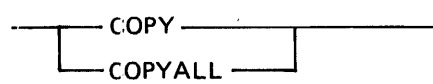
<first opt>



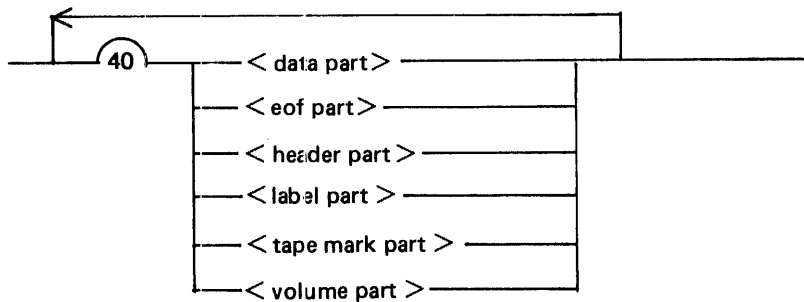
<pad opt>



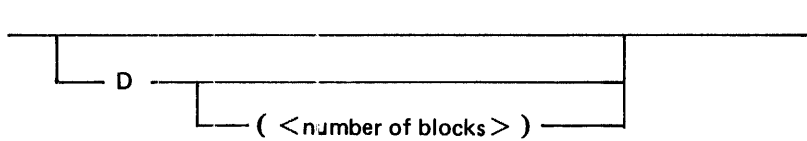
<copy opt>



<tape parts list>



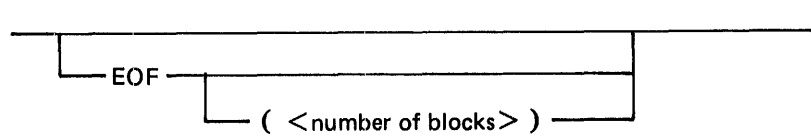
<data part>



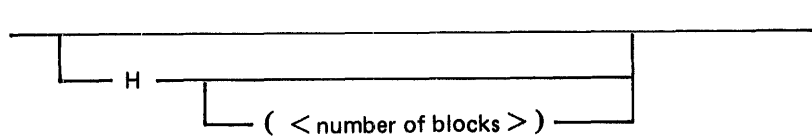


B 1000 Systems SSOG, Volume 2  
FOREIGN/TAPECOPY

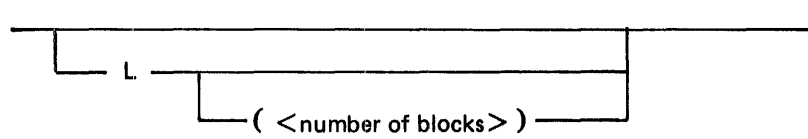
<eof part>



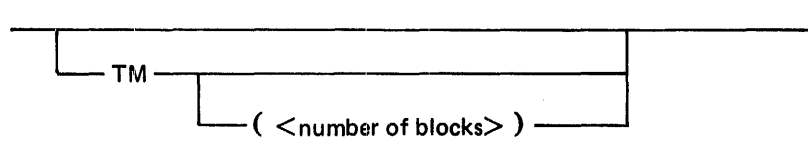
<header part>



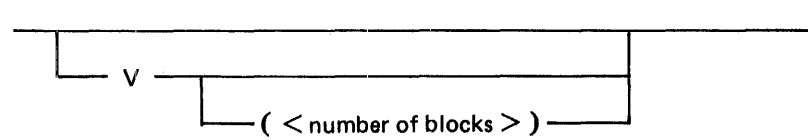
<label part>



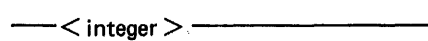
<tape mark part>



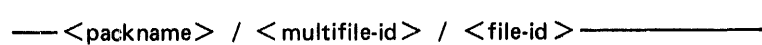
<volume part>



<number of files>



<filename>



B 1000 Systems SSOG, Volume 2  
FOREIGN/TAPECOPY

<packname>

— <name> —————|

<multifile-id>

— <name> —————|  
| <usercode> —————|

<file-id>

— <name> —————|

<usercode>

— ( <name> ) —————|

<number of blocks>

— <integer> —————|

<name>

— <letter> <name> —————|

<options>

**PURGE** <format list>

The PURGE option can be used with an operator-specified tape format to indicate which portions of the tape are not to be copied. The operator can specify that a copy of any portion(s) of the tape is not to be performed, excluding tapemarks (TM). Tapemarks are not copied but are used only as a means of delimiting files. The PURGE option cannot be used in conjunction with the SKIP or INC options.

Example:

```
PURGE V
PURGE H V EOF
```

**SKIP** <number of files>

The SKIP option is used to bypass the specified number of files before copying a file to the output device. A file is defined to be that data existing between single tapemarks. The SKIP option must be used with the INC option to select the required portions of the input tape. This option cannot be used with a tape format. A maximum of 100 SKIP and INC options can be specified in a given run.

**INC** <number of files>

The INC option specifies the number of files to be copied to the output device. A file is defined to be that data existing between single tapemarks. The INC option is used in conjunction with the SKIP option to select the specified portions of the input tape. This option cannot be used with a tape format. Only files with an INC option are copied if either the SKIP or INC option is used. In other words, the FOREIGN/TAPECOPY program stops processing at the end of the last SKIP or INC option.

**FIRST**

The FIRST option allows the operator to print only the first block of each file of the input tape. This option is used to check the contents of the input tape against the contents of the intended copy and can only be used when the output device is a line printer.

**PAD**

If padding is necessary, a program called TAPEX automatically pads with @00@. The PAD option specifies characters other than @00@ for padding. The operator-selected character must be specified in the form @<EBCDIC character><EBCDIC character>@, where <EBCDIC character> is a 4-bit hexadecimal number. The operator-selected character can also be specified for use in filling variable length blocks for fixed output. For example, if a blank is desired the appropriate input would be @40@.

**COPY**

The COPY option provides an exact duplicate of the input tape. Translation, blocking, tape formats, or other options are not allowed. Double tapemarks terminate the copy. The input and output tape densities specified must match.

**COPYALL**

The COPYALL option provides an exact duplicate of the input tape. Translation, blocking, tape formats, or other options are not allowed, and the physical end of either the input or output tape terminates the copy. Input and output tape densities must match.

## **ERROR MESSAGES**

**UNKNOWN ERROR, PLEASE INPUT INFORMATION AGAIN.**

An error occurred for which no error message is provided.

**INPUT TAPE MNEMONIC REQUIRED.**

**INVALID INPUT TAPE MNEMONIC.**

Input mnemonic must be a system defined unit mnemonic for a 7-or 9-track tape.

**INVALID OUTPUT DEVICE MNEMONIC.**

Output device must be DISK, PRINTER, or TAPE.

**PLEASE ANSWER "YES" OR "NO".**

A YES or NO answer was expected.

B 1000 Systems SSOG, Volume 2  
FOREIGN/TAPECOPY

INVALID TRANSLATE SPECIFIED.

A standard translation type or operator translation file must be specified.

NUMERIC VALUE REQUIRED FOR BLOCKING.

TOO MANY BLOCKING PARAMETERS SPECIFIED

A maximum of four blocking parameters, two for input and two for output, can be specified.

INVALID OPTION SPECIFIED.

MISMATCH OF PARENTHESES.

INVALID TAPE FORMAT MNEMONIC.

Valid tape format mnemonics are: D, EOF, H, L, TM, and V.

TAPE FORMAT EXCEEDS 60 MNEMONICS.

NUMERIC VALUE REQUIRED FOR "SKIP" AND "INC" OPTIONS.

INVALID PADDING CHARACTER SPECIFIED.

The pad character used must represent a 4-bit hexadecimal digit.

DOUBLE TAPEMARKS ENCOUNTERED.

Copying ends if a double tapemark is found.

NO OTHER INPUT SPECS ALLOWED WITH "DEFAULT" OPTION.

INPUT STRING EXCEEDED MAXIMUM LENGTH.

Card input must not exceed 20 cards.

INPUT TAPE NOT READY--PLEASE READY UNIT.

PARITY ERROR ON INPUT TAPE, PLEASE ENTER "YES" TO CONTINUE.

PROGRAM TERMINATING.

Program ending after fatal error.

TIMEOUT ERROR AFTER 5 TRIES.

BLOCK SIZE TOO SMALL FOR ACTUAL DATA. (ONE TIME WARNING)

B 1000 Systems SSOG, Volume 2  
FOREIGN/TAPECOPY

<MAXIMUM BLOCK-SIZE> REQUIRED FOR VARIABLE LENGTH RECORDS.

INPUT TAPE BLOCK LENGTH EXCEEDS 65528 BITS.

ENTER "YES" TO TRUNCATE BLOCK AND CONTINUE.

DYNAMIC MEMORY TOO SMALL. RE-EXECUTE PROGRAM WITH MORE DYNAMIC  
MEMORY.

BLOCK SIZE TOO LARGE FOR ACTUAL DATA. (ONE TIME WARNING)

A "TAPE FORMAT" MAY NOT BE SPECIFIED WITH "SKIP" OR "INC"  
OPTIONS. "SKIP" OR "INC" OPTION IGNORED.

A "TAPE FORMAT" MUST BE SPECIFIED WITH A "PURGE" OPTION.  
"PURGE" OPTION IGNORED.

INVALID INPUT/OUTPUT BLOCKING SPECIFIED.

Input/output blocking factors must be non-zero.

PROGRAM EXCEEDED MAX NUMBER OF ELEMENTS IN TAPE FORMAT.

Increase number of elements on tape format.

TAPE MARK ENCOUNTERED BEFORE NUMBER OF BLOCKS SPECIFIED WERE  
COPIED.

END OF TAPE ENCOUNTERED. MAY LOSE ENDING LABELS.

Program will terminate at end of tape.

FIRST BLOCK OF FILE HAS A LENGTH OF ZERO.

Cannot open a file with a zero length record.

USER SPECIFIED TAPE MARK NOT FOUND ON INPUT TAPE.

User specified input tape format does not match input tape format.

SIZE OF BLOCK EXCEEDS MAXIMUM LIMIT OF 65528 BITS.

"FIRST" OPTION ALLOWED WITH PRINTER OUTPUT ONLY.

END OF INPUT TAPE ENCOUNTERED.

Copying has stopped at end of input tape.

B 1000 Systems SSOG, Volume 2  
FOREIGN/TAPECOPY

END OF OUTPUT TAPE ENCOUNTERED.

Copying has stopped at the end of the output tape.

NO DATA FROM READ HEAD.

MUST SPECIFY "COPY" OPTIONS WITH OUTPUT DEVICE "TAPE".

OUTPUT DEVICE "TAPE" MUST BE SPECIFIED WITH "COPY" OPTIONS.

PARITY ERROR ON OUTPUT TAPE, PLEASE ENTER "YES" TO CONTINUE.

INPUT AND OUTPUT TAPE DENSITIES MUST MATCH FOR "COPY".

IRRECOVERABLE IO ERROR.

PARITY ERROR ON 7-TRACK TAPE. CHANGING PARITY AND RETRYING.

Parity error encountered. Parity will be changed and processing continued.

EXACT COPY NOT ALLOWED ON 7-TRACK TAPES.

## INTERNAL FILES

The following is a list of the name and function of each file used by the FOREIGN/TAPECOPY program.

<b>Internal File Name</b>	<b>Description</b>
CARDS	A card file containing input specifications
OUTPUTFILE	Modified to appropriate output device
INPUTFILE	A tape file containing input data
TRANSFILE	Translate file

## SECTION 20

### INITIALIZE/ANALYZER

The INITIALIZE/ANALYZER program is a normal-state utility program that creates a line printer listing of the addresses of disk sectors that were relocated or removed during initialization or reconfiguration. The disk pack or disk cartridge that is to be analyzed must have been initialized by a Mark 8.0 (or later) version of the SYSTEM/DISK.INIT or PACK/INIT program. The INITIALIZE/ANALYZER program can analyze a disk while other programs are accessing the same disk.

### OPERATING INSTRUCTIONS

The INITIALIZE/ANALYZER utility program is executed from the console keyboard. Several disks can be analyzed during a single execution of the program, one after another. One printer backup file is created that describes each disk analyzed.

Upon execution, the INITIALIZE/ANALYZER program issues the following prompt message on the operator display terminal (ODT):

ENTER INPUT DRIVE

The operator responds by entering an accept (AX) input message with the unit mnemonic that names the drive on which the disk to be analyzed is located. Unit mnemonics accepted by the INITIALIZE/ANALYZER program are DCx and DPx, where x is replaced by a letter that references an existing disk drive. For example, the letter A references drive 0.

The input drive prompt is repeated after the specified disk is analyzed. A null response to this prompt (a blank AX input message) causes the program to go to end of job (EOJ).

### PROGRAM OUTPUT

The printer listing generated describes each disk, naming each relocated sector and each removed sector, summarizes the number of relocated and removed sectors, and then lists label information that gives the identifier, serial number, type, and initialization date of the disk.

### ERROR MESSAGES

INVALID RESPONSE -TRY AGAIN

An invalid unit mnemonic was entered, or the disk named could not be analyzed. Re-enter the drive specification.

BAD LABEL OR INITIALIZED VERSION -<drive>

Either the disk has a bad label, or it was initialized with an initializer program version other than the Mark 8.0 release of the SYSTEM/DISK.INIT or PACK/INIT program. The disk must be re-initialized before the INITIALIZE/ANALYZER program can analyze the disk.

## SECTION 21

### LOGCONVERT

The LOGCONVERT program is a normal-state utility program that extracts information from the LOG/<integer> file and creates a new file, NEW.LOG/<integer>, containing log data in a format readable by COBOL and RPGII programs.

### OPERATING INSTRUCTIONS

The LOGCONVERT program processes a transferred SYSTEM/LOG file created automatically by the operating system (MCPII) or explicitly by the operator with the LG or TL input messages. The identifier of the log file to be processed is specified by entering the <integer> with a free-form format accept (AX) response, for which leading zeroes are optional. The currently-active log file, SYSTEM/LOG, cannot be processed with the LOGCONVERT program.

An interrogation AX input message of STATUS (or ST) can be entered at any time after processing begins. The LOGCONVERT program responds by displaying the current input record count, thus enabling the system operator to monitor program progress.

If the LOGCONVERT program finds an incorrectly-formatted record in the file it is reading, it displays a warning message (for the first five times only) and ignores the bad record. This is normally evidence of a corrupt file.

### OUTPUT FILE FORMAT

The output file NEW.LOG/<integer> contains 180-byte records, one record per block.

#### Record Types

Ten distinct record types, identified by the first letter of each record, can be found within the output file:

##### CLEAR/START record

One for each CLEAR/START operation performed during the scope of the log.

##### Schedule record

One for each job or task that is scheduled for execution.

##### BOJ record

One for each job or task that reaches beginning of job (BOJ).

##### Open record

One for each instance of a file being opened by a program.

##### Close record

One each time a file is closed by a program.

##### Data Management System Statistics record

One for each structure used within a particular DMSII data base when the data base is closed.

##### Data Management System Open/Close record

One for each instance of a DMSII program opening or closing a data base.



## B 1000 Systems SSOG, Volume 2 LOGCONVERT

### Peripheral Assignment/Release record

One each time an SDL program using special Initializer I/O communications, such as the DISK-MAP/UTILITY or SYSTEM/DISK.INIT programs, assigns a peripheral unit for its own exclusive use, or releases the peripheral back to the system.

### EOJ record

One for each program (1) reaching end of job (EOJ), (2) being terminated due to a CLEAR/START operation, or (3) being removed from the schedules (RS input message).

### Log Transfer record

The last record in the file, identifying the next log file by a sequential integer maintained by the operating system (MCPPII).

## General Information

Records in the log file are generated as the information that they contain becomes available, which means that those records pertaining to a given job can be scattered widely throughout the log file. To simplify operator processing, a sort operation is invoked to restructure the NEW.LOG/<integer> file after it is filled, so that all records pertaining to a given job appear together with the original order maintained. This sort can be bypassed by setting program switch 9 to 1 when executing the LOGCONVERT program.

The input and output files have the internal identifiers LOGFILE and NEWLOGFILE, respectively, and are each declared with two buffers. The NEWLOGFILE file is automatically directed to the second drive by the MCPPII when multiple system disk drives have been assigned with the SD input message. Installations that have a single system disk drive can minimize disk arm movement, and consequently achieve a substantial performance improvement, by file-equating the NEWLOGFILE file to a user disk through file-equation of the PACK.ID (PID) file attribute.

## Record Format Tables

Generic descriptions of the record formats found in the output file are given in tables 21-1 through 21-15. Specific COBOL, COBOL74, and RPGII data layouts follow these tables. Data type N represents packed numeric fields, four-bit digits with high-order sign, whereas data type A represents eight-bit EBCDIC alphanumeric fields. To minimize the size of these tables, all 30-character alphanumeric fields represent the standard <pack-id>/<family-id>/<file-id> triples of ten characters each. Likewise, all 16-digit numeric date/time fields are composed of three distinct subfields: a four-digit field containing the last two digits of the year, a four-digit field containing the Julian day, and an eight-digit field containing the time in tenths of seconds past midnight.

Table 21-1 shows the record format of the LOGCONVERT CLEAR/START record.

B 1000 Systems SSOG, Volume 2  
LOGCONVERT

**Table 21-1. LOGCONVERT CLEAR/START Record Format**

Field Name	Field Length	Data Type	Value and Function
Record Type	1	A	A = CLEAR/START record
Job Accounting Number	8	N	Of the next job to be scheduled
Job Number	6	N	Of the next job to be scheduled
Time	16	N	YYYYDDDDTTTTTTTT
MCPII Name	30	A	
Interpreter Name	30	A	If the MCP's interpreter
MCPII Version	10	A	MARK.LEVEL.PATCH (for example, VIII.0.12)
S-Memory Size	10	N	In bits
GISMO Name	30	A	
Micro MCPII Name	30	A	
New Log Flag	2	N	Indicates first CLEAR/START in log

Table 21-2 shows the record format for the LOGCONVERT SCHEDULE record.

**Table 21-2. LOGCONVERT SCHEDULE Record Format**

Field Name	Field Length	Data Type	Value and Function
Record Type	1	A	B = Schedule record
Job Accounting Number	8	N	
Job Number	6	N	
Time	16	N	YYYYDDDDTTTTTTTT
Program Name	30	A	
Object Program Name	30	A	If execute type is a compilation
Schedule Priority	4	N	0-15
Execute Type	2	N	Refer to table 21-15
Static Memory	10	N	In bits
Dynamic Memory	10	N	
Total Memory	10	N	Run Structure size
Charge Number	8	N	0-9999999
Filler	3	A	
Number of Files	4	N	
Interpreter Name	30	A	
Date of Compilation	16	N	
Parent Job Number	6	N	

B 1000 Systems SSOG, Volume 2  
LOGCONVERT

Table 21-3 shows the record format of the LOGCONVERT BOJ record.

**Table 21-3. LOGCONVERT BOJ Record Format**

Field Name	Field Length	Data Type	Value and Function
Record Type	1	A	C = Beginning of Job record
Job Accounting Number	8	N	
Job Number	6	N	
Time	16	N	YYYYDDDDTTTTTTTT
Program Name	30	A	
Object Program Name	30	A	If this is a compilation
Execute Type	2	N	Refer to table 21-15
Processor Priority	4	N	0-15
Memory Priority	4	N	0-15
Charge Number	8	N	0-9999999
Virtual Disk	6	N	In segments
Interpreter Name	30	A	

Table 21-4 shows the record format of the LOGCONVERT FILE OPEN record.

**Table 21-4. LOGCONVERT File Open Record Format**

Field Name	Field Length	Data Type	Value and Function
Record Type	1	A	D = Open record
Job Accounting Number	8	N	
Job Number	6	N	
Time	16	N	YYYYDDDDTTTTTTTT
File Number	4	N	0 if first file declared in program
Internal Name	10	A	
External Name	30	A	
Hardware Type	4	N	Refer to table 21-12
Variable Record File	2	N	As with all flags, 1 = true
Pseudo Reader File	2	N	
DMS	2	N	DMSII file
Emulator Tape	2	N	To be processed by emulator tape communicates in SDL

B 1000 Systems SSOG, Volume 2  
LOGCONVERT

**Table 21-4. LOGCONVERT File Open Record Format (Cont)**

Field Name	Field Length	Data Type	Value and Function
Record Size	6	N	In bits
Blocking Factor	6	N	
Maximum Block Size	6	N	For variable record files, RSZ * R.B
Unit Name	6	A	Explicit user peripheral designation
Number of Stations	4	N	
Access Mode	4	N	Refer to table 21-3
Number of Areas	4	N	Declared for disk file
Blocks per Area	8	N	
Autoprint Flag	2	N	
Filler	11	A	
File Type	4	N	Refer to table 21-13
Serial Number	6	A	
Queue Family Size	4	N	
Queue Max Messages	4	N	
Securitytype	2	N	0 = PUBLIC, 1 = PRIVATE
Securityuse	2	N	0 = INPUT/OUTPUT, 1 = INPUT, 2 = OUTPUT
Number of Buffers	4	N	
Memory Space Required	8	N	Bits: FIB + buffers + I/O Descriptor
Open Input Flag	2	N	
Open Output Flag	2	N	
New File Flag	2	N	
Open with Punch Flag	2	N	
Open with Print	2	N	
Open No Rewind Flag	2	N	
Open Reverse Flag	2	N	
Open Lock Flag	2	N	
Open Lockout Flag	2	N	

B 1000 Systems SSOG, Volume 2  
LOGCONVERT

Table 21-5 shows the record format of the LOGCONVERT FILE CLOSE record.

**Table 21-5. LOGCONVERT File Close Record Format**

Field Name	Field Length	Data Type	Value and Function
Record Type	1	A	E = Close record
Job Accounting Number	8	N	
Job Number	6	N	
Time	16	N	YYYYDDDDTTTTTTTT
File Number	4	N	0 is first file declared in program
Internal Name	10	A	
External Name	30	A	
Hardware Type	4	N	Refer to table 21-12
Variable Record File	2	N	As with all flags, 1 = true
Pseudo Reader File	2	N	
DMS	2	N	DMSII file
Emulator Tape	2	N	Was processed by emulator tape communicates in SDL
Record Size	6	N	In bits
Blocking Factor	6	N	
Maximum Block Size	6	N	For variable record files, RSZ * R.B
Unit Name	6	N	Explicit user peripheral designation
Number of Stations	4	N	
Access Mode	4	N	Refer to table 21-14
Number of Areas	4	N	Declared for disk file
Blocks per Area	8	N	
Autoprint Flag	2	N	
Record Count	8	N	Logical I/O's
Block Count	8	N	Physical I/O's
Errors	6	N	
File Type	4	N	Refer to table 21-13
Serial Number	6	A	
Queue Family Size	4	N	
Queue Max Messages	4	N	
Protection	2	N	
Protection I/O	2	N	
Number of Buffers	4	N	

B 1000 Systems SSOG, Volume 2  
LOGCONVERT

**Table 21-5. LOGCONVERT File Close Record Format (Cont)**

Field Name	Field Length	Data Type	Value and Function
Memory Space Required	8	N	Bits: FIB + buffers + I/O descriptor
Close Reel Flag	2	N	
Close Release Flag	2	N	
Close with Purge Flag	2	N	
Close Remove Flag	2	N	
Close with Crunch Flag	2	N	
Close No Rewind Flag	2	N	
Close Insecure	2	N	
Close with Lock Flag	2	N	
Close Conditional	2	N	
Close Rollout Flag	2	N	
DMS Audit Switch Flag	2	N	
Filler	1	A	
Close Unavailable	2	N	
Filler	7	A	
MCPII Close due to EOJ	2	N	
Close Backup Exceeded	2	N	
Filler	14	A	

Table 21-6 shows the record format of the LOGCONVERT DMS Statistics record.

**Table 21-6. LOGCONVERT DMS Statistics Record Format**

Field Name	Field Length	Data Type	Value and Function
Record Type	1	A	F = DMS Statistics record
Job Accounting Number	8	N	
Job Number	6	N	
Time	16	N	YYYYDDDDTTTTTTTT
Database Name	10	A	
Structure Number	4	N	
Invoke Number	4	N	
Structure Close Flag	2	N	
Structure Name	17	A	
File Name	30	A	
Hardware Type	4	N	

B 1000 Systems SSOG, Volume 2  
LOGCONVERT

**Table 21-6. LOGCONVERT DMS Statistics Record Format (Cont)**

Field Name	Field Length	Data Type	Value and Function
Unit Name	6	A	
Number of Random Finds	10	N	
Number of Sequential Finds	10	N	
Number of Inserts	10	N	
Number of Updates	10	N	
Number of Deletes	10	N	
Number of Sys. Changes	10	N	
Number of Exceptions	10	N	
Number of Logical Reads	10	N	
Number of Physical Reads	10	N	
Number of Logical Writes	10	N	
Number of Physical Writes	10	N	

Table 21-7 shows the record format of the LOGCONVERT DMSII open/close record.

**Table 21-7. LOGCONVERT DMS Open/Close Record Format**

Field Name	Field Length	Data Type	Value and Function
Record Type	1	A	G=DMS open or close record
Job Accounting Number	8	N	
Job Number	6	N	
Time	16	N	YYYYDDDDTTTTTTTT
Physical Database Name	10	A	
Logical Database Name	17	A	
Close Flag	2	N	0 = open, 1 = close
Update Flag	2	N	
Audit Flag	2	N	
Hardware Type	4	N	Refer to table 21-12
Unit Name	6	A	
Unit Name	6	A	
Pack Name	10	A	
Number of Update Open	10	N	
Number of Non-update Op	10	N	
SMCPH time	10	N	

B 1000 Systems SSOG, Volume 2  
LOGCONVERT

Table 21-8 shows the record format of the LOGCONVERT Peripheral Assignment/Release record.

**Table 21-8. LOGCONVERT Peripheral Assignment/Release**

Field Name	Field Length	Data Type	Value and Function
Record Type	1	A	H = Peripheral assign/release record
Job Accounting Number	8	N	
Job Number	6	N	
Time	16	N	YYYYDDDDTTTTTTTT
Hardware Type	4	N	Refer to table 21-12
Unit Name	6	A	The unit that was assigned/released
Action	2	N	0 = assigned, 1 = released

Table 21-9 shows the format of the LOGCONVERT EOJ record.

**Table 21-9. LOGCONVERT EOJ Record Format**

Field Name	Field Length	Data Type	Value and Function
Record Type	1	A	I = End of Job record
Job Accounting Number	8	N	
Job Number	6	N	
Time	16	N	YYYYDDDDTTTTTTTT
Program Name	30	A	
Object Program Name	30	A	If this was a compilation
Execute Type	2	N	Refer to table 21-15
End of Job Reason	2	N	0 = normal EOJ 1 = DS-ed 2 = program error (DS or DP condition) 3 = execution aborted by CLEAR/START 4 = RS-ed 5 = death in family 6 = In schedule at CLEAR/START
Processor Time	8	N	In tenths of seconds
Code Overlays	8	N	
Data Overlays	8	N	
Successor Program	30	A	Next program to run in a job stream
EOJ Diagnostic Message	60	A	Specific "DS or DP" message



B 1000 Systems SSOG, Volume 2  
LOGCONVERT

Table 21-10 shows the record format for the LOGCONVERT Comment record.

**Table 21-10. LOGCONVERT Comment Record Format**

Field Name	Field Length	Data Type	Value of Function
Record Type	1	A	J = Comment Record
Job Accounting Number	8	N	
Job Number	6	N	
Time	16	N	YYYYDDDDTTTTTTTT
Comment Length	4	N	Length of Characters
Comment Text	162	A	Text

Table 21-11 shows the record format of the LOGCONVERT Log Transfer record.

**Table 21-11. LOGCONVERT Log Transfer Record Format**

Field Name	Field Length	Data Type	Value and Function
Record Type	1	A	K = Log Transfer record
Filler	7	A	Hexadecimal @FF@s
Time	16	N	YYYYDDDDTTTTTTTT
Next Logfile Name	30	A	

Table 21-12 lists the LOGCONVERT Hardware Type Definitions.

**Table 21-12. LOGCONVERT Hardware Type Definitions**

Value	Device
0	Durnmy file
1	80-column data recorder
2	80-column card punch
4	Diskette (floppy disk)
5	96-column reader/punch/printer
6	Paper tape reader
7	Paper tape reader
8	Printer, other than B 1247-4 printer control
9	MICR reader-sorter
10	MICR reader-sorter
11	Disk file (any head-per-track unit)
12	Head-per-track disk

B 1000 Systems SSOG, Volume 2  
LOGCONVERT

**Table 21-12. LOGCONVERT Hardware Type Definitions (Cont)**

Value	Device
13	Disk cartridge (Disk Cartridge Controller 2/3)
14	Disk cartridge (Disk Cartridge Controller 1)
15	Disk pack
16	Disk pack (any available)
17	Disk (of any kind)
18	B 9470 Head-per-track disk
19	96-column card reader
20	Paper tape punch
21	80-column card reader
22	Hard copy ODT
23	CRT ODT
24	Nine-track NRZ tape
25	Seven-track tape
26	Nine-track PE tape
27	Any tape unit
28	Any nine-track tape unit
29	Any tape
30	Cassette
31	Printer with B 1247-4 printer control
32	DSC (Disk Packs)
33	Printer Control 7
36	PE tape (Magnetic Tape Control 6)
61	Queue
63	Remote

Table 21-13 lists the LOGCONVERT file types.

**Table 21-13. LOGCONVERT File Types**

Value	Meaning
0	Absolute MCP II
1	Log
2	Directory (entry points to sub-directory)
3	Control deck
4	Backup print file
5	Backup punch file
6	Dump file
7	Interpreter
8	Code file
9	Data file
10	Undefined
11	Variable-length data file
12	Intrinsic
13	Undefined
14	Undefined
15	DMS audit file
16	Undefined
17	Relative
18	Index Sequential - Global
19	Index Sequential - Data
20	Index Sequential - Index

Table 21-14 lists the LOGCONVERT access modes.

**Table 21-14. LOGCONVERT Access Modes**

<b>Value</b>	<b>Meaning</b>
0	Serial
1	Random
2	I/O sequential
3	Input/stackers data recorder
4	Input/output data recorder
5	Emulator tape
6	Delayed random
7	I/O sequential with extend

Table 21-15 lists the LOGCONVERT executes types.

**Table 21-15. LOGCONVERT Execute Types**

<b>Value</b>	<b>Meaning</b>
1	Regular execution
2	Compile and go
3	Compile for syntax
4	Compile to library
5	Compile and save
6	Execute phase of compile and go

## **SAMPLE PROGRAM RECORD DECLARATIONS**

The following are sample program record declarations for COBOL, COBOL74, and RPGII programs.

### **COBOL Record Formats**

Tables 21-16 through 21-26 are COBOL and COBOL74 declarations which correspond to the record formats previously detailed in tables 21-1 through 21-15. They can be used by COBOL and COBOL74 programs which process the output of the LOGCONVERT program.

B 1000 Systems SSOG, Volume 2  
LOGCONVERT

Table 21-16. Clear/Start

01	CLEAR-START RECORD.	
02	CS-REC-TYPE	PC X.
02	CS-JOB-ACCOUNTING-NUMBER	PC S9(7) CMP.
02	CS-JOB-NUMBER	PC S9(5) CMP.
02	CS-YEAR	PC S9(3) CMP.
02	CS-JULIAN-DAY	PC S9(3) CMP.
02	CS-TIME	PC S9(7) CMP.
02	CS-MCP-NAME	PC X(30).
02	CS-INTERP-NAME	PC X(30).
02	CS-MCP-VERSION	PC X(10).
02	CS-MAIN-MEMORY-SIZE	PC S9(9) CMP.
02	CS-GISMO-NAME	PC X(30).
02	CS-MMCP-NAME	PC X(30).
02	CS-NEW-LOG-FLAG	PC S9 CMP.

Table 21-17. Schedule

01	SCHEDULE-RECORD.	
02	SR-REC-TYPE	PC X.
02	SR-JOB-ACCOUNTING-NUMBER	PC S9(7) CMP.
02	SR-JOB-NUMBER	PC S9(5) CMP.
02	SR-YEAR	PC S9(3) CMP.
02	SR-JULIAN-DAY	PC S9(3) CMP.
02	SR-TIME	PC S9(7) CMP.
02	SR-PROGRAM-NAME	PC X(30).
02	SR-OBJECT-PROGRAM-NAME	PC X(30).
02	SR-SCHEDULE-PRIORITY	PC S9(3) CMP.
02	SR-EXECUTE-TYPE	PC S9 CMP.
02	SR-STATIC-MEMORY	PC S9(9) CMP.
02	SR-DYNAMIC-MEMORY	PC S9(9) CMP.
02	SR-TOTAL-MEMORY	PC S9(9) CMP.
02	SR-CHARGE-NUMBER	PC S9(7) CMP.
02	FILLER	PC X(3).
02	SR-NUMBER-OF-FILES	PC S9(3) CMP.
02	SR-INTERPRETER-NAME	PC X(30).
02	SR-YEAR-COMPILED	PC S9(3) CMP.
02	SR-JULIAN-DAY-COMPILED	PC S9(3) CMP.
02	SR-TIME-COMPILED	PC S9(7) CMP.
02	SR-PARENT-JOB-NUMBER	PC S9(5) CMP.

B 1000 Systems SSOG, Volume 2  
LOGCONVERT

**Table 21-18. Beginning-of-Job**

01	BOJ-RECORD.	
02	BOJ-RECORD-TYPE	PC X.
02	BOJ-JOB-ACCOUNTING-NUMBER	PC S9(7) CMP.
02	BOJ-JOB-NUMBER	PC S9(5) CMP.
02	BOJ-YEAR	PC S9(3) CMP.
02	BOJ-JULIAN-DAY	PC S9(3) CMP.
02	BOJ-TIME	PC S9(7) CMP.
02	BOJ-PROGRAM-NAME	PC X(30).
02	BOJ-OBJECT-PROGRAM-NAME	PC X(30).
02	BOJ-EXECUTE-TYPE	PC S9 CMP.
02	BOJ-PROCESSOR-PRIORITY	PC S9(3) CMP.
02	BOJ-MEMORY-PRIORITY	PC S9(3) CMP.
02	BOJ-CHARGE-NUMBER	PC S9(7) CMP.
02	BOJ-VIRTUAL-DISK	PC S9(5) CMP.
02	BOJ-INTERPRETER-NAME	PC X(30).

**Table 21-19. File Open**

01	OPEN-RECORD.	
02	OR-REC-TYPE	PC X.
02	OR-JOB-ACCOUNTING-NUMBER	PC S9(7) CMP.
02	OR-JOB-NUMBER	PC S9(5) CMP.
02	OR-YEAR	PC S9(3) CMP.
02	OR-JULIAN-DAY	PC S9(3) CMP.
02	OR-TIME	PC S9(7) CMP.
02	OR-FILE-NUMBER	PC S9(3) CMP.
02	OR-INTERNAL-NAME	PC X(10).
02	OR-EXTERNAL-NAME	PC X(30).
02	OR-HARDWARE-TYPE	PC S9(3) CMP.
02	OR-VARIABLE-RECORDS	PC S9 CMP.
02	OR-PSEUDO-READER-FILE	PC S9 CMP.
02	OR-DMS-FILE	PC S9 CMP.
02	OR-EMULATOR-TAPE-FILE	PC S9 CMP.
02	OR-RECORD-SIZE	PC S9(5) CMP.
02	OR-BLOCKING-FACTOR	PC S9(5) CMP.
02	OR-MAXIMUM-BLOCK-SIZE	PC S9(5) CMP.
02	OR-UNIT-NAME	PC X(6).
02	OR-NUMBER-OF-STATIONS	PC S9(3) CMP.
02	OR-ACCESS-MODE	PC S9(3) CMP.
02	OR-NUMBER-AREAS-DECLARED	PC S9(3) CMP.
02	OR-BLOCKS-PER-AREA	PC S9(7) CMP.
02	OR-AUTOPRINT	PC S9 CMP.
02	FILLER	PC X(11).
02	OR-FILE-TYPE	PC S9(3) CMP.

B 1000 Systems SSOG, Volume 2  
LOGCONVERT

**Table 21-19. File Open (Cont)**

02	OR-SERIAL-NUMBER	PC X(6).
02	OR-QUEUE-FAMILY-SIZE	PC S9(3) CMP.
02	OR-QUEUE-MAX-MESSAGES	PC S9(3) CMP.
02	OR-PROTECTION	PC S9 CMP.
02	OR-PROTECTION-IO	PC S9 CMP.
02	OR-NUMBER-OF-BUFFERS	PC S9(3) CMP.
02	OR-MEMORY-REQUIRED	PC S9(7) CMP.
02	FILLER	PC X(12).
02	OR-OPEN-INPUT	PC S9 CMP.
02	OR-OPEN-OUTPUT	PC S9 CMP.
02	OR-NEW-FILE	PC S9 CMP.
02	OR-WITH-PUNCH	PC S9 CMP.
02	OR-WITH-PRINT	PC S9 CMP.
02	OR-OPEN-WITH-NO-REWIND	PC S9 CMP.
02	OR-OPEN-REVERSE	PC S9 CMP.
02	OR-OPEN-LOCK	PC S9 CMP.
02	OR-OPEN-LOCKOUT	PC S9 CMP.

**Table 21-20. File Close**

01	CLOSE-RECORD.	
02	CR-REC-TYPE	PC X.
02	CR-JOB-ACCOUNTING-NUMBER	PC S9(7) CMP.
02	CR-JOB-NUMBER	PC S9(5) CMP.
02	CR-YEAR	PC S9(3) CMP.
02	CR-JULIAN-DAY	PC S9(3) CMP.
02	CR-TIME	PC S9(7) CMP.
02	CR-FILE-NUMBER	PC S9(3) CMP.
02	CR-INTERNAL-NAME	PC X(10).
02	CR-EXTERNAL-NAME	PC X(30).
02	CR-HARDWARE-TYPE	PC S9(3) CMP.
02	CR-VARIABLE-RECORDS	PC S9 CMP.
02	CR-PSEUDO-READER-FILE	PC S9 CMP.
02	CR-DMS-FILE	PC S9 CMP.
02	CR-EMULATOR-TAPE-FILE	PC S9 CMP.
02	CR-RECORD-SIZE	PC S9(5) CMP.
02	CR-BLOCKING-FACTOR	PC S9(5) CMP.
02	CR-MAXIMUM-BLOCK-SIZE	PC S9(5) CMP.
02	CR-UNIT-NAME	PC X(6).
02	CR-NUMBER-OF-STATIONS	PC S9(3) CMP.
02	CR-ACCESS-MODE	PC S9(3) CMP.
02	CR-NUMBER-AREAS-DECLARED	PC S9(3) CMP.
02	CR-BLOCKS-PER-AREA	PC S9(7) CMP.
02	CR-AUTOPRINT	PC S9 CMP.
02	CR-RECORD-COUNT	PC S9(7) CMP.
02	CR-BLOCK-COUNT	PC S9(7) CMP.
02	CR-ERROR-COUNT	PC S9(5) CMP.

B 1000 Systems SSOG, Volume 2  
LOGCONVERT

**Table 21-20. File Close (Cont)**

02	CR-FILE-TYPE	PC S9(3) CMP.
02	CR-SERIAL-NUMBER	PC X(6).
02	CR-QUEUE-FAMILY-SIZE	PC S9(3) CMP.
02	CR-QUEUE-MAX-MESSAGES	PC S9(3) CMP.
02	CR-PROTECTION	PC S9 CMP.
02	CR-PROTECTION-IO	PC S9 CMP.
02	CR-NUMBER-OF-BUFFERS	PC S9(3) CMP.
02	CR-MEMORY-REQUIRED	PC S9(7) CMP.
02	CR-CLOSE-REEL	PC S9 CMP.
02	CR-CLOSE-RELEASE	PC S9 CMP.
02	CR-CLOSE-PURGE	PC S9 CMP.
02	CR-CLOSE-REMOVE	PC S9 CMP.
02	CR-CLOSE-WITH-CRUNCH	PC S9 CMP.
02	CR-CLOSE-WITH-NO-REWIND	PC S9 CMP.
02	CR-CLOSE-INSECURE	PC S9 CMP.
02	CR-CLOSE-WITH-LOCK	PC S9 CMP.
02	CR-CLOSE-CONDITIONAL	PC S9 CMP.
02	CR-CLOSE-ROLLOUT	PC S9 CMP.
02	CR-DMS-AUDIT-SWITCH	PC S9 CMP.
02	FILLER	PC X(1).
02	CR-CLOSE-UNAVAIL	PC S9 CMP.
02	FILLER	PC X(7).
02	CR-MCP-CLOSE-DUE-TO-EOJ	PC S9 CMP.
02	CR-BACKUP-EXCEEDED	PC S9 CMP.

**Table 21-21. DMS Statistics**

01	DMS-STATISTICS-RECORD.	
02	DMS-RECORD-TYPE	PC X.
02	DMS-JOB-ACCOUNTING-NUMBER	PC S9(7) CMP.
02	DMS-JOB-NUMBER	PC S9(5) CMP.
02	DMS-YEAR	PC S9(3) CMP.
02	DMS-JULIAN-DAY	PC S9(3) CMP.
02	DMS-TIME	PC S9(7) CMP.
02	DMS-DATABASE-NAME	PC X(10).
02	DMS-STRUCTURE-NUMBER	PC S9(3) CMP.
02	DMS-INVOKE-NUMBER	PC S9(3) CMP.
02	DMS-NUMBER-RANDOM-FINDS	PC S9(9) CMP.
02	DMS-NUMBER-SEQUENTIAL-FINDS	PC S9(9) CMP.
02	DMS-STRUCTURE-CLOSE	PC S9 CMP.
02	DMS-STRUCTURE-NAME	PC X(17).
02	DMS-FILE-NAME	PC X(30).
02	DMS-HARDWARE-TYPE	PC S9(3) CMP.
02	DMS-UNIT-NAME	PC X(6).
02	DMS-NUMBER-OF-FINDS	PC S9(9) CMP.
02	DMS-NUMBER-OF-INSERTS	PC S9(9) CMP.
02	DMS-NUMBER-OF-UPDATES	PC S9(9) CMP.

