

TK50 DATA RELIAB TST

TEST NO.	TEST NAME	TEST TYPE	TEST RESULT	TEST DATE	TEST TIME	TEST USER	TEST STATUS	TEST COMMENTS
1
2
3
4
5
6
7
8
9
10
11
12
13
14
15
16
17
18
19
20
21
22
23
24
25
26
27
28
29
30
31
32
33
34
35
36
37
38
39
40
41
42
43
44
45
46
47
48
49
50
51
52
53
54
55
56
57
58
59
60
61
62
63
64
65
66
67
68
69
70
71
72
73
74
75
76
77
78
79
80
81
82
83
84
85
86
87
88
89
90
91
92
93
94
95
96
97
98
99
100

100 01 100
00 00 00
00 00 00

.REM @

1
2
3
4
5
6
7
8
9
10
11
12
13
14
15
16
17
18
19
20
21
22
23
24
25
26
27
28
29
30
31
32
33
34
35
36
37
38
39
40
41
42
43

IDENTIFICATION

PRODUCT CODE:	AC - T773A - MC
PRODUCT NAME:	CZTKBAO TK50 DATA RELIABILITY
PRODUCT DATE:	17 - APRIL - 1985
MAINTAINER:	TAPE OPTICAL DIAGNOSTIC ENGINEERING
AUTHOR:	BRIAN T. LEBLANC

THE INFORMATION IN THIS DOCUMENT IS SUBJECT TO CHANGE WITHOUT NOTICE AND SHOULD NOT BE CONSTRUED AS A COMMITMENT BY DIGITAL EQUIPMENT CORPORATION. DIGITAL EQUIPMENT CORPORATION ASSUMES NO RESPONSIBILITY FOR ANY ERRORS THAT MAY APPEAR IN THIS DOCUMENT.

NO RESPONSIBILITY IS ASSUMED FOR THE USE OR RELIABILITY OF SOFTWARE ON EQUIPMENT THAT IS NOT SUPPLIED BY DIGITAL OR ITS AFFILIATED COMPANIES.

COPYRIGHT (C) 1985 BY DIGITAL EQUIPMENT CORPORATION

THE FOLLOWING ARE TRADEMARKS OF DIGITAL EQUIPMENT CORPORATION:

DIGITAL	PDP	UNIBUS	MASSBUS
DEC	DECUS	DECTAPE	

TABLE OF CONTENTS

45		
46		
47		
48		
49	1	GENERAL INFORMATION
50	1.1	PROGRAM ABSTRACT
51	1.2	RUNTIME ENVIRONMENT REQUIREMENTS
52	1.3	RELATED DOCUMENTS AND STANDARDS
53	1.4	PASS/FAIL CRITERIA
54	1.5	DATA COMPARE FUNCTION
55	1.6	RESTRICTIONS
56		
57	2	OPERATING INSTRUCTIONS
58	2.1	USER DIALOGUE
59	2.2	HARDWARE QUESTIONS
60	2.2.1	DEFINITION OF HARDWARE QUESTIONS
61	2.3	SOFTWARE QUESTIONS
62	2.3.1	DEFINITION OF SOFTWARE QUESTIONS
63	2.4	CONVERSATION MODE TEST QUESTIONS
64	2.5	ALLOWABLE COMMANDS
65	2.6	SUPERVISOR RUNTIME FLAGS
66		
67	3	ERROR INFORMATION
68	3.1	ERROR REPORTING
69	3.2	COMMANDS
70	3.3	TYPE OF ERROR
71	3.4	STATUS ERRORS
72	3.5	ERROR LOG PACKETS
73	3.6	PROGRAM DETECTED ERROR CONDITIONS
74	3.7	DRIVE ERRORS
75	3.8	HARD ERROR REPORTS
76	3.9	SOFT ERROR REPORTS
77		
78	4	PERFORMANCE AND PROGRESS REPORTS
79	4.1	STATISTICS MATRIX
80	4.2	READ ERROR DEFINITION
81	4.3	WRITE ERROR DEFINITION
82	4.4	MISCELLANEOUS
83		
84	5	TEST DESCRIPTIONS
85	5.1	TEST 1 BASIC FUNCTION TEST
86	5.2	TEST 2 QUICK VERIFY WRITE/READ TEST
87	5.3	TEST 3 COMPLEX WRITE/READ TEST
88	5.4	TEST 4 WRITE INTERCHANGE TAPE
89	5.5	TEST 5 READ UNKNOWN TAPE
90	5.6	TEST 6 START/STOP WRITE/READ TEST
91	5.7	TEST 7 CONVERSATION MODE TEST

93
94
95
96
97
98
99
100
101
102
103
104
105
106
107
108
109
110
111
112
113
114
115
116
117
118
119
120
121
122
123
124
125
126
127
128
129
130
131
132
133
134
135
136
137
138
139
140
141
142
143
144
145
146
147
148
149

1 GENERAL INFORMATION

1.1 Program Abstract

The TK50 PDP11 Q-bus Data Reliability program will exercise the TK50 and establish the performance quality of each unit through the accumulation of statistics. Predetermined sequences of operations will permit read and write compatibility (Media Interchange testing) and data reliability testing. This program will be designed to run in a PDP11 Q-bus XXDP+ environment.

The Data Reliability program will detect functional faults, but will not provide diagnostic isolation to the field replaceable unit.

The PDP11 Q-bus TK50 Data Reliability program is intended for the following users:

1. Quality and user audit functions,
2. F A & T at our various facilities,
3. Field service personnel,
4. DEC customers who choose to provide their own maintenance.

Program uses include but are not limited to the following:

1. Determination of a unit's specific performance (error rate)
2. Fault detection,
3. Repair verification,
4. Installation verification,
5. Preventive maintenance software tool.