B 1000 Systems SSOG, Volume 2  
LOGCONVERT

**Table 21-21. DMS Statistics (Cont)**

02	DMS-NUMBER-OF-DELETES	PC S9(9) CMP.
02	DMS-NO-OF-SYSTEM-CHANGES	PC S9(9) CMP.
02	DMS-NUMBER-OF-EXCEPTIONS	PC S9(9) CMP.
02	DMS-NUMBER-LOGICAL-READS	PC S9(9) CMP.
02	DMS-NUMBER-LOGICAL-WRITES	PC S9(9) CMP.
02	DMS-NUMBER-PHYSICAL-READS	PC S9(9) CMP.
02	DMS-NUMBER-PHYSICAL-WRITES	PC S9(9) CMP.

**Table 21-22. DMS Open/Close**

01	DMS-OPEN-CLOSE-RECORD.	
02	DMSOC-RECORD-TYPE	PC X.
02	DMSOC-JOB-ACCOUNTING-NUMBER	PC S9(7) CMP.
02	DMSOCA-JOB-NUMBER	PC S9(5) CMP.
02	DMSOC-YEAR	PC S9(3) CMP.
02	DMSOC-JULIAN-DAY	PC S9(3) CMP.
02	DMSOC-TIME	PC S9(7) CMP.
02	DMSOC-PHYSICAL-DATABASE-NAME	PC X(10).
02	DMSOC-LOGICAL-DATABASE-NAME	PC X(17).
02	DMSOC-CLOSE-FLAG	PC S9 CMP.
02	DMSOC-UPDATE-FLAG	PC S9 CMP.
02	DMSOC-AUDIT-FLAG	PC S9 CMP.
02	DMSOC-HARDWARE-TYPE	PC S9(3) CMP.
02	DMSOC-UNIT-NAME	PC X(6).
02	DMSOC-PACK-NAME	PC X(10).
02	DMSOC-NUMBER-OF-UPDATE-OPS	PC S9(9) CMP.
02	DMSOC-NUMBER-NON-UPDATE-OPS	PC S9(9) CMP.
02	DMSOC-SMCP-TIME-TENTHS	PC S9(9) CMP.

**Table 21-23. Peripheral Assignment/Release**

01	PERIPHERAL-ASSIGN-RELEASE-RECORD.	
02	PAR-RECORD-TYPE	PC X.
02	PAR-JOB-ACCOUNTING-NUMBER	PC S9(7) CMP.
02	PAR-JOB-NUMBER	PC S9(5) CMP.
02	PAR-YEAR	PC S9(3) CMP.
02	PAR-JULIAN-DAY	PC S9(3) CMP.
02	PAR-TIME-TENTHS	PC S9(7) CMP.
02	PAR-HARDWARE-TYPE	PC S9(3) CMP.
02	PAR-UNIT-NAME	PC X(6).
02	PAR-ACTION	PC S9 CMP.



B 1000 Systems SSOG, Volume 2  
LOGCONVERT

**Table 21-24. End-of-Job**

01	EOJ-RECORD.	
02	EOJ-RECORD-TYPE	PC X.
02	EOJ-JOB-ACCOUNTING-NUMBER	PC S9(7) CMP.
02	EOJ-JOB-NUMBER	PC S9(5) CMP.
02	EOJ-YEAR	PC S9(3) CMP.
02	EOJ-JULIAN-DAY	PC S9(3) CMP.
02	EOJ-TIME-TENTHS	PC S9(7) CMP.
02	EOJ-PROGRAM-NAME	PC X(30).
02	EOJ-OBJECT-PROGRAM-NAME	PC X(30).
02	EOJ-EXECUTE-TYPE	PC S9 CMP.
02	EOJ-END-OF-JOB-REASON	PC S9 CMP.
02	EOJ-CPU-TIME-TENTHS	PC S9(7) CMP.
02	EOJ-NUMBER-OF-CODE-OVERLAYS	PC S9(7) CMP.
02	EOJ-NUMBER-OF-DATA-OVERLAYS	PC S9(7) CMP.
02	EOJ-SUCCESSOR-PROGRAM	PC X(30).
02	EOJ-DS-OR-DP-MESSAGE	PC X(60).

**Table 21-25. Log Transfer**

01	LOG-TRANSFER-RECORD.	
02	LT-RECORD-TYPE	PC X.
02	FILLER	PC X(7).
02	LT-YEAR	PC S9(3) CMP.
02	LT-JULIAN-DAY	PC S9(3) CMP.
02	LT-TIME-TENTHS	PC S9(7) CMP.
02	LT-NEXT-LOGFILE-NAME	PC X(30).

**Table 21-26. Comment**

01	COMMENT-RECORD.	
02	CM-REC-TYPE	PC X.
02	CM-JOB-ACCOUNTING-NUMBER	PC S9(7) CMP.
02	CM-JOB-NUMBER	PC S9(5) CMP.
02	CM-YEAR	PC S9(3) CMP.
02	CM-JULIAN-DAY	PC S9(3) CMP.
02	CM-TIME	PC S9(7) CMP.
02	CM-COM-LENGTH	PC S9(3) CMP.
02	CM-COMMENT	PC X(162).

**RPGII Record Formats**

Figures 21-1 through 21-10 are sample RPGII declarations which correspond to the record formats previously detailed in tables 21-1 through 21-15. They can be used by RPGII programs which process the output file of the LOGCONVERT program.

```

ILOGFILE NS 01 1 D1
I P 1 10CSRTYP
I P 2 50CSJAN
I P 6 80CSJOBN
I P 9 100CSYEAR
I P 11 120CSJDAY
I P 13 160CSTIME
I 17 46 CSMCP
I 47 76 CSINTP
I 77 86 CSVRSN
I P 87 910CSSMEM
I 92 121 CSGSMO
I 122 151 CSMHCP
I 152 152 CSNEWL
:
:
0---1---*---2---*---3---*---4---*---5---*---6---*---7---
6 0 0 0 0 0 0 0
:

```

Figure 21-1. Clear/Start

```

I NS 02 1 CB
I 1 1 SRTYPE
I P 2 50SJAN
I P 6 80SJOBNO
I P 9 100SYEAR
I P 11 120SJDAY
I P 13 160STIME
I 17 46 SPROG
I 47 76 SOBJ
I P 77 780SSCHPR
I P 79 790SEXTYP
I P 80 840SSTMEM
I P 85 890DYMEM
I P 90 940STOMEM
I P 95 980SCHGNO
:
:

```

**[TEXT DELETED]**

```

I P 102 1030SFILES
I 104 133 SINTRP
I P 134 1350SYRC
I P 136 1370SJDAYC
I P 138 1410STIMEC
I P 142 1440SRPNUM
:
:
0---1---*---2---*---3---*---4---*---5---*---6---*---7---
6 0 0 0 0 0 0 0
:

```

Figure 21-2. Schedule

B 1000 Systems SSOG, Volume 2  
LOGCONVERT

```

I      NS  03  1  D3                                     :
I                                          P  1  10BRTYPE      :
I                                          P  2  50BJAN       :
I                                          P  6  80BJOBNO     :
I                                          P  9  100BYEAR     :
I                                          P 11  120BJDAY     :
I                                          P 13  160BTIME     :
I                                          17  46  BPROG     :
I                                          47  76  80BJ      :
I                                          P 77  770BEXTYP   :
I                                          P 78  790BPPRI    :
I                                          P 80  810BMPRI    :
I                                          P 82  850BCHGNO   :
I                                          P 86  880BVDISK   :
I                                          89 118  BINTRP    :
:
0-----1-----2-----3-----4-----5-----6-----7-----:
6      0              0              0              0              0              0              0              0      :

```

Figure 21-3. Beginning-of-Job

B 1000 Systems SSOG, Volume 2  
LOGCONVERT

I	NS	04	1	CD						:
I					1	1	OR	TYPE		:
I					P	2	500	JAN		:
I					P	6	800	JOBNO		:
I					P	9	1000	YEAR		:
I					P	11	1200	JDAY		:
I					P	13	1600	TIME		:
I					P	17	1800	FILNO		:
I						19	28	OINTNM		:
I						29	58	OEXTNM		:
I					P	59	6000	HDWR		:
I					P	61	6100	VAR		:
I					P	62	6200	PSEUD		:
I					P	63	6300	DMS		:
I					P	64	6400	EMUTP		:
I					P	65	6700	DRECSZ		:
I					P	68	7000	BLOCK		:
I					P	71	7300	MAXBL		:
I						74	79	OUNI		:
I					P	80	8100	NUMST		:
I					P	82	8300	ACCSS		:
I					P	84	8500	AREAS		:
I					P	86	8900	BPORA		:
I					P	90	9000	AUTOP		:
I					P	102	10300	FILTP		:
I						104	109	OSERNO		:
I					P	110	11100	QFMSZ		:
I					P	112	11300	QMX		:
I					P	114	11400	QSEC		:
I					P	115	11500	QSUS		:
I					P	116	17000	QBUFFS		:
I					P	118	12100	MEM		:
I					P	134	13400	QINPUT		:
I					P	135	13500	QOUTPI		:
I					P	136	13600	QNEW		:
I					P	137	13700	QPUNCH		:
I					P	138	13800	QPRINT		:
I					P	139	13900	QNORWD		:
I					P	140	14000	QREVRS		:
I					P	141	14100	QLOCK		:
I					P	142	14200	QLKOUT		:
	0	1	2	3	4	5	6	7		:
	6	0	0	0	0	0	0	0		:

Figure 21-4. File Open

B 1000 Systems SSOG, Volume 2  
LOGCONVERT

I	NS	05	1	CE		:		
I			1	1	CRTYPE	:		
I			P	2	50CJAN	:		
I			P	6	80CJOBNO	:		
I			P	9	100CYEAR	:		
I			P	11	120CJDAY	:		
I			P	13	160CTIME	:		
I			P	17	180CFILNO	:		
I				19	28 CINTNM	:		
I				29	58 CEXTNM	:		
I			P	59	600CHDWR	:		
I			P	61	610CVAR	:		
I			P	62	620CPSEUD	:		
I			P	63	630CDMS	:		
I			P	64	640CEMUTP	:		
I			P	65	670CRECSZ	:		
I			P	68	700CBLOCK	:		
I			P	71	730CMAXBL	:		
I				74	79 CUNI	:		
I			P	80	810CNUMST	:		
I			P	82	830CACCSS	:		
I			P	84	850CAREAS	:		
I			P	86	890CBPERA	:		
I			P	90	900CAUTOP	:		
I			P	91	940CRECCT	:		
I			P	95	980CBLKCT	:		
I			P	99	1010CERRS	:		
I			P	102	1030CFILTP	:		
I				104	109 CSERNO	:		
I			P	110	1110CQFMSZ	:		
I			P	112	1130CQMX	:		
I			P	114	1140CSEC	:		
I			P	115	1150CSUS	:		
I			P	116	1170CBUFFS	:		
I			P	118	1210CNEM	:		
I			P	122	1220CREEL	:		
I			P	123	1230CRELES	:		
I			P	124	1240CPURGE	:		
I			P	125	1250CREMOV	:		
I			P	126	1260CCRNCH	:		
I			P	127	1270CNORWD	:		
I			P	128	1280CINSEC	:		
I			P	129	1290CLOCK	:		
I			P	130	1300CCONDT	:		
I			P	131	1310CRLOUT	:		
I			P	132	1320CAUDSW	:		
I			P	134	1340CUNAVL	:		
I			P	142	1420CBYMCP	:		
I			P	143	1430CBKEXC	:		
						:		
						:		
0	1	2	3	4	5	6	7	:
6	0	0	0	0	0	0	0	:

Figure 21-5. File Close

B 1000 Systems SSOG, Volume 2  
LOGCONVERT

I	NS	06	1	CF		:
I					1	1 DRTYPE
I					P 2	50DJAN
I					P 6	80DJOBNO
I					P 9	100DYEAR
I					P 11	120DJDAY
I					P 13	160DTIME
I					17	26 DDBNAM
I					P 27	280DSTRNO
I					P 29	300DINVNO

**[TEXT DELETED]**

I					P 31	310DSTRCL
I					32	48 DSTRNM
I					49	78 DFILNM
I					P 79	800DHDWR
I					81	86 DUNI
I					P 87	910DRANFD
I					P 97	1010DINSRT
I					P 102	1060DUPDAT
I					P 107	1110DDELET
I					P 112	1160DSYSCH
I					P 117	1210DEXCEP
I					P 122	1260DLOGRD
I					P 127	1310DLOGWR
I					P 132	1360DPHYRD
I					P 137	1410DPHYWR

0-----1-----2-----3-----4-----5-----6-----7-----  
6    0                    0                    0                    0                    0                    0                    0

Figure 21-6. DMS Statistics







## SECTION 22

### PACK/INIT

The PACK/INIT program is a stand-alone utility program designed to label, and to initialize, verify, or reconfigure disk packs for use on B 1000 systems. Pack types that can be created with the PACK/INIT program are System (S) and User (U).

All disks must be initialized before they can be used with B 1000 systems software. During initialization, the PACK/INIT program properly formats each sector by writing addresses, sync bits, data patterns, and error protection codes, and then reads the addresses and data to check for errors and to ensure that the sectors are not defective. Defective sectors are either relocated or removed. If any of the first 64 sectors of the disk are defective, that disk cannot be used with B 1000 system software. A removed sector is marked as unusable, and no alternate sector location (relocation) is provided. Sectors are removed only when more than five sectors in the same cylinder are bad, in which case the first five bad sectors are relocated, and the sixth and successive bad sector addresses are removed from the Master Available Table. In all other cases when a bad sector is found it is relocated to a spare sector.

The PACK/INIT program processes each cylinder of a disk individually to verify its integrity. The actual number of read and write operations performed on each cylinder is based on the characteristics of the disk device being processed. Cylinder 0, the outermost cylinder of the disk, is initialized first: label information and system tables are built in this cylinder at the end of the initialization process.

Disk characteristics are documented in appendix A of this volume, Disk Device Characteristics.

## OPERATING INSTRUCTIONS

The PACK/INIT program is a stand-alone utility program that must be loaded from a cassette and executed from the console keyboard. With dual processor systems, it is required that the operator push the console interrupt switch before attempting to execute the program. After the PACK/INIT program is loaded from the cassette, the program displays the following messages on the operator display terminal (ODT):

```
<ETX> REWIND CASSETTE
DISKPACK INITIALIZER <level and date>
IS CARD INPUT DESIRED? <YES OR NO>
```

Initialization specifications are then entered, either through the console keyboard or through a card reader. Several disks can be processed during a single execution of the PACK/INIT program, one at a time. A set of input specifications must be entered for each disk to be processed. The PACK/INIT program continues to read the specifications until it receives a blank input string when console keyboard input is used, or until it reaches end of file (EOF) when card input is used.

### Console Keyboard Execution

The PACK/INIT utility program prompts the operator for all initialization specifications.

During a single execution, the PACK/INIT program can process several disks, one at a time. After initializing, verifying, or reconfiguring each disk, the PACK/INIT program begins the prompt sequence again with the following message:

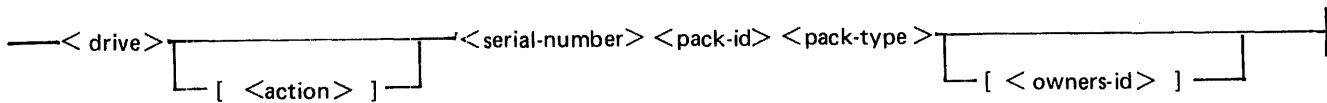
```
ENTER UNIT ID <DC? OR DP?>
```

A null or blank input response terminates the program.

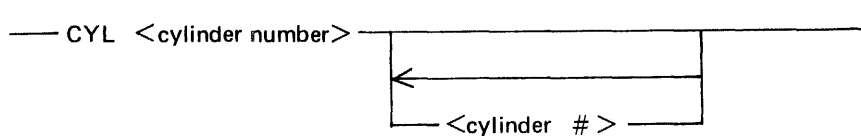
## Card Reader Execution

The PACK/INIT program accepts the disk initialization specifications from cards when the operator responds YES to the prompt IS CARD INPUT DESIRED?. The initialization specifications are not contained in a data deck; thus, neither a ?DATA or a ?END card is used.

All specifications are in free-form format (up to 96 columns) and must be separated by one or more spaces; comma (,) characters are not valid separators. The specification card format follows the same order as the prompt sequence and is described as follows. The semantics of the command are described in the section entitled User Interface Information.



When the action CV (cylinder verify) or CI (cylinder initialize) is specified, the following card format must be used to specify the cylinders to be processed.



If verification is requested (action is V), then only the <drive> and <action> specifications are read; the remainder of the specification card is ignored.

The initialization specifications used when the program is executed from cards are identical to those used when it is executed from the console keyboard. The specifications are described in the subsection entitled Initialization Specifications.

## Run Time Options

While the PACK/INIT program is executing, the operator can interrogate the program's status with an ST <ETX> inquiry. In response to this inquiry, the PACK/INIT program issues a message that describes the cylinder that it is currently processing and the number of sectors that it has relocated or removed thus far in its processing. The format and an example of this message follows:

```
CYLINDER = <# being processed>  PASS <current I/V pass>
RELOCATED = <#>  REMOVED = <#>
```

Example:

```
CYLINDER = 147  PASS = 3  RELOCATED = 0  REMOVED = 0
```

## INITIALIZATION SPECIFICATIONS

The following paragraphs are the initialization specifications.

### User Interface Information

#### DRIVE

The DRIVE message asks for a unit mnemonic which names the drive on which the disk to be processed is located (DC? or DP?).

Prompt: ENTER UNIT ID <DC? OR DP?>

#### ACTION

The ACTION message asks for the type of processing to be performed.

When the initialization action I, RI, RC, or V is requested, the PACK/INIT program processes the disk in sequential cylinder order. When the requested action is CI, or CV, the PACK/INIT program issues prompts during processing that ask the operator to specify the cylinders to be processed.

Prompt: ENTER ACTION: <I,V,RC,RI,CI,OR CV>

These abbreviations and their requested actions follow:

**I = Initialization**

Uses the default initialization characteristics for the device. The initialization characteristics are noted in Appendix A of this volume.

**V = Verification only**

**RC = Reconfiguration**

Verification with relocation or removal of bad sectors that are noted in the initialization table of the disk. The disk is purged.

**RI = Reinitialization**

Initializes using the default initialization passes, but relocates or removes any sector that has its address in the sector table built during the previous initialization of the disk.

**CI = Cylinder Initialization**

Same as initialization, except the operator must specify which cylinders are to be processed. A later prompt requests the cylinder number.

**CV = Cylinder Verification**

Same as verification, except the operator must specify which cylinders are to be processed. A later prompt requests the cylinder number.

#### NOTE

The actions RC, RI, CI, and CV are valid only when processing a disk that has previously been initialized with the Mark 8.0 (or later) version of the SYSTEM/DISK.INIT or PACK/INIT programs.

## SERIAL NUMBER

The SERIAL NUMBER message asks for a unique six-digit decimal number used to identify the disk. This number is used by the system when purging a pack and must be non-zero.

Prompt: ENTER 6 DIGIT SERIAL NUMBER

## PACK ID

The PACK ID message asks for a unique name that is assigned to the disk using six characters or less and containing no embedded blank characters. The system uses this name to reference the disk. When input specifications are entered from cards, omitting this entry causes an error for pack-type.

Prompt: ENTER PACK.ID

## PACK TYPE

The PACK TYPE message asks for the type of use to which this disk is dedicated and the type of access that is desired by the system software. The type is specified by a one-letter abbreviation of the allowable pack types, which are System, Unrestricted (User), and Restricted.

Prompt: ENTER PACK TYPE -<U,S,OR R>

## DATE

The DATE message asks for the current date in Julian date format. A Julian date is a five-digit field, with two digits for the year followed by three digits for the absolute day of the year.

Prompt: ENTER 5 DIGIT JULIAN DATE (YYDDD)

## OWNER'S NAME

The OWNER'S NAME message ask for an optional 14-character field available to identify the person responsible for maintaining the disk. The system does not reference this field.

Prompt: ENTER OWNER'S NAME

## OPTIONS

The OPTIONS message asks for one of several options to be entered. The options are available during initialization of a disk to provide a more flexible initialization and verification procedure and are described in the subsection entitled Initialization Options. The options allow the operator to extend or enhance the default initialization characteristics of the disk.

The OPTION prompt is repeated until a blank accept (AX) input message is received, at which point initialization, verification, or reconfiguration begins.

Prompt: ENTER OPTIONS

## CYLINDER NUMBER

The CYLINDER NUMBER message asks for the number of the cylinder that is to be processed. This information is required when either the cylinder initialize (CI) or cylinder verify (CV) action is specified. The PACK/INIT program continually processes the specified cylinders and reissues this prompt until it receives a blank input response. A single cylinder or a group of cylinders can be specified. The following example illustrates this prompt and valid responses.

Prompt: ENTER CYLINDER NUMBER

Responses: 25 40 <ETX>            or            300 <ETX>

When card input is used, the cylinder number specification must be on a separate card from the input specifications, and the keyword CYL must appear in columns 1-3, followed by the cylinder numbers to be processed. Several CYL cards can be used if needed.

## Initialization Options

The operator can extend or enhance the default initialization characteristics for a disk by using the initialization options. The REMOVE and LIMIT options control the total number of errors that are allowed to exist on a disk. The Marginal Sectors option describes sectors that are known to be bad and requests that they be relocated or removed. The Dollar Sign options control the physical I/O operations that occur when processing a disk.

### REMOVE Option

This option is valid only when initializing disk packs and allows the default error limit of five errors per cylinder to be overridden. If the REMOVE option is specified as an option, the first five bad sectors found in each cylinder are relocated. Additional sectors that are found to be in error are removed from the Master Available Table; thus, the pack is not rejected because the error limits have been exceeded when there are more than five bad sectors in a single cylinder. If the REMOVE option is specified, a pack is not rejected unless the total number of errors on the pack exceeds the default error limit or error limit specified by the LIMIT option. Refer to appendix A, Disk Device Characteristics, for a description of disk characteristics.

The REMOVE option is ignored when processing B 9499-6 disk packs.

The REMOVE option is requested by the keyword REMOVE. When executing the PACK/INIT program from cards, the keyword REMOVE must appear on a physically separate card from all other specifications. For execution from the console keyboard, the keyword is entered when the prompt ENTER OPTIONS appears.

### LIMIT Option

This option allows the default error limit totals to be overridden for disk packs. If the LIMIT option is requested, a disk is not rejected until the total error count for the disk exceeds the value specified in the option. The default error limit of five errors per cylinder might cause a pack to be rejected, unless the REMOVE option has also been requested.

The LIMIT option is requested with the following syntax:

LIMIT <integer >

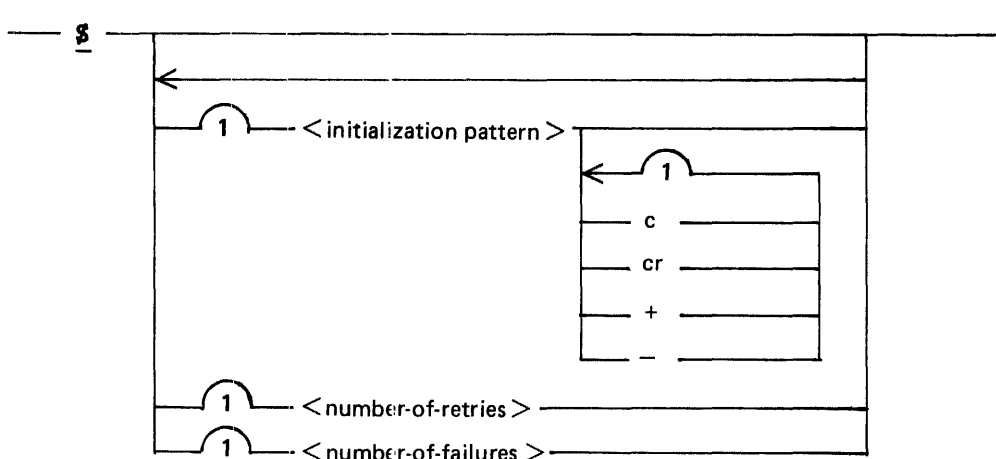
### Marginal Sectors Option

This option allows the operator to describe sectors that are known to be bad (known to require relocation or removal). Marginal sectors that are removed or relocated are included in the error count totals. Sector addresses specified in this option must be valid decimal addresses. More than one sector address can appear in each response to the ENTER OPTIONS prompt, or in each input specification card, with the restriction that the addresses are separated by spaces, not by commas.

If CI or CV is requested as the action (cylinder initialization or verification), then the operator must ensure that the specified marginal sector resides within a cylinder that is to be processed (has been included in a CYLINDER NUMBER input specification).

### Dollar Sign Options

These options allow the operator to verify with or without offset, to compare during verification, to change the number of retries on bad sectors, or to modify the number of failures that can occur before a sector is considered bad. The syntax of the dollar-sign option follows.



Semantics:

#### Initialization Pattern

The initialization pattern entry is a four-digit representation of a hexadecimal pattern (0000 through FFFF). The first dollar-sign statement, in a series of dollar-sign statements, which omits this entry causes the default pattern to be used for that pass (refer to the disk device characteristics for the default patterns). Any successive dollar-sign statements apply to verification only, and only those entries described in the following paragraphs are valid:

Offset: + or

The plus sign (+) or minus sign (-) entry is used for disk packs to indicate that in (+) or out (-) offset is to be used during verification. Default is no offset.

The term offset refers to a disk pack capability which is a means of causing the disk unit to create a critical head alignment useful in testing marginal disk conditions. This capability is not available in Design Level I B 9499-7/B 9499-8 Disk Pack Drives or the Design Level I Disk Pack Electronics Controller (DPEC). Specifying offset on this level disk pack has no effect on initialization.

B 1000 Systems SSOG, Volume 2  
PACK/INIT

**Number-of-retries**

The number-of-retries entry specifies the number of times during verification that the same I/O operation is attempted after an exception condition has occurred. Default is ten retries.

**Number-of-failures**

The number-of-failures entry specifies how many of the retries specified by the number-of-retries entry are allowed to fail before the sector is considered bad and is relocated or removed. Default allows one failure.

**NOTE**

If the default values for both number-of-retries and number-of-failures are used while verifying a pack, all ten retries must be successful to allow the sector to remain.

## **INFORMATIVE MESSAGES**

The PACK/INIT program issues messages that describe the processing that is being performed. The following sequence of messages are examples of the information in these informative messages when cylinder processing is requested.

```
INITIALIZATION BEGINS - CYL:0 -DPD
INITIALIZATION COMPLETE CYL:0 -DPD
INITIALIZATION BEGINS - CYL:1 -DPD
INITIALIZATION COMPLETE CYL:1 -DPD
ID=SNARFF SER#=022779 PACK TYPE=S
0 RELOCATED SECTORS
INITIALIZATION COMPLETE - DPD
```

The following sequence of messages exemplify the informative messages issued by the PACK/INIT program when the requested action is not CI or CV:

```
INITIALIZATION BEGINS - DPD
ID=SNARFF SER#=022779 PACK TYPE=S
0 RELOCATED SECTORS
```

## **RELOCATION NOTIFICATION**

When sectors are relocated, the PACK/INIT program issues informative messages regarding the relocation. The message is in the following format:

**THE FOLLOWING SECTORS ARE IN ERROR AND WILL BE RELOCATED**

```
ADDRESS HEX:04ABEC DEC:306156
ADDRESS HEX:050ECB DEC:33147
```

## ERROR MESSAGES

### INVALID ENTRY BLANK ID

A blank prompt response is not valid at this time.

### INVALID ENTRY <entry>

The prompt response is invalid; enter a valid response.

### INVALID SERIAL NUMBER

Enter a valid serial number. It must be a six-digit, non-zero decimal number.

### INVALID SECTOR NUMBER

The addresses of marginal sectors must be specified in decimal.

### THE FOLLOWING SECTORS ARE IN ERROR AND WILL BE RELOCATED

This informative message is issued when sectors on a disk pack are relocated.

### THE FOLLOWING SECTORS ARE IN ERROR AND THEIR TRACKS WILL BE REMOVED

This is an informative message issued when bad sectors are encountered while initializing a disk cartridge.

### IS <unit-id> TO BE INITIALIZED?

This is a precautionary measure to ensure that the operator has specified the correct disk. YES and NO are valid responses.

### WOULD YOU LIKE TO RETRY? <YES OR NO>

This message follows the DISK ERROR message and offers the operator the option of retrying an I/O operation that has previously failed. A YES response causes the operation to be retried. A NO response causes processing of the current disk to be terminated.

### DISK ERROR - RESULT = <result-descriptor>

An irrecoverable I/O error existed after ten (or the value set in a Dollar Sign option) retries.

### MUST RESTART TO CONTINUE

The PACK/INIT program cannot continue execution. Re-execute the program.

### WRITE LOCK OUT <unit-id>

Remove the write lock-out on the specified unit and execution continues.

### DISK NOT READY <unit-id>

Ready the specified unit and execution continues.



**PACK CANNOT BE USED WITH MCPH <unit-id>**

One of the first 64 sectors has been removed, or the Master Available Table is full. These are irrecoverable disk faults.

**ERROR CYL 0 <unit-id> <disk-address>**

This is an informative message only. A disk with relocated sectors within cylinder 0 can be used with the system software; however, a disk with removed sectors within cylinder 0 cannot be used with the system software.

**SECTOR REMOVED <disk-address>**

This is an informative message and is issued for each removed sector.

**COMPARISON ERROR <disk-address>**

The C or CR option is enabled, and a comparison error has been detected at the address specified.

**ERROR LIMITS HAVE BEEN EXCEEDED**

The maximum number of errors allowed per cylinder (if the REMOVE option was not specified), or the total number of errors allowed for the disk (either the default value or the value specified by the LIMIT option), was exceeded during verification. This message might also indicate that, due to removed sectors (REMOVE option), the Master Available Table is filled beyond capacity.

## SECTION 23

### QWIKLOG

The QWIKLOG program is a normal-state utility program designed to provide a convenient and compact analysis of job information contained within the MCP II log file, a disk file of system activity maintained by the MCP II. The LOG option must be set in order for the MCP II to maintain the LOG information (refer to the SL input message in volume 1, section 2).

The QWIKLOG program presents the results of its analysis in two distinct formats:

1. A tabular summary of selected information about each job.
2. A graph which chronologically depicts schedule and mix activity.

### OPERATING INSTRUCTIONS

By default, the QWIKLOG program analyzes the currently active log file named SYSTEM/LOG. The active log file can be transferred at any time by the operator (refer to the LG and TL input messages) or automatically by the MCP II upon reaching maximum capacity. This action renames SYSTEM/LOG as LOG/<integer> and creates a new SYSTEM/LOG file. Such a transferred log file can be analyzed with the following command.

```
EXECUTE QWIKLOG FILE LOGFILE NAME LOG/<integer>;
```

The 6-digit <integer> identifies the desired log file.

As an operator convenience, if the FILE statement (refer to volume 1, section 1) is used to change the NAME file attribute of the LOGFILE file to the single name of LOG, the QWIKLOG program issues a prompt requesting the <integer> to be entered in an accept (AX) input message. Leading zero (0) characters need not be provided and, as with all program input, the accept (AX) input message response can be entered in free-form format.

### PROGRAM SWITCHES

The QWIKLOG program recognizes switches 0 and 8:

#### Switch 0

Used to specify the width of the print line. A value of 0 implies 132 print positions, and a value of 1 implies 120 print positions. Users with 120-position line printers can permanently set this switch with the following control instruction:

```
MODIFY QWIKLOG SWITCH 0=1
```

#### Switch 8

By default, the QWIKLOG program prints both parts of the analysis for all jobs within the log file. If program switch 8 is set to 1, however, the QWIKLOG program requests information to control its actions.

## B 1000 Systems SSOG, Volume 2 QWIKLOG

### Prompts:

JOB SUMMARY? (Y/N)  
Enter Y, YES, N, or NO.

GRAPH? (Y/N)  
Enter Y, YES, N, or NO.

JOB NUMBER RANGE: "ALL" OR <from> [ <to> ]

The JOB NUMBER RANGE option allows selection of a specific range of jobs for analysis. The keyword ALL specifies that all jobs within the given log file are to be processed (as is done in the default case); otherwise, the first integer (<from>) is used as the lower bound for the job number range to be analyzed. Analysis continues to the end of the log file unless a second integer (<to>) is provided to indicate an upper bound. The specified range can span zero; that is, the <stopping job number> can be less than the <starting job number>. The input for all three QWIKLOG requests can be provided in a single AX input message, in which case processing commences immediately.

### Example:

```
% QWIKLOG =907 JOB SUMMARY? (Y/N)
907AX Y NO 800 860
```

If analysis is done upon the currently active SYSTEM/LOGFILE file, the QWIKLOG program stops upon reaching its own BOJ record in the log unless a lower value is specified for the <stopping job number>.

## WORK FILES

The work file SLOG can be file-equated to a user disk (with the PACK.ID [PID] file attribute) to minimize disk arm movement. For installations with multiple system disk drives (refer to the SD input message in volume 1, section 2), SLOG is automatically placed on the second system disk drive, again in an attempt to minimize disk arm movement. If a second user disk is available, the other work file, GRAFILE, can be file-equated to further reduce disk arm movement.

## DESCRIPTION OF OUTPUT

### Summary Listing

The following information is provided in the job summary portion of the QWIKLOG output. Fields inappropriate for a particular job are left blank. For example, a job that has been removed from the schedule (RS input message) has no BOJ or EOJ date and time.

#### Job Number

An asterisk (\*) preceding the job number indicates that the job was part of a job stream (refer to the THEN, AFTER, and AFTER.NUMBER control instruction attributes in volume 1, section 2).

#### Program name

#### Job Accounting Number

This is a sequential value associated with each job which, unlike the job number, is not reset at 9999, and hence provides a unique identification number for each job.

B 1000 Systems SSOG, Volume 2  
QWIKLOG

Object program name

If the execute type is a compilation. For a CLEAR/START operation, this field contains the MCP II version number.

Charge number

If non-zero.

Execute type

Items that can appear are EXECUTE, COMPILE & GO, COMPILE & SAVE, COMPILE TO LIBRARY, GO PART OF COMPILE & GO, and GO PART OF COMPILE & SAVE.

Priorities

Schedule, processor, and memory priorities.

Interpreter <family-name>

For example, COBOL.

Date and time scheduled

BOJ date and time

EOJ date and time

Elapsed execution time

Processor time consumed

Job termination type

Items that can appear are NORMAL, discontinued (DS or DP input message), SYNTAX ERRORS, ABORTED (by a CLEAR/START operation), RS-ED (removed from the schedule with the RS input message), FAM. DEATH, and RUNNING (which indicates that the job was still in the mix when the QWIKLOG program was executed or when the logfile was transferred).

Static memory assigned (in bits)

Dynamic memory assigned (in bits)

Total Run Structure size

In bits. Defined as <static memory> + <dynamic memory> + <run structure nucleus>. This figure does not include structures such as file buffers. Users interested in a more complete analysis of memory requirements for a given program should use the CODE/ANALYZER utility program. In the entry for a CLEAR/START, this field contains the system main memory size in KB (1 KB = 1024 bytes = 8192 bits).

Files

The number of files declared with this program.

Open and closes

The number of opens and number of closes performed by the program or automatically performed by the MCP.

B 1000 Systems SSOG, Volume 2  
QWIKLOG

Code overlays

The number of code overlays (program, interpreter, and MICRO.MCP) requested by the program.

Data overlays

The number of data overlays performed during execution.

VIRTUAL.DISK segments

Segments of VIRTUAL.DISK (disk for data overlays) assigned.

**Mnemonics**

The following mnemonics are used with the chronological graph:

Mnemonic	Meaning
*	CLEAR/START
S	Program Scheduled
E	Program Execution
C	Compiler Execution
R	Program Running (Program still running at time of analysis or transfer of log.)
X	Abnormal Termination (Spotlights an abnormal by substituting the X for C or E for the last minute of a job that was discontinued (DS or DP input message) or was aborted due to a CLEAR/START operation.

**Error Messages**

EMPTY LOG FILE... EOJ

[TRY AGAIN...] ENTER LOG NUMBER

If LOGFILE file is file-equated to the single name of LOG, the QWIKLOG program requests the log identification <integer> with this prompt. The phrase TRY AGAIN... prefaces the next request if the previous AX input message was invalid.

FIRST 3 HEXITS NEQ @1AA@  
INVALID RECORD TYPE

These two messages appear on the printer file labeled LINE, but not on the operator display terminal, the first five or fewer times an incorrectly-formatted record is found in the log being analyzed. A hexadecimal/EBCDIC printout of the offending record also appears. Processing continues, although results for the job to which the faulty record applies can be erroneous. This problem is indicative of disk file corruption.

B 1000 Systems SSOG, Volume 2  
QWIKLOG

HEY...I NEED SOME WORK...EOJ

This message is displayed if the operator specifies that neither a job summary nor a graph are to be produced.

SOFTWARE INCOMPATIBILITY: <level> QWIKLOG; <level> MCP...EOJ

The format of log records is subject to change between software releases. To guarantee valid results, the QWIKLOG program ascertains that it is being run with the proper MCPII and terminates with this message if a mismatch is detected.

INVALID INPUT "<TOKEN>"

The QWIKLOG program decoded operator input that was contextually invalid.

INVALID TRANSFER HEADER RECORD IN LOG...EOJ

The first record in the specified logfile was not the expected CLEAR/START or pseudo-CLEAR/START record.

LOG HAS NO JOBS IN SPECIFIED RANGE...EOJ

The QWIKLOG program found that the log file being analyzed had no jobs within the range specified by the operator.

LOG XFER RECORD IS GARBAGE...EOJ

When processing a transferred log file, the QWIKLOG program discovered that the last record in the file was not the expected log transfer record.

NO JOBS IN LOG...EOJ

The log file which the QWIKLOG program was processing contained no jobs.

TIMEOUT ON DISK I/O AT <absolute disk address>

A disk read operation timeout occurred when the QWIKLOG program was using INITIALIZER I/O operation to process the currently-active SYSTEM/LOG file. This message causes the QWIKLOG program to go to EOJ.

IRRECOVERABLE I/O ERROR ON LOG READ AT <absolute disk address>;

RESULT= <result descriptor>

An exception condition occurred when the QWIKLOG program was using INITIALIZER I/O to read the currently-active SYSTEM/LOG file. The QWIKLOG program goes to EOJ if an irrecoverable disk error occurs.

<file-name> NOT ON DISK...EOJ

The log file on which analysis was requested was not in the disk directory.

B 1000 Systems SSOG, Volume 2  
QWIKLOG

NULL INPUT INVALID

The operator entered an empty input line in response to an accept (AX) input message.

(STARTING/STOPPING) JOB NUMBER "<token>" ISN'T NUMERIC

A non-numeric response was entered for an <integer> specifying the job number range.

**Program Traps**

The following four program traps within the QWIKLOG program reflect probable software errors. A Burroughs technical representative should be provided with a machine-readable copy of the log file for analysis by Burroughs QWIKLOG development and support personnel. The dump that is produced is also to be printed and submitted.

BINARY SEARCH FAILED: PLEASE SUBMIT LOG TO BURROUGHS...EOJ

COULDN'T HANDLE CLEARSTART: PLEASE SUBMIT LOG TO BURROUGHS...EOJ

GRAFILE EOF: PLEASE SUBMIT LOG TO BURROUGHS...EOJ

LOST SLOG: PLEASE SUBMIT LOG TO BURROUGHS...EOJ

## SECTION 24

### SQUASH/USER.DISK

The SQUASH/USER.DISK program is a normal-state utility program that consolidates available storage areas on a user disk. In performing this consolidation, the squash algorithm moves data into holes in the available disk space, and then frees the previously used data space. When disk space is extremely fragmented, successive squashes can further consolidate the available area. The SQUASH/USER.DISK program uses an external sort program called SORT/VSORT. The SORT/VSORT program displays the number of records sorted. This number consists of all disk pieces, which can or cannot be moved.

The SQUASH/USER.DISK program can run while other programs are in the mix; however, no other program can access the disk that is being squashed. The disk being squashed is unavailable to the system when the squash operation begins. The SQUASH/USER.DISK program waits until there is no activity on the disk before beginning the squash.

### OPERATING INSTRUCTIONS

The SQUASH/USER.DISK program can be initiated with an explicit EXECUTE command, or it can be initiated implicitly with the MCP/II system control instruction SQ. Refer to SQ input message in volume 1, section 2 for further details. The explicit execution of SQUASH/USER.DISK is described in the following example:

```
EXECUTE SQUASH/USER.DISK
  SQUASH/USER.DISK = 1111 BOJ.
  % SQUASH/USER.DISK = 1111 ENTER UNIT TO BE SQUASHED:
    DP? OR DC?
  SQUASH/USER.DISK = 1111 ACCEPT.
1111 AX <unit mnemonic>
```

No further input is required. The program displays an appropriate message at the beginning of the job or if the job cannot be started due to programs still running on the designated disk drive. The following messages are displayed.

```
DO NOT DISTURB <unit-mnemonic> UNTIL FURTHER NOTICE
Displayed at the beginning of the job.
```

```
ACTIVITY MUST CEASE ON <unit-mnemonic> FOR DISK SQUASH TO CONTINUE
Displayed if specified pack is in use.
```

As soon as the specified unit becomes available (when all activity on the device stops) the SQUASH/USER.DISK program performs an integrity check on the disk and then squashes the disk.



B 1000 Systems SSOG, Volume 2  
SQUASH/USER.DISK

When the squash operation is complete, the SQUASH/USER.DISK program displays the following messages:

```
% SQUASH/USER.DISK = 1111      MAXIMUM CONTIGUOUS SPACE
                                AVAILABLE IS <integer> SECTORS
% SQUASH/USER.DISK = 1111      TOTAL SPACE AVAILABLE IS
                                <integer> SECTORS
% SQUASH/USER.DISK = 1111      SQUASH COMPLETE
% SQUASH/USER.DISK = 1111      <unit-mnemonic> LABELED <pack-id>
                                IS NOW AVAILABLE FOR USE

SQUASH/USER.DISK = 1111 EOJ.
```

## INTEGRITY CHECK

The SQUASH/USER.DISK program checks the disk for overlapping sectors before beginning the squash operation. If any overlapping sectors are found the SQUASH/USER.DISK program describes the size and location of each overlapping sector in an operator display terminal (ODT) message and then aborts the squash operation.

Examples of overlapping sector messages:

```
OVERLAPPING SECTORS ON UNIT DPB LABELLED DOCGRP
```

```
1 SECTOR OVERLAPPING STARTING AT SECTOR @000ABB@
1 SECTOR OVERLAPPING STARTING AT SECTOR @000ABD
128 SECTORS OVERLAPPING STARTING AT SECTOR @000D15@
```

Overlapping areas cause the program to abort before performing the squash operation. The following message is displayed when this error occurs:

```
<integer> OVERLAPPING AREAS --SQUASH ABORTED
```

The SQUASH/USER.DISK program also checks the available table for out-of-sequence or zero-length entries. These are flagged on the first KA listing (generated by the SQUASH/USER.DISK program), and the program aborts before performing the squash operation with the following message:

```
BAD AVAILABLE TABLE --SQUASH ABORTED
```

## RUN-TIME OPTIONS

While the squash operation is in progress, the operator can set several options by entering unsolicited console keyboard accept (AX) input message that have the following syntax:

```
<job #> AX <option>
```



Descriptions of valid options and the actions they request follow.

#### SIZE

The program displays the current address, the maximum contiguous disk space available, and the total disk space available. The format of these messages is identical to the space messages displayed prior to EOJ. The values displayed reflect the contiguous and total space available at the time the SIZE interrogation is received.

#### STOP

The program terminates as soon as possible without corrupting data. The SQUASH/USER.DISK program must never be discontinued (DS or DP input message); the STOP run-time option must be used instead.

#### NEED <n>

The program terminates as soon as the <n> contiguous sectors are made available. This option does not ensure that <n> contiguous sectors are made available; if they are made available, however, the program terminates without consolidating (squashing) the disk any more.

#### TRACE <n>

The program generates trace output on a line printer listing. The value of <n> must be either 0, 1, 2, or 3. Refer to the following subsection entitled Trace Options for further details.

## TRACE FUNCTIONS

The TRACE option allows the operator to follow the progress of the squash. One of three different types of trace functions can be requested. The value <n> entered with the TRACE keyword selects the trace function.

0

Nothing is traced. This is the default value of trace. Entering TRACE 0 turns off previously requested trace options.

1

Each piece of data is traced when it is moved. Example of TRACE 1 output:

AVAILABLE TABLE MOVED FROM @00005F@ TO @000042@

FILE HEADER MOVED FROM @00007B@ TO 00004B@

SUB-DIRECTORY MOVED FROM @00008A@ TO @00004F@

FILE AREA MOVED FROM @00009A@ TO @000053@

2

Each disk I/O (read or write) operation is traced. Example of TRACE 2 output:

READ @0000EC@

WRITE @00006F@

READ @00006E@

WRITE @00006E@

3

Each disk I/O (read or write) operation is traced, and the contents of the read/write buffer are listed on the trace output. Example of TRACE 3 output:

```
READ @000030@ F21000031000000000 F210000300F2100007800008F....
```

```
WRITE @000030@ F21000031000000000 F210000300F2100007900008E....
```

## PROGRAM OUTPUT

The SQUASH/USER.DISK program generates a summary printer listing and a squashed input disk.

### Summary Listing

The SQUASH/USER.DISK program generates a line printer listing that describes the disk before and after the squash is performed. This summary listing is similar to two KA listings, with additional information describing the available disk areas printed at the beginning of the listing.

TRACE information, if requested, is included in this listing. Also, out-of-sequence or zero-length available table entries are flagged on this listing if the SQUASH/USER.DISK program discovers table errors during its integrity check.

If no line printer is available, a backup print file containing the information is created.

### Squashed Disk

After the squash operation is complete, the user disk contains the same data that it did prior to the squash, although the data can be in a physically different location on the disk. The available tables are updated during the squash, with the available areas consolidated whenever possible.

## ERROR RECOVERY

An abnormal termination of the SQUASH/USER.DISK program due to a CLEAR/START operation, a DS or DP input message, or program failure, might corrupt the disk being squashed. How much information, if any, is lost depends on the state of the program at the time it was aborted.

The DISKMAP/UTILITY program must be run to determine what errors, if any, there are. Using the DISKMAP/UTILITY output, along with a KA listing of the disk before and after the squash operation, the operator can determine what was being moved at the time of failure, and can attempt recovery of lost data by removing any bad areas and using IAD (installation allocated disk) files to recover lost information. The order of the squash operation follows.

1. The data is moved.
2. Relatives are notified of the move (for example, the area-address field of a disk file header must be updated when one of its areas is moved).
3. The old space is returned to the disk available table.
4. The new space is removed from the disk available table.

## ERROR MESSAGES

The following paragraphs describe the error messages.

### Nonfatal errors

Nonfatal errors generate a message that describes the error. When the error exists in an input specification string, the SQUASH/USER.DISK program requests a response and waits for an accept (AX) input message. The operator must correct the error, re-enter the incorrect specifications, or terminate the program.

#### <unit> IS AN INVALID UNIT NAME

Valid unit names are DP<letter> and DC<letter>. These mnemonics reference the disk drive on which the disk to be squashed is located. DP<letter> indicates a disk pack device; DC<letter> indicates a disk cartridge device.

#### <unit> COULD NOT BE ASSIGNED

The unit requested could not be assigned to the program. Make sure that the unit is on-line and that there is no activity on the disk. While a disk is being squashed, no program can access data on the disk.

#### <unit> LABELLED <name> IS NOT A USER DISK

The SQUASH/USER.DISK program can only squash user disks. Refer to the SQ input message in volume 1, section 2 for information on how to squash system disks.

#### <option> IS NOT A KNOWN OPTION

An invalid run-time option was requested in an unsolicited console keyboard accept (AX) input message. See Run-Time Options subsection for valid options.

#### VALUE OF <value> FOR <option> IS NOT NUMERIC

TRACE and NEED require numeric values. Re-enter the request.

#### TRACE CANNOT BE SET TO <value>

TRACE can only be set to 0, 1, 2, or 3.

#### NOT READY ON <unit> -- PLEASE CORRECT AND AX WITH OK

The unit being squashed has become not ready. The unit must be readied for the squash to continue. The SQUASH/USER.DISK program waits for an accept (AX) input message before continuing. Data is not corrupted.

#### WRITE LOCKOUT ON <unit> -- PLEASE CORRECT AND AX WITH OK

A write lockout has occurred on the unit being squashed. The unit must be write-enabled before the squash operation can continue. The SQUASH/USER.DISK program waits for an accept (AX) input message before continuing. Data is not corrupted.

## Fatal Errors

When the SQUASH/USER.DISK program discovers an uncorrectable error it issues an error message describing the error, and then aborts with the message SQUASH ABORTED.

Following an abnormal termination of the SQUASH/USER.DISK program, the unit being squashed is not readied. The operator must check the integrity of the disk before using it. Refer to the Error Recovery subsection for details.

### IO UNCORRECTED AFTER 10 RETRIES

An irrecoverable disk I/O error occurred. This error indicates that the disk itself is corrupted.

### INVALID FILE HEADER AT SECTOR <hex address>

A file header contained invalid information. Two possible causes of this error follow.

1. More than 105 areas were requested for the file.
2. The number of assigned areas for a file is greater than the number of areas requested.

### PROGRAM ERROR

The squash aborted due to a programmatic error. The SQUASH/USER.DISK program generates a memory dump of its memory areas before going to EOJ. Save the memory dump, and call a Burroughs technical representative.

### <integer> OVERLAPPING AREAS

Refer to the Integrity Check subsection.

### BAD AVAILABLE TABLE

Refer to the Integrity Check subsection.

## SECTION 25 SSLOAD/MAKCAS

The SSLOAD/MAKCAS program is a normal-state utility program that creates CLEAR/START and stand-alone utility cassettes. Cassettes can be created only on B 1800 systems that have a magnetic tape I/O control and on B 1700 systems that have a magnetic tape cassette subsystem in addition to the console cassette drive. The cassette system must be in I/O mode when the SSLOAD/MAKCAS program is executed.

### OPERATING INSTRUCTIONS

The SSLOAD/MAKCAS utility program operates in two distinct modes. When switch 0 = 0, the utility program accepts its input from the console keyboard through accept (AX) input messages and must be used to create CLEAR/START cassettes. When switch 0 = 1, the SSLOAD/MAKCAS program accepts its file specification data from an input file labeled CARDS and assumes by default that the input file is a card deck.

An EXECUTE SSLOAD/MAKCAS statement initiates the program, and switch settings must be specified when the execute statement is entered.

### PROGRAM SWITCHES

Switch settings control the execution of the SSLOAD/MAKCAS program. When a value is not specified for a switch, the default value of 0 is assumed. Valid switch settings for the SSLOAD/MAKCAS program are given in table 25-1.

**Table 25-1. SSLOAD/MAKCAS Program Switches**

Switch	Value	Meaning
0	0	Input is through the ODT.
0	1	Input is through the card reader.
1	0	Create a B 1000 format cassette.
1	1	Create a B 1830/B 1825 format cassette.
1	2	Create a B 1830/B 1825 format cassette using the triple check feature.
2	0	Create a CLEAR/START cassette. Enter CLEAR/START with an accept message.
2	1	Create a stand-alone utility cassette containing an <u>SDL</u> utility program. An input file must identify the software to be loaded.
9	0	Create, but do not verify the cassette.
9	1	Create and then verify the cassette. Do not remake in case of error.
9	2	Create and then verify the cassette. If an error is found, remake and reverify the cassette.

*ML program*

## CLEAR/START CASSETTES

To create a CLEAR/START cassette, execute the SSLOAD/MAKCAS program and enter the file identifier CLEAR/START through the console keyboard with an accept (AX) input message. One CLEAR/START cassette is created for each accept (AX) input message; however, it is the operator's responsibility to ensure that a cassette is scratched and the drive is ready as each cassette is made. A blank accept message terminates the SSLOAD/MAKCAS program.

Example:

```
EX SSLOAD/MAKCAS

  SSLOAD/MAKCAS = 1257 BOJ...
  % SSLOAD/MAKCAS = 1257 ENTER FILE IDENTIFIER
  SSLOAD/MAKCAS = 1257 ACCEPT.

1257AXCLEAR/START

1257AX

  SSLOAD/MAKCAS = 1257 EOJ.
```

## STAND-ALONE UTILITY CASSETTES (SDL UTILITY PROGRAMS)

### Instructions

To create a stand-alone utility cassette, execute the SSLOAD/MAKCAS program with switch 0 = 1, and identify the input specification file. The input file identifies all software (including the loader, GISMO routine, interpreter, and utility program) to be loaded onto the cassette. The input file must have the information in the exact format described in the following, one specification per record, each beginning in column 1.

```
L CASSETTE/LOADER
G GISMO/SA
I SDL/INTERP1U
S <file identifier>
```

The CASSETTE/LOADER, GISMO/SA, and SDL/INTERP1U programs are the standard system software, and these program names must not be changed. The <file identifier> specifies the name of the SDL utility program to be loaded on the cassette. All programs to be loaded must exist on disk. Only one SDL utility program can be written to each cassette.

### Stand-alone SDL Utility Programs

<file identifier>	Program Description
CART/INIT	Disk Cartridge Initializer
PACK/INIT	Disk Pack Initializer
DISK/DUMP	Sector by sector disk dump
STANDALONE/DISK.DUMP	File by file disk dump
COLDSTART/DISK	Disk coldstart
COLDSTART/TAPE	Tape coldstart
STANDALONE/INTERCHANG	Create an interchange disk pack

### Sample Execution Code

The following code can be punched on specification cards and used to execute the SSLOAD/MAKCAS program.

```
? EXECUTE SSLOAD/MAKCAS SW 0 1;  
? DATA CARDS  
L CASSETTE/LOADER  
G GISMO/SA  
I SDL/INTERP1U  
S DISK/DUMP  
? END
```

### ERROR CORRECTION

If any one of the input L, G, I, or S specification cards is missing, or names a non-existent disk file, the SSLOAD/MAKCAS program notifies the operator and asks for the correct file-identifier to be entered. The requested specification card, followed by an ?END card must then be read through the card reader from which the SSLOAD/MAKCAS program is receiving its data.

### ERROR MESSAGES

INVALID FILE TYPE <type>

The <type> specified in column one of an input specification card is not L, G, I, or S. Correct and re-enter the card.

FILE NOT ON DISK-<file identifier>

The file named could not be located on disk. Make this file available, or request another file, and re-enter the corresponding specification card through the card reader assigned to the SSLOAD/MAKCAS program.

MISSING <file type> FILE

The specification card for the file-type indicated was not found. Read the correct card through the card reader assigned to the SSLOAD/MAKCAS program.

<file identifier> IS NOT A VALID CODE FILE

The specified SDL program is segmented, and thus cannot be used to make a stand-alone SDL cassette. Stand-alone programs cannot be segmented.

DISK PARITY ERROR - UNABLE TO VERIFY

The temporary disk file used during verification has a parity error.

VERIFY ERROR REC# <record number>

An error was encountered during verification of the record indicated.

REMAKE NOT REQUESTED

A verification error was encountered, but the cassette was not remade because switch 9 = 1.

INVALID REMAKE - DISK PARITY ERROR ENCOUNTERED

The temporary disk file used for remaking the cassette is bad.



## SECTION 26

### STANDALONE/DISK.DUMP

The STANDALONE/DISK.DUMP program is a stand-alone utility program that copies files from a disk cartridge or pack to another disk cartridge or pack of the same or larger capacity. The input disk can be a user disk (type U or R) or a single system disk (type S). A system disk that has not had a CLEAR/START operation performed on it is not allowed as input, nor are any system disks that are part of a multiple system pack configuration. The input disk is copied file-by-file to produce a squashed output disk of the same type as the input disk (S, R, or U).

Multi-pack files are copied only if the input and output disk serial numbers are the same.

Recovery from errors detected during the copy process occurs whenever possible. An irrecoverable error in a disk directory entry or file header causes the associated file to be skipped, but the other files on the disk are copied.

The STANDALONE/DISK.DUMP program can, as an option, compare the copied files to the original files so that comparison errors can be identified.

#### NOTE

When copying a system disk, AB (autobackup) attributes are checked for correctness but not copied. This is because a newly copied disk could be taken to another system where port and channel values for the line printer(s) are different.

## OPERATING INSTRUCTIONS

The STANDALONE/DISK.DUMP program does not operate under MCPII control and must be loaded and executed through the cassette reader in the system console. With dual processor systems, it is required that the operator push the console interrupt switch before attempting to execute the program.

Execute the STANDALONE/DISK.DUMP program in the following manner:

1. Place the STANDALONE/DISK.DUMP cassette in the cassette reader. The BOT light must be on at this time.
2. Place the console keyboard on-line.
3. Set the system MODE pushbutton to the TAPE position, and press the CLEAR and START pushbuttons, respectively. This procedure loads the bootstrap loader from the cassette tape and halts the processor. The L register must be equal to @AAAAA@ at this time.
4. Set the MODE pushbutton to the RUN position and press the START pushbutton. (Do not press the CLEAR pushbutton.) This loads the STANDALONE/DISK.DUMP program.

When the cassette tape has been read, the STANDALONE/DISK.DUMP program begins operation with the following message displayed on the Operator Display Terminal (ODT):

```
STANDALONE/DISK.DUMP MARK <level-number >  
ENTER INPUT DRIVE - <DC? OR DP? >
```

Entering a blank or null input string at this point terminates the program.

B 1000 Systems SSO, Volume 2  
STANDALONE/DISK.DUMP

After a valid response, the input drive <pack-id>, serial number, and type is displayed, followed by the message:

ENTER OUTPUT DRIVE - <DC? OR DP?>

After a valid response, the output drive <pack-id>, serial number, and type is displayed, followed by the message:

TO VERIFY - IS COPY OF <input-drive> TO <output-drive> DESIRED?

A NO response causes the prompt sequence to begin again. A YES response is followed by the message:

IS DATA COMPARISON DESIRED? <YES OR NO>

Data only (no directories or file headers) is compared when this option is requested.

After a valid response, the following message is displayed, and the copy process begins:

DUMP BEGINS

Upon completion without errors, the following message is displayed:

DUMP COMPLETE

At this time, the entire prompt sequence is repeated. Entering a blank input message causes the STANDALONE/DISK.DUMP program to end with the following message:

END STANDALONE/DISK.DUMP

## **ERROR MESSAGES**

### **DISK ERROR - RESULT IN "T"**

The T register contains the error result descriptor. The operator must press the START pushbutton to retry.

### **INVALID RESPONSE - TRY AGAIN**

The input message entered was in error. Enter a valid response in another input message.

### **BAD FILE HEADER <file-id> ADDRESS <disk-address>**

An invalid disk file header was detected on the specified input file. The file is not copied to the output disk, but the dump continues.

### **INSUFFICIENT DISK SPACE FOR <file-id> INVALID DUMP**

The output disk does not have enough space available to contain the specified file. The STANDALONE/DISK.DUMP program must be restarted using an output disk with a larger capacity.

### **NO DISK DEVICE ON SYSTEM**

No disk cartridge or pack control could be located on the system.

B 1000 Systems SSOG, Volume 2  
STANDALONE/DISK.DUMP

<family-id> INVALID SUB DIRECTORY

An invalid subdirectory was detected for the specified <family-id>. No files having that <family-name> can be copied.

<file-id> IS A MULTI PACK FILE - CANNOT COPY

The serial numbers of the input and output packs are not the same; therefore, copying a multi-pack file is not allowed.

<file-name> BAD AREA-ADDRESS <disk-address>

An area of the specified file had irrecoverable errors during the copy process. The copy is completed on the bad area, but the output data needs to be manually verified to insure integrity.

DISK NOT READY <drive>

A not ready exception was detected on the specified disk unit. The operator must insure that the disk is ready, then press the START button to retry the operation.

WRITE LOCKOUT <drive>

A write lockout exception was detected on the specified disk unit. The operator must insure that the disk is write enabled, then press the START button to retry the operation.

DISK NOT PRESENT <drive>

The disk drive specified in the input message could not be located.

BAD PACK LABEL - CANNOT USE <drive>

The label on the specified output pack/cartridge is invalid. The operator must rerun the copy using a pack/cartridge with a valid label.

<file-name> IS NOT AN IAD FILE ON <drive>

This message warns that an Installation Allocated Disk (IAD) file on the input drive is copied to the specified disk drive, but that the output copy is no longer marked as an IAD file.

BAD NAME TABLE - COLDSTARTING

This message indicates that the MCP II Name Table on the input system disk is invalid. The STANDALONE/DISK.DUMP program automatically attempts to perform a disk-to-disk coldstart (as is done by the COLDSTART/DISK program) operation, in order to create a usable output system disk.

COMPARISON ERROR <file-name> ADDRESS <disk-address>

The specified files failed to compare following the copy operation.

B 1000 Systems SSOG, Volume 2  
STANDALONE/DISK.DUMP

<file-id> MISSING OR DUPLICATE FILES

This message can only occur during the coldstart operation caused by an invalid Name Table. It indicates that the operator-specified system file (when the default <file-identifier> was not present) could not be found on the input disk, or had already been designated as another system file.

<drive> CANNOT BE USED ON A <system-id>

One of the SDL Interpreter files was found to be missing during copy of a system disk. This message is a warning that a CLEAR/START operation cannot be performed with the specified disk on the specified type of processor.

PACK LABEL BAD - USER PACK ASSUMED <drive>

The label on the input disk is invalid. The STANDALONE/DISK.DUMP program assumes that the disk type is U and continues the copy operation.

<drive> IS NOT A USABLE SYSTEM DISK

The input disk is part of a multiple system disk configuration, or is a system disk that has not had a CLEAR/START operation performed on it.

## SECTION 27

### SYSTEM/BACKUP

The SYSTEM/BACKUP program is a system utility program that retrieves backup files from disk or tape, and then prints or punches the complete file. Backup files are output files whose device type is specified as printer, punch, or backup when the file is created.

#### OPERATING INSTRUCTIONS

Primary execution of the SYSTEM/BACKUP program is through the PB input message. Refer to volume 1, section 2 for syntax details. The AB (autobackup) input message executes the SYSTEM/BACKUP program automatically. An explicit execute command (for example, EXECUTE SYSTEM/BACKUP) also initiates the program. All three methods of executing the SYSTEM/BACKUP program access the same code file.

An explicit execute must specify a single input file. This file must be file-equated to the file BACK-FILE.

Program switches and run-time control options govern the execution of the SYSTEM/BACKUP program. Run-time control options are entered with the PB input message syntax. The PB syntax can include switch value settings by following the PB input message with a switch statement. Switches are permanently set by modifying the object-code of the SYSTEM/BACKUP program to the desired switch settings.

Examples:

```
PB <backup file>; SWITCH 3 = 1;  
MODIFY SYSTEM/BACKUP SWITCH 3 = 1;
```

#### NOTE

The execution-time options available with the PB input message and the run-time options of the SYSTEM/BACKUP program are not the same. No execution-time control options are allowed when the SYSTEM/BACKUP program is initiated through the autoprint (AB input message) mechanism.

#### PROGRAM SWITCHES

The SYSTEM/BACKUP program recognizes program switches 1, 2, 3, 6, 7 and 8. The default setting for each switch is zero (0). The options are described in table 27-1.

B 1000 Systems SSOG, Volume 2  
SYSTEM/BACKUP

**Table 27-1. SYSTEM/BACKUP Program Switches**

Switch	Value	Meaning
1	0	Carriage control information in the backup file is recognized.
1	1	Single spacing overrides any carriage control in the backup file.
2	0	User-named backup files are not removed after they are printed or punched.
2	1-15	User-named backup files are removed after they are printed or punched.
3	0	System-named backup files are removed after they are printed or punched.
3	1-15	All files are saved.
6	0	A simple page-skip is performed when end of page is reached while printing the file.
6	1-15	The end-of-file action specified for the file is performed when end of page is reached.
7	0	When autoprint (AB) is set, SYSTEM/BACKUP goes to EOJ as soon as all backup files in the autoprint queue are printed.
7	1-14	When autoprint (AB) is set, SYSTEM/BACKUP waits up to five minutes for a new backup file to be released before going to EOJ. The five minutes wait is not cumulative, which means that the autoprint queue must be empty for five consecutive minutes before SYSTEM/BACKUP goes to EOJ.
7	15	When autoprint (AB) is set, the SYSTEM/BACKUP program waits indefinitely for a new backup file to be released.
8	0	Banners are omitted.
	1-15	Banners are printed if the file is labeled and the FORM option is not set. The banner includes the usercode (if present) under which the program was executed, the external file name in the creating program, and the actual file name on disk, if different from the external file name.

## RUN-TIME CONTROL OPTIONS

The run-time control operation specifies or queries the status of the program during its execution with unsolicited accept (AX) input messages entered at the ODT. These control options affect only the file currently being printed and override any options specified in the original execute statement or PB command.

If more than one copy of the SYSTEM/BACKUP program has been initiated by the AB input message, only the copy which accepts the message (determined by the job number in the AX input message) is affected by the run-time option.

Descriptions of the run-time options and the results of their uses follow:

### TEACH or HELP

Displays an explanation of the AX input messages and switch settings.

### WHERE

Displays the number of records written and the end-of-file (EOF) count for the current copy of the file being written.

### COPIES

Specifies how many copies of a backup file remain to be printed or punched.

### SINGLE

Causes single spacing to override any carriage control specified in the backup file.

### DOUBLE

Causes double spacing to override any carriage control specified in the backup file.

### HOSTNAME <host-name>

Causes a backup file to be transferred to a specified host name. If the SAVE option has been specified, the backup file is saved on the receiving host system after transfer. If the NOLIST option has been specified, the backup file is not printed or punched on the receiving host system.

### + or - <integer>

Spaces forward (+) or backward (-) <integer> records in the file. A warning is printed on the printer listing. This option is valid only with printer output.

### SAVE

Does not remove the backup file after it is written.

### RE or REMOVE

Removes the backup file from disk after writing it.

### QT or QUIT

Terminates writing of the current backup file. The remaining backup files are still written.

### QT END or QUIT END

Terminates the SYSTEM/BACKUP program after the current file has finished printing. This overrides the five minute wait option that can be set by program switch 7.

### UPPER

Causes the translation of all lower-case characters to be translated to upper-case characters before printing the line.

## LOWER

Resets the UPPER option to allow the printing of lower-case characters.

## Examples:

```
PB #1234 HOSTNAME = SL8 COPIES = 2
PB #5678 COPIES = 5 SAVE
PB DOCUMENT/UTILITY UPPER DOUBLE (SAVE)
```

## AUTOPRINT FACILITY

The autoprnt facility causes printer backup files to be routed automatically to a line printer as soon as the backup file is released. When autoprnt (AB input message) is active, each line printer assigned to autoprnt (refer to the AB input message in volume 1, section 2 for details) has its own copy of the SYSTEM/BACKUP program executing. If five printers are assigned to autoprnt, then there are five copies of the SYSTEM/BACKUP program in the job mix.

The names of the existing printer backup files are entered into a queue file labeled AUTOPRINT. As each file is processed, its file parameter block (FPB) is copied into the FPB of file OUTFILE. Processing is the same as with a simple PB input message, and the backup files are removed after printing. The autoprnt mechanism can be overridden for a particular backup file at the time the backup file is created, or anytime the file is open, by specifying NO AUTOPRINT.

## ERROR MESSAGES

<file-id> - EOF REACHED, NO RECORDS PROCESSED.

Either the file had no records or the selection options specified could not be satisfied.

<file-id> - FILE NOT PRESENT.

The specified file is not on disk. Removing a backup file before it is printed can cause this error. Also, failing to specify file-security information for secured backup files can cause this error.

## PRINTER/PUNCH BACKUP FILE FORMAT

Backup files produced by the MCPPII are in a fixed record length, fixed blocking factor format. The record size assigned to the backup file is two bytes larger than the record size of the original printer or punch file and one byte larger than the previous backup format. The extra bytes are used for carriage control and stacker select information. The record size for tape backup files is made modulo six bytes by the MCPPII due to the hardware requirements of 7-track and phase-encoded tapes. Backup files on tape cassette are not allowed.

The minimum allowable backup record size that can be written to disk is three bytes. This includes two bytes of control information plus one data byte. The block size of the backup file is computed by the MCPPII so that maximum usage of disk space is accomplished. The SYSTEM/BACKUP program always opens the input backup file with default in order to read it with the proper record size and blocking factor.

The first record of a backup file contains control information for the SYSTEM/BACKUP program. If the backup file record size is less than the size of any of the control records, the MCPPII produces multiple records, dividing the information between the records as necessary.



## CONTROL INFORMATION

The control information begins in the first record of the backup file and consists of the fields (all numbers are in binary) listed in table 27-2.

**Table 27-2. Control Information Record**

<b>Start Pos:Length</b>	<b>Description</b>
[0:24]	Logical record size (in bits) of the backup file.
[24:6]	Increment (in bits) that was added to the record size for tape backup fields. This increment can be from 0 to 40 (always zero for disk files).
[30:18]	Length (in bits) of the control information.
[48:24]	Length (in bits) of the file parameter block (FPB) (refer to the subsection entitled FPB).
[72:24]	Length (in bits) of the label (refer to the subsection entitled Label).
[96:4]	Label type of the backup file. A value of 0001 indicates unlabeled. All other values indicate a labeled file.
[100:4]	Format type of the backup file. A value of 0001 indicates a Mark 9.0 release backup file.

### FPB

A copy of the original printer/punch file parameter block (FPB) begins in the record following the control information.

### Label

A copy of the beginning label for the printer/punch file (even for an unlabeled file) begins in the record following the FPB. The first two bytes of this record contain the appropriate carriage control information (refer to the subsection entitled Carriage Control Field).

### Data Record

The remainder of the file following the label record consists of printer/punch data records. A data record which contains all zeroes is ignored. It is a filler record created by execution of a CLOSE ROLL-OUT operation on the backup file.

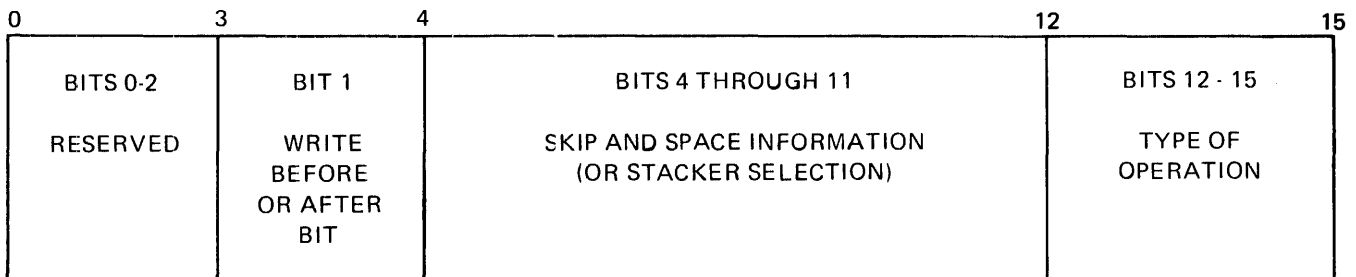
### Carriage Control Field

The first two bytes (16 bits) of each data record comprise the carriage control field, the contents of which specify the spacing, skipping, or stacker selection required. For punch backup files without stacker selection the control field is not used. The low-order four bits of the control field, referred to as TYPE, determine the specific action to be taken. For example, a value of 0011 is TYPE 3. The five field types are shown in table 27-3.

**Table 27-3. TYPE Field Definition**

Type	Function
0	Print the data buffer with channel skipping or single or double spacing as specified in bits 4 through 11 of the 2-byte carriage control field.
1	Punch the data buffer, and route the card to the stacker specified in bits 4 through 11 of the 2-byte carriage control field.
2	Space the number of lines specified by bits 4 through 11 of the 2-byte carriage control field. The data buffer is ignored. A maximum of 255 spaces is allowed.
3	Skip to the channel specified in bits 4 through 11 of the 2-byte carriage control field. The data buffer is ignored.
4	Print the data buffer, and space as specified in bits 4 through 11 of the 2-byte carriage control field. A maximum of 255 spaces is allowed.

The format of the carriage control field is shown in figure 27-1. As previously stated, bits 12 through 15 contain the type of operation. Bits 4 through 11 contain space and skip information for printing or stacker information for punching. Bit 3 is the write before or after bit. When this bit is zero the MCPPII prints before spacing. When this bit is one, MCPPII prints after the spacing. The first three bits (0 through 2) are reserved.



G18246

**Figure 27-1. Format of the Carriage Control Field**

B 1000 Systems SSOG, Volume 2  
SYSTEM/BACKUP

Tables 27-4 and 27-5 display the meaning of the space and skip information (bits 4 through 11) for the various field types.

**Table 27-4. Values for Printer Spacing/Skipping**

Type	Value	Function
0 or 3	@00@	No paper advance
0 or 3	@01@	Skip to channel 1
0 or 3	@02@	Skip to channel 2
-	-	-
-	-	-
-	-	-
0 or 3	@0C@	Skip to channel 12
0 or 3	@0D@	Skip to next channel
0 or 3	@0E@	Space one line
0 or 3	@0F@	Space two lines
2 or 4	@00@-@FF@ (0-255)	Space number of lines indicated (0-255)

**Table 27-5. Values for Punch Stacker-Selection**

Type	Value	Description
1	@00@	Reject Stacker
1	@01@	Stacker 1
1	@02@	Stacker 2
-	-	-
-	-	-
-	-	-
1	@06@	Stacker 6

The following examples show the value of the 16-bit carriage control field for some commonly used operations.

Examples:

Value	Meaning
@0000@	Print data buffer; no paper advance.
@0010@	Print data buffer; then skip to channel one
@1010@	Skip to channel one; then print data buffer.
@00E0@	Print data buffer; then single space.
@10E0@	Single space; then print data buffer.
@0013@	Skip to channel one (no print).
@0052@	Space five lines (no print).
@0012@	Space one line (no print).
@0000@	Punch and stacker-select (reject pocket).
@0041@	Punch and stacker-select (pocket number 4).

## SECTION 28

### SYSTEM/BUILDTRAIN

The SYSTEM/BUILDTRAIN program is a normal-state utility program that creates translation tables for the B 1247-4 Train Printer Control (control-id @3E@). The generated translation tables are contained in a file on the system disk labeled SYSTEM/TRAINABLE. Each installation should create a SYSTEM/TRAINABLE file that contains the character sets used at the installation.

Refer to the LT input message in volume 1, section 2 for further information on specifying translation tables for the train printers.

#### TRANSLATE TABLE FORMAT

Each translation table generated by the SYSTEM/BUILDTRAIN program consists of 256 one-byte elements. Each element contains the hexadecimal value of the link position on the train module for the corresponding graphic. For example, byte 193 (zero-relative) in a translator contains the link position for the graphic A (internal EBCDIC representation @C1@).

For example, the 96-character EBCDIC translator (train-id number 016) contains link position values from @00@ through @60@ in the byte positions corresponding to the internal representations of the graphics on the train module. All other positions in the translator contain a code representing the invalid character.

The invalid character code is represented by a hexadecimal character corresponding to the link position of the graphic to be printed plus @80@. The graphic usually printed as the invalid character is the question mark (?) character. On the 96-character EBCDIC print module, the question mark graphic is located in link position 17 (@11@); therefore, the invalid character code for the 96-character EBCDIC translator is @91@ (@11@ + @80@).

If a link position value is greater than any link existing on the train module, a PRINT CHECK exception condition occurs. Thus, the 96-character EBCDIC print module link positions @61@-@7F@ and @E1@-@FF@ result in a PRINT CHECK exception and should not be specified in the translator.

#### OPERATING INSTRUCTIONS

The SYSTEM/BUILDTRAIN program is executed with a standard EXECUTE statement and is controlled by program switches 1, 3, and 4. By default, the file INPUT/PC5.TABLES is used by SYSTEM/BUILDTRAIN as an input file, and must be on disk when the program is executed unless the operator changes the external identifier of the internal file INPUT with a FILE statement (refer to volume 1, section 2).

## PROGRAM SWITCHES

The SYSTEM/BUILDTRAIN program recognizes the program switches 1, 3, and 4. The function of each switch is described in table 28-1.

Table 28-1. SYSTEM/BUILDTRAIN Program Switches

Switch	Value	Function
1	0	(Default) Creates a new SYSTEM/TRAINABLE file and generates a summary printer listing of the translate tables.
1	1	Generates a summary printer listing of the translate tables, but does not create a new SYSTEM/TRAINABLE file.
3	0	No action.
3	1	Lists the input file on the line printer. Valid only when SW 1 = 0.
4	0	No action.
4	1	Punches the input file as a card deck. Valid only when SW 1 = 0.

## INPUT RECORD FORMAT

When building new translation tables, the SYSTEM/BUILDTRAIN program expects the input file labeled INPUT/PC5.TABLES to be located on the system disk. No additional input specification needs to be entered if the standard tables are to be created. The internal name for this translate table specification file is INPUT.

For each translate table to be included in the SYSTEM/TRAINABLE file, a set of sixteen (16) input records is required as input to the SYSTEM/BUILDTRAIN program. The format of each of these records is described next.

Columns	Contents
1-20	Train name
22-24	Train-id number
26-28	Character set size
30-31	Printer type (00=400/750 LPM, 01=1100/1500 LPM)
33-64	Link positions (16 per record)
66-67	Sequence number (01-16)
70-80	Optional date (format: 01 JAN 1979)

The train name can be any identifier. The train-id number must be the identification generated by the train module for 1100/1500 LPM printers and is always less than 128. For 400/750 LPM printers, the train-id number can be any value that is greater than 127 and less than 256.

@80 = PC5 = 128<sub>10</sub>

PC6 @Cae = 192<sub>10</sub>

Each link position entry (columns 33 through 64) contains two hexadecimal characters that describe the location on the print train module of the graphic representing the internal EBCDIC configuration. Thus, record #01 gives the link positions for EBCDIC characters @00@ through @0F@, record #02 gives the link positions for EBCDIC characters @10@ through @1F@, and so forth.

Link position entries equal to or greater than @80@ are used to specify unprintable characters. If an internal EBCDIC character translates to a link position equal to or greater than @80@, an INVALID CHARACTER exception result descriptor is returned from the print operation. The actual graphic printed is the link position specified minus @80@.

It is possible to print a graphic other than the question mark as the invalid character, by specifying the value of the link position of the desired graphic plus @80@ as the invalid character code. For example, to print the space graphic (which has a link position of @00@) as the invalid character on the 96-character EBCDIC train module, a link position of @80@ (@00@ + @80@) is substituted for every occurrence of the @91@ link position in the input record set.

It is also possible to print a legitimate character in place of the invalid character. This suppresses generation of the exception result descriptor. For example, to print the question mark graphic as the invalid character (and suppress reporting of the INVALID.CHARACTER result descriptor), substitute @11@ for every occurrence of @91@ in the input record set.

If the resulting link position is greater than the number of characters on the train module, a PRINT CHECK exception result descriptor is returned from the print operation.

## STANDARD TRANSLATE TABLES

A set of specifications for the standard printer translate tables is supplied with the SYSTEM/BUILDTRAIN program in a disk file labeled INPUT/PC5.TABLES. The SYSTEM/TRAINTABLE file can be generated from these standard tables directly by executing the SYSTEM/BUILDTRAIN program with no modifications, or individual tables can be modified to suit installation requirements.

The five versions of the 96-character EBCDIC translate table are included as examples of the manner in which specific tables can be generated and tailored to individual installation requirements. EBCDIC96 is the standard 96-character EBCDIC translator, having both upper and lower case graphics. It prints the question mark (?) graphic for the invalid character. EBCDIC96.UPPER.CASE and EBCDIC96.LOWER.CASE also print the question mark graphic for the invalid character; however, EBCDIC96.UPPER.CASE option prints all alphabetic characters in upper case and EBCDIC96.LOWER.CASE option prints all alphabetic characters in lower case. EBCDIC96.UPPER.CASEB and EBCDIC96.LOWER.CASEB options function in a similar manner; however, they both print the space graphic (" ") for the invalid character.

Multiple translation tables with the same train-id number (but unique train names) can be contained in the same SYSTEM/TRAINTABLE file. The most commonly used version of the translation table should be the first one specified in the input file because it is the table loaded automatically by the MCPPI when the printer first becomes ready following a CLEAR/START operation; it is also the table loaded when the train-id number is specified in the LT input message (refer to the LT input message in volume 1, section 2), as shown in the following example:

```
LT LPA 016
```

A specific version of such multiple translation tables can be designated by using the train name in the LT input message:

```
LT LPA EBCDIC96.UPPER.CASEB
```

B 1000 Systems SSOG, Volume 2  
SYSTEM/BUILDTRAIN

Such a translation table remains loaded until the next CLEAR/START operation or until explicitly changed by another LT input message.

NOTE

It is not possible to designate a translate table for the 1100/1500 LPM printers where the train-id number does not match the identification number contained in the train module.

The 400/750 LPM train printers do not have automatic train identification. The proper translation table must be explicitly specified with an LT input message.

The standard printer translation tables supplied are named in tables 28-2 and 28-3.