This program will exercise up to 4 TK50's in a round robin manner. It will require 28KW of memory. One default pass will be when a tape cartridge (600') has been started at the beginning of tape (BOT) marker and has passed all available tape to the end of tape (EOT) marker over the tape head, twice. One End Of Pass (EOP) will require approximately 1 hour and 10 minutes for each unit under test.

1.2 Runtime Environment Requirements

Run time environment requirements include:

1. XXDP+ Diagnostic Supervisor
2. PDP11 family Q-bus CPU,
3. 28KW of memory,

150
151
152
153
154
155
156
157
158
159
160
161
162
163
164
165
166
167
168
169
170
171
172
173
174
175
176
177
178
179
180
181
182
183
184
185
186
187
188
189
190
191
192
193
194
195
196
197
198
199
200
201
202
203
204
205
206

4. an XXDP Load Device,
5. Console Terminal,
6. 1 to 4 TK50 drives with controllers,
7. 1 scratch tape cartridge / TK50

1.3 Related Documents And Standards

The TK50 Data Reliability program will run under the XXDP+ operating system, and will be Supervisor compatible. The program, with the supervisor will run on all PDP11 Q-bus processors.

This program will conform to the following documents:

1. EL-ENDIA-11 "PDP11 Diagnostic Design Guide",
2. PDP Diagnostic Quality Assurance Checklist,
3. Software Development Policies And Procedures Manual,
4. DEC Std 100,
5. UNIBUS/Q-bus Storage Systems Port Spec Version 2.1
6. Magnetic Tape Storage Control Protocol Spec Version 1.6
7. Mass Storage Control Protocol Spec Version 1.2

1.4 Pass/Fail Criteria

A unit under test will not pass the data reliability mode of testing if any of the following error conditions have occurred during the test cycle:

1. Any irrecoverable write errors detected as documented in the TK50 product specification,
2. Any irrecoverable read errors detected as documented in the TK50 product specification,
3. Irrecoverable hardware errors have occurred,
4. CRC recoverable read errors which exceed TBD errors in 10 to the 11th bits read
5. ECC recoverable read errors which exceed TBD errors in 10 to the 11th bits read

207
208
209
210
211
212
213
214
215
216
217
218
219
220
221
222
223
224
225
226
227
228
229
230
231
232
233
234
235
236
237

If less than the required data has been transferred, the confidence that the unit has met the error rate is diminished. That is to say if the program is run in a quick verify mode, the unit may be accepted as error free but only with a low degree of confidence.

1.5 Data Compare Function

The time required to perform 100% software data comparisons is entirely prohibitive for streaming tape drives. This problem is further exacerbated by the asynchronous nature of command execution under TMSCP and program size limitations which dictate the allocation of a single read data buffer.

To minimize the impact of all this, tests 2 and 3 (the only tests which will perform software data compares) will do software data compares on every 4th record. To avoid the problem of performing data compares on a dynamic read buffer, 3 records will be read from tape using the Access command.

1.6 Restrictions

This program is not intended for use as an isolation tool to detect a fault to the single Field Replaceable Unit (FRU). As such, it will not contain scope loops for that purpose. The parameter selection process, discussed later in this document, is meant to be used only for functional fault detection and unit isolation.

239
240
241
242
243
244
245
246
247
248
249
250
251
252
253
254
255
256
257
258
259
260
261
262
263
264
265
266
267
268
269
270
271
272
273
274
275
276
277
278
279
280
281
282
283
284
285
286
287
288
289
290
291
292
293
294
295

2 OPERATING INSTRUCTIONS

2.1 User Dialogue

The following user dialogue will be provided by the diagnostic to allow the operator to establish certain operational parameters of the program.

2.2 Hardware Questions

This set of questions must be answered by operator when the program is first started.

CHANGE HARDWARE (L) ? no default

NUMBER OF UNITS (D) ?

UNIT x
TKIP ADDRESS (O) 774500 ?
T/MSCP UNIT NUMBER (O) 0 ?

x = Number of unit the p table is
being built for.

Unit specific prompting will continue for a maximum of 4 times, depending on the users response to the "NUMBER OF UNITS" question.

2.2.1 Definition Of Hardware Questions -

CHANGE HARDWARE - If you want to change the hardware p table to be used in the testing this question must be answered yes. This question must be answered with a yes on the initial start of the program.

NUMBER OF UNITS - Number of units to test in decimal.

TKIP ADDRESS - The base address for this unit.

T/MSCP UNIT NUMBER - The unit number of the controller board as specified by MSCP.

2.3 Software Questions

Answering of the software questions is always optional. Default values for a specific question can be obtained simply by typing a <CR>.

CHANGE SW (L) ? no default

ENABLE TIME OF DAY CLOCK (L) N ?
INPUT HOUR IN 24 HOUR FORMAT (OMIT LEADING ZERO) (D) 0 ?
INPUT MINUTES (OMIT LEADING ZERO) (D) 0 ?
CHANGE CONTROLLER PARAMETERS (L) N ?
ENABLE CONTROLLER ERROR CORRECTION (L) Y ?
ENABLE CONTROLLER ERROR RECOVERY (L) Y ?
ENABLE PAD BLOCKING (L) Y ?
CHANGE PRINTING PARAMETERS (L) N ?
ENABLE SOFT ERROR REPORT PRINTING (L) N ?