**Table 28-2. 1100/1500 LPM Train Printer**

Train Name	Train-id Number	Train-id Description
EBCDIC18	001	18-character EBCDIC
FORTRAN48.NONSTD	002	48-character FORTRAN
B300.B50048	003	48-character B300/B500
EBCDIC48	004	48-character EBCDIC
EBCDIC72	005	72-character EBCDIC
UKB3500.72	006	72-character EBCDIC (UK B3500)
UKB6500.72	007	72-character EBCDIC (UK B6500)
PORTUGAL.72	008	72-character EBCDIC (Latin/Portugal)
SPAIN.72	009	72-character EBCDIC (Latin/Spain)
FINLAND.72	010	72-character EBCDIC (Finland)
DENMARK.72	011	72-character EBCDIC (Denmark/OCR-B)
BCL72	012	72-character BCL
TURKEY.72	013	72-character EBCDIC (Turkey)
SWEDEN.72	014	72-character EBCDIC (Sweden)
ASCII72	015	72-character ASCII
EBCDIC96	016	96-character EBCDIC
EBCDIC96.UPPER.CASE	016	96-character EBCDIC
EBCDIC96.UPPER.CASEB	016	96-character EBCDIC
EBCDIC96.LOWER.CASE	016	96-character EBCDIC
EBCDIC96.LOWER.CASEB	016	96-character EBCDIC
KATAKANA	017	96-character KATAKANA (Japan)
EBCDIC.A72	018	72-character Alphabetized EBCDIC
EBCDIC.N72	019	72-character Numericized EBCDIC
RPG48	020	48-character RPGII
OCR.A72	021	72-character OCR-A
OCR.B72	022	72-character OCR-B
FORTRAN48	036	48-character FORTRAN
THAI144	052	144-character EBCDIC (Thailand)

B 1000 Systems SSOG, Volume 2  
SYSTEM/BUILDTRAIN

**Table 28-3. 400/750 LPM Train Printer**

<b>Train Name</b>	<b>Train-id Number</b>	<b>Train-id Description</b>
FORTRAN48	130	48-character FORTRAN
FORTRAN48NONSTD	130	48-character FORTRAN non-standard
B300/B500.48	131	48-character B300/B500
EBCDIC3.48	132	48-character EBCDIC-3
RPG48	140	48-character RPGII
EBCDIC96	144	96-character EBCDIC
KATAKANA	145	96-character KATAKANA (Japan)
EBCDIC3.16	254	16-character EBCDIC-3
EBCDIC3.64	255	64-character EBCDIC-3



## SECTION 29

### SYSTEM/COMPARE

The SYSTEM/COMPARE program is a normal-state utility program that compares pairs of files that exist on cards, tape, or disk. This program prints and identifies by record number all non-matching record pairs. Such non-identical records are printed in EBCDIC and hexadecimal representations with all differing four-bit hexadecimal digits flagged with an asterisk.

The files to be compared can be on any combination of disk, card, or tape media. Should both files be on disk, a special mode of operation can be selected wherein both files are compared sector-by-sector rather than record-by-record, allowing maximum speed.

Other features include automatic termination of comparison after encountering a specified number of errors, interrogation of the number of records currently processed, and explicit termination of a comparison operation at any time.

If the files being compared have differing record lengths, the SYSTEM/COMPARE program emits a warning message and pads the shorter records with blanks.

### OPERATING INSTRUCTIONS

Input to the SYSTEM/COMPARE program can be provided in either of two ways. An operator desiring to compare only one pair of files can execute SYSTEM/COMPARE and file-equate the desired files to internal file identifiers A and B. The specified comparison is performed and SYSTEM/COMPARE goes to end of job (EOJ).

Example:

```
EXECUTE SYSTEM/COMPARE FILE A NAME=COBOL;FILE B NAME  
USERPACK/COBOL/;
```

Use of the SYSTEM/COMPARE program in this single compare mode precludes the ability to use the sector-by-sector mode of operation or to be able to specify the maximum number of comparison errors before termination (a default of 15 is used).

If files A and B are not changed by the NAME file attribute of the FILE statement, the SYSTEM/COMPARE program expects commands to be entered by means of accept (AX) input messages. Input is requested after each pair of files has been processed until the program is terminated by a blank accept (AX) input message. The input syntax is similar to that of the DMPALL program to maximize ease of use.

## PROGRAM SWITCHES

The SYSTEM/COMPARE program recognizes the program switches 0 and 9. Switch 0 specifies the number of characters on the printer chain, and switch 9 changes result-information from zero-relative to one-relative quantities.

Switch 0 specifies the number of characters on the printer train. This affects the EBCDIC error listing because characters not printable based on this switch setting are translated to blanks. Users with one printer or multiple printers with the same character set will find it convenient to permanently assign the appropriate value to switch 0, based on the following:

Switch 0 Value	Character Set
0	48-character (default)
1	64-character
2	72-character
3	96-character

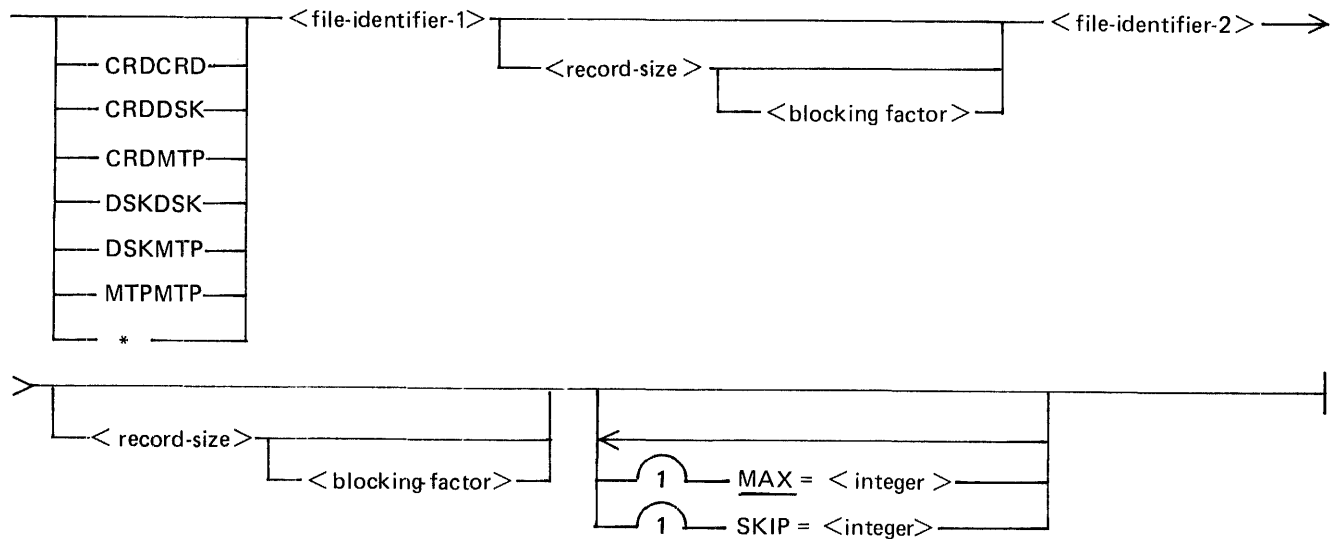
If the value of switch 0 exceeds 3, a 48-character set is assumed.

By default, the SYSTEM/COMPARE program expresses results as zero-relative quantities, meaning that the first bit of the first record is defined as bit zero of record zero. However, if switch 9 is set to 1, quantities are expressed in a one-relative manner; that is, the first bit of the first record is defined as bit one of record one. The appropriate origin is printed as part of the error listing heading.

## COMMAND SPECIFICATIONS

Commands must conform to the following syntax:

Syntax:



The shorthand notation for the routine type uses the following abbreviations: DSK (Disk), CRD (Card), or MTP (Magnetic Tape/Cassette). If the routine-type specification is omitted, DSKDSK is assumed. The asterisk is used to specify comparison of two disk files sector-by-sector. A warning message is emitted if a record size or blocking factor is specified for either file in the sector-by-sector mode.

B 1000 Systems SSOG, Volume 2  
SYSTEM/COMPARE

The file-identifiers must be valid file names such as ANNUAL, (CRUCIAL)/DOCUMENT, or USER-PACK/"First Try"/TABULATION. A file identifier whose first component is all numeric, such as 1234/TEST, or is equivalent to a <routine-type> (when none is specified), must be enclosed in quotes to distinguish it as a <file-identifier>. For example, the preceding file-id must be specified as "1234"/TEST.

Both <record-size> and <blocking-factor> are integer values. If a <blocking-factor> is specified, a <record-size> must also be specified to avoid ambiguity. The <blocking-factor> is normally used only for unlabeled tapes or for blocked card files.

The integer appearing after the keyword MAX specifies the maximum number of errors to list before terminating the current file comparison. The default value is 15. Any value specified must be non-zero.

The integer appearing after the keyword SKIP causes the SYSTEM/COMPARE program to space over the first <integer> records or sectors in each file before beginning the file comparison.

## SAMPLE EXECUTION STRINGS

MTPDSK SALES/AUGUST MASTERFILE

\* (SOURCE)/QWIKLOG BACKUPCOPY/QWKLOGSRCE

MTPMTP UL\_1 80 9 UL\_2 80 4 MAX=5 SKIP 150

DSKDSK FIRST/USER BACKUP/COPY MAX=1

FORTTRAN RELEASE/FORTTRAN/SKIP=1

## RUN-TIME INTERROGATION

Console keyboard requests for additional comparisons can be provided at any time; commands are tanked if a comparison is currently in progress.

The following run-time interrogation requests can also be provided at any time during execution of the SYSTEM/COMPARE program; they are not tanked, but are processed immediately, whether or not additional comparison specifications have been provided.

ST or STATUS	Displays current record and error count.
QUIT or ABORT	Terminates current comparison.
CLEAR or CLLP	Closes the printer file LINE to free it for other users.
TRAIN or PRINTER [=]	Displays the value, or changes that value dynamically, of the character set to which
[(48)]	mismatched records are translated.
[(64)]	Characters not appearing on the specified
[(72)]	print train are translated to blanks.
[(96)]	

The occurrence of a parity error on either record of a pair being compared causes an informative message to be emitted on the printer file and the comparison of that pair of records to be suppressed.  
1138542

## OUTPUT MESSAGES

The SYSTEM/COMPARE program provides a comprehensive set of error, warning, and informative messages. Error messages indicate invalid operator input and generally include the syntactic token which evoked the error message.

### Fatal Errors

Fatal errors represent irrecoverable problems with the execution environment of the SYSTEM/COMPARE program and cause termination of the program. The fatal error messages and related information follow.

INSUFFICIENT DYNAMIC MEMORY TO EXECUTE SYSTEM/COMPARE;

RERUN WITH ME GEQ 4320 BITS

The SYSTEM/COMPARE program makes substantial use of dynamic memory; 4320 bits are necessary as a minimum value. However, there is no increase in efficiency if it is run with a higher value unless the program specifically requests a greater amount.

SOFTWARE INCOMPATIBILITY: <level> SYSTEM/COMPARE; <level>

MCP...EOJ

The SYSTEM/COMPARE program is concerned with certain MCP II structures that may change between software releases. In order to ensure correct execution, the SYSTEM/COMPARE program verifies that it is being run under control of the proper MCP II and terminates with this message in case of a mismatch.

SYSTEM ERROR: VARIABLE RANDOM FILES - DS OR DP.

Take a program dump and submit the dump with KA listings of the two input files to a Burroughs technical representative.

### Errors

The following error messages identify incorrect operator input and cause the SYSTEM/COMPARE program to ignore that input message.

"48", "64", "72", OR "96" EXPECTED HERE

The value specified for the PRINTER (or TRAIN) option is not one of the four valid integers.

Examples:

TRAIN = 56

PRINTER = 74

INSUFFICIENT DYNAMIC MEMORY TO COMPARE <file name> TO

<file name>; RERUN WITH ME GEQ <integer> BITS.

The SYSTEM/COMPARE program builds its input buffers in dynamic memory; comparison of files with unusually large record sizes results in a need for more than the default 4320 bits. The value needed to perform a given comparison is 16 times the larger record size in bytes. The default value thus suffices to compare files up to 270 bytes (4320 divided by 16) in record length. Users who have frequent need to compare files with longer records are advised to modify the SYSTEM/COMPARE program to the minimum appropriate value for dynamic memory using the MEMORY program attribute.

B 1000 Systems SSOG, Volume 2  
SYSTEM/COMPARE

INTEGER EXPECTED FOR <error limit>

Following the keyword MAX, and the optional equal sign, the SYSTEM/COMPARE program scanned a non-numeric token. Example:

CRDCRD PUNCHED INPUT MAX=TEN

INTEGER EXPECTED FOR <skip value>

Following the keyword SKIP, and the optional equal sign, the SYSTEM/COMPARE program scanned a non-numeric token. Example:

\* WITH.INTRO DOCUMENT SKIP MIX=5

INTEGER EXPECTED FOR TRANSLATION ID

A non-numeric token was scanned after the keyword PRINTER (or TRAIN), aside from the optional equal sign (=) character. Example:

TRAIN = EBCDIC

<file-id> IS AN EMPTY DISK FILE

The SYSTEM/COMPARE program does not process a disk file having an end-of-file (EOF) record of zero.

"MAX", "SKIP", OR <EMPTY> EXPECTED HERE

The SYSTEM/COMPARE program parsed the second file name, and a record size and blocking factor if present, but then scanned an invalid keyword. Example:

MTPMTP DATA BACKUP 45 8 MSX=1

MISSING FILE ID'S

No input beyond the <routine type> was provided. Example:

DSKDSK

MISSING FINAL QUOTE IN FILE NAME

An unmatched pair of quotation marks was found, which presumably was intended to delimit a <file identifier> with special characters. Example:

DSKMTP PASSBOOK/ACCOUNTS "5%INTEREST MAX=3

MISSING 2ND FILE ID

The input command specifications contained only one <file identifier>. Example:

\* ONLY/ONE

NAME > 10 CHARACTERS

The MCP II permits a maximum of 10 characters in each component of a <file identifier>; the SYSTEM/COMPARE program enforces this limit. Example:

CRDDSK BANKBOOK PASSBOOK\_\_SAVINGS/ACCOUNT SKIP 10 MAX=1

NAME CAN'T START WITH SLASH

A <file identifier> erroneously started with the slash (/) character. Example:

CRDDSK CARDDECK /DISK/COMPARISON

B 1000 Systems SSOG, Volume 2  
SYSTEM/COMPARE

**NO DISK FILE <file-id>**

The disk file specified was not present. Note that this message terminates a comparison attempt for a disk file only. A NO FILE condition on card or tape causes the MCPPII to request the appropriate file.

**NON-CONGRUENT AREAS ON RANDOM FILES**

This message can appear only for random disk files, meaning files with interspersed empty areas. Either the records per area value differed for the two files, or the pattern of allocated and unallocated areas did not match.

**RANDOM DISK FILE <file-id> CAN BE COMPARED ONLY TO ANOTHER  
RANDOM DISK FILE**

It is meaningless to attempt to compare a disk file having interspersed unused areas to a serial type file, disk or otherwise.

**VARIABLE-RECORD DISK FILE <file-id> CAN BE COMPARED ONLY  
TO ANOTHER VARIABLE-RECORD DISK FILE**

The specialized format of variable records makes it illogical to attempt comparison to a file with fixed-length records.

**Warning Messages**

The following warning messages alert the operator to an action or decision by the SYSTEM/COMPARE program.

**COMMAND QUEUE FULL...SPECS IGNORED**

More than 10 commands queued while a comparison is in progress. In the unlikely event that this tank size is insufficient, the file TANK can be file-equated or modified to a higher value with the QUEUE.MAX.MESSAGES file attribute. Example:

EXECUTE SYSTEM/COMPARE FILE TANK QMX = 15;

**EXTRANEIOUS GARBAGE IGNORED**

Unrecognized input occurred in a valid query (non-command) accept (AX) input message.

Example:

STATUS PLEASE

**PREVIOUS SKIP/MAX CLAUSE IGNORED**

A duplicate SKIP or MAX clause was scanned. The SYSTEM/COMPARE program uses the last clause entered. Example:

\* STATISTICS CURRENT/(SYSTEM)/DATA MAX 5 SKIP 400 MAX = 3

**RECORD AND/OR BLOCK SIZE SPECS IGNORED IN SECTOR MODE**

B 1000 Systems SSOG, Volume 2  
SYSTEM/COMPARE

When processing disk files sector-by-sector, the actual record and block sizes of those files are superfluous and irrelevant. Example:

\* FIRST/FILE 50 3 SECOND/FILE/COMPARED

"SKIP" SPEC IGNORED FOR RANDOM DISK FILES

The meaning of skipping records in a file with empty disk areas cannot be satisfactory to all users. The SYSTEM/COMPARE program avoids the problem by not using a SKIP specification.

UNEQUAL RECORD SIZES: <file-identifier-A> = <record-size-A>;

<file-identifier-B> = <record-size-B>

The files to be compared had unequal record sizes. The SYSTEM/COMPARE program pads the shorter records with blanks.

VALUE OF ZERO IGNORED IN (MAX/SKIP) CLAUSE

The integer provided with these two keywords must be positive; a SKIP of zero records is the default, whereas a maximum error count of zero has no logical validity.

VARIABLE RECORD DISK FILES WILL BE COMPARED IN SECTOR MODE

To eliminate special handling of variable-length record disk files, the SYSTEM/COMPARE program compares them sector-by-sector, whether or not an asterisk (\*) is specified as the <routine type>.

### Informative Messages

<integer> CHARACTER TRANSLATION ENABLED

This message results from either a query or dynamic translation change using the PRINTER (or TRAIN) command. Examples:

TRAIN

% SYSTEM.COMPARE =3744 48 CHARACTER TRANSLATION ENABLED

TRAIN 72

% SYSTEM/COMPARE =3744 72 CHARACTER TRANSLATION ENABLED

COMPARISON TERMINATED

This message is the acknowledgement of a QUIT (or ABORT) request.

EOF ON <file-id>

The two files being compared had differing numbers of records. The SYSTEM/COMPARE program terminates the comparison when EOF on the shorter file is reached.

\*\*\* ERROR LIMIT REACHED \*\*\*

The specified (or default) number of comparison errors occurred before EOF was reached.

B 1000 Systems SSOG, Volume 2  
SYSTEM/COMPARE

(<integer>/NO) ERRORS IN <integer> (RECORDS/SECTORS)

The results of each comparison are displayed in this message. This is also the response to a STATUS (or ST) query. Example:

NO ERRORS IN 814 RECORDS

NO COMPARISON IN PROGRESS

ST, STATUS, QUIT, or ABORT was entered while the SYSTEM/COMPARE program was waiting for operator input with no comparison being performed.

"SKIP" SPEC POSITIONED (<file-id>/BOTH FILES) PAST EOF

The given SKIP specification exceeded the number of records or sectors (as appropriate) in one or both files being compared.

SPECS

The SYSTEM/COMPARE program is idling, waiting for operator input.



## SECTION 30 SYSTEM/COPY

The SYSTEM/COPY program is a generalized library maintenance program that copies disk files and library (COPY) tape files and prints the directory of library (COPY) tapes. The SYSTEM/COPY program produces Burroughs-standard library tapes, referred to as COPY tapes.

### NOTE

SYSTEM/COPY tapes are not compatible with SYSTEM/LOAD.DUMP tapes.

The SYSTEM/COPY program has operator-interface error handling capabilities which specify recovery options when I/O errors occur. The SYSTEM/COPY program also notifies the operator when the COPY command is not completed as specified. When fatal syntax errors are encountered in the command specification string, the program issues appropriate error messages and terminates without copying any files.

Program switches allow the operator to control message routing, duplicate file checking, and the use of special characters by the SYSTEM/COPY program.

## OPERATING INSTRUCTIONS

The SYSTEM/COPY program can add or copy files, or print the directory of a COPY library tape. ADD, COPY, and DIR are MCPPII library maintenance instructions which implicitly invoke the execution of the SYSTEM/COPY program. The syntax of each of these commands is described in volume 1, section 2 and in the following pages.

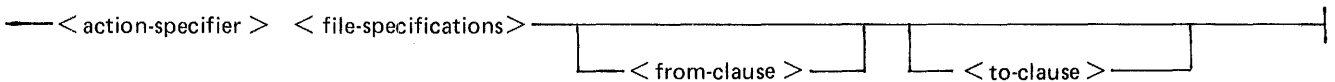
## SYNTAX OF COMMAND SPECIFICATION

The syntax, semantics, and examples of actual command specifications are documented in this section, along with a description of the specification format. The valid commands ADD, COPY, and DIR are presented in alphabetical order.

### Syntax Overview

The following syntax diagram is an overview of the SYSTEM/COPY command syntax for the ADD and COPY commands.

Syntax:



# COPY & COMPARE

B 1000 Systems SSOG, Volume 2  
SYSTEM/COPY

is  
obsolescent?

Semantics:

< action-specifier >

Refer to the Action Specifier in this section.

< file-specifications >

Refer to File Specification Clause in this section.

< from-clause >

Refer to From Clause in this section.

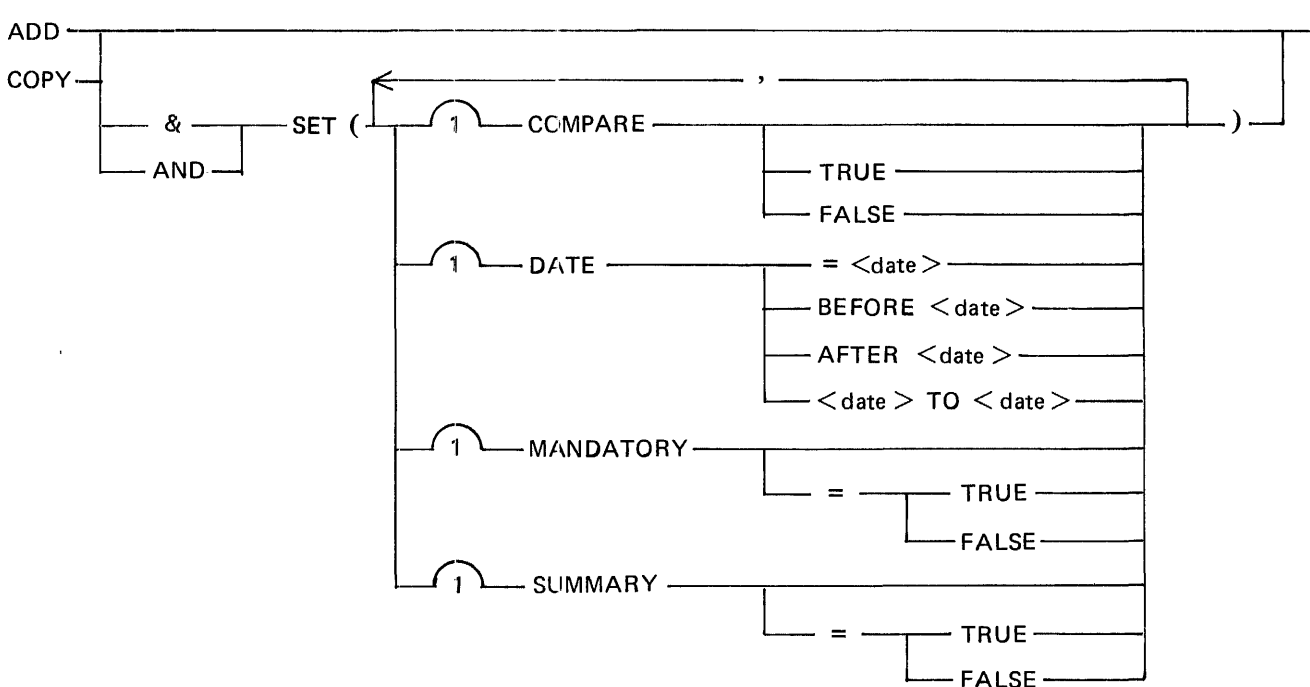
< to-clause >

Refer to To Clause in this section.

## Action Specifier

This syntax is valid only for the ADD and COPY commands.

Syntax:



## ADD

The ADD command directs the SYSTEM/COPY program to copy the named files to disk only if a file by the same name does not already exist on the disk. If several files are named in a single ADD command, and a duplicate file error occurs with one file, all files except the file causing the duplicate file condition are copied.

Example:

```
ADD FILE1 TO YRDISK(KIND=DISK);
```

## COPY

The COPY command directs the SYSTEM/COPY program to copy the named files in the same order as they are specified in the COPY statement, except that all copies from tape are done in the order that the files appear on the tape. If a duplicate file condition occurs during a copy to disk, the SYSTEM/COPY program removes the old file and replaces it with the new file. The new file is referenced by the disk directory entry.

Examples:

```
COPY FILE1 FROM MYDISK(KIND=DISK);  
COPY FILE2 TO FROM MYTAPE(KIND=TAPE) TO DISK(KIND=DISK);
```

## Options List

The operator specifies task options by using an options list with the ADD command. The options list, when used, must follow the keyword specification and precede the list of files to be copied. The options list must be enclosed in parentheses, and all option specifications must be separated by commas. Only one occurrence of each option is allowed.

### COMPARE Option

The COMPARE option causes the SYSTEM/COPY program to verify that files are copied correctly. When performing the data comparison on tape files, the SYSTEM/COPY program makes efficient use of the system's ability to backspace tapes. Both disk and tape files are verified by reading the data and the header of the original file and of the copied file and comparing them. Any comparison errors are noted in a library maintenance message. Files are compared immediately after they are written.

The COMPARE function of the SYSTEM/COPY program can be used independently of the COPY function. When used this way, the COMPARE command syntax is similar to the COPY command syntax except for the restriction that only one input and one output unit can be specified. For example, multiple COMPARE statements are required to verify the results of a COPY from more than one tape to a disk. The SYSTEM/COPY program produces a listing of the file compared, flagging any errors.

Examples:

```
ADD & SET(COMPARE) SYS/= TO MYDISK(KIND=DISK);  
COPY & SET(COMPARE) SYS/= TO MYTAPE(KIND=TAPE);  
COMPARE SYS/= TO MYDISK(KIND=DISK);  
COMPARE SOURCE/= FROM TAPE1(KIND=TAPE) TO DISK1(KIND=DISK);
```

To compare tape files that cross a reel boundary, the COMPARE statement must be used independently of the copy function. If the input file originates on other than the first reel of the multireel file, the attribute VOLUMEINDEX must be specified. The attribute VOLUMEINDEX specifies which reel of an input tape should be used in a copy or compare function.

Example:

```
COMPARE FILE3 FROM MYTAPE(KIND=TAPE,VOLUMEINDEX=3)  
TO MYDISK(KIND=DISK)
```

### DATE Option

Specifying the DATE option causes only those files whose update dates fall within the specified range to be copied or compared. If the update date of the file is zero (the last update date occurred prior to the Mark 9.0 release), then the creation date of the file is used instead of the update date. The SYSTEM/COPY program checks each file specified in the file list and ignores all files that do not match the date selection criteria. Dates must be in the form mm/dd/yy. Values for the DATE option follow.

Value	Meaning
<date>	Only files updated on the specified date are selected.
BEFORE <date>	Only files updated before the specified date are selected.
AFTER <date>	Only files updated after the specified date are selected.
<date1> TO <date2>	Only files whose update dates fall within the specified range are selected. The range is inclusive of the starting and stopping dates.

#### Examples:

```
ADD & SET(DATE BEFORE 6/1/81) SYS/= TO MYDISK(KIND=DISK);  
ADD & SET(COMPARE, DATE 02/10/82 TO 03/10/82) SYS/=  
FROM MYTAPE(KIND=TAPE) TO YRDISK(KIND=DISK);  
COPY & SET(DATE BEFORE 6/1/81) SYS/= TO MYTAPE(KIND=TAPE);  
COPY & SET(COMPARE, DATE 02/10/82 TO 03/10/82) SYS/=  
TO YRDISK(KIND=DISK);
```

### MANDATORY Option

The MANDATORY option controls the action to be taken when a disk input file cannot be found. By default, the SYSTEM/COPY program generates an error message and continues with the next copy request when a missing file is encountered. However, if the MANDATORY option is set, operator intervention is required to correct all missing file conditions except those encountered after the SYSTEM/COPY program has started to copy a file. If a file name is listed which contains an equal sign (=), the SYSTEM/COPY program requires that at least one file matches that name.

~~The MANDATORY option has no effect on missing tape files.~~ Regardless of the setting of this option, the SYSTEM/COPY program always hangs and waits for the directory file for each input tape. All files not included in the tape directory are reported not found.

#### Examples:

```
ADD AND SET(MANDATORY) FILEA, FILEB, FILEC FROM DISK1(KIND=DISK)  
TO DISK2(KIND=TAPE);  
COPY AND SET(MANDATORY) FILEA, FILEB, FILEC FROM DISK1(KIND=DISK)  
TO TAPE1(KIND=TAPE);  
COPY & SET(MANDATORY) SYS/= TO MYDISK(KIND=DISK);
```

### SUMMARY Option

The SUMMARY option produces a listing of the copy messages and the results of the program run.

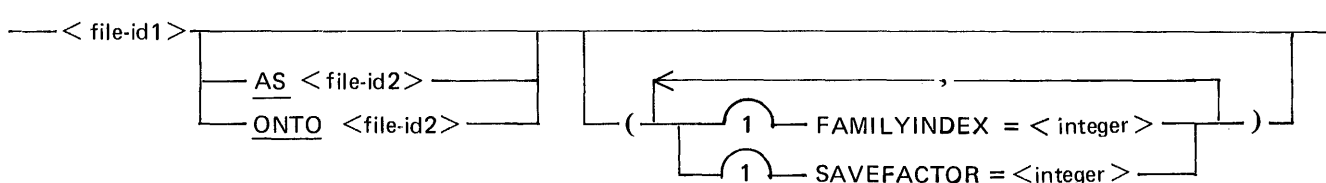
Examples:

```
ADD AND SET(SUMMARY) SYS/= TO DISK1(KIND=DISK);
COPY AND SET(SUMMARY) SYS/= TO DISK1(KIND=DISK);
```

### File Specification Clause

The file specification clause specifies file attributes, renames a file, or copies the file onto an Installation Allocated Disk (IAD) file. The syntax for the File Specification clause follows.

Syntax:



### <file-id>

The <file-id> must be a valid file identifier. The volume (user disk or tape name), if needed, is always specified in a FROM clause, and must not be included as part of the file-id. Program switch 2 must equal 0 for tape file-ids to allow special characters.

The format of disk file identifiers is the same as that recognized by the MCP II. Disk file identifiers consist of one or two 10-character fields, separated by a slash (/) character. The equal sign (=) character is the last identifier.

The file identifiers on tape must conform to the Burroughs standard format. Valid tape file-ids consist of one to thirteen 17-character fields separated by slash (/) characters, and allow an equal sign (=) character as the last identifier. However, when copying tape files to disk, the file-id must conform to the disk file identifier format. If the tape file-id does not meet this restriction, the AS option of the file specification clause must be used, or the copy operation cannot be performed.

### AS

The AS option renames the file, giving the newly copied file the name specified by the <file-id> following the keyword AS.

### ONTO

The ONTO option directs the SYSTEM/COPY program to copy the file to the absolute location assigned by the IAD file specified by the <file-id>. If the <file-id> does not name an IAD file, the SYSTEM/COPY program issues an advisory warning message and does not perform the copy. The IAD file and the file being written to it must have exactly the same file specifications or the copy is disallowed. The name of the IAD file remains unchanged, but its file type, EOF pointer, and creation date are updated to correspond with the new data in the file.

**FAMILYINDEX**

The FAMILYINDEX attribute specifies the electronic unit (device number) on which the file is located, and refers to head-per-track disk devices. An integer value must be specified; mnemonics cannot be used.

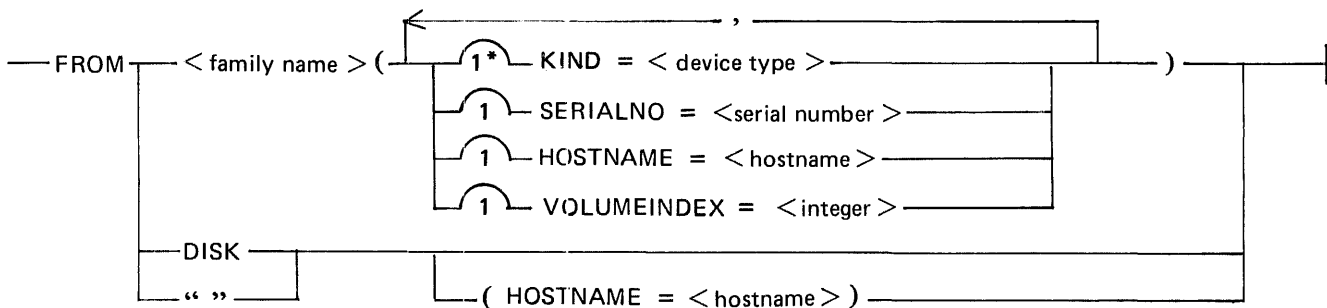
**SAVEFACTOR**

The SAVEFACTOR attribute specifies the value of the save factor field in the disk file header of the output file. Expired files, those files whose last access date was prior to the save factor, are marked with an asterisk but not removed on B 1000 systems. Expired files can be copied to other systems.

**FROM Clause**

The FROM clause of the copy specification string is optional and is used to specify the volume the SYSTEM/COPY program is to use as its input source. The syntax for the FROM clause follows.

Syntax:



Semantics:

The FROM clause applies to all file identifiers that precede the clause but are not separated by a TO clause or by another FROM clause. Only one FROM clause per file is allowed. A maximum of 32 FROM clauses are allowed in a single COPY statement.

When the FROM clause is omitted, the SYSTEM/COPY program assigns a default source based on file-security options. If the file referenced is a non-secured file, the SYSTEM/COPY program uses the system disk as the default input volume. If the file referenced is a secured file, the SYSTEM/COPY program uses the user-disk assigned to the usercode as the default input volume. An implicit FROM DISK (DISK specifies the system disk) is never assumed for a file that either is non-secured and within the scope of another FROM clause, or for a secured file that has a default pack associated with it.

Examples that use FROM clauses:

COPY A FROM TAPE1(KIND = TAPE)

Copies file A from TAPE1 to the system disk.

COPY A, B, C FROM TAPE1(KIND = TAPE)

Copies files A, B, and C from TAPE1 to the system disk.

COPY A AS B FROM TAPE1(KIND = TAPE)

Copies file A from TAPE1 to the system disk, and names the new file B.

COPY A FROM TAPE1(KIND = TAPE), B FROM TAPE2(KIND = TAPE)

TO TAPE3(KIND = TAPE)

Copies file A from TAPE1 to TAPE3 (not the system disk); then copies file B from TAPE2 to TAPE3.

<family-name>

The <family-name> must be a valid single-name file-identifier that is 10 characters or less. Disk volume names can contain special characters if the name is enclosed in quotation marks; tape volume names can never contain special characters (characters other than A through Z and 0 through 9).

**DISK and " "**

Both DISK and " " specify the system disk.

#### **SERIALNO Attribute**

The <serial number> can be specified for each device. The value of <serial number> must be an alphanumeric string whose length is six characters or less. This attribute does not direct the output to a specific unit; it provides further identification of a tape to the SYSTEM/COPY program when tape files with the same name are used. The serial number assigned to a tape with an SN input message (refer to volume 1, section 2), and the SERIALNO attribute of the SYSTEM/COPY program are distinct and unrelated values. The <serial number> used by the SYSTEM/COPY program defaults to zero when this attribute is not specified.

#### **KIND Attribute**

The SYSTEM/COPY program uses the default (KIND = DISK) when (1) the program is executed with switch 9 = 1, (2) the hardware attribute KIND is omitted in all references to a volume, (3) and a volume other than the system disk is specified. Otherwise, the hardware attribute KIND must be specified for each volume.

The value of <device-type> must be one of the following mnemonics:

<device type>	Meaning
DISK	
TAPE	(any tape)
TAPEPE	(any 9-track PE tape)
TAPE7	(any 7-track tape)
TAPE9	(any 9-track tape)
	(NRZ if no type specifically requested)
TAPEGCR	(any group-coded-recognition magnetic tape)

**HOSTNAME Attribute**

The HOSTNAME attribute specifies the system on which a volume is to be found on a Burroughs Network Architecture (BNA) network.

**VOLUMEINDEX Attribute**

The VOLUMEINDEX attribute specifies which reel of a multivolume input tape is to be used for the copy. The SYSTEM/COPY program skips previous reels and searches in the directory of the specified reel. However, if the tape directory crosses a reel boundary, files on the continuation reel cannot be accessed by specifying VOLUMEINDEX = <integer>, where <integer> is the number of the continuation reel. The reel on which the directory originated must be specified.

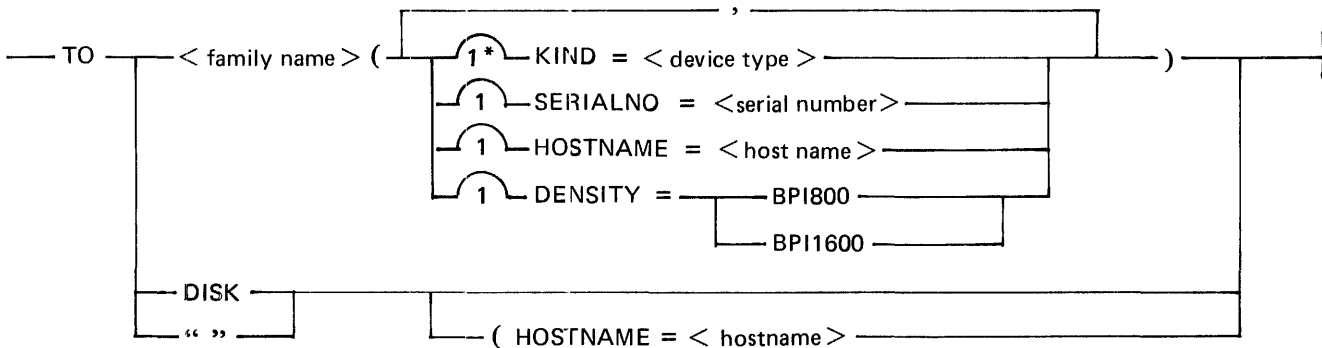
Examples:

```
COPY A TO DISK1(KIND=DISK), TO TAPE1(KIND=TAPE, SERIALNO=12);
COPY B FROM DISK2(KIND=DISK, HOSTNAME=LIB) TO TAPE3(KIND=TAPE);
COPY C FROM DISK TO TAPE2(KIND=TAPEPE);
COPY D FROM TAPE4(KIND=TAPE, VOLUMEINDEX=4);
```

**TO Clause**

The TO clause of the copy specification string is optional, and it specifies the volume that the SYSTEM/COPY program is to use for output. The syntax for the TO clause follows.

Syntax:



**Semantics:**

TO clauses are allowed only at the end of the COPY statement and apply to all files specified in the COPY statement. A maximum of eight TO clauses can be specified for each file. When more than one TO clause is associated with a file, the TO clauses must be adjacent; embedded file-identifiers change the reference of the TO clauses.

The SYSTEM/COPY program uses the system disk as the default output volume when the TO clause is omitted. Files that are within the scope of an explicit TO clause are never assigned an implicit TO DISK (DISK specifies the system disk) clause.



Examples using TO clauses:

**COPY A**

Invalid because the source and destination fields are the same.

**COPY A AS B**

Copies file A from the system disk to file B on the system disk.

**COPY A,B FROM TAPE1(KIND = TAPE)**

Copies files A and B from TAPE1 to the system disk.

**COPY A AS B TO DISK2(KIND = DISK), TO TAPE1(KIND = TAPE)**

Copies file A from the system disk to file B on DISK2, and to file B on TAPE1.

**COPY A TO TAPE1(KIND = TAPE)**

Copies file A from the system disk to file A on TAPE1; there is no attempt to copy file A to the system disk.

**COPY A TO TAPE1(KIND = TAPE), B TO TAPE2(KIND = TAPE)**

Invalid because multiple copy requests are not allowed. The correct syntax is:

**COPY A TO TAPE1(KIND = TAPE)**

**COPY B TO TAPE2(KIND = TAPE)**

Two separate SYSTEM/COPY jobs are started. The first copies file A from the system disk to TAPE1. The second copies file B from the system disk to TAPE2.

**COPY A TO TAPE1(KIND = TAPE), TO TAPE1(KIND = TAPE)**

Creates two output tapes labeled "TAPE1", each containing file A.

**COPY A TO DISK1(KIND = DISK), TO DISK2(KIND = DISK)**

Copies file A from the system disk to DISK1, with a second copy overwriting the first.

**<family-name>**

The <family-name> must be a valid single-name file-identifier that is 10 characters or less. Disk volume names can contain special characters if the name is enclosed in quotation mark (") characters; tape volume names can never contain special characters (characters other than A through Z and 0 through 9).

**DISK and " "**

Both DISK and " " specify the system disk.

**SERIALNO Attribute**

The <serial number> can be specified for each device. The value of <serial number> must be an alphanumeric string whose length is six characters or less. This attribute does not direct the output to a specific unit; it provides further identification of a tape to the SYSTEM/COPY program when tape files with the same name are used. The serial number assigned to a tape with an SN input message (refer to volume 1, section 2), and the SERIALNO attribute of the SYSTEM/COPY program are distinct and unrelated values. The <serial number> used by the SYSTEM/COPY program defaults to zero when this attribute is not specified.

B 1000 Systems SSOG, Volume 2  
SYSTEM/COPY

**KIND Attribute**

The SYSTEM/COPY program uses the default (KIND = DISK) when (1) the program is executed with switch 9 = 1, (2) the hardware attribute KIND is omitted in all references to a volume, and (3) a volume other than the system disk is specified. Otherwise, the hardware attribute KIND must be specified for each volume.

The value of <device-type> must be one of the following mnemonics:

<device type>	Meaning
DISK	
TAPE	(any tape)
TAPEPE	(any 9-track PE tape)
TAPE7	(any 7-track tape)
TAPE9	(any 9-track tape)
	(NRZ if no type specifically requested)
TAPEGCR	(any group-coded-recognition magnetic tape)

**HOSTNAME Attribute**

The HOSTNAME attribute specifies the system on which a volume is to be found on a Burroughs Network Architecture (BNA) network.

**DENSITY Attribute**

If the volume has KIND = TAPE, then the DENSITY attribute can be used to specify the recording density. Valid values on the B 1000 system are BPI800 and BPI1600.

Examples:

```
COPY AND SET(COMPARE) FILE1 TO MYTAPE(KIND = TAPE, DENSITY = BPI1600)

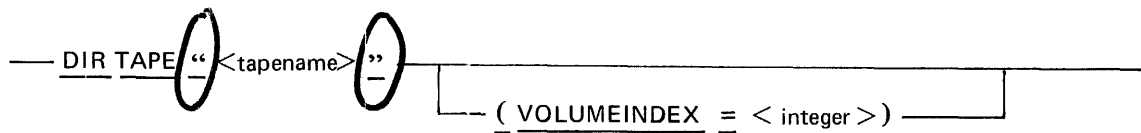
COPY & SET(COMPARE) FILE2 FROM MYTAPE3(KIND = TAPEPE, VOLUMEINDEX=3)
TO " " (HOSTNAME = HUB)

COPY FILE4 FROM YOURTAPE(KIND=TAPE, SERIALNO = 1982)
TO MYDISK(KIND = DISK)
```

**DIR Command**

The DIR command causes the SYSTEM/COPY program to print the directory of the specified COPY library tape. The syntax for the DIR command follows.

Syntax:



Every reel of a multireel library tape contains a directory that lists all the files on the current and subsequent reels of the tape file for reference by the SYSTEM/COPY program.

<tapename>

The <tapename> must be a valid tape identifier consisting of a single identifier with a maximum of 10 characters. Special characters, that is, characters other than A through Z and 0 through 9, are not allowed in the <tapename>. Program switch 2 does not allow special characters to be used in <tapename> volume identifiers.

#### **VOLUMEINDEX Attribute**

There are two ways to access the directory on tape reels other than the first reel of a multivolume COPY tape. Specifying the VOLUMEINDEX attribute with the DIR command allows the SYSTEM/COPY program to print the directory on the specified reel without searching on previous reels. However, if the tape directory crosses a reel boundary, files on the continuation reel cannot be accessed by specifying VOLUMEINDEX = <integer>, where <integer> is the number of the continuation reel. The reel on which the tape directory originated must be specified.

If VOLUMEINDEX is not specified and the operator desires the directory on other than the first reel of a multivolume input tape, the operator must dynamically modify the SYSTEM/COPY program. When the SYSTEM/COPY program issues the message

```
<tapename>/TAPDIR1977 NOT FOUND
```

The operator must enter the following commands.

```
<job #> DY FI INFILE REEL <reel number>  
<job #> OK
```

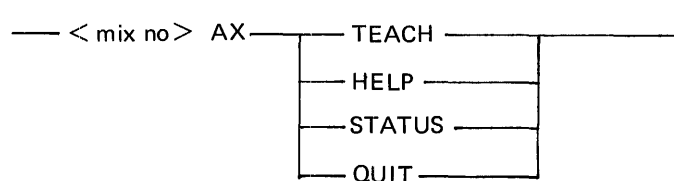
Examples:

```
DIR TAPE "MYTAPE";  
DIR TAPE "MYTAPE" (VOLUMEINDEX=3);
```

## **AX COMMANDS**

While copying or comparing files, the SYSTEM/COPY program checks for accept (AX) input messages, making it possible for the operator to query the status of the program. The valid AX input messages are TEACH/HELP, STATUS, and QUIT. The syntax for the accept (AX) input messages follows.

Syntax:



## TEACH/HELP Command

Entering the TEACH or HELP command with the AX input message while the SYSTEM/COPY program is processing causes a list of valid AX input messages to be displayed.

Example:

```
1841AX TEACH
% SYSTEM/COPY = 1841 SYSTEM/COPY ACCEPTS THE FOLLOWING COMMANDS:
% SYSTEM/COPY = 1841 TEACH /HELP - DISPLAY THESE INSTRUCTIONS
% SYSTEM/COPY = 1841 STATUS - DISPLAY CURRENT PROGRAM STATUS
% SYSTEM/COPY = 1841 QUIT - DISCONTINUE PROCESSING THE CURRENT
```

## STATUS Command

Entering the STATUS command with an AX input message while the SYSTEM/COPY program is processing causes the name of the file currently being copied or compared and the number of records already processed to be displayed.

Example:

```
1843AX STATUS

% SYSTEM/COPY = 1843 COMPARING (JMS)/FILE1 FROM M (2940/6780 SEGMENTS)
```

## QUIT Command

Entering the QUIT command with the AX input message while the SYSTEM/COPY program is processing causes the program to discontinue processing of the current file and go to the next file. If the output file was on tape, a partial file remains. If the output file was on disk, the partially copied file is removed.

Example:

```
1844AX QUIT
COPY 1844 : "(JMS)/FILE1" NOT COPIED FROM "M" AS "(JMS)/FILE2"
COPY 1844 : 1 FILE NOT COPIED
SYSTEM/COPY = 1844 EOJ. TIME = 14:53:35.8
```

## PROGRAM SWITCHES

Five program switches govern the execution of the SYSTEM/COPY program: switches 0, 1, 2, 3, and 9. Switch defaults are always zero and the valid switch settings are described in table 30-1. Changes to the settings must be specified immediately after the command specification string. After the SYSTEM/COPY program has gone to beginning of job (BOJ), it is not responsive to changes in the switch settings. The operator can modify the object code of the SYSTEM/COPY program when only switch settings other than the default are to be used. The following example sets the SYSTEM/COPY program switches for the current execution only.

Example:

```
COPY <command specification string>; SW 1 = 1;
```

B 1000 Systems SSOG, Volume 2  
SYSTEM/COPY

The following example permanently sets the SYSTEM/COPY program switches by permanently modifying the code file.

```
MODIFY SYSTEM/COPY; SWITCH 1 = 1;
```

**Table 30-1. SYSTEM/COPY Program Switches**

Switch	Value	Meaning
0	0	Syntax and semantic error messages are displayed on the ODT only.
0	1	Syntax and semantic error messages are displayed on the line printer only.
0	2	Syntax and semantic error messages are displayed on both the ODT and the line printer.
1	0	Library maintenance messages are displayed on the ODT if the MCPII COPY option is set.
1	1	Library maintenance messages are listed on the line printer. If the MCPII COPY option is set, messages are also displayed on the ODT.
2	0	Special characters are allowed in tape file-ids (values other than A-Z and 0-9). Tape volume names never allow special characters.
2	1	Only alphanumeric characters are allowed in tape file names in accordance with Burroughs standard (only values A-Z and 0-9).
3	0	Only first level (unexpanded) screening for duplicate file-id screening is performed.
3	1	Screening for duplicate file-ids is performed on expanded file-ids if duplicate names are possible based on results of unexpanded screening.
9	0	Hardware KIND must be specified for each volume specified.
	1	Disk is the default hardware KIND.

\* ODT = Operator Display Terminal

## TIME REQUIRED FOR COPY

The SYSTEM/COPY program generally requires more time when copying from tape to disk because the buffer size for output to disk has certain constraints upon it that the buffer size for output to tape does not. For output to tape, the buffer size is 10 sectors (1800 bytes). For output to disk, the sectors per area must be evenly divisible by the number of disk sectors that the buffer contains. To determine the number of disk sectors that a buffer contains, a modulo function is performed on the sectors per area of the file, starting at 10 and working down to 1, if necessary. When the modulo function yields a result of 0, the number used in the modulo function determines the number of sectors per buffer. For example, if the sectors per area of a file is 72:

$$\begin{aligned}72 \text{ MOD } 10 &= 2 \\72 \text{ MOD } 9 &= 0\end{aligned}$$

The buffer contains nine sectors. As the buffer size increases, the number of I/O operations increases, and thus, the elapsed time for the COPY increases.

## SAMPLE INPUT STRINGS FOR COPYING

The following paragraphs describe the sample input strings.

### User Disk to Tape

Assume the following usercode/default-pack-id pairs:

```
USER - PACK
YOURS - PACK1
MINE - PACK2
```

Examples:

```
COPY & SET(COMPARE) (USER)/= FROM PACK(KIND=DISK)
  TO TAPE1(KIND=TAPE);
  The default pack-id can be explicitly specified.
```

```
COPY & SET(COMPARE) (YOURS)/= AS (MINE)/= TO TAPE2(KIND=TAPE);
```

```
COPY & SET(COMPARE) A/B, (USER)/FILE1, (MINE)/=
  TO TAPE3(KIND=TAPEPE);
```

### System Disk to Tape

The keyword DISK or " " in the FROM clause, or the lack of a FROM clause when non-secured files are referenced, indicates that the system disk is the input medium.

Examples:

```
COPY & SET(COMPARE) MCP11, GISMO, SYSTEM/INIT, SDL/INTERP1S,
  SDL/INTERP1M, SYSTEM/COPY, SDL.INTRIN/AGGREGATE, FILE/LOADER,
  MCP11/MICRO.MCP TO SYSTEM(KIND=DISK);
```

```
COPY & SET(COMPARE) THIS/THAT FROM DISK TO BACKUP(KIND=TAPE);
COPY & SET(COMPARE) (NEWUSER)/= FROM " " TO USERS(KIND=DISK);
  The default pack-id associated with NEWUSER must be the system disk.
```

## Disk to Disk

The keyword DISK or " " references the system disk; also, if a TO or FROM clause is missing and non-secured files are referenced, the system disk is assumed by default to be the source or destination medium.

Examples:

```
COPY & SET(COMPARE) INFO/= FROM MYPACK(KIND=DISK)
  TO YRPACK(KIND=DISK);
```

```
COPY & SET(COMPARE) IMPORTANT/DATA FROM MYPACK(KIND=DISK);
COPY & SET(COMPARE) IMPORTANT/DATA FROM MYPACK(KIND=DISK) TO " ";
```

## Tape-to-Tape

The following is an example of tape-to-tape copy:

Input:

```
COPY & SET(COMPARE) TFILE1, TFILE2 FROM TAPE1(KIND=TAPE),
  TFILE4 FROM TAPE10(KIND=TAPE)
  TO TAPE2(KIND=TAPEPE)
  TO TAPE3(KIND=TAPE7)
  TO TAPE4(KIND=TAPE);
```

Result:

	Volume Name	File Name
INPUT	TAPE1	TFILE1, TFILE2
	TAPE10	TFILE3, TFILE4
OUTPUT	TAPE2 - PE tape	TFILE1, TFILE2, TFILE3, TFILE4
	TAPE3 - 7-track	TFILE1, TFILE2, TFILE3, TFILE4
	TAPE4 - Tape	TFILE1, TFILE2, TFILE3, TFILE4

## SYSTEM COPY OUTPUT

### File Formats

The format of the newly copied file is identical to the format of the source file from which it is created, except for pseudo decks. When pseudo decks are copied, the SYSTEM/COPY program changes the file type from PSEUDO to DATA. All pertinent file attributes, including CREATION - DATE, are copied from the header of the input file to the header of the output file. Refer to volume 1, section 2 for a description of file headers. Copying the file from disk to tape or from tape to disk changes only the medium of the file; no other attributes are affected. The only exception occurs when a file is copied to or from a tape that has been scratched in a non-EBCDIC mode. In this case, the EBCDIC code recognized by the B 1000 system is translated into the code-format indicated by the code with which the tape was scratched. Refer to the SN input message in volume 1, section 2 for details.

## ERROR HANDLING

### I/O Errors

The SYSTEM/COPY program attempts to recover from I/O errors on library files whenever possible. Where error-recovery options are offered, the SYSTEM/COPY program cannot continue until an action is specified. The program offers the operator the following error-recovery options:

AX "CONTINUE" TO IGNORE THE ERROR  
AX "QUIT" TO ABANDON ONLY THIS FILE  
AX "ABORT" TO ABORT ALL COPIES USING THIS UNIT

When a parity error occurs, the SYSTEM/COPY program displays a message that advises the operator of the error, and the program then offers the three error-recovery options. When critical errors occur, the SYSTEM/COPY program advises the operator, and then offers only the QUIT and ABORT error-recovery options. If an error occurs while writing an output tape directory, the SYSTEM/COPY program automatically aborts the copy of all files to the faulty tape unit, issues an advisory message, and purges the tape.

When the ABORT option is requested, the unit being aborted is purged and becomes a scratch unit. When an abort operation is performed automatically because an error occurred when writing labels and headers, the unit is automatically purged.

All records on COPY library tapes are marked as reliable or unreliable. Specifying the CONTINUE error-recovery option after the occurrence of an input parity error causes the output tape record in question to be marked as unreliable; specifying QUIT or ABORT causes the last record written to the file to be marked unreliable. If the SYSTEM/COPY program encounters any errors while writing a file and the COMPARE option was requested in the COPY command specification, the SYSTEM/COPY program ignores the COMPARE request for the file suspected to have errors, and does not compare it.

When the SYSTEM/COPY program reads a COPY library tape, it reports all unreliable input records with the following message, and then becomes a scratch unit:

BLOCK NUMBER <n> ON TAPE <tapename> CONTAINS UNRELIABLE DATA

When an abort operation is performed automatically because an error occurred when writing labels and headers, the unit is automatically purged.

### In-Use Files

When the SYSTEM/COPY program attempts to copy a file to disk, and a file with the same name already exists and is in use, the MCP II issues the message FILE NOT REMOVED--IN USE. The operator must respond with a MR, RM, DP, or DS input message (documented in volume 1, section 2) before the SYSTEM/COPY program can continue execution. When the SYSTEM/COPY program attempts to copy a file from a disk, and the file is in use by another program, the file is not copied, even though the name of the file was entered in the tape directory.

*iMerde!*



## Missing Files

### Disk Input Units

If the specified disk volume is on-line and the file cannot be found, the advisory message FILE NOT FOUND is issued and the file is not copied. The SYSTEM/COPY program continues execution. If the specified disk volume is off-line, the SYSTEM/COPY program issues the following message sequence and waits for operator response.

```
PACK <volume name> IS NOT ON-LINE  
PLEASE MOUNT THE PACK AND AX "READY"  
OR AX "SKIP" TO SKIP ALL COPIES USING THIS PACK.
```

### Tape Input Units

Missing tape input files can be caused by two different problems. First, when a tape is off-line, the SYSTEM/COPY program waits for the file <tape name>/TAPDIR1977 to be made available. In this case, the operator must either ready the requested tape, or use the IL input message to specify another tape for the SYSTEM/COPY program to process. Either response must make the first reel of a COPY library tape available to the SYSTEM/COPY program.

The missing tape input file error condition can also occur when a file identifier is included in the COPY tape directory, but the file has not actually been copied to the tape. This discrepancy between the tape directory and actual tape contents occurs when the program that originally built the COPY library tape was discontinued (DS or DP input message) before it finished copying all the files but after it built the directory.

In the preceding case, there are two possible corrective actions. If the ending label of the tape has not been corrupted, then the operator can respond with the <job #>OF input message to skip the requested copy of the missing file. The second possible operator action is to re-execute the copy, omitting the missing files from the command specification copy request.

## Duplicate File Identifier Checking

The SYSTEM/COPY program screens its command specification string for duplicate file identifiers. The screening prevents redundant copies of the same file, destructive copies to a file specified as both input and output, and premature copies that use a file that is created during this execution of SYSTEM/COPY as an input file from which another file copy is requested during this same execution.

The SYSTEM/COPY program always scans the original command specification string for duplicate file identifiers before any equal sign (=) characters in the file names are expanded. Duplication errors, such as the following, are detected at this time.

```
COPY FILE1, FILE2 AS FILE1 TO T(KIND=TAPE);  
COPY FILE1, FILE2, FILE1 TO T(KIND=TAPE);
```

Some duplication errors, however, cannot be detected until the equal sign (=) characters in the file names are expanded. The SYSTEM/COPY program optionally screens the expanded command specification string for duplicate file names if the operator requests this second level of checking by setting program switch 3 to the value 1.

When duplicate file names are discovered in the command specification string, the following rules of precedence apply in determining which files are copied and which copy requests are ignored. These rules of precedence used by the SYSTEM/COPY program select, in the the case of duplicate requests, the file that is accessible if no checks for duplicate file copies are performed. For example, if a tape contains two files with the same identifiers, only the first is accessible. When two files with duplicate identifiers are written to disk, one file is lost because there can be only one disk directory entry for each file-identifier. The SYSTEM/COPY program copies the disk file named last in the command specification string.

#### Rules of Precedence

1. Disk output files - The last file listed in the command specification string has priority over duplicate files that precede it.
2. Tape output files - The first file listed in the command specification string has priority over all duplicate files that follow it.
3. All other cases - In all other cases of file identifier duplication, the first file copy request listed in the command specification string has precedence over all ensuing copy requests that make duplicate reference to the file.

A descriptive Syntax and Semantic Warning Message is issued by the SYSTEM/COPY program when file duplication errors are encountered during scanning of the command specification string. File duplication errors cause the SYSTEM/COPY program to ignore only the duplicate copy requests; all non-duplicate copy requests are performed without interruption. Redundant requests to copy files to library tapes are discovered before beginning the copy; thus, neither the COPY library tape nor the tape directory contain duplicate file entries.

The advisory messages describing duplicate file errors are listed and described in the subsection entitled Syntax and Semantics Warning Messages.

#### **Error Messages Displayed on the ODT**

The error messages issued by the SYSTEM/COPY program are classified as Library Maintenance Messages, Syntax and Semantics Error Messages, Syntax and Semantics Advisory Messages, and Error Handling User Interface Messages. Only the Syntax and Semantics Error Messages describe fatal errors. Both the Library Maintenance Messages and the Syntax and Semantics Advisory Messages are advisory messages only. The Error Handling User Interface Messages were previously described in the subsection entitled Error Handling.

#### Syntax and Semantics Error Messages (Fatal Errors)

##### **"ADD" AND "ONTO" BOTH SPECIFIED**

The ADD command and ONTO option are incompatible requests. The keyword COPY must be used when the ONTO attribute is specified.

##### **"ADD" NOT ALLOWED FOR TAPE**

Since scratch tapes (which contain no files) must be used as tape output devices for the SYSTEM/COPY program, the ADD command is an invalid request.

##### **ATTRIBUTES SPECIFIED FOR SYSTEM DISK**

The operator must not specify volume attributes for the system disk.

##### **BAD ATTRIBUTE VALUE**

The SYSTEM/COPY program encountered a left parenthesis "(" character that was not followed by a recognizable attribute keyword. A common cause of this error is the failure to enclose a user-code in quotation mark (") characters.

**BLANK ID IN NAME**

A blank file identifier is not valid (SYS/" " is not valid).

**DISK FILE NAME HAS MORE THAN 2 IDS**

The SYSTEM/COPY program recognizes only the family-id and the file identifier as part of a disk file identifier. The volume identifier must be specified in a FROM or TO clause.

**DISK FILE NAME HAS MORE THAN 10 CHARS IN ID**

Disk file names recognized by the SYSTEM/COPY program must conform to the format accepted by the MCP II. Each identifier must be 10 characters or less, and a file identifier that contains any special characters must be enclosed in quotation mark (") characters.

**EXPANDED NAME TOO LONG**

While expanding the equal sign (=) characters in file identifiers, the SYSTEM/COPY program encountered a disk file identifier consisting of more than 2 identifiers, or a tape file identifier consisting of more than 13 identifiers. A common cause of this error is the improper use of trailing equal sign (=) characters.

**INVALID USE OF ASTERISK**

The asterisk (\*) character, used on the B 1000 system to assign an exact identifier to a file, has been used in an invalid situation. An asterisk (\*) character must be immediately adjacent to a file identifier.

**MISSING ATTRIBUTE VALUE**

The SYSTEM/COPY program could not find a value following an attribute keyword.

**MISSING FINAL PARENTHESIS**

The SYSTEM/COPY program could not find an ending right parenthesis ")" character to match a left parenthesis "(" character when processing a COPY command specification string.

**MISSING KEYWORD**

The SYSTEM/COPY program began execution, but could not find a valid command keyword at the beginning of the command specification string. COPY, ADD, and DIR are valid command keywords. The option AND COMPARE can be requested, but blanks must appear between the keywords.

**MISSING NAME**

The SYSTEM/COPY program expected a file identifier as input but could not find one. File identifiers must appear before their associated FROM or TO clauses.

**MISSING VOLUME NAME**

The keyword TO or FROM was encountered but was not followed by a volume identifier. When a blank is used to denote the system disk, it must be enclosed in quotation marks (" "). If a volume is named either TO or FROM, the identifier must be enclosed in quotation marks. Likewise, if a volume (other than the system disk) is named DISK, the identifier must be enclosed in quotation marks when referencing the volume with the SYSTEM/COPY program.

**NAME HAS TRAILING SLASH**

The volume identifier must always be specified in a TO or FROM clause; thus, the use of a trailing (ending) slash (/) character in a file identifier is invalid, because the trailing slash (/) character construct indicates that the format of the identifier is <volume-id>/<family-id>, without a third file identifier.

B 1000 Systems SSOG, Volume 2  
SYSTEM/COPY

NUMERIC VALUE EXPECTED

The SYSTEM/COPY program expected a numeric value as a parameter but found a non-numeric value. Mnemonics cannot be substituted for device numbers.

NULL ID IN NAME

Neither blanks nor null fields are valid file identifiers. Two adjacent slashes delimit an invalid null identifier.

NON STANDARD NAME FOR TAPE FILE

Tape file identifiers can have a maximum of 13 distinct names. Each name can consist of a maximum of 17 alphanumeric characters. Identifiers that contain special characters, whether within quotation mark (") characters or not, are nonstandard file identifiers. When program switch 2 is set to 0, special characters are allowed in tape file identifiers.

NOT COPIED - SECURITY VIOLATION OR FILE REMOVED

The file was found but could not be opened, either because of a security violation or because the file was removed by another operation.

NOT COPIED - COPY OVER SELF NOT ALLOWED

The copy operation was not done because it would have destroyed the input file.

NOT COPIED - FILE IN USE

The file was in use by another program and could not be opened read-only by the SYSTEM/COPY program.

NOT COPIED - ERROR IN GET/CHANGE ATTRIBUTE REQUEST

The copy operation was aborted because a GET or CHANGE attribute request failed. This occurs only when copying to a foreign host.

"ONTO" NOT ALLOWED FOR TAPE

Installation Allocated Disk (IAD) files cannot be created on tapes; thus, the ONTO clause is meaningless, and hence is disallowed.

QUOTED STRING INCOMPLETE

The SYSTEM/COPY program began processing a string enclosed in quotation mark (") characters but could not find an ending quotation mark (") character.

SAME TAPE SPECIFIED AS INPUT AND OUTPUT

The SYSTEM/COPY program never uses a file for both input and output during a single copy request. If different tapes with the same names are to be used for input and output, the SERIALNO attribute should be used to distinguish the tapes. The SYSTEM/COPY program does not read the serial number assigned by the SN input message; the SERIALNO volume attribute describes a serial number known only to the SYSTEM/COPY program.

TAPE NAME MUST BE IN QUOTES

The SYSTEM/COPY program expects the tape name specified in a DIR command to be enclosed within quotation mark (") characters.

Example:

DIR TAPE "<tapename>"

#### TOKEN TOO LONG

The SYSTEM/COPY program cannot process an input command specification if any words (tokens) in the specification string contain more than 17 characters. Blank characters must be included between the command keywords. For example, the blank characters in the command COPY & COMPARE are required.

#### UNEXPECTED DELIMITER

The SYSTEM/COPY program unexpectedly encountered a special character that it regards as a delimiting character when processing the command specification string. Any file or volume identifier that includes special characters must be enclosed in quotation mark (") characters, or these characters can be interpreted as delimiters. Special characters are defined as all characters other than A through Z and 0 through 9.

#### UNEXPECTED TEXT FOLLOWING DIR REQUEST

The DIR command specification does not allow comments to be included in the specification string. Any comments must be separated from the DIR command with a percent sign (%) character.

#### UNKNOWN ATTRIBUTE

The SYSTEM/COPY program was expecting an attribute description but encountered an unknown keyword.

#### UNMATCHED EQUALS

When the AS clause is specified in a COPY request, the source and destination file names must contain compatible equal sign characters. A trailing equal sign (=) character encompasses all equal sign (=) characters and file identifiers that follow the last name specified. For example, the tape file names A/B/C and A/B/E/F would both be included when A/= is specified as the file identifier. The equal sign (=) characters are compatible; however, in the string COPY A/= AS B/= /C/= the equal sign (=) characters are not compatible. The number of identifiers in the file name can vary. The string COPY A/= AS B/C/= is a valid copy request.

#### <unit> HEADER LEVEL VERSION MISMATCH

Major software changes made to the SYSTEM/COPY program can result in an incompatibility between COPY library tapes created by different versions of the SYSTEM/COPY program. Information in file headers can prevent access of an incompatible tape when a level mismatch is detected.

#### <unit> NOT CREATED BY B 1000 COPY

The COPY library tape referenced was not created on a B 1000 computer by the SYSTEM/COPY program.

B 1000 Systems SSOG, Volume 2  
SYSTEM/COPY

<unit> NOT A LIBRARY TAPE

The tape unit named as an input volume in the command specification string was not created by the SYSTEM/COPY program and is not a library tape.

VOLUME NAME CONTAINS SPECIAL CHARS

When a disk volume name contains special characters (any character other than A through Z or 0 through 9), that name must be enclosed in quotation mark (") characters. Tape volume names can never contain special characters.

VOLUME NAME HAS MORE THAN 10 CHARS

No volume identifier can contain more than 10 characters. Tape names must conform to this restriction, although tape file identifiers can contain up to 17 characters in each identifier.

Syntax and Semantics Advisory Messages

ATTRIBUTE NOT YET IMPLEMENTED

The SYSTEM/COPY program recognized the specified file attribute as valid; however, the program cannot accept the attribute as input.

"COPY AS" NAME SAME AS ORIGINAL

The copy command specification string requests that a file be copied and given its original name. The AS clause is redundant and is not allowed.

COPY TO SELF IGNORED

A redundant copy was requested.

DESTRUCTIVE COPY TO <unit> IGNORED

The command specification string requested a copy operation that would either overwrite a file used as input elsewhere in the specification string, or attempted to use a file created during this execution of the SYSTEM/COPY program as an input file elsewhere in the command specification string.

DUPLICATE COPY TO <unit> IGNORED

A redundant copy was requested.

DUPLICATE OUTPUT UNIT IGNORED

An output file was listed in more than one TO clause for a single file.

EMPTY ATTRIBUTE LIST

The SYSTEM/COPY program encountered an empty pair of parenthesis "()" characters while processing the copy command specification string.

TOO MANY INPUT UNITS - UNIT IGNORED

Only one FROM clause can be associated with an input file.

### TOO MANY OUTPUT UNITS -UNIT IGNORED

The maximum number of TO clauses that can be associated with an input file is eight. Multiple phrases cannot be used to circumvent this restriction. For example, the string COPY A TO T1, TO T2, TO T3, TO T4, TO T5, TO T6, TO T7, TO T8, TO T9 generates the advisory message TOO MANY OUTPUT UNITS.

### TOO MANY SLASHES

Disk file identifiers can contain one slash (/) character. Tape file identifiers can contain 12 slash (/) characters.

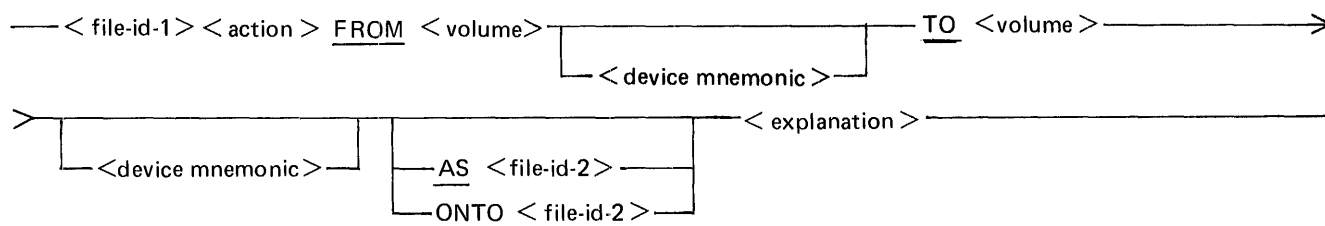
### TOO MANY UNITS FOR ONE STMT -UNIT IGNORED

In a single copy command specification, a maximum of 255 distinct volumes, including both input and output units, can be described and referenced.

### Library Maintenance Messages

Library maintenance messages are displayed on the operator display terminal (ODT) only when the COPY MCPPII option is set. When program switch 1 = 1, all library maintenance messages are listed on the line printer; the file labeled LINE can be modified to be a printer backup file to avoid allocation of the line printer to the SYSTEM/COPY program throughout the copy process. Program switch 0 must be set to 1 if error messages are to be included in this listing.

Library maintenance messages are displayed in the following format.



#### Semantics:

The `< actions >` which can appear in SYSTEM/COPY library maintenance messages are named in the following list under the heading Messages Denoting Actions. The `< explanations >` which can further describe the `< action >` are named in the succeeding list under the heading Explanatory Messages.

#### Messages Denoting Actions

##### COPIED

The file named was copied. If no explanation accompanies this message, the file was copied successfully, although there is no inherent implication that the copied file has been compared with the original file. Accompanying explanations can note errors that occurred while copying or comparing the new file.

##### COPIED AND COMPARED

The file named was copied successfully, and no errors were found during the requested comparison.

#### NOT COPIED

The file named was not copied. An explanation included in the message describes the reason that the file was not copied.

#### COPIED BUT NOT COMPARED

The file named was copied, but the file could not be compared although the COMPARE option was specified. Tape files that are split across reel boundaries cannot be compared.

#### Explanatory Messages

#### NOT FOUND

The SYSTEM/COPY program could not find the file named in the command specification string.

#### FILE REMOVED BEFORE COPY WAS COMPLETE

The file named was found when the copy operation began. If the copy destination is a library tape, the file name is included in the tape directory, and a dummy file is created on the tape. If the copy destination is a disk, the copy request is ignored.

#### ONTO FILE NOT FOUND

The IAD file named in the ONTO option of the File Specification clause could not be found, and the copy request is ignored.

#### INCOMPATIBLE ONTO FILE

The IAD file named in the ONTO option of the File Specification clause in the command specification string is incompatible with the input file. When the ONTO option is used, the input and the output (IAD) file must have identical attributes.

#### MULTI-PACK FILE

Files that exist as multi-pack files cannot be copied. The copy request is ignored.

#### TOO MANY UNITS IN USE

Only eight output files can be in use by the SYSTEM/COPY program at a given time, although a maximum of 255 total output units can be named in the command specification. Because the SYSTEM/COPY program accesses each input file only once, at which time it makes all requested copies of that file, a situation might arise which requires more than eight output units to be in use by the SYSTEM/COPY program at one time. This error is independent from the restriction that a single file can only be copied to eight output units.

#### REQUIRES INPUT TAPE REWIND

The SYSTEM/COPY program reads an input tape only once. If a copy function requires a tape to be rewound, the SYSTEM/COPY program cannot perform the copy. This error arises when the operator attempts to copy a file from a tape and assigns the copied files new identifiers. For example, the copy request COPY A FROM TAPE1, A AS A2 FROM TAPE1 generates this error and error message.



### BAD FILE HEADER

The input file (<file-id-1>) named in the <action> clause of the error message has a bad file header and cannot be copied.

### COMPARISON ERROR

The file named in the <action> clause of the error message was copied, but an error was detected while comparing the input and copied files. Accuracy of the data is questionable.

### IO ERRORS

I/O errors (any exception condition) were encountered during the copy. When I/O errors are encountered, the SYSTEM/COPY program offers the operator error recovery options. When this explanation accompanies the COPIED message, it indicates that the operator chose the error recovery option CONTINUE. The records in which the errors occurred are marked as unreliable. When this explanation accompanies the NOT COPIED message, it indicates that the copy of this file was not performed, because either the operator selected the QUIT or ABORT error-handling option, or the I/O errors were irrecoverable.

An MCPPII-generated error message which describes the I/O error in detail precedes this SYSTEM/COPY error message.

### **COMPARE Function Error Line Printer Listing**

When errors occur during a COMPARE function, the SYSTEM/COPY program gives an explanation of the comparison error in a line printer listing. In the message format, <file> is either SOURCE FILE or COPY FILE. The following comparison error messages can be given.

#### HEADER COMPARISON ERROR

DATA COMPARISON NOT DONE

Certain size-related fields in the header did not match; therefore, the data was not compared.

#### HEADER COMPARISON ERROR IGNORED

[DATA SUCCESSFULLY COMPARED]

A comparison error in the header was not significant enough to prevent the comparison of the data.

<file> NOT FOUND

The specified file was not located.

<file> IN USE

The file is being updated by another program.

<file> INVALID HEADER

<file> INVALID DIRECTORY ENTRY

#### DATA COMPARISON ERROR

<file> IO ERROR

<file> UNRELIABLE TAPE DATA

<file> REQUIRES TAPE REWIND

COMPARISON TERMINATED BY "QUIT"

<file> UNEXPECTED END-OF-FILE

The end of the file was encountered on <file> before the end-of-file (EOF) record on the other file.

CANNOT COMPARE MULTI-PACK FILES

A COMPARE function alone cannot be done. However, a COPY & COMPARE function can be done in order to compare the file.

CANNOT COMPARE VARIABLE FILES

A COMPARE function alone cannot be done. However, a COPY & COMPARE function can be done in order to compare the file.

CANNOT COMPARE RELATIVE FILES

A COMPARE function alone cannot be done. However, a COPY & COMPARE function can be done in order to compare the file.

<file> IN DIRECTORY BUT NOT ON TAPE

The file was available at the time the tape directory was created but the file was removed when the data transfer to the tape occurred.

## SECTION 31

### SYSTEM/DISK.DUMP

The SYSTEM/DISK.DUMP program is a normal-state utility program that can copy the contents of disk cartridges or disk packs to disk cartridges or disk packs of the same or larger capacity. Neither the input nor the output disk can be in use by another program when the dump takes place. Thus, the system disk cannot be copied to another disk with this utility program.

The SYSTEM/DISK.DUMP program can perform two types of full disk copies. These full disk copies (often called a disk dump) are described in the following paragraphs.

1. A sector-by-sector copy produces an identical copy of the source disk. The stand-alone utility program DISK/DUMP also performs a sector-by-sector copy. Irrecoverable I/O errors cause the SYSTEM/DISK.DUMP program to terminate.
2. A file-by-file copy produces a squashed output disk. The stand-alone utility program STAND-ALONE/DISK.DUMP also performs a file-by-file copy. The file-by-file copy allows a disk with a bad label to be copied. When the SYSTEM/DISK.DUMP program is unsuccessful in recovering from I/O errors, the error data is copied to the new disk, and the address of the error is displayed on the operator display terminal (ODT) in an error message. If the irrecoverable I/O error occurs in the disk directory or in a file header, the associated file is not copied.

## OPERATING INSTRUCTIONS

The SYSTEM/DISK.DUMP program is executed from the console keyboard and accepts dump specifications through accept (AX) input messages. The program issues prompts for all required information.

## DUMP SPECIFICATIONS

The SYSTEM/DISK.DUMP program needs the following dump specifications to begin execution. When the program is executed from the console keyboard, prompts request the necessary input specifications. The operator must respond with an accept (AX) input message that gives the requested information. The format of an accept message is "<job number>AX <information>". When execution is through a card reader, the specifications must be punched to cards, maintaining the order in which they are presented in the following paragraphs.

### INPUT DRIVE

A unit mnemonic that names the drive on which the source disk is mounted. The dump cannot begin while the input disk is in use by any other program. After the completion of each dump, the SYSTEM/DISK.DUMP program repeats the prompt sequence, beginning with the following message:

Prompt: ENTER INPUT DRIVE <DC? OR DP?>

A blank response terminates the program.

### COPY TYPE

The COPY TYPE message asks for the file-by-file or sector-by-sector copy type. When the input disk is a user disk, or the input disk has a bad label, the SYSTEM/DISK.DUMP program offers the operator a choice of copy types. A YES response to the prompt requests a file-by-file copy; a NO response requests a sector-by-sector copy.

Prompt: IS FILE BY FILE COPY DESIRED? <YES OR NO>

B 1000 Systems SSOG, Volume 2  
SYSTEM/DISK.DUMP

### OUTPUT DRIVE

The OUTPUT DRIVE message asks for a unit mnemonic that names the drive on which the output disk is located. The dump cannot begin while the output disk is in use by any other program.

Prompt: ENTER OUTPUT DRIVE <DC? OR DP?>

### VERIFICATION

The VERIFICATION message asks for optional verification of data can be requested when a file-by-file copy is performed.

Prompt: IS VERIFICATION REQUIRED? <YES OR NO>

## PROGRAM OUTPUT

The following paragraphs describe the program output.

### Output Disk

After a successful dump, the output disk is the same pack type as the input disk, and the data on the disks is identical, except for the serial number and label information. If a sector-by-sector copy was performed, the output disk is identical to the input disk. If a file-by-file copy was performed, the output disk is a squashed copy of the input disk, and the disk directory has been reconstructed.

### Informative Messages

When verification is requested, the following message indicates that the VERIFY pass is beginning:

VERIFICATION BEGINS

After completion of the copy (or copy and compare) and assuming no errors, the following message is displayed on the ODT:

DUMP COMPLETE <input-drive> <serial-number>  
TO <output-drive> <serial-number>

After the SYSTEM/DISK.DUMP program has completely processed a disk, whether or not the dump was successful, it repeats the prompt sequence.

### NOTE

Neither the serial number nor label information is copied from the input disk to the output disk.

### Error Messages

DISK ERROR - RESULT = <result-status>

An I/O error was not corrected after 10 retries.

IS RETRY DESIRED? <YES OR NO>

This message follows the DISK ERROR message. A YES response causes the operation to be retried; NO causes termination of the program.

B 1000 Systems SSOG, Volume 2  
SYSTEM/DISK.DUMP

ENTER DESIRED NUMBER OF RETRIES

Follows an affirmative response to the IS RETRY DESIRED? message. The entry of a blank or zero terminates the program.

INVALID RESPONSE - TRY AGAIN

Either the response to a prompt was other than YES, Y, NO, N, DCx, or DPx, or the requested drive is not available.

TEMP TABLE FILLED - <input-drive>

Temporary areas cannot currently be assigned on the input drive. This problem can be alleviated by performing a CLEAR/START operation.

DISK NOT READY

Ensure that both the input and output disks are ready and enter "<job #> OK".

WRITE LOCKOUT

Remove the write lockout on the output disk drive and enter "<job #> OK".

REMOVED SECTORS ON DRIVE <drive>

A disk with removed sectors must be copied in the file-by-file mode. This message indicates that a sector-by-sector copy was requested.

PACK LABEL BAD - <drive>

If this message names the input drive, a file-by-file copy must be requested. If the message is for an output drive, the disk must be initialized.

I/O ERROR - <drive>

A parity, timeout, address parity, extended result descriptor, or address error occurred and all retries failed.

RESULT = <result-status>

Follows the I/O ERROR message and indicates the type of error.

<file-id> IS A MULTI PACK FILE AND CANNOT BE COPIED

If the serial numbers of the input and output packs do not agree, a multi-pack file cannot be copied.

BAD DIRECTORY <output drive>

The new directory cannot be read; the dump must be restarted using a different output disk.

B 1000 Systems SSOG, Volume 2  
SYSTEM/DISK.DUMP

**BAD MAIN DIRECTORY** <disk-address> <drive>

Invalid input directory at indicated address. No file whose directory entry is within that segment is copied.

**BAD SUB DIRECTORY** <disk-address> <drive>

Invalid input subdirectory at indicated address. No file whose directory entry is within that subdirectory is copied.

**BAD HEADER** <file-id> <disk-address> <drive>

Invalid input disk file header found; the associated file is not copied.

**BAD SECTOR** <file-id> <disk-address> <drive>

The data contained in this sector of the input disk could not be read without the occurrence of an exception condition. The data has been written to the output disk as is and must be manually verified.

**NOT ASSIGNED -- DS OR DP**

The requested output unit is not on-line.

## SECTION 32

### SYSTEM/DISK.INIT

The SYSTEM/DISK.INIT program is a normal-state utility program designed to label, initialize, verify, or reconfigure both disk cartridges and disk packs for use on B 1000 systems. Disk types that can be created with the SYSTEM/DISK.INIT program are System (S), User (U), and Interchange (I).

All disks must be initialized before they can be used with B 1000 system software. During initialization and verification, the SYSTEM/DISK.INIT program properly formats each sector by writing addresses, sync bits, data patterns, and error protection codes, and then reads the addresses and data to check for errors and to ensure that the sectors are not defective. If any of the first 64 sectors of the disk have to be removed, that disk cannot be used with B 1000 system software. Sectors are removed in the following two circumstances: (1) For a disk cartridge, if a sector is found to be bad, the entire track in which it resides is removed from the Master Available Table. (2) For a disk pack, if more than five sectors in the same cylinder are bad, the sixth and successive bad sectors are removed from the Master Available Table; otherwise, when a bad sector is found it is relocated to a spare sector.

The SYSTEM/DISK.INIT program processes each cylinder of a disk individually. The actual number of reads and writes performed on each cylinder is based on the initialization and verification characteristics of the disk device being processed. Cylinder 0, the outermost cylinder of the disk, is initialized first: label information and system tables are built in this cylinder at the end of the initialization process.

Disk device characteristics and initialization patterns are documented in appendix A of this volume.

## OPERATING INSTRUCTIONS

The SYSTEM/DISK.INIT program can be executed from the card reader or from the console keyboard. The program must be explicitly executed with the command EXECUTE SYSTEM/DISK.INIT. Once the program has begun processing a disk, that disk cannot be accessed by the system until the SYSTEM/DISK.INIT program has finished processing it. An OL inquiry reports the disk as IN USE BY SYSTEM/DISK.INIT while the program is running. After the disk is processed (initialized, verified, or reconfigured), the system attempts to recognize it and bring it on-line; however, if the pack type is System, the disk cannot be recognized and remains off-line.