296 ENABLE READ SOFT ERRORS ONLY (L) Y ?
297 CLEAR MEDIA TABLE ON EVERY PASS (L) N ?
298 ENABLE PRINTING OF MEDIA DEFECTS TABLE (L) N ?
299 ENABLE PROGRAM VARIABLES DUMP ON ERROR (L) N ?
300 ENABLE CLEAR STATS ON FATAL ERROR (L) N ?
301 CHANGE TEST PARAMETERS (L) N ?
302 DATA PATTERN (D) 0 ?
303 RUN TEST 3 ONLY (L) Y ?
304 ENABLE DATA COMPARES IN TEST 5 (L) Y ?
305 ENABLE PRINT READ BUFFER IN TEST 5 (L) N ?
306 CHANGE COMMAND SEQUENCE (L) N ?
307
308
309

2.3.1 Definition Of Software Questions -

310
311
312 ENABLE TIME OF DAY CLOCK (L) N ?
313

314
315 The default is to not enable the clock. This question allows the
316 operated to start a program clock to track time on a 24 hour basis
317 during the running of the program. The clock will remain fairly
318 accurate as long as the program is running. Any time you stop the
319 program the clock will stop running. It is therefore necessary to
320 reset the time whenever the program is started.
321

322
323 INPUT HOUR IN 24 HOUR FORMAT (OMIT LEADING ZEROS) (D) 0 ?
324

325 Input the hour in a decimal number leaving off any leading zeros.
326

327
328 INPUT MINUTES (OMIT LEADING ZEROS) (L) 0 ?
329

330 Input the minutes in a decimal number leaving off any leading zeros.
331

332
333
334 CHANGE CONTROLLER PARAMETERS (L) N ?
335

336 The default answer (no) prohibits the asking of the controller
337 parameter questions. To change the controller parameters type a Y.
338

339
340 ENABLE CONTROLLER ERROR CORRECTION (L) Y ?
341

342 If answered "yes" (default) the program will enable the controller's
343 error correction algorithms for read errors.
344

345
346 ENABLE CONTROLLER ERROR RECOVERY (L) Y ?
347

348 If answered "yes" (default) the program will enable the controller's
349 error recovery algorithms for write and read errors.
350

351
352 ENABLE PAD BLOCKING (L) Y ?

353
354
355 If answered "yes" (default) the program will enable the controller's
356 pad blocking algorithms to assist in streaming
357
358
359
360 CHANGE PRINTING PARAMETERS (L) N ?
361
362 The default answer (no) prohibits the asking of the printing parameter
363 questions. To change the printing parameters type a Y.
364
365
366 ENABLE SOFT ERROR REPORTS (L) N ?
367
368 The default answer (no) inhibits the printing, but not the tallying of
369 soft errors as reported by the subsystem. Answering the question
370 "yes" will result in detailed error reports on the terminal for each
371 recoverable data error.
372
373
374 ENABLE READ SOFT ERRORS ONLY (L) N ?
375
376 This question will only be asked when the above question is answered
377 no. This question allows the operator to enable print outs on read
378 soft errors only. The default answer is to inhibit all soft error
379 printouts.
380
381
382 CLEAR MEDIA TABLE ON EVERY PASS (L) N ?
383
384 The default answer (no) allows the tallying of media defects over
385 multiple passes. By answering the question yes, the operator can then
386 print the table on every pass and see how the defects are affected by
387 passing over the heads.
388
389
390 ENABLE PRINTING OF MEDIA DEFECTS TABLE (L) N ?
391
392 The default answer (no) inhibits the printing, but not the tallying of
393 media defects as reported in the soft error reports by the subsystem.
394 If the default answer is used the table may still be printed by giving
395 the PRINT command at the supervisor prompt (DS>) after the termination
396 of the program. Answering the question "yes" will cause the printing
397 of the table after every pass and after a control C (C) is issued.
398
399
400 ENABLE CLEAR STATS ON FATAL ERROR (L) N ?
401
402 The default answer (no) allows the accumulation of statistics from
403 pass to pass. An answer of "yes" results in the clearing of a devices
404 statistical matrix following any error that results in the unit's
405 being dropped from the test sequence for the rest of the current pass.
406 This action is intended for use primarily by Springfield volume
407 manufacturing.
408
409
409 ENABLE PROGRAM VARIABLES DUMP ON FATAL ERROR (L) N ?

410
411 This question is intended as a program and subsystem debug tool.
412 Answering the question "yes" will cause the program to print out the
413 contents of approximately 1K words of critical memory locations. This
414 is a time consuming process and this question should be defaulted
415 under ordinary circumstances.
416
417
418
419 CHANGE TEST PARAMETERS (L) N ?
420
421 The default answer (no) prohibits the asking of the test parameter
422 questions. To change the test parameters type a Y.
423
424 DATA PATTERN (0) 0 ?
425
426 This question allows the user to select a data pattern from the table
427 of patterns provided by the program. (See the Data Pattern section
428 below.) The default answer, "0", causes the program to cycle through
429 all the data patterns. Answering the question with a number from 1 5
430 will causes the program to use that pattern only. A number higher
431 than 5 will cause the question to be repeated.
432
433
434 RUN TEST 3 ONLY (L) Y ?
435
436 Answering this question "Y" (default) will automatically cause the
437 program to run test 3 only; i.e., it will no longer be necessary to
438 use the /TES:3 switch to the start command. Please note that this
439 question will effectively override the /TES: switch if the user
440 wishes to run a test other than 3. That is, if the user wants to run
441 test 4 he must specify the /TES:4 switch AND answer this question "N".
442
443
444 ENABLE DATA COMPARES IN TEST 5 (L) N ?
445
446 The default answer (no) disallows the data compare function during
447 test 5. This would have to be the case when running with a truly
448 unknown tape. The option (yes) is given to the operator so that when
449 a tape is written in a known manner using this program the operator
450 can then run test 5 using data compares.
451
452
453 ENABLE PRINT READ BUFFER IN TEST 5 (L) N ?
454
455 Answering this question "yes" will cause a printout of all data read
456 from tape in test 5. The data will be presented on a record basis.
457 This is a time consuming process, and this question should be
458 defaulted except in special cases.
459
460 CHANGE COMMAND SEQUENCE (L) N ?
461
462 Answering this question "Y" will cause the program to prompt the user
463 for a sequence of commands to be used in Test 7. (See Test 7 below.)
464 If defaulted, this is the last software question asked.
465
466

467
468
469
470
471
472
473
474
475
476
477
478
479
480
481
482
483
484
485
486
487
488
489
490
491
492
493
494
495
496
497
498
499
500
501
502
503
504
505
506
507
508
509
510
511
512
513
514
515
516
517
518
519
520
521
522
523

2.4 Conversation Mode Test Questions

Answering of these question is optional. These questions will not be asked unless the operator has answered the CHANGE COMMAND SEQUENCE question with a yes. A total of seven commands may be entered by the operator.