### Console Keyboard Execution

When the SYSTEM/DISK.INIT program is executed from the console keyboard, the program prompts the operator for all initialization specifications. All operator input should be in the following format:

<job #>AX response to prompt

The prompt sequence is described in the subsection entitled Initialization Specifications.

During a single execution, the SYSTEM/DISK.INIT program can process several disks, one at a time. After initializing, verifying, or reconfiguring each disk, the SYSTEM/DISK.INIT program begins the prompt sequence again with the following message:

ENTER UNIT ID <DC? OR DP?>

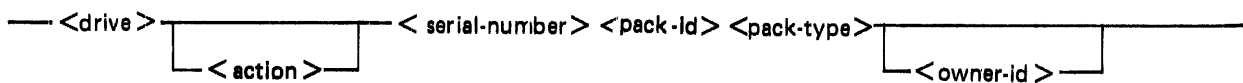
A blank or null input response to this prompt terminates the program.

## Card Reader Execution

The control card format used to execute the SYSTEM/DISK.INIT program from the card reader follows.

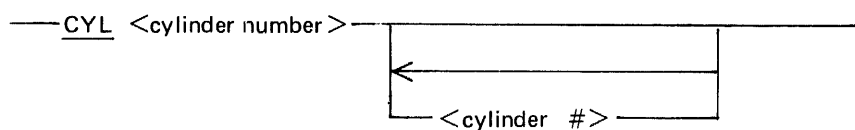
```
? EXECUTE SYSTEM/DISK.INIT FILE SPEC NAME <spec-file-id>  
? DATA <spec-file-id>  
  <specification cards>  
? END
```

All specifications are entered in free-form format (up to 96 columns). The first six specifications must be on a single card, and must be separated by one or more spaces; comma (,) characters are not valid separators. The initialization options and the cylinder specifications must be entered on separate cards. The first specification card maintains the same order as the prompt sequence and must be formatted as follows.



The semantics of the command are described in the following subsection entitled Initialization Specifications.

When the action CV (cylinder verify) or CI (cylinder initialize) is specified, the following card format must specify the cylinders to be processed.



If verification is requested (action V), then only the <drive> and <action> specifications are read; the remainder of the specification card is ignored.

The initialization specifications used when the SYSTEM/DISK.INIT program is executed from cards are identical to those used when it is executed from the console keyboard. The specifications are described in the subsection entitled Initialization Specifications.

## Run-Time Options

While the SYSTEM/DISK.INIT program is executing, the operator can interrogate its status with a <job #>AX ST inquiry. In response to this query, the SYSTEM/DISK.INIT program issues an ODT message that describes the cylinder that it is currently processing and the number of sectors that it has relocated or removed thus far in its processing. The format of this message follows:

```
CYLINDER = <# being processed> PASS <current I/V pass>  
RELOCATED = <#> REMOVED = <#>
```

Example:

```
CYLINDER = 243 PASS = 3 RELOCATED = 1 REMOVED = 0
```



## INITIALIZATION SPECIFICATIONS

### User Interface Information

#### Drive

A unit mnemonic which names the drive on which the disk to be processed is located (DC? or DP?).

Prompt: ENTER UNIT ID <DC? OR DP?>

#### Action

Specifies the type of processing to be performed. When the initialization action I, RI, RC, or V is requested, the SYSTEM/DISK.INIT program processes the disk in sequential cylinder order. When the requested action is CI or CV, the program issues prompts during processing that ask the operator to specify the cylinders to be processed.

Prompt: ENTER ACTION: <I,V,RC,RI,CI,OR CV>

These abbreviations request the following actions:

#### I = Initialization

Uses the default initialization characteristics for the device, followed by verification. The initialization characteristics are noted in appendix A of this volume.

#### V = Verification only

#### RC = Reconfiguration

Verification with relocation or removal of bad sectors. The disk is purged.

#### RI = Reinitialization

Initializes the disk using the default initialization/verification passes described in appendix A of this volume and automatically relocates or removes any sector that has its address in the sector table built during the previous initialization of the disk.

#### CI = Cylinder Initialization

Same as RI (reinitialization) except that the operator must specify which cylinders are to be processed. A later prompt requests the cylinder number.

#### CV = Cylinder Verification

Same as V (verification) except that the operator must specify which cylinders are to be processed. A later prompt requests the cylinder number.

#### NOTE

The actions RC, RI, CI, and CV are valid only when processing a disk that has previously been initialized with the Mark 8.0 version (or later) of the SYSTEM/DISK.INIT or PACK/INIT programs.

B 1000 Systems SSOG, Volume 2  
SYSTEM/DISK.INIT

Serial Number

A unique six-digit decimal number identifies the disk. This number must be non-zero and identifies a disk that is to be purged.

Prompt: ENTER 6 DIGIT SERIAL NUMBER

Pack Id

A unique name, six characters or less and containing no embedded blanks, that is assigned to the disk. The system uses this name to reference the disk. When input specifications are entered through the card reader, omitting this entry causes an error for pack-type.

Prompt: ENTER PACK.ID

Pack Type

Specifies the type of use to which this disk is dedicated and the type of access that is desired by the system software. The type is specified by a one-letter abbreviation of the allowable pack types, which are System, Unrestricted (User), Restricted, and Interchange. Only B 9499-7/8 disk packs are allowed to be initialized as Interchange packs; accordingly, the prompt message varies, depending on the type of disk being initialized.

Prompt: ENTER PACK TYPE - <U,S,OR R>  
(for disk types other than B 9499-7/8)

Prompt: ENTER PACK TYPE - <U,S,R,OR I>  
(for disk types B 9499-7/8)

Owner's Name

An optional 14-character field available to identify the person responsible for maintaining the disk. The system does not reference this field.

Prompt: ENTER OWNER'S NAME

Options

Several options are available during initialization of a disk to provide a more flexible initialization and verification procedure; they are described in the subsection entitled Initialization Options. The options allow the operator to extend or enhance the default initialization characteristics of the disk.

The OPTION prompt is repeated until a blank accept (AX) input message is received, at which point initialization, verification, or reconfiguration begins.

Prompt: ENTER OPTIONS

Cylinder Number

Requests the number of the cylinder that is to be processed. This information is required when either the cylinder initialize (CI) or cylinder verify (CV) action is specified. The SYSTEM/DISK.INIT program continually processes the specified cylinders and then reissues this prompt until it receives a blank input response. A single cylinder or a group of cylinders can be specified. The following are examples of this prompt and valid responses.

Prompt: ENTER CYLINDER NUMBER

Responses: <job #>AX 25 40      or      <job#>AX 300

When card input is used, the cylinder number specification must be on a separate card from the input specification, and the entry CYL must appear in columns 1 through 3, followed by the cylinder numbers to be processed. Additional cards can be used if needed.

### **Initialization Integrity Protection**

When the action CI is requested, the SYSTEM/DISK.INIT program prevents a disk from being only partially initialized by checking the disk initialization table maintained on the disk for the initialization status. If any sector on the disk is not initialized, the SYSTEM/DISK.INIT program automatically completes the initialization before processing any operator-specified cylinders. The ENTER CYLINDER NUMBER prompt is not issued until the program ensures that the disk is completely initialized.

This feature of the CI action allows a restart of either disk initializer program (SYSTEM/DISK.INIT and PACK/INIT) when a new disk is being initialized. The restart of the program processes only those sectors that are not initialized.

### **Initialization Options**

The operator can choose to extend or to enhance the default initialization characteristics for a disk by using the initialization options. The REMOVE and LIMIT options control the total number of errors that are allowed to exist on a disk. The Marginal Sectors option describes sectors that are known to be bad, and requests that they be relocated or removed. The Dollar Sign option controls the physical I/O operations that occur when processing a disk.

#### REMOVE Option

This option is valid only when initializing disk packs and allows the default error limit of five errors per cylinder to be overridden. If the REMOVE option is specified as an option, the first five bad sectors found in each cylinder are relocated. Additional sectors that are found to be in error are removed from the Master Available Table; thus, the pack is not rejected because the error limits have not been exceeded when there are more than five bad sectors in a single cylinder. If the REMOVE option is specified, a disk is not rejected unless the total number of errors on the disk exceeds the default error limit (refer to appendix A of this volume for a description of Disk Device Characteristics), or the error limit specified in the LIMIT option.

The REMOVE option is ignored when initializing disk cartridges or B 9499-6 disk packs; however, the REMOVE option is implicitly set since these disks do not have spare sectors that can be used for relocation. On disk cartridges, when a sector is found to be in error, the entire track in which the sector is located is removed from the Master Available Table.

The REMOVE option is requested by the keyword REMOVE. For execution from the card reader, the keyword must appear on a physically separate card from all other specifications. For execution from the console keyboard, the keyword is entered when the prompt ENTER OPTIONS appears.

#### LIMIT Option

This option allows the default error limit totals to be overridden for disk packs and disk cartridges. If the LIMIT option is requested, a disk is not rejected until the total error count for the disk exceeds the value specified in the option. For disk packs, the default error limit of five errors for each cylinder might cause a pack to be rejected, unless the REMOVE option has also been requested.

The LIMIT option is requested with the following syntax:

LIMIT <integer>

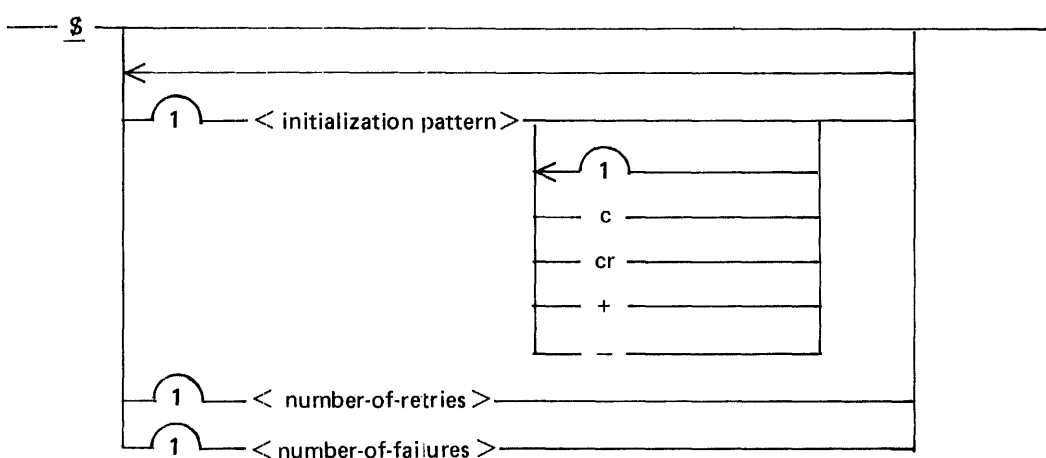
### Marginal Sectors Option

This option allows the operator to describe sectors that are known to be bad and to require relocation or removal. Marginal sectors that are removed or relocated are included in the error count totals. Sector addresses specified in this option must be valid decimal addresses. More than one sector address can appear in each accept response to the ENTER OPTIONS prompt or in each input specification card, with the restriction that the addresses must be separated by spaces.

If CI or CV (cylinder initialize or verify) is requested as the action, then the operator must ensure that the specified marginal sector resides within a cylinder that is to be processed (has been included in a CYLINDER NUMBER input specification).

### Dollar Sign Option

This option causes verification with or without offset, comparison during verification, modification on the number of retries on bad sectors, or modification on the number of failures that can occur before a sector is considered bad. The syntax of the dollar-sign option follows:



#### Semantics:

##### Initialization Pattern

The initialization pattern entry is a four-digit representation of a hexadecimal pattern within the range 0000 through FFFF. The first dollar-sign statement in a series of dollar-sign statements that omits this entry causes the default pattern to be used for that pass (refer to the disk device characteristics for the default patterns). Any successive dollar-sign statements apply to verification only, and only those entries described in the following are valid.

##### Offset: + or -

The plus sign (+) or minus sign (-) entry is used for disk packs to indicate that in (+) or out (-) offset is to be used during verification. Default is no offset.

The term offset refers to a disk pack capability which is a means of causing the disk unit to create a critical head alignment useful in testing marginal disk conditions. This capability is not available in Design Level I B 9499-7/B 9499-8 Disk Pack Drives or the Design Level I Disk Pack Electronics Controller (DPEC). Specifying offset on this level disk pack has no effect on initialization.

**Compare: C or CR**

The C or CR entry requests the options compare or compare-and-remove during verification of a disk cartridge. This option compares the data pattern that is written to the disk with the data pattern read during verification.

**Number-of-retries**

The number-of-retries entry specifies the number of times during verification that the same I/O operation is attempted after an exception condition has occurred. Default is ten retries.

**Number-of-failures**

The number-of-failures entry specifies how many of the retries specified by the number-of-retries entry are allowed to fail before the sector is considered bad and is relocated or removed. Default is one failure.

**NOTE**

If the default values for both number-of-retries and number-of-failures are used while verifying a user disk, all ten retries must be successful to allow the sector to remain.

## **PROGRAM OUTPUT**

The following paragraphs describe the program output.

### **Informative Messages**

The SYSTEM/DISK.INIT program issues a message that describes the type of processing that is beginning. These messages are identical to those included in the summary listing and vary with the type of processing (action) requested.

Also, the number of the printer backup file to which the summary listing has been directed is given at this time.

### **Summary Listing**

The SYSTEM/DISK.INIT program generates a printer listing that summarizes all activity that takes place during processing. This listing is directed to a printer backup file; the backup file number is given in an ODT message issued after the SYSTEM/DISK.INIT program processes its input specifications. The messages in this listing are identical to those messages displayed on the ODT by the SYSTEM/DISK.INIT program during its execution.

The following example illustrates the information contained in this listing when cylinder processing is requested.

```
INITIALIZATION BEGINS -CYL:0 -DPD
INITIALIZATION COMPLETE CYL:0 -DPD
INITIALIZATION BEGINS -CYL:1 -DPD
INITIALIZATION COMPELTE CYL:1 -DPD
ID=SNARFF SER#=022779 PACK TYPE=S
0 RELOCATED SECTORS
INITIALIZATION COMPLETE -DPD
```

B 1000 Systems SSOG, Volume 2  
SYSTEM/DISK.INIT

For processing that is not cylinder oriented, the following example illustrates the summary listing produced by the SYSTEM/DISK.INIT program.

```
INITIALIZATION BEGINS -DPD
ID=SNARFF SER#=022779 PACK TYPE=S
0 RELOCATED SECTORS
INITIALIZATION COMPLETE -DPD
```

### **Error Messages**

#### **INVALID ENTRY BLANK ID**

A blank prompt response is not valid at this time.

#### **INVALID ENTRY <entry>**

The prompt response entered was invalid; re-enter the response.

#### **INVALID SERIAL NUMBER**

Re-enter the serial number. It must be a six-digit, non-zero decimal number.

#### **INVALID SECTOR NUMBER**

The addresses of marginal sectors must be specified in decimal.

#### **THE FOLLOWING SECTORS ARE IN ERROR AND WILL BE RELOCATED**

This is an informative message issued when sectors on a disk pack are relocated.

#### **THE FOLLOWING SECTORS ARE IN ERROR AND THEIR TRACKS WILL BE REMOVED**

This is an informative message issued when bad sectors are encountered while initializing a disk cartridge.

#### **IS <unit-id> TO BE INITIALIZED?**

This is a safety measure to ensure that the operator has specified the correct disk. YES and NO are valid responses.

#### **WOULD YOU LIKE TO RETRY? <YES OR NO>**

This message follows the DISK ERROR message and offers the operator the option of retrying an I/O operation that has previously failed. A YES response causes the I/O operation to be retried. A NO response terminates processing of the current disk.

#### **DISK ERROR - RESULT = <result-descriptor>**

An irrecoverable I/O error existed after ten (or the value set in a Dollar Sign option) retries.

B 1000 Systems SSOG, Volume 2  
SYSTEM/DISK.INIT

**MUST RESTART TO CONTINUE**

SYSTEM/DISK.INIT cannot continue execution. Re-execute the program.

**WRITE LOCK OUT <unit-id>**

Reset the write lock-out pushbutton switch on the specified unit and execution continues.

**DISK NOT READY <unit-id>**

Ready the specified unit.

**PACK CANNOT BE USED WITH MCP II <unit-id>**

One of the first 64 sectors has been removed, or the Master Available Table is full. These are irrecoverable errors.

**ERROR CYL 0 <unit-id> <disk-address>**

This is an informative message.

**SECTOR REMOVED <disk-address>**

This is an informative message.

**COMPARISON ERROR <disk-address>**

The C or CR option is enabled, and a comparison error was detected.

**ERROR LIMITS HAVE BEEN EXCEEDED**

The maximum number of errors allowed for each cylinder (if the REMOVE option was not specified), or the total number of errors allowed for the disk (either the default value or the value specified by the LIMIT option), was exceeded during verification. This message also indicates that, due to removed sectors (REMOVE option), the Master Available Table is filled beyond capacity.

## SECTION 33

# SYSTEM/ELOGOUT

The SYSTEM/ELOGOUT program is a system utility program designed to analyze the Field Engineering Maintenance Log (SYSTEM/ELOG file) and produce highly organized, meaningful reports of the results. Errors are reported in chronological order by unit-mnemonic with appropriate totals summarized at the end of the output listing.

### OPERATING INSTRUCTIONS

#### Execution

The ET input message transfers the SYSTEM/ELOG file, creating a file labeled ELOG/<integer> and a new (empty) SYSTEM/ELOG file. The SYSTEM/ELOGOUT program is executed automatically by the MCPPII following the transfer and is file-equated to the transferred ELOG file.

If it is desired to analyze a previously-transferred ELOG file, the SYSTEM/ELOGOUT program must be executed as follows:

```
EXECUTE SYSTEM/ELOGOUT FILE ELOG.FILE NAME=ELOG/<integer>;
```

#### Program Switches

The program switches 0, 1, and 2 control certain functions of the SYSTEM/ELOGOUT program and are described as follows.

Switch	Description
0	Causes the input ELOG file to be closed with PURGE after the input phase has been completed. By default, the input ELOG file is not removed from disk. This switch must be set to 1 if the PURGE is desired.
1	Causes the entire output listing to be single-spaced, and suppresses all page skipping and multiple spacing operations. When not set, the default printer spacing is employed.
2	Causes the system serial number to be printed at the beginning of each individual report. No action is taken if no system serial number has been specified.

Also, the program switches specify the system serial number, as follows:

```
MODIFY SYSTEM/ELOGOUT SWITCH = <system number>
```

If it is necessary to specify the system serial number as well as other program switch controlled functions, the serial number must be specified first in the control string, as follows:

```
MODIFY SYSTEM/ELOGOUT SWITCH = 513;SWITCH 0 = 1;SWITCH 1 = 1
```



## DESCRIPTION OF OUTPUT

If the system serial number is specified, it is printed on the first page of the output listing.

A number of reports are produced by the SYSTEM/ELOGOUT program, providing a detailed analysis of errors by unit and hardware type. The output of these reports is described in the following paragraphs.

### System Information

The first portion of the output line printer listing produced by the SYSTEM/ELOGOUT program consists of information describing the system and the ELOG file being analyzed. The information includes the system serial number (if one was specified in the program switches) plus the name and period of time covered by the ELOG file being analyzed.

The dates and times of all CLEAR/START operations (if any) that occurred during the period of time covered by the ELOG are listed in the order they occurred.

A description of the system I/O configuration which lists each hardware device together with its device-id (control-id in hexadecimal) and hardware address (port/channel/unit) is included at least once. Additional descriptions are printed for each CLEAR/START operation where the I/O configuration differs from that at the previous CLEAR/START operation.

A description of the system software being used is included at least once. Additional descriptions are printed for each CLEAR/START operation where the software configuration differs from that of the previous CLEAR/START operation.

### Operator Messages

Messages entered in the ELOG file by the system operator (refer to the EM input message described in volume 1) are listed chronologically, together with the date and time of entry.

### Memory Error Report

Memory parity errors and related information are reported in the line printer listing generated by the SYSTEM/ELOGOUT program. The report produced for B 1700 and B 1830/B 1825 systems differs from the report produced for other B 1000 systems. The content of each report is described in the following paragraphs.

In the report for B 1700 and B 1830/B 1825 systems, memory parity errors are listed in chronological order. The report includes the date and time of the failure, the job number of the program that was terminated, the program's BASE and LIMIT addresses, and the address of the parity error. If the parity error could not be located by GISMO (possibly indicating a READ-OUT-OF-BOUNDS error by the program), the parity address is FFFFFFFF.

Memory parity errors for B 1000 systems other than the B 1830/B 1825 are reported differently due to the use of error correcting memory. The report includes the date and time of the error, the hexadecimal representation of the processor ELOG register (which reports the memory error to the software), and the decoded representation of this register. This decoded information enables the Burroughs field engineer to quickly isolate and repair the failure.

For memory parity errors that cannot be corrected, the job number of the program that was terminated and its BASE and LIMIT addresses are also included.

NOTE

B 1000 correctable memory errors do not cause a system or program failure, nor any measurable decrease in system throughput. Replacement of marginal memory chips is to be scheduled for a time that does not interfere with normal system operation.

**Console Cassette Error Report (B 1800 and B 1900 only)**

This report is produced if any data errors are detected during programmatic loading of a cassette through the cassette reader in the B 1800 and B 1900 system console (for example, with the SYSTEM/LOAD.CAS program). The report includes the date and time of the error, the job number of the program reading the cassette, and the program's BASE and LIMIT addresses.

**Unit Error Report**

Peripheral unit errors are reported and summarized by unit-mnemonic and, where multiple controls are connected to the same unit (for example, a PE/NRZ tape exchange), by hardware address. Each error reported on the unit is listed chronologically, together with the following information:

1. The date and time of the error.
2. The result descriptor in hexadecimal.
3. The I/O operation code and its mnemonic description (R = Read, W = Write, S = Space).
4. The disk address (disk devices only).
5. The data length (in bytes) requested in the I/O descriptor. the
6. The actual length (in bytes) of the data transferred.
7. The number of retries attempted (often shown as zero for MCP II operations, even though retries are actually attempted).
8. A code indicating the success or failure of the recovery operation.
9. The job number of the program encountering the error.
10. The label of the file (always).
11. The pack-id (for disk devices).
12. The serial number (for tapes and disk packs/cartridges).
13. The reel number (tapes only).
14. The Extended Result Description (if any).

If an Extended Result Descriptor is present, it is decoded on the following lines into an alphanumeric description. An asterisk next to the job number indicates that the ELOG entry was made by the MCP II at the request of the specified program.

Following the chronological error listing, an analysis of the result descriptors is printed. The result descriptors are listed in both hexadecimal and binary format, together with an alphanumeric description of the error. Each result descriptor is listed once in the summary, regardless of the number of times it occurred in the chronological error listing.

For disk devices, an additional report analyzes the errors which are grouped by disk address, sorted in ascending order, and summarized by address and actual data transfer length (in sectors). Each line represents the errors for a particular disk address and actual length pair, giving the total number of retries (retry counts shown as zero in the chronological listing are counted as one retry), and decoding the beginning and ending disk addresses involved in the operation into their actual hardware locations. This decoding can enable the Burroughs field engineer to locate not only the source of the failure, but also to make reasonable inferences as to errors that can be caused by a single read/write head, even though the addresses can be widely scattered across an entire disk surface.

B 1000 Systems SSOG, Volume 2  
SYSTEM/ELOGOUT

**Unit Error Summary**

This report summarizes the peripheral unit errors listed in the Unit Error Reports, providing totals for each unit-mnemonic (and hardware address, if applicable). For each device listed, the total errors occurring on input (read and space operations) and output (write operations), as well as the total number of memory access errors, are presented in tabular form.

The Unit Error Summary also includes totals for any console cassette errors or main memory errors that occurred.

## SECTION 34

### SYSTEM/FILE.INIT

The SYSTEM/FILE.INIT program is a utility program that can allocate and initialize a relative file for COBOL74 programs. The format and the initialization of the B 1000 implementation of relative files as defined for the COBOL74 language are described in this section.

#### RELATIVE FILES

A relative file consists of records identified by relative record numbers or keys. The file can be thought of as composed of a serial string of areas, each capable of holding a logical record. Each of these areas is addressed by a relative record number. For example, the tenth record is the record addressed by the relative record number ten. The tenth record is in the tenth record area, regardless of whether records have been written in the first through the ninth record areas.

These record areas are of fixed length, the value of which is specified by the file attribute MAXREC-SIZE. Each record area can contain one record. A record area is empty if it contains no valid record. Full record areas can be emptied by deleting the record they contain, making the contents inaccessible through the normal mechanism. A DELETE verb is implemented in the operating system for this purpose.

The record areas are grouped into blocks of one or more records, and like the record size, the number of records in each block cannot be changed. Appended to the beginning of the block is a field known as the Block Control Information (BCI). The BCI consists of one bit per record area, plus enough filler bits to make the size of the BCI modulo eight. The bits in the BCI field are used to indicate the presence or absence of a valid record in the associated record area. A value of one indicates that a record is present in the associated area. The SYSTEM/FILE.INIT program allocates the area and initializes all of the necessary presence bits, setting them to a value of zero. As a consequence of this design, the size of a block in a relative file is equal to the number of records in each block times the record size plus the BCI.

#### INITIALIZATION

The execution of this program is one means of accomplishing initialization; however, the MCP II has the ultimate responsibility of ensuring that areas are initialized.

The manner in which the file is accessed determines the method of initialization the MCP II uses. If the access mode of the file is sequential, the MCP II allocates the area and the logical I/O routines initialize each block before access. In this case, use of the SYSTEM/FILE.INIT program is not required. If the access mode is random or dynamic, and if a new disk area is allocated to the file, the MCP II automatically executes the SYSTEM/FILE.INIT program to ensure that the BCI is initialized. If this is the case, the operator program which caused the area to be allocated is stopped and the SYSTEM/FILE.INIT program is executed at the same priority level as the requesting program. If a relative file being accessed in the sequential mode is closed with the end-of-file (EOF) pointer not at the end of an area, the MCP II causes the execution of the SYSTEM/FILE.INIT program to initialize the remainder of that area before the CLOSE operation is completed.

If the SYSTEM/FILE.INIT program is called by the MCP, as previously described, the requesting program is not allowed to execute until the SYSTEM/FILE.INIT program has completed the initialization of the area. The mechanism used for calling the relative initialization program is similar to that used

for a program sort. The program attempting the write operation (which requires the use of the SYSTEM/FILE.INIT program) is suspended. Upon successful completion of the initialization, the program is reinstated. Even if an irrecoverable error is detected during initialization, the program is still reinstated. Any USE procedures declared for the file are executed at this time.

## OPERATING INSTRUCTIONS

Operating instructions are not required by the SYSTEM/FILE.INIT program when it is executed automatically by the MCP. Only when initializing a file prior to an actual production run is it necessary to provide information to the SYSTEM/FILE.INIT program. Operators can initialize the file prior to the time it is actually required to ensure that the required disk space is available and to minimize the amount of processing actually required when a record is added to a new disk area. The latter consideration can be important in an on-line environment where it is desirable to minimize response time.

## OPERATOR EXECUTION

Operators are responsible for providing file attributes for each file to be initialized. Attributes are entered from one of three sources determined by program switch 0. These attributes are shown in table 34-1. Keywords are used for card image or Operator Display Terminal (ODT) input. File attributes are set by entering keyword-value pairs of the form <keyword> = <value>. Input is as follows: a TITLE specification followed by its attributes, followed by another TITLE specification and its attributes. Input is basically in free-form format, the only restriction being that keyword-value pairs must not be split across record boundaries. Since the SYSTEM/FILE.INIT program handles only relative files, RELATIVE should be specified as the file kind value.

Table 34-1. File Attributes

Attribute	Keyword	Default
File name	TITLE	none
Record size	RECSIZE RECORD.SIZE RSZ	175
Records per block	RECORDS.BLOCK R.B	1
Blocks per area	BLOCKS.AREA B.A	100
Number of areas	AREAS ARE	25
File kind	FILEKIND FILE.TYPE	none

An example of the instructions needed to execute the SYSTEM/FILE.INIT program follows.

Example:

```
?EXECUTE SYSTEM/FILE.INIT;SWITCH 0=1;
?DATA CARDS
TITLE =MY/FILE2 RECSIZE = 80 BLOCKSIZE = 2
TITLE =MY/FILE3 TITLE =MY/FILE4 AREAS = 105
TITLE =PACKID/MY/FILE5
RECSIZE = 80
BLOCKSIZE = 2
AREASIZE = 4
AREAS = 105
FILEKIND = RELATIVE
?END
```

If only one file is to be initialized, a file-equate is probably the easiest means of executing the program. A sample execution follows. The defaults shown in table 34-1 apply to any attributes not specified in the FILE cards. Switch 0 is defined under Switch Settings.

Example:

```
? EXECUTE SYSTEM/FILE.INIT; SWITCH 0=0;
? FILE RELFILE NAME TEST/F1 RECORD.SIZE 80 RECORDS.BLOCK 19
? BLOCKS.AREA 10 FILE.TYPE = RELATIVE; % AREAS DEFAULT TO 25
```

## MCPII INTERNAL USE

When the MCPII automatically executes the program, it passes three items of information to the initialization program by means of a file equation to RELFILE. The instructions that perform the call are:

```
? EXECUTE SYSTEM/FILE.INIT SWITCH 9=1;
? FILE RELFILE NAM <file name> AREAS <area #>;
```

The three items of information are:

1. The file name.
2. The area to be initialized.
3. The MCPII request that only one area is to be initialized (indicated by program switch 9 set to a nonzero integer as shown in the previous instructions).

## SWITCH SETTINGS

Program switch 0 determines the source of the file attributes.

SW0=0 indicates that the source is a file equation to RELFILE.

SW0=1 indicates that the source is card input.

SW0=2 through 15 indicate that the source is ODT input.

Program switch 9, set to any nonzero number, specifies that: (1) the program was initiated automatically by the MCPII, (2) only one area is to be initialized, and (3) the file is a relative file. This switch is reserved for use by the MCPII and operators are prohibited from setting it.

## ERROR MESSAGES

When the program is executed automatically by the MCP II, error messages are not displayed; however, three conditions can be returned to the calling program.

1. The file is not present on disk.
2. The file is locked (by some other program).
3. An irrecoverable I/O error occurred on initialization.

These conditions are stored in the file status variable for the appropriate file. The calling program must take action based on these conditions. The first two conditions are unlikely to occur because the calling program is either creating the file (in the case of a new file), or accessing an existing file. The third condition is more likely to occur; therefore, it is advisable that the calling program be able to handle the error condition or be discontinued.

When the program is executed from the ODT or from the card reader, the following error messages are subject to display.

INVALID <token name> -- <value>

The name of the token entered was a valid attribute name but the value associated with it exceeded the limits of the MCP II on that attribute. The values allowed for the relevant attributes are presented in volume 1, section 2 of the B 1000 Systems System Software Operation Guide.

INVALID TOKEN -- <name>

The attribute specified by <name> is not recognizable by the program.

EQUAL SIGN EXPECTED

The equal sign (=) character between the attribute and its value is mandatory.

VALUE EXPECTED AFTER =

No value was found in the input string after the equal sign (=) character in an attribute specification. When the input is from cards, the attribute value must be contained on the same card as the attribute name.

UNEXPECTED TOKEN -- <name> -- TITLE EXPECTED

The program was looking for a TITLE attribute and encountered a different attribute name.

<file name> NOT INITIALIZED FILE MISSING

<file name> NOT INITIALIZED FILE LOCKED

<file name> NOT INITIALIZED EXCEPTION ON WRITE

The file was not initialized due to the cause displayed. The first two messages cannot occur when the program is executed from the ODT or from the card reader, because in these cases, the program always creates a new file; thus, the program cannot be used to initialize one area of an existing file.

## INTERNAL FILES

The name and function of each file used by the SYSTEM/FILE.INIT program follows.

<b>Internal File Name</b>	<b>Function</b>
RELFILE	Disk file
CARDS	Card file



## SECTION 35

### SYSTEM/ICMD.INIT

The SYSTEM/ICMD.INIT program is a normal-state utility program that is designed to initialize, verify, and label the industry-compatible mini-disk (diskette or floppy disk). The program assigns addresses to the appropriate sectors, writes a fixed data pattern in the sector data area, and reads this pattern back to ensure that the sectors are not defective.

If a sector cannot be read after 10 retries or cannot be written after two retries (whether the error is a sector address error or a data error), the sector is considered bad and the entire track is relocated to a spare track. A diskette is considered bad and unusable if there are more than two defective tracks.

### OPERATING INSTRUCTIONS

The SYSTEM/ICMD.INIT program operates under control of the MCP II and receives input through accept (AX) input messages entered from the console keyboard. A listing is produced containing the test data pattern used, relocated tracks, and any errors that occurred during program execution.

Executing the program with switch 9 = 1 allows a operator-specified test pattern to be entered. If switch 9 = 0, the default pattern of @6363@ is used.

The following messages are displayed during execution and require an operator response:

ENTER UNIT NAME, FD<?>

Respond with the unit-mnemonic of the diskette to be initialized.

DISKETTE ON UNIT <unit-mnemonic> IS ALREADY INITIALIZED, ENTER  
Y FOR RE-INIT (BLANK FOR EOJ)

This message is displayed only if the specified diskette is already initialized. A Y response performs the requested initialization; otherwise, the program terminates.

FOR INTERLACED ADDRESSES ENTER Y (BLANK FOR SEQUENTIAL)

A Y response interlaces the sector addresses in the following order: 1, 14, 2, 15, 3, 16, ..., 12, 25, 13, 26.

ENTER PATTERN - 4 HEX NUMBERS (BLANK FOR DEFAULT OF @6363@)

This message is displayed only if the SYSTEM/ICMD.INIT program is executed with switch 9 = 1. Four hexadecimal digits are required for the response.

DISKETTE GOOD

This message is displayed twice if the initialization is successful and the diskette is good. The standard pattern of @E5E5@ is written on the diskette when the second message is displayed.

FOR RETRY PATTERN ENTER Y (BLANK FOR EOJ)

This message is displayed only if the SYSTEM/ICMD.INIT program is executed with switch 9 = 1. A Y response causes the program to request a new test pattern. A blank causes the ending pattern of @E5E5@ to be written.

ENTER 1 TO 6 ALPHANUMERIC CHARACTERS FOR VOLUME ID OR LEAVE BLANK

Respond with the new diskette volume-id (a blank results in a blank volume-id).

## ERROR MESSAGES

### INVALID UNIT NAME, RE-ENTER

The <unit-mnemonic> was not in the form FDx.

### \*\*ERROR\*\*PORT AND CHANNEL NOT ASSIGNED

Unit cannot be accessed. The program is terminated, and must be re-executed.

### ILLEGAL PATTERN - RE-ENTER 4 HEX NUMBERS (BLANK FOR DEFAULT OF @6363@)

Each character must be a valid hexadecimal digit.

### PATTERN TOO LONG, RE-ENTER

Pattern must be four hexadecimal digits in length.

### \*\*I/O OPERATION NOT COMPLETE

The diskette control has control errors.

### <unit-mnemonic> NOT READY, ENTER Y FOR RETRY (BLANK FOR EOJ)

The diskette is not ready.

### \*\*ERROR\*\*MORE THAN 2 BAD TRACKS

Diskette is bad and unusable. The program is terminated.

## SECTION 36

### SYSTEM/IS.MAINT

The SYSTEM/IS.MAINT program is a normal-state utility program that performs library maintenance functions for index sequential files.

Index sequential files consist of one cluster file, one direct file, and any number of index files.

The cluster file has the same name as the index sequential file and contains a dictionary that lists the names of all associated subfiles (direct and index files).

## OPERATING INSTRUCTIONS

### Execution

This program can be executed from (1) the Operator Display Terminal (ODT), (2) a card reader, or (3) a pseudo-deck. Program commands can be entered from the ODT or from a data file. If program input is through the ODT, the program is executed by entering:

```
EXECUTE SYSTEM/IS.MAINT
```

When the program goes to beginning of job (BOJ), it responds with accept messages requesting program input. If program input is from a data file, a file equation must be entered at the time of the execute as shown in the following examples:

```
EXECUTE SYSTEM/IS.MAINT;FILE SOURCE NAME CARD.FILE CARDS
```

```
EXECUTE SYSTEM/IS.MAINT;FILE SOURCE NAME IS/INPUT DISK
```

### Input From a Data File

Input from a data file requires that the internal file SOURCE be file-equated to the desired input file at execution time. The file is opened using input blocking and each record in the file corresponds to a single accept (AX) input message from the ODT. Command prompts that normally appear are not displayed.

### Input from the ODT

The name of the cluster file upon which maintenance is to be performed is entered in response to the command prompt ENTER FILE NAME. If the file cannot be found or is invalid, the program informs the B 1000 system operator and repeats the accept operation. If the file is found and it is a valid cluster file, the program responds by displaying the message ENTER COMMAND and then waits for an accept operation. All commands entered from this point are performed on the previously specified cluster file.

B 1000 Systems SSOG, Volume 2  
SYSTEM/IS.MAINT

The following example shows program execution from the ODT.

Example:

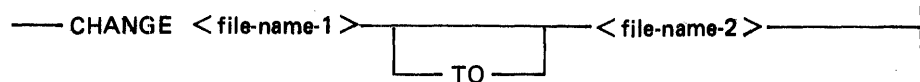
```
EXECUTE SYSTEM/IS.MAINT;
SYSTEM/IS.MAINT =38 BOJ. PP=4, MP=4 TIME = 10:30:37.7
% SYSTEM/IS.MAINT =38 ENTER FILE NAME
SYSTEM/IS.MAINT =38 ACCEPT
38 AX CL/FILE
%SYSTEM/IS.MAINT =38 ENTER COMMAND
SYSTEM/IS.MAINT =38 ACCEPT
38 AX LIST ODT
% SYSTEM/IS.MAINT =38 FILE NAME : CL/FILE
% SYSTEM/IS.MAINT =38 FILE TYPE : INDEX SEQUENTIAL
% SYSTEM/IS.MAINT =38 # SUBFILES : 2
% SYSTEM/IS.MAINT =38
% SYSTEM/IS.MAINT =38 KEY/1
% SYSTEM/IS.MAINT =38 CL/DATA
% SYSTEM/IS.MAINT =38 ENTER COMMAND
SYSTEM/IS.MAINT =38 ACCEPT
38 AX CHANGE KEY/1 CL/KEY1
"KEY/1" CHANGED TO "CL/KEY1"
% SYSTEM/IS.MAINT =38 1 STRUCTURE CHANGED
% SYSTEM/IS.MAINT =38 ENTER COMMAND
SYSTEM/IS.MAINT =38 ACCEPT
38 AX NEXT
% SYSTEM/IS.MAINT =38 ENTER FILE NAME
SYSTEM/IS.MAINT =38 ACCEPT
38 AX EMPLOYEE/INFO
% SYSTEM/IS.MAINT =38 ENTER COMMAND
SYSTEM/IS.MAINT =38 ACCEPT
38 AX VERIFY
SYSTEM/IS.MAINT =38 "LINE" = "BACKUP.PRT/19"
SYSTEM/IS.MAINT =38 "BACKUP.PRT.19" RELEASED
% SYSTEM/IS.MAINT =38 VERIFICATION COMPLETE -3 ERRORS
% SYSTEM/IS.MAINT =38 ENTER COMMAND
SYSTEM/IS.MAINT =38 ACCEPT
38 AX
SYSTEM/IS.MAINT =38 EOJ. TIME = 10:32:53.9
```

## COMMAND SYNTAX OVERVIEW

Commands are entered from the ODT in response to the prompts, or from a data file. The following paragraphs describe each of the SYSTEM/IS.MAINT program commands.

### CHANGE Command

Syntax:



The CHANGE command changes the name of the file in the disk directory and modifies the data in the cluster file to reflect the new name. In all cases, the names of files that are part of an index sequential file can be changed by this command as opposed to using the similar Master Control Program II (MCPII) construct. The names used within this instruction can take either of these forms:

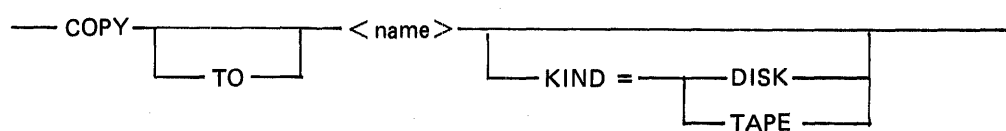
<pack-id>/<multifile-id>/<file-id>

<pack-id>/<multifile-id>/=

Both names must have the same pack-id. The first name used must be on disk and in the cluster file dictionary, whereas the second name must be neither on disk nor in the cluster file dictionary.

### COPY

Syntax:



The COPY option generates a COPY message for the MCPII which causes the cluster file and all associated subfiles to be copied by the SYSTEM/COPY program. The SYSTEM/IS.MAINT program resumes execution when the SYSTEM/COPY program has gone to end of job (EOJ). If any subfiles are missing, the command is not processed. Index sequential files can be reloaded with the following MCPII instruction.

COPY =/= FROM <tape name>

Subfiles can also be distributed to different packs as part of the load operation by means of the syntax provided in the SYSTEM/COPY program. To accomplish this, the operator must know the names of all the files involved and their associated keys. It is also necessary to modify the cluster file to reflect the names of the packs that now contain each of the files. Refer to UPDATE in this subsection for additional information for updating files.

## END, EOJ, or <job#>AX<null>

Syntax:

```
—END —————|
—EOJ —————|
—< job-number> AX <null > —————|
```

Entry of the commands END or EOJ, or the entry of an accept (AX) input message with a null character terminates the program.

## HELP

Syntax:

```
—HELP —————|
```

In response to the HELP command the program displays its operating instructions on the ODT. This command functions in the same manner as the TEACH command.

## LIST

Syntax:

```
—LIST —————|
      |             |
      | ODT         |
      | PRINTER    |
      |             |
```

The LIST command instructs the program to produce a listing of the name and version dates of the cluster file, direct file, and all index files. If ODT is specified as the list destination, the listing is displayed on the ODT. If PRINTER is specified, a line printer listing is produced. The default destination is the line printer.

## NEXT

Syntax:

```
—NEXT —————|
```

The NEXT command provides the capability to reference another cluster file. The program repeats the process of displaying the prompt ENTER FILE NAME and performs the accept operation. After the new cluster file name is specified, command entry proceeds.

## REMOVE

Syntax:

— REMOVE —————|

The REMOVE command removes the cluster file and all of its associated subfiles, the names of which are contained in the cluster file. For a file to be removed, it must have the correct version date and no users. If a subfile is missing, the program displays the following message.

NOT ALL STRUCTURES ARE PRESENT  
DO YOU WANT THE REMOVE TO CONTINUE? - YES OR NO

If the operator enters a NO response, the program asks for another command. If the response is YES, the program removes the files that are present. After completing the remove operation, the program repeats the process of displaying an ENTER FILE NAME prompt and performs the associated accept operation.

## TEACH

Syntax:

— TEACH —————|

In response to the TEACH command, the program displays its operating instructions on the ODT. This command functions in the same manner as the HELP command.

## UPDATE

Syntax:

— UPDATE <file-name> [ ] TO [ ] <file-name> —————|

The UPDATE command modifies data in the cluster file dictionary to reflect the new names of direct or index files. The names can take one of these forms:

<pack-id>/<multifile-id>/<file-id>

<pack-id>/<multifile-id>/=

<pack-id>/=/=

If the first name used cannot be found in the cluster file, or if the second name is already present in the cluster file, an error message is displayed. The program makes no effort to determine if these files are actually on disk.

## VERIFY

Syntax:

— VERIFY —————|

The VERIFY command determines if all the subfiles are present in the disk directory and if their version dates are correct. A hardcopy summary of the verification results is listed on the line printer and the results are also displayed on the ODT.

## ERROR MESSAGES

Error messages for the SYSTEM/IS.MAINT program are displayed on the ODT and are preceded by the word ERROR or WARNING.

Those error messages preceded by the word ERROR are displayed when the command is entered incorrectly and consequently cannot be processed. Messages in this category and their descriptions are listed in alphabetical order.

### ERROR - BOTH NAMES ARE THE SAME

The file names used in the UPDATE or CHANGE commands cannot be the same.

### ERROR - CLUSTER FILE IN USE BY ANOTHER PROGRAM

No other users can be accessing the cluster file while the SYSTEM/IS.MAINT program is accessing it.

### ERROR - CLUSTER FILE IS CORRUPTED

The cluster file dictionary of subfiles could not be read.

### ERROR - CLUSTER FILE NOT FOUND

In response to the ENTER FILE NAME prompt, a cluster file was specified that could not be found. The operator must ensure that the file is on the system.

### ERROR - COMMAND NOT ALLOWED IN THIS CONTEXT

The program expected the entry of a file name. Commands acting on a file are not allowed.

### ERROR - COMMAND REQUIRES PARAMETERS

The command just entered required parameters that were omitted. Refer to the appropriate command syntax.

### ERROR - DIFFERENT PACK-IDS NOT ALLOWED IN CHANGE

The pack-ids used in the CHANGE command must be the same. Refer to the CHANGE command syntax.



**ERROR - ID EXCEEDS 10 CHARACTERS**

The pack-id, multifile-id, or file-id name cannot contain more than 10 characters each. Refer to <file id> of the File Specification Clause in the SYSTEM/COPY section for B 1000 file naming conventions.

**ERROR - INVALID COMMAND, TRY "TEACH" FOR LIST OF VALID COMMANDS**

Either the command was erroneously entered or the command is not recognized by this program. Enter the TEACH or HELP command for a listing of valid commands.

**ERROR - MCPH DENIED COPY REQUEST**

The MCPH failed to process this command, possibly because of invalid file or tape names. Refer to the SYSTEM/COPY program for further explanation.

**ERROR - MISSING ENDING QUOTE**

The file name requiring quoted text lacks the final quotation character.

**ERROR - MORE PARAMETERS EXPECTED**

The program expected more parameters than were entered. Refer to the command syntax or operating instructions.

**ERROR - NAME CONTAINS ILLEGAL DELIMITER**

Special characters within a file name such as the semicolon (;) require enclosing quotation marks. Ensure that file names conform to B 1000 file-naming conventions.

**ERROR - NAMES HAVE UNMATCHED EQUALS**

The UPDATE or CHANGE commands have been used improperly and the file names have mismatched equal sign (=) characters. Refer to the command semantics for additional information.

**ERROR - NOT AN INDEX SEQUENTIAL FILE**

The file is not index sequential.

**ERROR - ONLY "ODT" OR "PRINTER" ALLOWED AFTER "LIST"**

The ODT or line printer are the only permissible output destinations. Refer to the command syntax.

**ERROR - ONLY THREE IDS ALLOWED IN NAME**

The file name can contain only the pack-id, multifile-id, and file-id. Refer to the B 1000 file-naming conventions.

**ERROR - STRUCTURE IS NOT PART OF THIS FILE**

The structure was erroneously specified.

B 1000 Systems SSOG, Volume 2  
SYSTEM/IS.MAINT

**ERROR - UNEXPECTED TEXT FOLLOWING <COMMAND or NAME>**

More than one command was encountered. Only one command is allowed for each accept (AX) input message.

**ERROR - "/" IS INCORRECTLY USED**

File names such as /A/B or A//B, for example, are not allowed. Ensure that the use of the slash (/) character conforms with file-naming conventions.

**ERROR - "=" NOT ALLOWED AS <PACK-ID, MULTIFILE-ID, or FILE-ID>**

In some cases, the equal sign (=) special character is not allowed as a replacement for the pack-id, multifile-id, and/or file-id. Refer to the command syntax.

Error messages preceded by the word WARNING are displayed when a command being processed encounters problems with part of the task. The commands REMOVE, CHANGE, and UPDATE are subject to this type of error message because action can be required on more than one file. These error messages have the following format.

**WARNING - <file name 1> NOT < REMOVED, CHANGED, or UPDATED >  
[ TO <file name 2>] - <explanation>**

The accompanying <explanation> can be any of the following:

MCPII DENIED REQUEST  
NOT ON DISK  
PACK NOT ON-LINE  
ALREADY IN DICTIONARY  
ALREADY ON DISK  
VERSION MISMATCH  
HAS USERS

## INTERNAL FILES

The following is a list of the name and function of each file used by the SYSTEM/IS.MAINT program.

Internal File Name	Function
IS_FILE	The cluster file that is currently being accessed.
COPY	Used to pass copy requests to the SYSTEM/COPY program.
SOURCE	If file-equated to a name other than IS.MAINT/IS.MAINT/IS.MAINT, this file is used for input commands instead of the ODT.

## SECTION 37

### SYSTEM/LDCONTRL

The SYSTEM/LDCONTRL (LOAD CONTROL) program is a normal-state utility program that creates pseudo-reader files. Physical card decks are the primary source for pseudo file input. Pseudo-reader files are often referred to as pseudo decks.

A pseudo deck (file) is a special disk file (FILE.TYPE=3) that is processed as though it were a card deck. All system control messages in a pseudo deck are processed by the control card driver in the MCPPII; thus, only the first 72 columns of each record are processed. Each record of the input file is represented by a 96-column card image in the pseudo deck.

Invalid characters in control records are replaced by an EBCDIC question mark character (@6F@) in the pseudo deck.

Pseudo decks have a file identifier of DECK/<nnnn>, where <nnnn> is a unique five-digit number assigned by the MCPPII. Numbers are assigned sequentially, beginning with 00001 and incremented as each new file is assigned. The counter is reset when there are no pseudo decks in the disk directory, or when the count reaches 9,999.

The CD, ED, IL, LD, OL, RD, and RN input messages operate on pseudo decks. Refer to volume 1, section 2 for documentation on these commands.

## OPERATING INSTRUCTIONS

Primary operation of the SYSTEM/LDCONTRL utility program is through the LD input message. An explicit EXECUTE SYSTEM/LDCNTRL command is also valid.

The SYSTEM/LDCONTRL program expects a card data file identified as ? DATA CTLDCK. This data file contains card files to be loaded into pseudo decks; one or more card files can be included in the input data file CTLDCK. Each card file must be identified with a ? DATA <filename> card, and ended with a ? END card. The input data file CTLDCK must be terminated with a ? ENDCTL card.

The SYSTEM/LDCONTRL program recognizes all end-of-deck (? END) control cards. Each time an end-of-deck control card is encountered, the SYSTEM/LDCONTRL program closes the file that it is currently creating; thus, each card file is loaded as a separate pseudo deck. All control statements, including the end of deck card, that are included in each card deck are written to the pseudo deck along with all normal data cards. Control statements are cards that have an invalid character or a question mark (?) character in column 1.

Because the SYSTEM/LDCONTRL program accepts the control message ? END as program input, the special message ? ENDCTL must be used to terminate the input data file CTLDCK.

B 1000 Systems SSOG, Volume 2  
SYSTEM/LDCONTRL

An example of an input data file CTLDCK follows.

```
? DATA CTLDCK
? EXECUTE A/B;
? PRIORITY = 5;
? DATA CARDS
  data for file CARDS
? END %end of deck 1
? DATA INPUT
  data for file INPUT
? END
.
.
.
? ENDCTL
```

## GENERAL RULES

1. A card file labeled ?CTLDCK cannot be read by a pseudo reader.
2. A maximum of 9,999 cards can be loaded into a single pseudo deck.
3. When reading from a pseudo reader, the MCP II recognizes a question mark (?) character in column 1 of a card image as the control card indicator. Any invalid character in the input card file is translated to a question mark (?) character when the pseudo file is created.
4. The STREAM and TERMINATE cards replace DATA and END cards in the input data file CTLDCK. In this case, the following four restrictions apply.
  - a. The STREAM or TERMINATE card must contain an invalid character in column 1, and the keyword STREAM or TERMINATE must be the first token on the card after the invalid character in column 1.
  - b. The file-id on the STREAM and TERMINATE cards must match, and the identifier on each TERMINATE card must match the identifier on the immediately preceding STREAM card, or the TERMINATE card is included in the pseudo deck.
  - c. No embedded (nested) STREAM or TERMINATE cards are allowed.
  - d. Any invalid character found in column 1 of the card file delimited by the STREAM and TERMINATE cards is translated to the null character (@00@) rather than the question mark (?) character; thus, these card images are not recognized as control cards when the pseudo deck is read.
5. Pseudo decks can only be removed by the ED and RD input messages (refer to volume 1, section 2). The only way to prevent a pseudo deck from eventually being allocated to a pseudo reader and being read is to remove the deck.
6. When the SYSTEM/LDCONTRL program is ZIP-executed, either explicitly or implicitly (with the EXECUTE or LD input message), job spawning information is included on the first record of each pseudo deck. If the spawning program has gone to EOJ when the pseudo deck is read, the MCP II removes the pseudo deck without processing it.
7. Pseudo deck characteristics are:

Record size: 288 bytes (Three 96-character card images)  
Blocking factor: 1 record per block  
Records per area: 100  
Multifile-id: DECK  
File type: 3 = pseudo deck
8. The internal identifier of the input file is CARD.IN.

## SECTION 38

### SYSTEM/LOAD.CAS

System programs such as compilers, interpreters, object code, and system software can be loaded from a cassette to the system disk by using the SYSTEM/LOAD.CAS program. Cassettes used for input to this program must be created using the CASSETTE/MAKER utility program.

This program is executed from the Operator Display Terminal (ODT) using a standard EXECUTE statement. After the program goes to beginning of job (BOJ), an accept (AX) input message is issued requesting the name of the file to be loaded. The file name can be entered as <multifile-id>/<file-id>, <multifile-id>/=, or /=. The file name must conform to the file-naming conventions prescribed for B 1000 systems. After the program loads the specified file, it requests additional file entries. A null entry terminates the program.

Example:

```
EXECUTE SYSTEM/LOAD.CAS
SYSTEM/LOAD.CAS =73 BOJ.
% SYSTEM/LOAD.CAS =73 ENTER NAME OF FILE TO LOAD
SYSTEM/LOAD.CAS =73 ACCEPT.
73 AX COBOL/=
```

After a file or files have been loaded, the cassette must be rewound. If rewinding is neglected, an error message is issued when a subsequent program load is attempted. If more than one cassette is needed, the following message is issued.

```
MOUNT NEXT CASSETTE AND <job #>AX
```

## ERROR MESSAGES

The following error messages are generated by the SYSTEM/LOAD.CAS program. The error messages appear alphabetically with their descriptions.

### BAD HEADER

The reel number could not be read due to a bad header.

```
CASSETTE READ ERROR - <file name> NOT LOADED
```

The hash-total is incorrect; possible causes are:

Bad cassette

Cassette not rewound

```
<file name> NOT LOADED DUE TO OPERATOR INTERVENTION
```

File not loaded.

B 1000 Systems SSOG, Volume 2  
SYSTEM/LOAD.CAS

INCORRECT CASSETTE

PLEASE LOAD REEL NO. <reel #> AND <mix #> AX

INVALID FILE NAME

The file name did not conform with B 1000 file-naming conventions; either there were unmatched quotation mark (") characters, more than 10 characters in the file-id, or the equal sign (=) character was used as only a multifile-id and not as a file-id as well.

INVALID FIRST CHARACTER IN FILE NAME

The percent sign (%), blank, semicolon (;), comma (,), and virgule (/) characters are not allowed as the first character in a file name.

<file name> NOT FOUND

The file was not on cassette.

## **SECTION 39**

### **SYSTEM/LOAD.DUMP**

The SYSTEM/LOAD.DUMP program is a normal-state utility program that copies a file or group of files from a SYSTEM/LOAD.DUMP tape to disk. A SYSTEM/LOAD.DUMP tape is a tape that was created by the SYSTEM/LOAD.DUMP program prior to the Mark 9.0 system software release. The program does not create tapes since the SYSTEM/COPY program performs that function.

#### **OPERATING INSTRUCTIONS**

Primary operation of the SYSTEM/LOAD.DUMP program is through the AD and LOAD commands. Refer to documentation on these commands in volume 1, section 2 for details.

#### **PROGRAM SWITCHES**

The SYSTEM/LOAD.DUMP program recognizes program switches 0 and 1; however, switch 0 is used only for debugging the SYSTEM/LOAD.DUMP program and should not be used.

Program switch 1 governs redundancy checking. By default, this switch is off, and the SYSTEM/LOAD.DUMP program does not check to see if the same file-identifier is included more than once in the control specification list.

#### **CONTROL SPECIFICATIONS**

The syntax for specifying the files to be copied and the tape and disk to be used during an explicit execute of the SYSTEM/LOAD.DUMP program, initiated with the AD and LOAD commands, is documented in volume 1, section 2.

## SECTION 40

### SYSTEM/LOGOUT

The SYSTEM/LOGOUT program is a system utility program which produces a formatted analysis of the system's operating log. The MCP II maintains the system log when the LOG option is set. This program accesses a transferred copy of the system log. Refer to the LG and LN input messages in volume 1, section 2 for information on transferring the system log file.

### OPERATING INSTRUCTIONS

Primary operation of the SYSTEM/LOGOUT program is through the LG and LN input messages (refer to volume 1, section 2). An EXECUTE SYSTEM/LOGOUT command can also initiate the program. The internal file-id of the logfile is LOGFILE.

### PROGRAM OUTPUT

The SYSTEM/LOGOUT program writes directly to a line printer. If no line printer is available, a backup print file is generated. Output includes analysis of CLEAR/START log records, job information (BOJ, EOJ) log records, DMSII log records, and file information log records.

#### CLEAR/START Records

The SYSTEM/LOGOUT program generates three lines of output for each CLEAR/START record in the log file. This output describes the date and time of the CLEAR/START operation, the name and version of the MCP II being used and the amount of S-memory in bits, and the names of the interpreter and GISMO being used by the MCP.

#### Job Information Records

The SYSTEM/LOGOUT program combines several log records to generate information on each job processed by the system. Job information output describes:

1. The job number of the program that is executing.
2. The charge number used to execute the job.
3. The full name of the program.
4. The BOJ date and time.
5. The elapsed time for the program, in seconds.
6. Any abnormal job termination messages.
7. The type of termination (for example, NORMAL, ABORTED).
8. The termination (EOJ) date and time.
9. The processor time used by the program, in seconds.
10. The compilation date and time of the executed program.
11. The processor and memory priorities assigned at BOJ.
12. The date and time that the program was scheduled.
13. The number of code overlays performed by the MCP II for the program.
14. The interpreter used.
15. The type of execution (for example, EXECUTE, COMPILE).
16. The number of files declared in the program.
17. The amount of static memory required by the program.
18. The amount of dynamic memory (data overlay space) used by the program.
19. The total memory used by the program for its run structure (Base/Limit area).
20. The number of data overlays required by the program.



B 1000 Systems SSOG, Volume 2  
SYSTEM/LOGOUT

**File Information records**

The log file contains information about files used during execution of a program. Files that are not accessed are not logged. The SYSTEM/LOGOUT program generates the following information about each file.

1. The internal file name.
2. The external file identifier.
3. The record size, in bytes.
4. The hardware device type assigned to the file.
5. The number of buffers assigned to the file.
6. Either (a) the number of records per block (fixed length files), or (b) the maximum size of a physical block (variable length records).
7. The file type (for example, CODE, RANDOM).
8. For disk files only, the number of blocks per disk area and the number of disk areas. For non-disk files, this field is blank.
9. The memory space required to open this file.
10. For tape files only, the serial number of the last reel of tape accessed by the file. For non-tape files, this field is blank.
11. The number of records in the file.
12. The number of blocks in the file.
13. The time and type of each OPEN performed.
14. The time and type of each CLOSE performed.
15. The total elapsed time that the file was open.
16. The hardware unit name where the file now resides. For temporary files, this field is blank.

## SECTION 41

### SYSTEM/MAKEUSER

The SYSTEM/MAKEUSER program is a normal-state utility program used to create, access, or modify the (SYSTEM)/USERCODE file, a file describing all usercode and password combinations that are valid on the system. The (SYSTEM)/USERCODE file maintains all information needed by the MCP II to support the file security mechanism, including usercodes and their associated passwords, default disk-identifiers, charge numbers, maximum allowable run-time priorities, and default security codes. A maximum of 1023 usercode/password combinations is allowed in the (SYSTEM)/USERCODE file.

### OPERATING INSTRUCTIONS

All communication and control of the SYSTEM/MAKEUSER program is performed through the operator display terminal (ODT), except for certain automatic features described in the following paragraphs. Accept (AX) input messages provide commands and input data to the program. Success/failure messages are displayed on the ODT after each command has been processed, as follows:

EXECUTE SYSTEM/MAKEUSER

SYSTEM/MAKEUSER = 1 BOJ....  
SYSTEM/MAKEUSER = 1 ACCEPT.

1AX <command> <optional comment>

%SYSTEM/MAKEUSER = 1 <success/failure message>

The <optional comment>, if present, must begin with a percent sign (%) character.

Several automatic features are included in the SYSTEM/MAKEUSER program and are described next.

1. When a card file labeled NEW/USERCODES (such as is produced by the PUNCH command) is present at BOJ or after execution of any command, the SYSTEM/MAKEUSER program automatically performs a CREATE command followed by a list of the newly created (SYSTEM)/USERCODE file. The SYSTEM/MAKEUSER program then goes to EOJ. No operator action or intervention is required. Refer to the CREATE command for information describing the required input file.
2. When the SYSTEM/MAKEUSER program is executed with SWITCH = L, a listing of the current (SYSTEM)/USERCODE file is produced without operator intervention, and the SYSTEM/MAKEUSER program then goes to EOJ.

Example:

EXECUTE SYSTEM/MAKEUSER SWITCH=L;

3. When the SYSTEM/MAKEUSER program is executed with SWITCH = P, the current (SYSTEM)/USERCODE file is punched to a card deck labeled NEW/USERCODES. No operator intervention is allowed, and the SYSTEM/MAKEUSER program goes to EOJ after the card deck is punched.

Example:

EXECUTE SYSTEM/MAKEUSER SWITCH=P;

## FILE SECURITY ATTRIBUTES

The valid file security attributes are described in table 41-1. along with their default values. The user-code (US) and password (PW) attributes have no default and must be specified explicitly for every user-code included in the (SYSTEM)/USERCODE file. A null ("") password is valid, but must be explicitly specified.

Table 41-1. SYSTEM/MAKEUSER Security Attributes

Option	Description	Default Value
US = <usercode>	Usercode	No default
PW = <password>	Password	No default
PACK = <pack-id>	Default pack-id	System disk (" ")
CHG = <integer>	Default charge number	Zero (0)
PRI = <integer>	Maximum priority	Seven (7)
*PRIV	PRIVILEGED indicator	Not PRIVILEGED
*NONPRIV	NON-PRIVILEGED indicator	Not PRIVILEGED
PUBLIC	File security	PRIVATE
PRIVATE	File security	PRIVATE
SL = <integer>	Security level	Zero (0)
MAXTIME = <integer>	Default maximum execution time	Zero (0) (infinite)
HOSTNAME = <identifier>	BNA hostname	*NONE

The <usercode> can be up to ~~seven~~ <sup>8</sup> characters in length. A <usercode> of SYSTEM is not allowed. Restrictions in the <usercode> format are:

1. An asterisk must not be the beginning character.
2. A left parenthesis must not begin or a right parenthesis end the <usercode>.
- ~~3. The less than sign (<) character is always an invalid character in a <usercode>.~~
4. Special characters (other than A through Z and 0 through 9) are permitted in a <usercode>; however, unpredictable results can occur if special characters are misused.

B 1000 Systems SSOG, Volume 2  
SYSTEM/MAKEUSER

The <password> can be up to ten characters in length; it can also be null (""), indicating that no password is required for the associated <usercode>.

Multiple entries with the same <usercode> are allowed; however, each must have a unique <password>. In addition, multiple entries with the same <usercode> must all specify the same default <pack-id> and security (PUBLIC or PRIVATE).

The security level attribute defines the multifile identifiers that a usercode can access. The values can be 0, 1 or 2. A value of 0 allows access to any multifile identifier. A value of 1 allows access to a multifile identifier that is a usercode, for example (PAYROLL). A value of 2 allows access to a multifile identifier that is the same as the usercode supplied with the input command or the same as the usercode under which the job is executing.

The MAXTIME attribute is used by the MCP II in determining the maximum amount of processor minutes for a job. The maximum value of <integer> is 27962.

The HOSTNAME attribute relates only to BNA host services. <identifier> is a string of up to 17 characters for use as a BNA hostname. Lower-case characters are translated to upper-case characters and the underscore (\_\_) character is translated to a minus sign (-) character. The values \*ANY or \*NONE can be used instead of an identifier.

\*ANY means that the associated usercode/password is valid from any BNA host as well as from the local host.

\*NONE means that this usercode/password is invalid from all BNA remote hosts. The usercode/password is still valid from the local host.

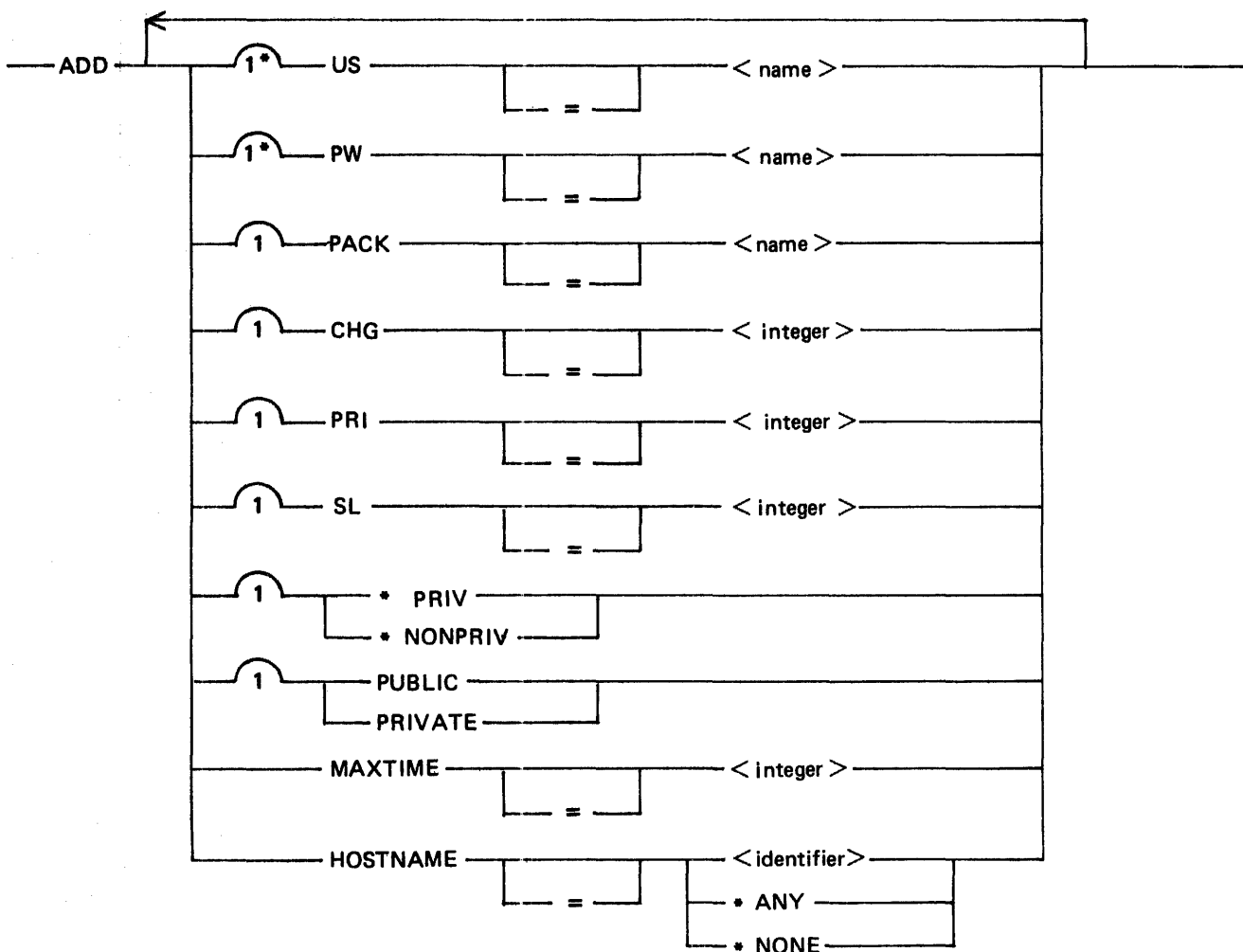
*HELP*

## COMMANDS

The syntax, semantics, and examples of actual input commands are documented in this section, along with a description of the input specification format. Valid commands are ADD, CHANGE, COPY, CREATE, DEBUG, DELETE, DISPLAY, END, EOJ, LIST, PUNCH, and PURGE. The commands are presented in alphabetical order.

### ADD Command

The ADD command allows entries to be added to the (SYSTEM)/USERCODE file. The command has the following syntax:



B 1000 Systems SSOG, Volume 2  
SYSTEM/MAKEUSER

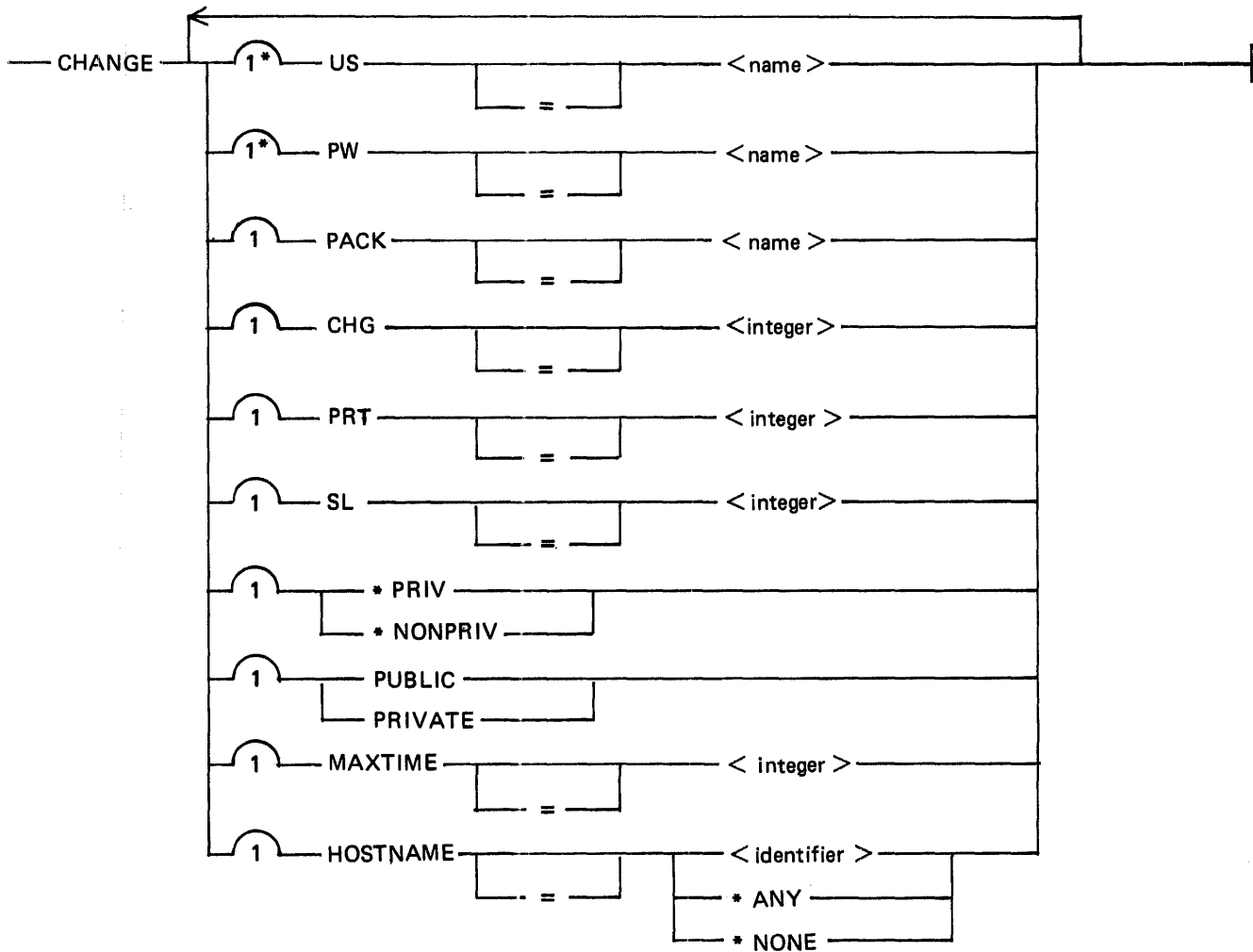
The usercode (US) and password (PW) options are both required; any other options described in the subsection entitled File Security Attributes are allowed.

Examples:

1AX ADD US=NEWUSER PW=NEWPW CHG=1234 PRI=5 PACK=USER	
1AXADD CHG=50 US=SITE PW=""	
1AXADD CHG=50 US=SITE PW=""	
1AX ADD US=SITE PW=PRIV PRI=15	%System disk is default
1AXADD US=FINANCE PW=VP	%All other options default
1AX ADD US REMOTE PW BNA HOSTNAME *ANY	%*ANY means all remote BNA %hosts
1AX ADD US STUDENT PW 3 MAXTIME = 3	%Maximum execution time %is 3 minutes
1AX ADD US PAYROLL PW PAY SL = 1	%Allows access to a <u>%multifile that is a usercode</u>

## CHANGE Command

The CHANGE command allows the attributes of existing usercodes to be modified. The command has the following syntax:



The usercode-specifier must have one of the following formats:

`<usercode>/<password>`

`<usercode>/""`

`<usercode>/=`

The first two formats are used to change a specific entry in the (SYSTEM)/USERCODE file (the "" option is for usercodes having a null password). The "=" option is used to change all entries having the specified `<usercode>`.

B 1000 Systems SSOG, Volume 2  
SYSTEM/MAKEUSER

Only the attributes specified in the CHANGE command are modified. All file security attributes except the usercode (US) can be modified with the CHANGE command, with the following two restrictions.

1. The password (PW) option is not allowed if the /= format of the usercode-specifier is used.
2. The PUBLIC, PRIVATE, \*PRIV and \*NONPRIV options are allowed only when the /= form of the usercode-specifier is used.

WARNING

The CHANGE command must not be used when other programs are executing.

*same as ddote?*

Examples:

1AX CHANGE FINANCE/VICEPRES TO PW=VP CHG=1234 PRI=10

1AXCHA SITE/= TO CHG=55555

1AXCHANGE NEWUSER/"" TO PW=NEWPW PACK=""

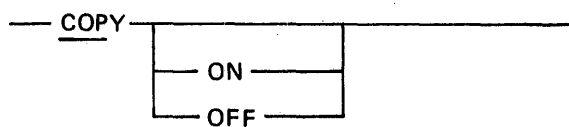
1AX CHANGE STUDENT/1 TO MAXTIME = 3

1AX CHANGE CLASS/= TO HOSTNAME \*NONE

% All usercodes are now denied  
% execution from all remote BNA  
% hosts.

### COPY Command

The COPY command controls listing of all ODT input and output messages on the line printer. The command has the following syntax:



By default, the COPY option is ON, and all SYSTEM/MAKEUSER ODT messages are listed on the line printer. When the COPY option is turned off, the printer listing is suppressed.

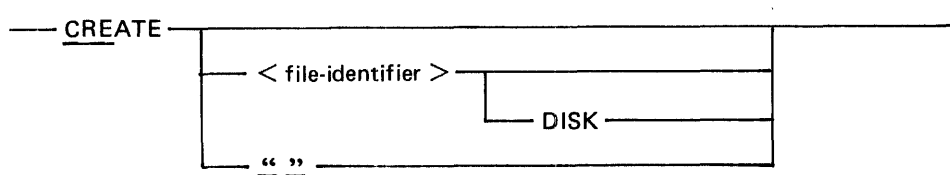
Example:

1AXCOPY OFF



## CREATE Command

The CREATE command causes a new (SYSTEM)/USERCODE file to be built which can be entered into the US slot of the Name Table with the CM input message (refer to volume 1, section 2). The command has the following syntax:



If no options follow the keyword CREATE, then the input file-identifier is assumed to be CARD. The input file is assumed to be a card file unless the keyword DISK is specified.

The null ("") CREATE option allows users to create a usercode file through the ODT alone. The input specification CREATE "" produces a usercode file that contains one entry: the privileged usercode/password pair DEFAULT/PRIVILEGED. This privileged usercode is valid for all usercode operations.

The SYSTEM/MAKEUSER program enters the new (SYSTEM)/USERCODE file into the US slot of the Name Table. If a (SYSTEM)/USERCODE file already exists, it is removed from disk and replaced by the new (SYSTEM)/USERCODE file. The SYSTEM/MAKEUSER program and the SYSTEM/ODT program must be the only programs executing when the CREATE command is used to replace an existing (SYSTEM)/USERCODE file, or the replacement is not performed.

The input file for the CREATE option contains input specifications in a free-form format, 80-character record. Each file security attribute that is specified must be preceded by the keyword that identifies the attribute. Options can be entered in any order, but all attributes for any usercode must be contained on one record. The usercode (US) and password (PW) attributes must be included on every record. Other attributes can be included as desired; if omitted, the default values apply. Refer to File Security Attributes in this section for a description of the attributes.

### NOTE

The CREATE command can be executed only when SYSTEM/MAKEUSER is the only program executing. If any other programs are executing, the newly created (SYSTEM)/USERCODE file is NOT entered into the US slot of the Name Table.

Examples of input records in the card file CARD follow:

```
US=USERA PW=X CHG=123456 PRI=4 PACK=USERPACK *PRIV
```

```
CHG=1000 PRI=8 US=USERB PW=ME
```

```
US=USERA PW="" PACK=USERPACK
```

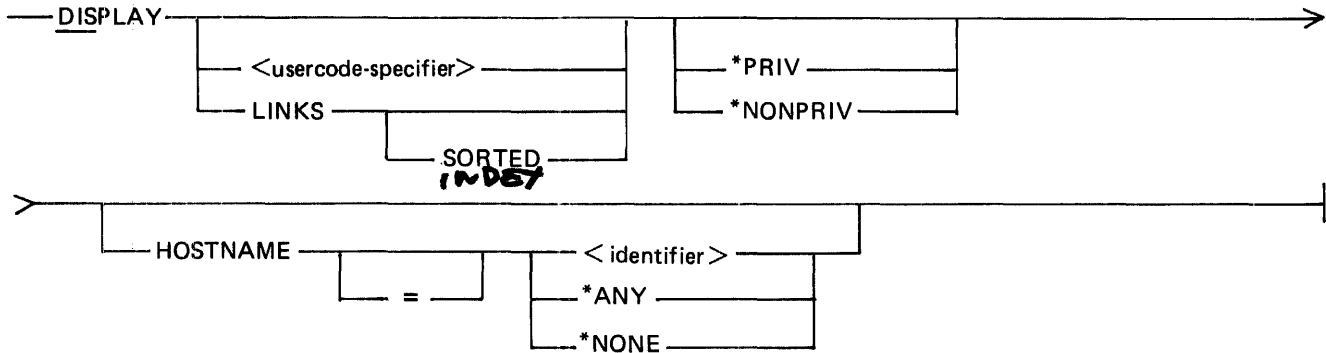
```
US=SITE PW="" CHG=999999 PRI=15 *PRIV PUBLIC
```

```
US=USERB PW=PW3 PACK=""
```



## DISPLAY Command

The DISPLAY command displays the existing (SYSTEM)/USERCODE file on the ODT. The command has the following syntax:



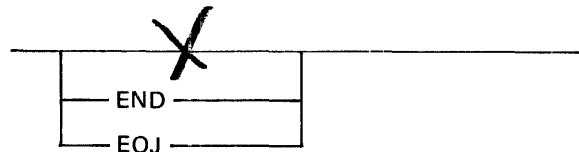
If no options are specified, the DISPLAY command causes the entire file to be displayed. If the <usercode-specifier> is entered, only the specified usercodes are displayed. The use of the \*PRIV option allows the display of only privileged usercodes. The HOSTNAME option allows usercodes on the specified BNA host to be displayed. If the LINKS option is specified, the usercode file is displayed on the ODT by usercode index. The SORTED option causes the usercode file to be displayed in alphabetical order with the link fields.

Example:

DISPLAY HARRY/= *NONPRIV	%Display those usercode/password %pairs of HARRY which are %nonprivileged.
DISPLAY *PRIV HOSTNAME *ANY	%Display only those entries %which are privileged and valid %from the remote host.
DIS LINKS SORTED	%Display the entire usercode file %in alphabetical order with the %link fields.

## END/EOJ Command

The END/EOJ command causes the SYSTEM/MAKEUSER program to go to end of job. The command has the following syntax:

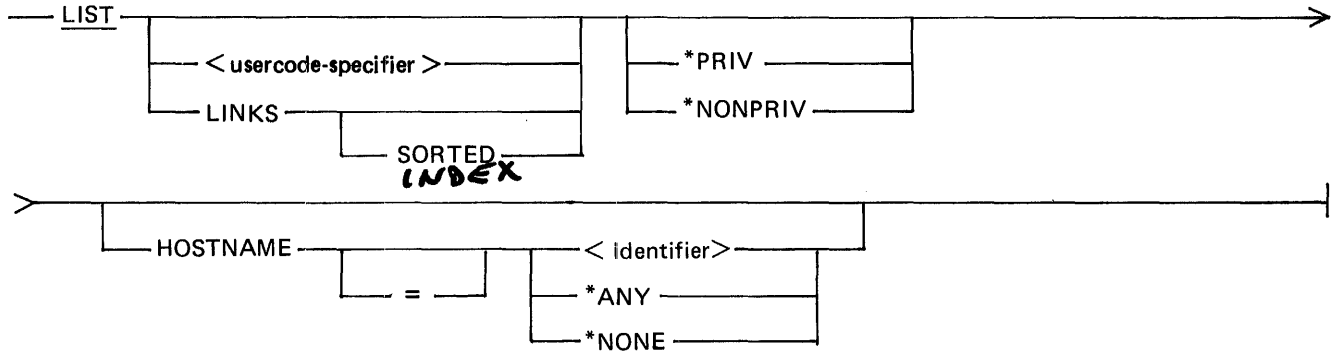


Example:

2AX END % THIS IS THE SAME AS "EOJ"

## LIST Command

The LIST command allows listing of all or part of the existing (SYSTEM)/USERCODE file on the line printer. The command has the following syntax:



If no option is specified with the LIST command, the (SYSTEM)/USERCODE file is sorted by usercode and the entire file is listed on the line printer. The < usercode-specifier > option allows the listing of only the designated entries. If the LINKS option is specified, the usercode file is listed on the line printer by usercode index. The SORTED option causes the usercode file to be listed on the line printer in alphabetical order with the link fields. If \*PRIV is specified, only the entries for usercodes marked as PRIVILEGED are listed. Specifying \*NONPRIV causes only those usercodes marked NONPRIVILEGED to be listed on the line printer. If a usercode family is specified in the LIST command, the usercode family can be listed only by usercode index.