Test 7 is intended to give the user the ability to create a specific sequence of commands. Note that Test 7 will not support the entire TMSCP command repertoire, it is limited primarily to the tape motion commands. To run Test 7, the user must issue a STA/TES:7 and must answer "Run Test 3 Only" with a N(0). The user must also answer "Change Command Sequence" with a Y (yes). Understand that the program does not check for legality of command sequences issued by the user, the onus is on the user to perform this check.

The following questions will be asked by the program to prompt the user for his input.

CMD/1 (0) 160 ?

The user enters the octal value for the desired command from the list shown below. Please note that the command values are those defined by the diagnostic, not by TMSCP. The default value for the first command is a rewind.

DATA PATTERN (0) 1 ?

The user should enter the octal value of the desired data pattern from the table of patterns shown above. If the command does not use a data pattern, any number entered here is ignored.

PATTERN #	DESCRIPTION
-----	-----
0	ROTATE THROUGH ALL DATA PATTERNS
1	ALL 1'S
2	ALL 0'S
3	WORST-CASE MFM PEAK SHIFT (110)
4	ALTERNATE 1'S AND 0'S
5	RANDOM DATA
6	MW PEAK SHIFT (1110)
7	COMBINATION OF PATTERN 3 AND 5
200	NO DATA PATTERN USED

ITEM COUNT (BYTE, RECORD, OBJECT) (0) 0 ?

The purpose of this field varies with the type of command. For example, for write and read commands, the user may specify the record size, in decimal bytes. If the command is a reposition command, the user may specify the number of records, objects or file marks. There are also two special commands provided which use this value in unique

ways. For a branch command, the user would specify the command number to which (s)he wishes to branch. For the delay command, the value entered here is the relative delay length, with larger numbers producing longer delays. User experimentation may be required to produce desired delay.

ITERATION COUNT (D) 1 ?

This field allows the user to specify how many times the command should be issued before the program issues the next command. The value is entered in decimal.

Additional Commands

This same sequence of four questions will be repeated up to 6 more times, allowing the operator to create a command table with seven unique commands. The only noticeable difference in question format is that each time the command question is asked, its relative position in the Test 7 command table is identified.

2.5 Allowable Commands

The following commands are supported by Test 7. Please remember that the octal values are defined by the program and have no numerical correlation to TMSCP command opcodes. Also note that the diagnostic does not check for legality of the value entered or for valid command sequences. Operator error in either of these cases could result in bizarre program behavior.

Octal	Command	Description
10	RD	Read forward
20	WR	Write
30	CMP	Compare host data
40	ACC	Access
50	SPC	Space records
51	SCR	Space records reverse
60	SKP	Skip tape marks
61	SKR	Skip tape marks reverse
70	SPD	Space objects
71	SPR	Space objects reverse
100	WTM	Write tape mark
160	REW	Rewind
300	BR	Branch - item count specifies destination
310	DLY	Delay - item count specifies relative delay
377	END	End of sequence - necessary if sequence has less than 7 commands

2.6 Supervisor Run Time Flags

This program will support all of the PDP11 Diagnostic Supervisor flags except for those mentioned here.

LOE Loop on Error This flag will not be supported by this program.

524
525
526
527
528
529
530
531
532
533
534
535
536
537
538
539
540
541
542
543
544
545
546
547
548
549
550
551
552
553
554
555
556
557
558
559
560
561
562
563
564
565
566
567
568
569
570
571
572
573
574
575
576
577
578
579
580

581
582
583
584
585
586
587

Data reliability programs do not lend themselves to implementation of error loops.

IDR - Inhibit Drop Units - This flag will not be supported by this program due to the devices sequential operation. If an error of fatal extent happens on the device there is no way to continue running in any meaningful way.

589
590
591
592
593
594
595
596
597
598
599
600
601
602
603
604
605
606
607
608
609
610
611
612
613
614
615
616
617
618
619
620
621
622
623
624
625
626
627
628
629
630
631
632
633
634
635
636
637
638
639
640
641
642
643
644
645

3 ERROR INFORMATION

3.1 Error Reporting

TKDR provides a variety of information in its error printouts, most of which is self-explanatory. The following information is intended to clarify certain messages and abbreviations used.

3.2 Commands

All error printouts will contain a field indicating the command on which the error was detected. Refer to the TMSCP specification for detailed descriptions of these commands. Also, please note that commands currently used by TKDR are indicated by an asterisk.

RD*	read
WRT*	write
CMP	compare host data
ACC*	access
SPC*	space records (position)
SKP*	skip tape marks (position)
SPO*	space objects (position)
WTM*	write tape mark
ERS	erase
ERG	erase gap
AVL*	available
ONL*	online
SUC	set unit characteristics
REW*	rewind (position)
ABO	abort
GCS*	get command status
GUS*	get unit status
SCC*	set controller characteristics

The following two "commands" are used by TKDR for special purposes and are not actually sent as commands to the subsystem:

NUL	null - used by program to while waiting for last responses to real commands
INT	initialize - used by program to invoke the UQ Port init sequence

3.3 Type Of Error

Each error message includes one line of text intended to describe the type of error detected. There are three distinct sources of information used by the program to generate the text message: the status field of an end packet; an error log packet; and program detected error conditions.