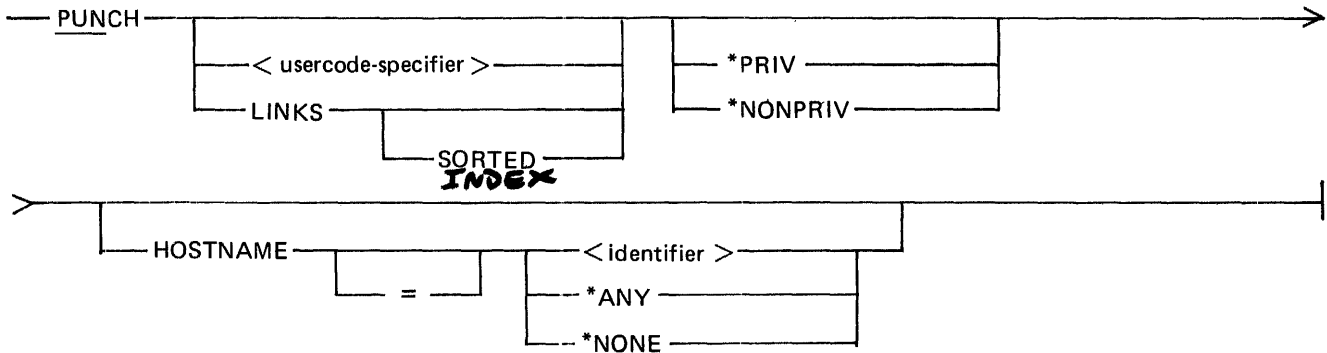
The pack-id listed for all usercodes which have the default pack assignment of the system disk is \*SYS DISK\*.

Examples:

1AX LIST	
1AXLIS USER1/=	
1AXLIST SITE/PRIV	
1AX LIS USERA/""	
1AXLIST *PRIV	
1AX LIST HOSTNAME HUB	% Lists only the entries which are % valid from the remote host called % HUB.
1AX LIST REX/= HOST SA1	% Lists only those entries of REX % which are valid from the remote % host SA1.
1AX LIST *NONPRIV HOSTNAME USER3	% Lists only those entries which are % nonprivileged and which are valid % from remote BNA host USER3.

### PUNCH Command

The PUNCH command punches a card deck containing the current (SYSTEM)/USERCODE file. The PUNCH option can be requested as an automatic function by executing the SYSTEM/MAKEUSER program with SWITCH = P. The command has the following syntax:



The output card file is labeled NEW/USERCODES and is acceptable as input to the CREATE command (either explicit or automatic). The usercode-specifier and \*PRIV options have the same meaning as they do when used with the LIST command.

Example:

1AX PUNCH

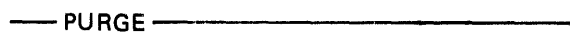
275AX PUNCH USER1/=

17AX PUNCH SITE/PRIV

2468AX PUNCH \*PRIV

### The PURGE Command

The PURGE command removes the file named (SYSTEM)/USERCODE and deletes the usercode file from the US-slot in the name table. The PURGE command is used when the SYSTEM/MAKEUSER program is the only job running. The PURGE command has no options. The command has the following syntax:



## ERROR MESSAGES

UNKNOWN COMMAND "<command>", TRY ONE OF "CHANGE, ADD, DELETE  
CREATE, LIST, DEBUG, PUNCH, COPY, END, ~~OR~~ EOJ." *PURGE, DISPLAY, HELP?*  
COMMANDS MUST BE FROM 3 TO 6 CHARACTERS IN LENGTH.

Enter a valid command or command abbreviation.

UNRECOGNIZED COMMAND/KEYWORD <command> OR TOO MANY PARAMETERS  
FOR THIS COMMAND. *<keyword list>*

Enter a valid command with all required options specified.

PARAMETER REQUIRED AND NOT FOUND FOLLOWING <input parameter>.

REQUIRED PARAMETERS WERE OMITTED FOR THIS COMMAND.

NUMBER FIELD FOR "CHARGE NUMBER" [or "PRIORITY"] TOO LARGE.

The maximum allowable value for CHARGE is 9999999, and the maximum allowable value for  
PRIORITY is 15.

"(SYSTEM)/USERCODE" FILE NOT ON DISK, COMMAND IGNORED.  
A CREATE IS REQUIRED.

If the NEW/USER~~C~~ODES file is not available as a disk file or as a card file, use the null option  
of the CREATE command to produce a single entry usercode file that describes the usercode/pass-  
word DEFAULT/PRIVILEGED.

"(SYSTEM)/USERCODE" FILE LOCKED, COMMAND IGNORED.

MAXIMUM FILE SIZE OF 1024 ENTRIES EXCEEDED, COMMAND TERMINATED.

INVALID USERCODE "<usercode>" ENTRY DISCARDED.

DIFFERENT PACK NAME FOR SAME USERCODE

<usercode> ENTRY DISCARDED.

FILE NAME MUST BE SPECIFIED WITH "DISK" OPTION, COMMAND  
IGNORED.

If DISK is specified with the CREATE command, a <file-identifier> must also be supplied.

PACK NAME IS INVALID FOR CARD FILES, COMMAND IGNORED.

If a <pack-id> is included in the <file-identifier> specified in the CREATE command, DISK  
must also be specified.

B 1000 Systems SSOG, Volume 2  
SYSTEM/MAKEUSER

INPUT FILE SPECIFIED IS NOT ON DISK, COMMAND IGNORED.

The disk file specified in the CREATE command does not exist.

NO USERCODE FILE PRESENT, COMMAND IGNORED.

A CREATE must be performed to create a usercode file.

SPECIFIED ENTRY DOES NOT EXIST, COMMAND IGNORED.

The requested <usercode-specifier> is not in the (SYSTEM)/USERCODE file.

NUMERIC FIELD CONTAINS NON-NUMERIC CHARACTERS.

INVALID DELIMITER "<delimiter>", COMMAND IGNORED.

CHANGE TO PACKNAME REQUIRES "<usercode>/=".

"<usercode>/<password>" ALREADY EXISTS.

Cannot ADD a <usercode>/<password> entry that already exists in the (SYSTEM)/USERCODE file.

CANNOT CREATE NEW USERCODE FILE WHEN OTHER JOBS ARE RUNNING.

The SYSTEM/MAKEUSER program must be the only program in the job mix if the existing usercode file is to be replaced with a new usercode file during execution of a CREATE command. The OVERRIDE option is not valid.

CANNOT CREATE USERCODE FILE WITH NO ENTRIES. USE CREATE "" FOR DEFAULT.

SECURITY MISMATCH -MIXED "PRIVATE" AND "PUBLIC" NOT ALLOWED.

CHANGE TO SECURITY REQUIRES "<usercode>/=".

CANNOT CHANGE ALL PASSWORDS.

CANNOT SPECIFY BOTH "PUBLIC" AND "PRIVATE".

CANNOT SPECIFY BOTH "PRIV" AND "NONPRIV".

REMOTE EXECUTION DENIED.

ILLEGAL USERCODE.

*more?*

*Can only DELETE from ODT*

## SECTION 42

### SYSTEM/MARK.SEGS

The SYSTEM/MARK.SEGS program is a normal-state utility program that marks program code segments as "important" or "unimportant" for use with the Extended Segment Decay facility of the MCP II Priority Memory Management System. By default, all code segments are marked by the compilers as unimportant. Segment and segment-page numbers are noted on the compilation listing of the program. This information can also be obtained by using the CODE/ANALYZER program to list an analysis of the code segment assignments of the program.

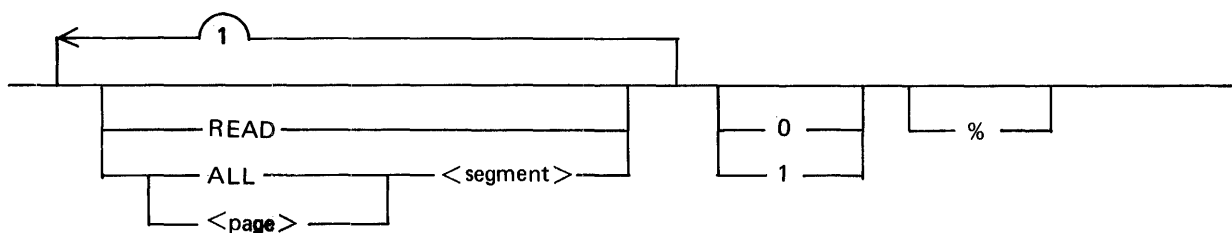
Refer to volume 1, appendix A, MCP Memory Management, for further details on Extended Segment Decay and Priority Memory Management.

### OPERATING INSTRUCTIONS

The SYSTEM/MARK.SEGS program marks only code (object) files. The file identifier of the file to be marked must be file-equated to the name CODE when the program is executed. The program expects to receive specification strings from a card file labeled CARDS; however, if that file is not present when the SYSTEM/MARK.SEGS program begins execution, the program prompts for input specifications and expects to receive the specification strings in accept (AX) input messages from the console keyboard. Card input can be forced by entering the keyword READ (R) as the first AX input message.

### INPUT SPECIFICATIONS

The SYSTEM/MARK.SEGS program expects free-form format input in the first 72 columns of each input specification card. Multiple specifications can be entered in each input card or accept message. A percent sign (%) character causes scanning on that input string to be terminated, allowing comments to be included. A description of the specification string syntax follows.



The keyword READ forces input specifications to be read from cards. This keyword must be used only if the SYSTEM/MARK.SEGS program is executed from the operator display terminal (ODT) and input specifications are to be entered via a card reader.

ALL causes every code segment in the program to be marked as important (1), or unimportant (0). When used, this option must be the first input specification (after READ).

The <page> and <segment> specifications allow individual segments to be marked. These numbers must be specified as integer values in either decimal or hexadecimal notation. The numbers must be enclosed in parentheses and must be followed by a 1 or a 0.



B 1000 Systems SSOG, Volume 2  
SYSTEM/MARK.SEGS

The SYSTEM/MARK.SEGS program ignores invalid (page, segment) specifications; furthermore, it terminates scanning the current specification string at the point where it encounters invalid specifications.

Page numbers are required if the source language of the program includes page numbers in its segment notation. Page numbers are optional only when they do not apply to the source language.

A null or blank specification string generates a printer listing of the status of all segments of the specified code file and terminates the SYSTEM/MARK.SEGS program.

Example:

```
? EX SYSTEM/MARK.SEGS
? FILE CODE NAME TEST/
PROGRAM
? DATA CARDS
(0,1) 1           % Specification
(2, @A@) 1 (1,7) 1 % Strings
(@F@,5) 0
? END
EX SYSTEM/MARK.SEGS FI CODE NAME PAYROLL/APPLICATION
  SYSTEM/MARK.SEGS = 7385.BOJ
  SYSTEM/MARK.SEGS = 7385 ACCEPT
7385AXREAD
.
.
.
SYSTEM/MARK.SEGS = 7385.EOJ
```

## PROGRAM OUTPUT

The SYSTEM/MARK.SEGS program generates a listing that notes errors in the specification strings; additionally, the report on each segment's status, if requested, appears in this listing. This listing goes directly to a line printer if one is available; otherwise, the report is saved in a backup file.

## SECTION 43 SYSTEM/ODT

The SYSTEM/ODT program handles all required I/O operations on the Operator Display Terminal (ODT). The program is loaded automatically by the COLDSTART process, entered in the appropriate slot in the Name Table, and executed automatically by the MCPPII during the CLEAR/START process.

It is recommended that the SYSTEM/ODT program be run at a higher priority than any other program. Operators can adjust this priority in accordance with individual needs. For example, if response from remote terminals is the most important activity on the system, the priority of the network controller can be set to a value equal to or higher than the priority of the SYSTEM/ODT program.

### NOTE

Because the SYSTEM/ODT program is a normal-state program, the response to ODT commands can be slightly slower if system activity is heavy. To prevent degradation of response, the program is placed at one end of memory to prevent checker-boarding. The SYSTEM/ODT program never goes more than 1.5 seconds before looking for work to do, and it is frozen in memory. Furthermore, the highest priority in its list of tasks is to accept operator input. Even if the system is so busy that ODT output response is significantly degraded, the program generally accepts operator input and conveys it to the MCPPII for action.

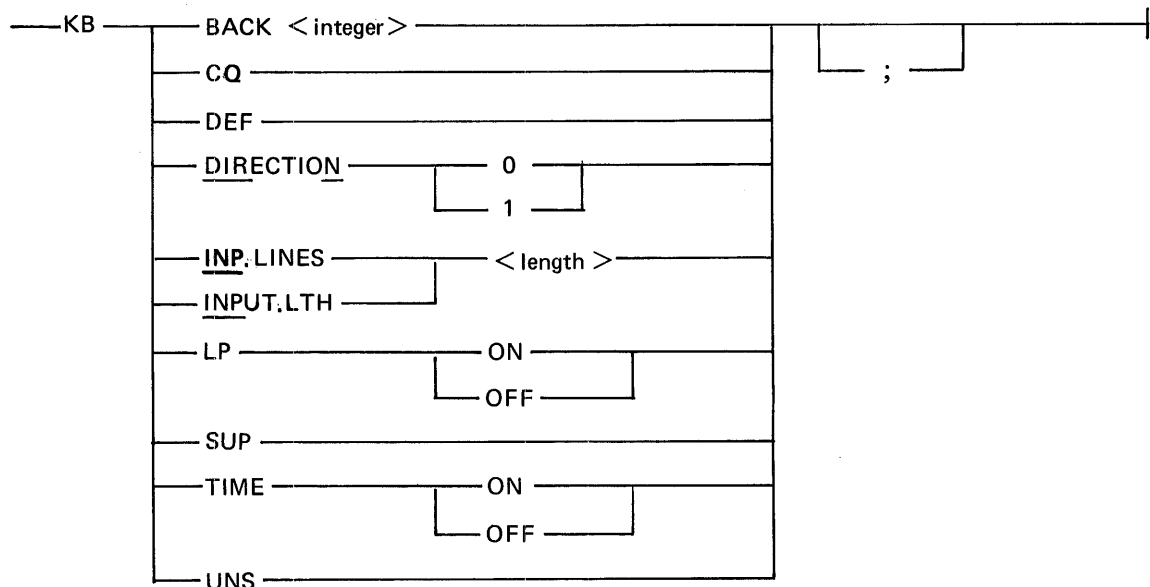
## OPERATING INSTRUCTIONS

No operating instructions are required for this program. It is executed automatically by the MCPPII during the CLEAR/START operation. Should the program terminate for any reason, this occurrence is recognized by the MCPPII and the program is automatically executed again.

## KB INSTRUCTION

The KB system control instruction specifies certain characteristics for displaying the ODT information.

Syntax:



Semantics:

**BACK**

The keyword **BACK** causes the last <integer> number of sectors in the 200-sector ODT queue file to be formatted and displayed, a screenful at a time, on a CRT ODT. No messages are displayed on a TTY ODT. For both ODT types, if **LP ON** has been specified, the same messages are directed to a line printer file by the **SYSTEM/ODT** program.

**CQ**

The keyword **CQ** causes all the messages stored in the ODT queue file to be removed.

**DEF**

The keyword **DEF** restores the default ODT settings and is valid only for CRT ODT devices. The default settings are **INPUT.LINES 2**, **DIRECTION 0**, **UNS**, **LP OFF**, and **TIME OFF**.

**DIRECTION**

The keyword **DIRECTION** specifies the order in which ODT messages are to appear on the screen and is valid only for CRT ODT devices. If **KB DIRECTION 0** is specified, the most recent message appears at the top of the ODT output area, with previous messages appearing in reverse order downwards. If **KB DIRECTION 1** is specified, the most recent message appears at the bottom line of the ODT output area, with previous messages appearing toward the top.

**INPUT.LINES** or **INPUT.LTH**

The keywords **INPUT.LINES** or **INPUT.LTH** changes the size of the input area reserved at the top of the screen and is valid only for CRT ODT devices. The default size of the input area is two lines. Specifying **KB INPUT.LINES <integer>** causes the number of reserved lines to be any value that ranges from 2 to 10, inclusive.

Changing the size of the input area affects the size of the output area. Increasing the size of the input area decreases the size of the output area, proportionately.

On the CRT ODT device, the input area remains displayed on the screen and in the memory of the CRT until a different message is entered. Therefore, if the operator has made an error in a long input message, after being told of the error by the **MCP II**, the operator does not have to retype the entire message. By positioning the cursor to the portion of the input message in error, the error can be corrected. After the error has been corrected, the message can be retransmitted in its error-free form. This feature is not applicable to all system control instructions.

**LP**

The keysymbol **LP** applies to both TTY and CRT ODT devices. Specifying **KB LP ON** causes a line printer file to be opened and all ODT message traffic to be written to the line printer file as well as to the ODT device. Specifying **KB LP OFF** restores normal ODT-only display. The default is **LP OFF**. The line printer file is closed when **KB LP OFF** is entered.

**OFF**

The keyword **OFF** is valid in the **LP** or **TIME** options in the **KB** system control instruction and resets the **LP** or **TIME** option.

**ON**

The keyword **ON** is valid in the **LP** or **TIME** options in the **KB** system control instruction and sets the **LP** or **TIME** option.

**SUP**

The keysymbol **SUP** causes all messages on the screen, except input and output messages entered at the ODT device, to be suppressed.

#### TIME

The keyword TIME functions for both TTY and CRT ODT devices and causes the time in which the input or output message was entered to be included on the ODT. The time portion of the message is displayed in front of the message and is always included on each message stored in the ODT queue, regardless of whether the option is set or not. Specifying KB TIME ON sets this option. Specifying KB TIME OFF resets this option.

#### UNS

The keysymbol UNS causes all messages, including ZIP, QUEUE, control cards from card readers and pseudoreaders, and so forth, to be displayed on the ODT.

#### 0

The keynumber 0 is valid for the DIRECTION option in the KB system control instruction and causes the most recent messages to appear at the top of the ODT output area, with previous messages appearing in reverse order, downwards.

#### 1

The keynumber 1 is valid for the DIRECTION option in the KB system control instruction and causes the most recent messages to appear at the bottom of the ODT output area, with previous messages appearing at the top.

#### integer

This field can be any integer within the range of 0 to 199, inclusive, and specifies the last sector in the 200-sector ODT queue file in which to begin re-displaying the ODT messages. <integer> is valid only for the BACK option in the KB system control instruction.

#### length

This field can be any integer within the range of 2 to 10, inclusive, and specifies the number of input lines to reserve at the top of the CRT ODT device for input messages. <integer> is valid only for the INPUT.LINES and INPUT.LTH options in the KB system control instruction.

#### Examples:

KB BACK 20;

KB TIME ON;

KB LP ON;

KB SUP;

KB DIRECTION 1;

## FILES

The SYSTEM/ODT.DRIVER file contains four file declarations. The following information is being presented for reference only. Modification of the attributes of the files is strictly prohibited in most cases.

The SYSTEM/ODT program communicates with the MCP II through two queue files. One of the queue files is used to store system output messages that are to be displayed on the screen; messages that are entered by the operator are stored in the other queue file. The SYSTEM/ODT program performs all I/O operations on the ODT through direct communication with GISMO. The SYSTEM/ODT program eliminates the backspace character(s) from messages entered by way of a Teletype ODT.

B 1000 Systems SSOG, Volume 2  
SYSTEM/ODT

In addition to the two queue files, the SYSTEM/ODT program also contains a declaration for a line printer file. This file is activated by a KB LP ON instruction. Also, since the line printer file is handled in the same manner as any other line printer file, the ODT messages that are written to the line printer can now be directed to a backup medium. Most of the file attributes associated with the line printer file, the internal name of which is LINE, can be modified by the operator.

The SYSTEM/ODT program contains one disk file that is used only when the MCPPII option ODTL is set. Refer to volume 1, section 2 of B 1000 Systems System Software Operation Guide. The ODT log is now a conventional disk file written by the SYSTEM/ODT program. In previous versions of the software, this function was also handled by the MCPPII.

Modification of most of the attributes of any of the three files, exclusive of the line printer file, is prohibited. If any of the attributes are set to an improper value (one that was not compiled into the program) the entire system can be rendered inoperable and a COLDSTART operation is necessary to effect recovery.

## MEMORY REQUIREMENTS

The SYSTEM/ODT program is frozen in memory. The following memory requirements are approximate and can vary to a small extent.

Run Structure	10,200 bytes
Q.IN File	100 bytes
Q.OUT File	100 bytes
Segment Dictionary	<u>35 bytes</u>
Total	10,435 bytes

Under normal circumstances, only the files listed in the preceding example are open. Operating with KB LP ON adds roughly 350 bytes to the memory requirements. Operating with the ODT.LOG option set adds approximately 350 bytes. Space requirements for input and output messages which are stored in the queue files mentioned also add to the memory requirements by small, varying amounts.

## SECTION 44

### SYSTEM/ODTLOGOUT

The SYSTEM/ODTLOGOUT program transfers, formats, and prints a log of Operator Display Terminal (ODT) messages. The program works with the contents of the logfile and produces a formatted hardcopy version of ODTLOG.

The program provides the capability of printing ODT messages pertinent to a specific time frame without printing the entire ODT log. This option is invoked by setting program switch 1 equal to 1.

### OPERATING INSTRUCTIONS

Execution of the SYSTEM/ODTLOGOUT program is performed using the ODT and takes one of several forms, depending on the task.

To transfer the log to disk, enter:

TL ODT

The MCPPII commands LG and LN transfer, analyze, and print the ODTLOG in its entirety. To execute, enter:

LG ODT or LN ODT

To transfer, analyze, and print messages within a specified time frame, enter:

LG ODT; SW1 1 or LN ODT; SW1 1

To print a log of ODT messages after it has been transferred (for example, after a TL ODT command), enter:

EX SYSTEM/ODTLOGOUT FI LOGFILE NAME ODTLOG/<file number> DISK;

To print messages pertinent to a specific time frame (after the log has been transferred), enter:

EX SYSTEM/ODTLOGOUT FI LOGFILE NAME ODTLOG/<file number> DISK; SW1 1

### SWITCH SETTINGS

There are two program switches that affect the operation of this program: Switch 0 and switch 1. If program switch 0 is set to an integer other than 0, the logfile is printed and then purged. If program switch 1 is set to 1, the program accepts user specifications for the date and time periods to be printed.

After the program is executed with program switch 1 set to 1, pertinent messages are displayed on the ODT. If the message ENTER STARTING AND ENDING DATES (MM/DD-MM/DD) is displayed, the operator responds with either of the following:

<mx>AX <month>/<day>  
<mx>AX <month>/<day>-<month>/<day>

B 1000 Systems SSOG, Volume 2  
SYSTEM/ODTLOGOUT

Example:

<mx>AX 06/13  
<mx>AX 06/13-06/14

After the message ENTER STARTING AND ENDING TIMES (HH:MM-HH:MM) is displayed, the operator replies:

<mx>AX <hour>:<minutes>-<hour>:<minutes>

Example:

<mx>AX 15:02-16:30

## ERROR MESSAGES

The SYSTEM/ODTLOGOUT program generates the following error messages:

### INVALID LOG ENTRY FOUND

A log entry was encountered with a message that exceeded the normal limits.

### ERROR - NO DATES SPECIFIED

The program found no dates in reply to its displayed request for parameters.

NO OUTPUT WAS PRINTED, THIS ODTLOG FILE DOES NOT CONTAIN THE SPECIFIED DATE(S)

### ERROR - NO TIME FRAME SPECIFIED

The program found no hours or minutes in reply to its displayed request for parameters.

### I/O ERROR

An I/O error was encountered during a read of the LOGFILE.

### INVALID MESSAGE SIZE

The ODTLOG entry was greater than 180 bytes.

## INTERNAL FILES

The following is a list of the name and function of each file used by the SYSTEM/ODTLOGOUT program.

Internal File Name	Function
LOGFILE	Stores transferred ODTLOG, input to program.
LINE	Printer file.

## SECTION 45

### TAPECOPY

The TAPECOPY program is a normal-state utility multifile program that duplicates, compares, merges, and concatenates non-library, non-Load Dump multifile or single-file data tapes. It allows selective deletion of any of the files processed and allows production of multiple copies of an output tape.

The TAPECOPY program accepts input files from tape or disk.

ANSI standards do not allow a blank file-id on a multifile tape; however, a blank file-id is allowed on a single-file tape.

The TAPECOPY program does not accept library tapes (produced by the SYSTEM/COPY program) or tapes created by the SYSTEM/LOAD.DUMP program. To copy or merge COPY library tapes, the SYSTEM/COPY program must be used. To copy, merge, or compare Load Dump tapes, the SYCOPY program must be used.

### OPERATING INSTRUCTIONS

Execution and control of the TAPECOPY program is through the console keyboard. Upon execution, the TAPECOPY program displays the following message:

```
ENTER <ACTION: CPY, CMP, CCM, MRG, OR CAT>  
      <DEVICE: T(APE) or D(ISK)> <TAPE COUNT>
```

### INPUT SPECIFICATIONS

The response to the previous message must have the following format:

```
<job #> AX <action> [ <device> ] [ <integer> ]
```

The following five possible actions can be specified in the response:

#### CPY

Prepares <integer> tape copies of the input, with input files coming from a tape labeled SOURCE or disk files with a <family-id> of SYS. The input tape can be IL-ed if the name is other than SOURCE. If the <family-id> of the disk files is other than SYS, modify the TAPECOPY code as follows:

```
MODIFY TAPECOPY FI DISK.FILE NAME <family-id>;
```

#### CMP

Compares (only) <integer> copies with the tape or disk files. The tapes to be compared are expected to have names of SOURCE and OUT.SYS, but they can be IL-ed. Disk files are expected to have a <family-id> of SYS, but this can be changed by modifying TAPECOPY as shown previously. Files with different record sizes can be compared; in such a case, the shorter record is padded on the right with blanks.

#### CCM

Creates <integer> copies of the source tape or disk files and compares all copies with the original. If any tape is not identical with the original, that tape is purged and the system operator is notified by a console display message. The TAPECOPY program continues to compare the remaining tapes.



B 1000 Systems SSOG, Volume 2  
TAPECOPY

**MRG**

Merges the files from a source tape with the files on a second tape or a family of disk files and creates <integer> copies. The source tape is expected to be labeled SOURCE, and the update tape is expected to be labeled UPDATE. Both tapes can be IL-ed if the tape identifiers are other than the default. If the update files are on disk, they are assumed to have the same <family-id> as the label of the source tape. The TAPECOPY program can be modified as shown in the preceding example to change the <family-id>, if necessary. The update file takes precedence in the case of duplicate files and is the file placed on the output tape.

**CAT**

Concatenates the input files onto one tape file with <integer> copies. The output tape is given the same identifier as the first input file. Subsequent input tapes are expected to be labeled NEXT-FILE but can be IL-ed if necessary. Only the first file on each tape is used. To terminate the concatenation process, enter the following message when a new tape is requested:

<job #> OF Input disk files are expected to have a <family-id> of SYS, but this can be changed by modifying the TAPECOPY program as shown previously. The following message appears on the operator display terminal (ODT) for each disk file to be concatenated.

ENTER FILE ID OF NEXT FILE

A blank (null) accept (AX) terminates the requests. If the TAPECOPY program is executed with switch 2 = 1, the <file-id>s are read from a card file labeled TCARDS, with one <file-id> per card.

From one to seven tapes can be specified by <integer>. If <integer> is omitted, one tape is assumed by default. If <device> is omitted, TAPE is assumed.

The disk files associated with all functions are expected to be on the system disk that has a <family-id> of SYS. Both the <pack-id> and the <family-id> can be changed by using the FILE statement. The <internal-file-id> of the disk file is DISK.FILE.

## PROGRAM SWITCHES

The TAPECOPY program responds to the values of program switches 0, 1, and 2 as shown in table 45-1.

**Table 45-1. TAPECOPY Program Switches**

Switch	Value	Meaning
0	1	Causes a summary listing of the actions taken by and the files accessed by TAPECOPY to be printed on the line printer.
1	1	Allows files to be deleted. The file-identifiers are read from a card file labeled TCARDS, one file-id to a card.
1	2	Allows files to be deleted. The file-identifiers are entered from the console keyboard.
2	1	Valid only when concatenating files. Allows the file identifiers to be specified in the card file labeled TCARDS.
3	1	Causes TAPECOPY to read all command specifications from the card file TCARDS rather than from the ODT.

At the completion of processing, the TAPECOPY program displays the following messages:

<integer> TAPE(S) <action taken>

<integer> TAPE(S) FAILED TO COMPARE/PURGED

The second message is displayed only if any tapes failed to compare or were purged.

## APPENDIX A DISK DEVICE CHARACTERISTICS

### DISK CARTRIDGE CHARACTERISTICS

#### Physical Characteristics

	B 9480-12	B 9481-12	B 9482-32
Bytes-per-sector	180	180	180
Sectors-per-track	32	32	64
Tracks-per-cylinder	2	2	2
Cylinders-per-cartridge	203	406	406
Sectors-per-cartridge	12,992	25,984	51,968
Bytes-per-cartridge	2,338,560	4,677,120	9,354,240
Transfer Rate (MBITS/SEC)	1.55	1.55	3.10
Rotational Speed (RPM)	1,500	1,500	1,500

#### Initialization Error Limits

Errors-per-cartridge	406	812	812
----------------------	-----	-----	-----

#### Initialization Characteristics

All three types:

Initialize with @6363@ pattern

Read/Verify one pass

### B 9484 (205 AND 206) DISK PACK CHARACTERISTICS

#### Physical Characteristics

	B 9484-25	B 9484-55
Bytes-per-sector	180	180
Sectors-per-track	90	90
Tracks-per-cylinder	5	5
Cylinders-per-pack	408	815
Usable cylinders-per-pack	407	814
Sectors-per-pack	183,600	366,700
Usable sectors-per-pack	181,115	363,230
Bytes-per-pack	33,048,000	66,015,000
Usable bytes-per-pack	32,600,700	65,201,400
Spare sectors-per-cylinder	5	5
Spare Sectors-per-pack	2,040	4,075
Usable spare sectors-per-pack	2,035	4,070
Transfer Rate (MBITS/SEC)		9.86
Rotational Speed (RPM)		3,672

B 1000 Systems SSOG, Volume 2  
Disk Device Characteristics

**Initialization Error Limits**

Errors-per-cylinder	5	5
Errors-per-pack	2,035	4,070

**Initialization Characteristics**

Both types:

Initialize with @6363@ pattern  
Read/Verify three passes (0, +, -)  
Initialize with @9C9C@ pattern  
Read/Verify three passes (0, +, -)

NOTE

The preceding 0, +, and - refer to verification with no offset (0), offset in (+), and offset out (-).

**B 9499 (215, 225 AND 207) DISK PACK CHARACTERISTICS**

**Physical Characteristics**

	B 9499-8	B 9499-7	B 9499-6
Bytes/sector	180	180	180
Sectors/track	60	60	90
Tracks/cylinder	20	20	8
Cylinders/pack	203	406	1,564
Usable cylinders/pack	203	406	1,563
Sectors/pack	243,600	487,200	1,126,080
Usable sectors/pack	242,585	485,170	1,117,545
Bytes/pack	43,848,000	87,696,000	202,694,400
Usable bytes/pack	43,665,300	87,330,600	201,158,100
Spare sectors/cyl	5	5	5
Spare sectors/pack	1,015	2,030	7,820
Usable spare secs/pack	1,015	2,030	7,815
Transfer Rate (MBITS/SEC)		5.0	10.71
Rotational Speed (RPM)		2,400	3,872

**Initialization Error Limits**

Errors-per-cylinder	5	5	-
Errors-per-pack	1,015	2,030	-

## Initialization Characteristics

### B 9499-8/7

Initialize with @6DB6@ pattern  
Read/Verify three passes (0, +, -)  
Initialize with @6363@ pattern  
Read/Verify three passes (0, +, -)

### B 9499-6

Initialize with @0000@ pattern  
Read/Verify three passes (+, -, 0), pattern @6DB@  
Read/Verify three passes (+, -, 0), pattern @DB6@  
Read/Verify three passes (+, -, 0), pattern @B6D@

#### NOTE

The preceding 0, +, and - refer to verification with no offset (0), offset in (+), and offset out (-).

## APPENDIX B SYNTAX CONVENTIONS

Railroad diagrams show how syntactically valid statements can be constructed.

Traversing a railroad diagram from left to right, or in the direction of the arrow heads, and adhering to the limits illustrated by bridges produces a syntactically valid statement. Continuation from one line of a diagram to another is represented by a right arrow (--->) appearing at the end of the current line and the beginning of the next line. The complete syntax diagram is terminated by a vertical bar (|).

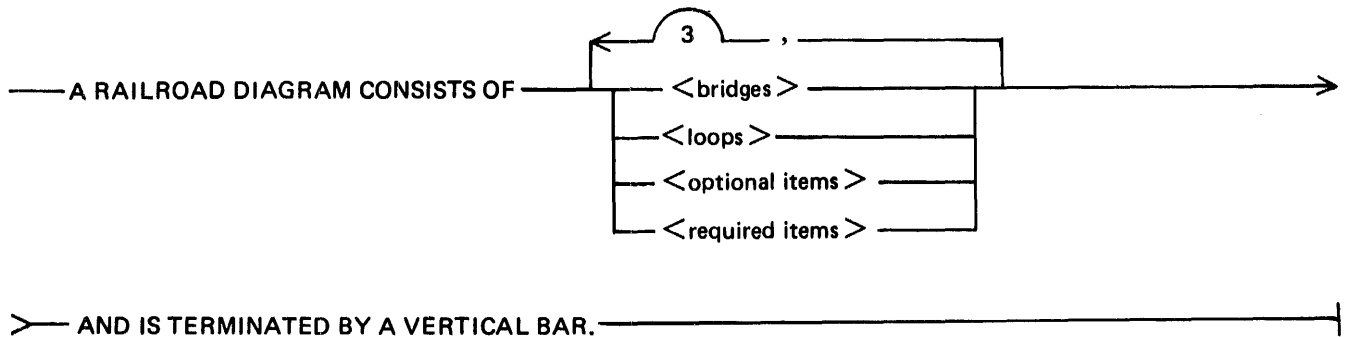
Words appearing in all upper-case characters are keywords and must appear as presented. Words in lower-case characters and embraced in left and right broken bracket characters are symbols for operator-supplied words or values.

Where keywords can be abbreviated, the abbreviation is denoted by the letters underlined in the keywords. Each keyword, keyword abbreviation, or operator-supplied item must be delimited from another such word by at least a single space or by a special character. Parentheses, commas, periods, and colon characters can be used as delimiters. Redundant spaces are ignored.

Items contained in broken brackets (< >) are syntactic variables that are further defined or are an indication the operator must supply requested information.

Upper-case keywords must appear literally. Minimum abbreviations are underlined.

Example:



G50051

The following syntactically valid statements can be constructed from the previous diagram:

A RAILROAD DIAGRAM CONSISTS OF <bridges> AND IS TERMINATED BY A VERTICAL BAR.

A RAILROAD DIAGRAM CONSISTS OF <optional items> AND IS TERMINATED BY A VERTICAL BAR.

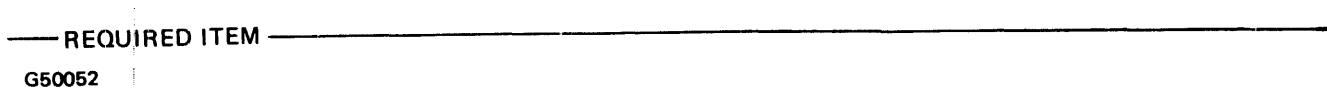
A RAILROAD DIAGRAM CONSISTS OF <bridges>, <loops> AND IS TERMINATED BY A VERTICAL BAR.

A RAILROAD DIAGRAM CONSISTS OF <optional items>, <required items>, <bridges>, <loops> AND IS TERMINATED BY A VERTICAL BAR.

## REQUIRED ITEMS

No alternate path through the railroad diagram exists for required items.

Example:

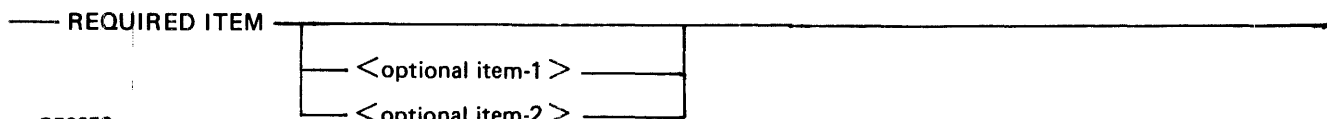


G50052

## OPTIONAL ITEMS

Items shown as a vertical list indicate that the operator must make a choice of the items specified. An empty path through the list allows the optional item to be absent.

Example:



G50053

The following valid statements can be constructed from the previous diagram:

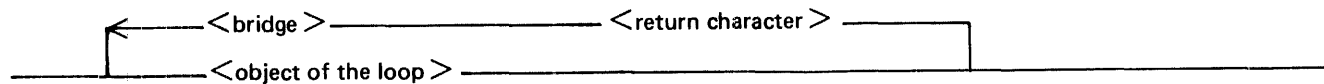
REQUIRED ITEM

REQUIRED ITEM <optional item-1>

REQUIRED ITEM <optional item-2>

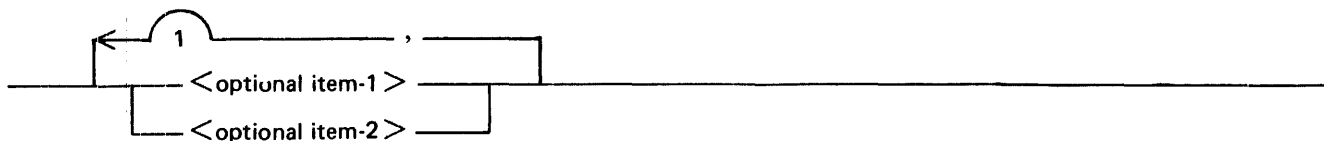
## LOOPS

A loop is a recurrent path through a railroad diagram and has the following format:



G50054

Example:



G50055

The following statements can be constructed from the previous railroad diagram:


- < optional item-1 >
- < optional item-2 >
- < optional item-1 >, < optional item-1 >
- < optional item-1 >, < optional item-2 >
- < optional item-2 >, < optional item-1 >
- < optional item-2 >, < optional item-2 >


A loop must be traversed in the direction of the arrow heads, and the limits specified by bridges cannot be exceeded.

## BRIDGES

A bridge illustrates the minimum or maximum number of times a path can be traversed in a railroad diagram.

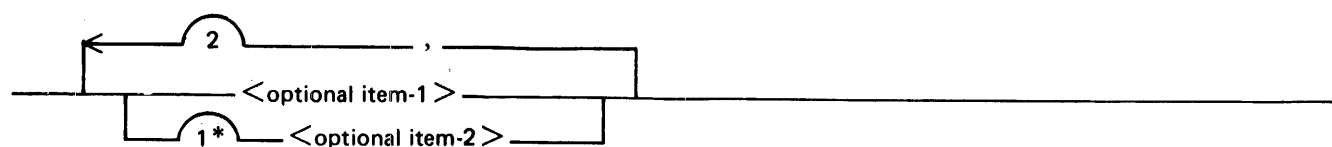
The following are the two forms of bridges.

 n is an integer that specifies the maximum number of times the path can be traversed.

 n is an integer that specifies the minimum number of times the path must be traversed.

**G50056**

Example:



**G50057**

The loop can be traversed a maximum of two times; however, the path for < optional item-2 > must be traversed at least one time.

The following statements can be constructed from the previous railroad diagram:

- < optional item-2 >
- < optional item-1 >, < optional item-2 >
- < optional item-2 >, < optional item-2 >, < optional item-1 >
- < optional item-2 >, < optional item-2 >, < optional item-2 >



**Documentation Evaluation Form**

Title: B 1000 Systems SSOG, Volume 2  
\_\_\_\_\_

Form No: 1138542  
Date: October 1982

Burroughs Corporation is interested in receiving your comments and suggestions, regarding this manual. Comments will be utilized in ensuing revisions to improve this manual.

Please check type of Suggestion:

- Addition                       Deletion                       Revision                       Error

Comments:

\_\_\_\_\_  
\_\_\_\_\_  
\_\_\_\_\_  
\_\_\_\_\_  
\_\_\_\_\_  
\_\_\_\_\_  
\_\_\_\_\_  
\_\_\_\_\_  
\_\_\_\_\_  
\_\_\_\_\_  
\_\_\_\_\_  
\_\_\_\_\_  
\_\_\_\_\_  
\_\_\_\_\_  
\_\_\_\_\_  
\_\_\_\_\_  
\_\_\_\_\_  
\_\_\_\_\_  
\_\_\_\_\_  
\_\_\_\_\_  
\_\_\_\_\_

From:

Name \_\_\_\_\_  
Title \_\_\_\_\_  
Company \_\_\_\_\_  
Address \_\_\_\_\_  
\_\_\_\_\_  
Phone Number \_\_\_\_\_ Date \_\_\_\_\_

Remove form and mail to:  
Burroughs Corporation  
Documentation Dept., TIO - West  
1300 John Reed Court  
City of Industry, CA 91745  
U.S.A.