3.4 Status Errors

646 These messages are derived from the status field of an end packet
 647 and correspond directly to the status codes as defined in the TMSCP
 648 specification.

649
 650 Invalid command issued
 651 Command aborted
 652 Unit offline
 653 Unit available error
 654 Unit write protected
 655 Data compare error
 656 Data error
 657 Host buffer access error
 658 Controller error
 659 Drive error
 660 Formatter error
 661 BOT encountered
 662 Tape mark encountered
 663 Data record truncated
 664 Position lost
 665 Serious exception
 666 Logical EOT encountered
 667

668 3.5 Error Log Packets

669 Certain messages will be generated as a result of receiving the
 670 "diagnostic mode" error log packet.

671 Retriable Data
 672 Hard CRC
 673 Data Underrun
 674 Data Overrun
 675 ECC Corrected
 676 CRC Error on ECC Block
 677
 678
 679
 680
 681
 682
 683

684 3.6 Program Detected Error Conditions

685 In addition to reporting errors detected by the subsystem, TKDR
 686 may generate additional error reports based on problems it detects.
 687 These error conditions are presented and defined here.

688 Invalid status received - the contents of the status field of an end
 689 packet is not a valid status as defined by
 690 TMSCP
 691
 692 Port-detected error - examination of the SA register indicated an
 693 error condition exists within the controller
 694
 695 Program command timeout the program received no end packet from the
 696 subsystem within the predefined command time-
 697 out.
 698
 699 Response out of sequence the program received an end packet for a
 700
 701
 702

703 sequential command other than the oldest out-
 704 standing command.
 705
 706
 707 Port initialization failed - the port failed to make an expected step
 708 transition during the UQ-Port init sequence.
 709
 710 Software data compare the program's data compare routine detected a
 711 miscomparison of read data to expected data.
 712
 713 Record length short - the data record read from tape was shorter than
 714 the record length expected.
 715

3.7 Drive Errors

716
 717
 718 On occurrence of a Drive Error, status code of 13(8), the error
 719 log packet will now contain a status code which is the drive error
 720 byte as returned by the drive. This value will be placed in the DRV
 721 CODE field of the error log packet.
 722

723 To understand the precise nature of the error condition it will
 724 be necessary to correlate the value presented in the printout against
 725 the table below.
 726

727	Octal	Hex	Description
728	1	01	Write lock violation
729	2	02	Drive fault
730	4	04	Communication exception (timeout, etc.)
731	6	06	Wrong track error (following a turnaround)
732	10	08	No cable or drive powered off
733	20	10	Synchronization failure - write/read
734	23	13	
735	44	22	
736	45	23	
737	47	27	
738	201	81	Failure to load to BOT
739	202	82	Failure to unload tape into cartridge
740	203	83	General motor or tach failure
741	204	84	Motor A failure
742	205	85	Motor B failure
743	206	86	Drive lost control of tape or bad tach
744	207	87	Excessive drag in tape transport
745	210	88	Failure to stop tape or remain stopped
746	211	89	Cartridge insert error
747	212	8A	Cartridge extract error
748	213	8B	CU attempted to move tape with drive in error
749	214	8C	Deceleration timeout error
750	215	8D	Second attempt to balance reels in init failed
751	220	90	8155 RAM memory failure in self-test
752	221	91	8155 timer failure
753	222	92	Read amplit (Hd 1) too low in calibrate
754	223	93	Read amplit (Hd 2) too low in calibrate
755	225	95	EOT sensed in R/W/S
756	226	96	BOT sensed in R/W/S
757	227	97	Drive block address overflow

760	230	98	Drive block address underflow
761	231	99	Servo error excessive speed variations
762	231	9A	Failure in tracking - currently not used
763	233	9B	Command error - not recognized
764	234	9C	Illegal command - incompatible with drive state
765	235	9D	Write lock error
766	236	9E	Write gate at wrong time
767	237	9F	No write gate for calibration track write
768	240	A0	Error sensing cal track 1 - bad head?
769	241	A1	Error sensing cal track 2 - bad head?
770	242	A2	Detection of edges of cal trk 1 out of spec
771	243	A3	Detection of edges of cal trk 2 out of spec
772	244	A4	Offset of cal trk 2 from 1 is too great
773	245	A5	Search for bottom edge of tape failed
774	246	A6	Bottom tape edge tolerance error
775	247	A7	Drive is overheating
776			
777	250	A8	No current in LED of BOT sensor (cable?)
778	251	A9	Hall switch sense lines Motor A questionable
779	252	AA	Tachometer failure

3.8 Hard Error Reports

Hard error reports, if not user disabled, will be generated anytime an error recovery process does not successfully complete.

Hard Error reports will typically be of the following format:

```
CZTKB HRD ERR 00014 ON UNIT 00 TST 003 SUB 000 PC: 020460
HARD DATA ERROR
COMMAND: RD      T/MSCP UNIT: 000(0)
PASS: 1(D)      DATA PAT: 01(0)
RECORD BYTE COUNT: 457(D)
OBJECT CNT : 000000026352(0)
```

RESPONSE PACKET

HIGH WORD	LOW WORD
000000(0)	026532(0)
000000(0)	000000(0)
000C50(0)	010240(0)
000000(0)	000733(0)
000000(0)	000000(0)
000000(0)	000000(0)
000000(0)	000000(0)
000000(0)	001413(0)
000000(0)	000733(0)

NOTE

Some error reports will not include a Response Packet field. For example a Command Timeout Error, by definition, results only when no response to a command has been received prior to expiration of the programs watch dog timer.

760
761
762
763
764
765
766
767
768
769
770
771
772
773
774
775
776
777
778
779
780
781
782
783
784
785
786
787
788
789
790
791
792
793
794
795
796
797
798
799
800
801
802
803
804
805
806
807
808
809
810
811
812
813
814
815
816

817
818
819
820
821
822
823
824
825
826
827
828
829
830
831
832
833
834
835
836
837
838
839

3.9 Soft Error Reports

Soft error reports, if not user disabled, will be generated anytime an error recovery process is successful. The soft error report will include the number of retries necessary in order to successfully complete the current operation. Soft Error reports will typically be of the following format:

```
CZTKB SFT RD ERR 00014 ON UNIT 00 TST 003 SUB 000 PC: 020460
ECC RECOVERED DATA ERROR
COMMAND: RD      T/MSCP UNIT: 000(0)
PASS:  1(D)     DATA PAT: 01(0)
OBJECT CNT : 000000026352(0)
TAP OBJ CNT: 000000026352(0)
TRK NUM:  6(D)  LEVEL:  0(D)  RETRIES:  1(D)
LOG BLK NUM:  0(D)  PHYS BLK NUM:  9932(D)
DRV CODE:  000(0)  DRV FLGS:  041(0)
DRV STATE: 000000(0)  INTERN STATUS:  002(0)
TAP CNT 0:  227(0)  TAP CNT 1:  015(0)
TAP CNT 2:  035(0)  RD/WR STATE:  000000(0)
OPER FLGS: 000000(0)
```

841
842
843
844
845
846
847
848
849
850
851
852
853
854
855
856
857
858
859
860
861
862
863
864
865
866
867
868
869
870
871
872
873
874
875
876
877
878
879
880
881
882
883
884
885
886
887
888
889
890
891
892
893
894
895
896
897

4 PERFORMANCE AND PROGRESS REPORTS

4.1 Statistics Matrix

	READ		WRITE			
	CH 1	CH 2	CH 1	CH 2		
SOFT DATA ERRORS						
RETRY RECOVERED	X	X	X	X		
ECC CORRECTED	X	X	N.A.			
HARD DATA ERRORS	X	X	X	X		
CRC ON ECC BLOCK	X	X	N.A.			
DATA COMPARE ERRORS		X	N.A.			
DATA UNDERRUN		N.A.	X			
DATA OVERRUN		X	N.A.			
MISPOSITIONS		X	X			
OTHERS		X				
TIMES DROPPED		X				
BYTES WRITTEN	X,XXX,XXX,XXX					
BYTES READ	X,XXX,XXX,XXX					
	TRK	PHY BLK	HWR	HRD	SWR	SRD
	0	26	0	0	1	0
	0	2474	0	0	1	0
	1	126	0	0	1	0
	1	10374	0	0	1	0

4.2 Read Error Definition

1. SOFT DATA ERRORS

- 0 Retry Corrected - ECC disabled or repositioning was required because >1 block in ECC group was bad.
- 0 ECC Corrected - CRC error occurred on data block but ECC has corrected it

2. Hard Data Errors - Maximum retries exhausted and data not recovered.

3. CRC Error on ECC Block - Data was read successfully, but CRC error occurred

4. Data Compare - No hardware detected errors, but the data compare failed. on an associated ECC block.

5. Data Overrun - The controller did not have sufficient buffer space for read data.

898
899
900
901
902
903
904
905
906
907
908
909
910
911
912
913
914
915
916
917
918
919
920
921
922
923
924
925

4.3 Write Error Definition

1. Retry Recovered - Operational write algorithm was enabled and controller successfully recovered from a write error. (In this case, media-induced write errors will appear in this category.)
2. Hard Data Errors - Write retries exhausted and block not successfully written.
3. Underrun - Controller ran out of write data blocks prior to a record boundary.

4.4 Miscellaneous

1. Mispositions - Times the drive lost position on tape.
2. Others - This is a tally of all errors not specifically called out in the error matrix.
3. Times Dropped - Times the drive has been dropped by the program.

927
928
929
930
931
932
933
934
935
936
937
938
939
940
941
942
943
944
945
946
947
948
949
950
951
952
953
954
955
956
957
958
959
960
961
962
963
964
965
966
967
968
969
970
971
972
973
974
975
976
977
978
979
980
981
982
983

5 TEST DESCRIPTIONS

5.1 Test 1 Basic Function Test

This test will execute a subset of the available commands on the unit under test. It serves as a quick verify test to ascertain that the unit can move tape and write/read predictably, without error. The subset of legal commands will be issued in a coherent manner.

The testing sequence, performed once will be as follows:

1. Execute online
2. Rewind to ensure that tape is at BOT,
3. Write two tapemarks, just after BOT,
4. Backspace two tapemarks,
5. Space forward to LEOT,
6. Rewind,
7. Write, using increasing byte counts, rotating through all data patterns, using decreasing file lengths. Files to be separated by tape marks.
8. Write LEOT after previous sequence,
9. Rewind,
10. Read records of the first file,
11. Space records over the second and third files,
12. Space objects over the fourth file,
13. Read records of the fifth file,
14. Skip reverse over four tape marks,
15. Skip forward one tape mark,
16. Read the second file set,
17. Space objects over the third record set,
18. Read the fourth record set,
19. Space objects to LEOT,
20. Space objects reverse to Just after BOT,
21. Skip four tape marks,
22. Space records over the fourth record set,

- 984
985
986
987
988
989
990
991
992
993
994
995
996
997
998
999
23. Skip a tape mark,
 24. Read the sixth record set,
 25. Skip two tape marks,
 26. Space objects reverse to the end of the second file set,
 27. Skip a tape mark,
 28. Read the third file set,
 29. Rewind tape,

5.2 Test 2 Quick Verify Read/Write Test

This test rewinds the tape, then executes the following sequence:

1. Write record set,
2. Write LEOT,
3. Rewind,
4. Reposition to just written record set,
5. Read the current record set,
6. Skip to LEOT.

for 5 iterations or until fatal error is encountered. This test permits retries, fixed record length (4096 bytes decimal), fixed number of records/set (250), and predetermined data patterns. This test will execute in a round-robin manner.

5.3 Test 3 Complex Read/Write Test

This test rewinds the tape, and executes the following sequence:

1. Write N records,
2. Write a tape mark,
3. Repeat 1 and 2 until EOT is reached,
4. Write 2 tape marks (LEOT),
5. Rewind,
6. Read N records,
7. Space 1 record (should see unexpected tape mark)

1000
1001
1002
1003
1004
1005
1006
1007
1008
1009
1010
1011
1012
1013
1014
1015
1016
1017
1018
1019
1020
1021
1022
1023
1024
1025
1026
1027
1028
1029
1030
1031
1032
1033
1034
1035
1036
1037
1038
1039
1040

1041
1042
1043
1044
1045
1046
1047
1048
1049
1050
1051
1052
1053
1054
1055
1056
1057
1058
1059
1060
1061
1062
1063
1064
1065
1066
1067
1068
1069
1070
1071
1072
1073
1074
1075
1076
1077
1078
1079
1080
1081
1082
1083
1084
1085
1086
1087
1088
1089
1090
1091
1092
1093
1094
1095
1096
1097

8. Repeat 6 and 7 until LEOT.

Number of records (N), and record size will be randomly selected. This sequence will permit hardware retries, if enabled by the operator. This test will run until EOT, LEOT or fatal error is detected. All data patterns including random data will be used in this test.

5.4 Test 4 Write Interchange Tape

This test will rewind the tape, then write until EOT or a fatal error is encountered. This test will keep track of the number of records and files written. If a fatal error is encountered, a message will report it, the tape on the unit will be rewound, and the unit prevented from executing further write operations.

5.5 Test 5 Read Unknown Tape

This test will rewind a tape, then read until EOT, LEOT or fatal error is encountered. This test will keep track of the number of records and files read. If a fatal error is encountered, a message will report it, the tape on the unit will be rewound, and the unit prevented from executing further read operations.

NOTE

Tests 4 and 5 can be used to perform a media interchange test for multiple drives. The program will not attempt to make any determination as to whether the unit that wrote the tape or the unit reading the tape is at fault for any errors.

5.6 Test 6 Start/Stop Write/Read Test

This test rewinds the tape, then executes the following sequence:

1. Write record set, stopping between each record,
2. Write a tape mark,
3. Repeat steps one and two until two tracks have been written,
4. Write LEOT,
5. Rewind,
6. Read the record set stopping between each record,
7. Skip a tape mark,
8. Repeat steps six and seven until LEOT is detected,

1098
1099
1100
1101
1102
1103
1104
1105
1106
1107
1108
1109
1110
1111
1112
1113
1114
1115
1116
1117

9. Rewind.

Until fatal error is encountered. This test permits retries, fixed record length (8096 bytes decimal), fixed number of records/set (250), and predetermined data patterns. This test will execute in a round-robin manner.

5.7 Test 7 Conversation Test

Conversation mode will run with or without error reports. The user can select, from a list of commands, a sequence which can be used to emulate a known failure mode. Between commands, the user can specify unique delays, ranging from 10 to 250 ms. The user can follow each tape command with integer values, the first indicating the byte/record/file count and the second indicating the # of repetitions necessary for that command.

@

```

1123
1134 .TITLE PROGRAM HEADER AND TABLES
1135 .SBTTL PROGRAM HEADER
1161
1163 000000 .ENABL ABS,AMA
1164 002000 = 2000
1166
1167 002000 BGNMOD
1168
1169 ;**
1170 ; THE PROGRAM HEADER IS THE INTERFACE BETWEEN
1171 ; THE DIAGNOSTIC PROGRAM AND THE SUPERVISOR.
1172 ; -
1173
1174 002000 POINTER BGNSW,BGNSFT,BGNRPT,ERRTBL,BGNDU,BGNSETUP
1175
1183
1184
1185 002000 HEADER CZTKB,A,0,15000,1,0
002000 L$NAME:: ;DIAGNOSTIC NAME
002000 103 .ASCII /C/
002001 132 .ASCII /Z/
002002 124 .ASCII /T/
002003 113 .ASCII /K/
002004 102 .ASCII /B/
002005 000 .BYTE 0
002006 000 .BYTE 0
002007 000 .BYTE 0
002010 L$REV:: ;REVISION LEVEL
002010 101 .ASCII /A/
002011 L$DEPO:: ;0
002011 060 .ASCII /0/
002012 L$UNIT:: ;NUMBER OF UNITS
002012 000001 .WORD T$PTHV
002014 L$TIML:: ;LONGEST TEST TIME
002014 015000 .WORD 15000
002016 L$HPCP:: ;PTR. TO H.W. QUES.
002016 046236 .WORD L$HARD
002020 L$SPCP:: ;PTR. TO S.W. QUES.
002020 046324 .WORD L$SOFT
002022 L$HPTP:: ;PTR. TO DEF. H.W. PTABLE
002022 002204 .WORD L$HW
002024 L$SPTP:: ;PTR. TO S.W. PTABLE
002024 002212 .WORD L$SW
002026 L$LADP:: ;DIAG. END ADDRESS
002026 115736 .WORD L$LAST
002030 L$STA:: ;RESERVED FOR APT STATS
002030 000000 .WORD 0
002032 L$CO:: .WORD 0
002032 000000 .WORD 0
002034 L$DTYP:: ;DIAGNOSTIC TYPE
002034 000001 .WORD 1
002036 L$APT:: ;APT EXPANSION
002036 000000 .WORD 0
002040 L$DTP:: ;PTR. TO DISPATCH TABLE
002040 002124 .WORD L$DISPATCH
002042 L$PRIO:: ;DIAGNOSTIC RUN PRIORITY

```

002042	000000		.WORD	0	
002044		L\$ENVI::			;FLAGS DESCRIBE HOW IT WAS SETUP
002044	000000		.WORD	0	
002046		L\$EXP1::			;EXPANSION WORD
002046	000000		.WORD	0	
002050		L\$MREV::			;SVC REV AND EDIT #
002050	004		.BYTE	C\$REVISION	
002051	000		.BYTE	C\$EDIT	
002052		L\$EF::			;DIAG. EVENT FLAGS
002052	000000		.WORD	0	
002054	000000		.WORD	0	
002056		L\$SPC::			
002056	000000		.WORD	0	
002060		L\$DEVP::			; POINTER TO DEVICE TYPE LIST
002060	002174		.WORD	L\$DVTYP	
002062		L\$REPP::			;PTR. TO REPORT CODE
002062	034210		.WORD	L\$RPT	
002064		L\$EXP4::			
002064	000000		.WORD	0	
002066		L\$EXP5::			
002066	000000		.WORD	0	
002070		L\$AUT::			;PTR. TO ADD UNIT CODE
002070	000000		.WORD	0	
002072		L\$DUT::			;PTR. TO DROP UNIT CODE
002072	040700		.WORD	L\$DU	
002074		L\$LUN::			;LUN FOR EXERCISERS TO FILL
002074	000000		.WORD	0	
002076		L\$DESP::			;POINTER TO DIAG. DESCRIPTION
002076	002142		.WORD	L\$DESC	
002100		L\$LOAD::			;GENERATE SPECIAL AUTOLOAD EMT
002100	104035		EMT	E\$LOAD	
002102		L\$ETP::			;POINTER TO ERR_TBL
002102	013262		.WORD	L\$ERR_TBL	
002104		L\$ICP::			;PTR. TO INIT CODE
002104	037004		.WORD	L\$INIT	
002106		L\$CCP::			;PTR. TO CLEAN UP CODE
002106	040372		.WORD	L\$CLEAN	
002110		L\$ACP::			;PTR. TO AUTO CODE
002110	040370		.WORD	L\$AUTO	
002112		L\$PRT::			;PTR. TO PROTECT TABLE
002112	021036		.WORD	L\$PROT	
002114		L\$TEST::			;TEST NUMBER
002114	000000		.WORD	0	
002116		L\$DLY::			;DELAY COUNT
002116	000000		.WORD	0	
002120		L\$HIME::			;PTR. TO HIGH MEM
002120	000000		.WORD	0	

1187
 1188
 1189
 1190
 1191
 1192
 1193
 1194 002122
 002122 000007
 002124
 002124 040714
 002126 043116
 002130 043612
 002132 044270
 002134 044650
 002136 045170
 002140 045646

.SBTTL DISPATCH TABLE

;
 ;**
 ; THE DISPATCH TABLE CONTAINS THE STARTING ADDRESS OF EACH TEST.
 ; IT IS USED BY THE SUPERVISOR TO DISPATCH TO EACH TEST.
 ;--

DISPATCH 7
 .WORD 7
 L\$DISPATCH:;
 .WORD T1
 .WORD T2
 .WORD T3
 .WORD T4
 .WORD T5
 .WORD T6
 .WORD T7

1195
 1196
 1197 002142
 002142
 002142

103 132 124

DESCRIPT <CZTKBAO DATA RELIABILITY>
 L\$DESC:;
 .ASCIZ /CZTKBAO DATA RELIABILITY/
 .EVEN

1198
 1199
 1200
 1201
 1202
 1203 002174
 002174
 002174

124 113 065

DEV TYP <TK50>
 L\$DVTYP:;
 .ASCIZ *TK50*
 .EVEN

1204

1206
 1207
 1208
 1209
 1210
 1211
 1212
 1213
 1214 002202
 002202 000002
 002204
 002204
 1215
 1216 002204 174500
 1217 002206 000000
 1218
 1219 002210
 002210

```
.SBTTL  DEFAULT HARDWARE P-TABLE

; **
; THE DEFAULT HARDWARE P-TABLE CONTAINS DEFAULT VALUES OF
; THE TEST-DEVICE PARAMETERS.  THE STRUCTURE OF THIS TABLE
; IS IDENTICAL TO THE STRUCTURE OF THE HARDWARE P-TABLES.
; --

          BGNHW  DFPTBL
          .WORD  L10000-L$HW/2

L$HW::
DFPTBL::

          174500          ;TKIP ADDRESS
          0               ;T/MSCP UNIT NUMBER

          ENDPHW

L10000:
```



```

1221          .SBTTL  SOFTWARE P-TABLE
1222
1223          ;**
1224          ; THE SOFTWARE P-TABLE CONTAINS THE VALUES OF THE PROGRAM
1225          ; PARAMETERS THAT CAN BE CHANGED BY THE OPERATOR.
1226          ;--
1227
1228 002210          BGNSW  SFPTBL
          002210 000041  .WORD  L10001-L$SW/2
          002212          L$SW::
          002212          SFPTBL::
1229
1230 002212          000  CLOCK::          .BYTE  0          ;ENABLE TIME OF DAY CLOCK
1231 002213          000  HOURS::         .BYTE  0          ;HOURS FOR TIME OF DAY CLOCK
1232 002214          000  MINUTE::        .BYTE  0          ;MINUTES FOR TIME OF DAY CLOCK
1233 002215          000  SECOND::       .BYTE  0          ;SECONDS FOR TIME OF DAY CLOCK
1234 002216          000  SUBSEC::       .BYTE  0          ;SUBSECONDS FOR TIME OF DAY CLOCK
1235
1236 002217          000  CONTPA::        .BYTE  0          ;CHANGE CONTROLLER PARARMETERS
1237 002220          001  SERREC::       .BYTE  1          ;ENABLE ERROR CORRECTION FLAG
1238 002221          001  SERCOR::       .BYTE  1          ;ENABLE ERROR RECOVERY FLAG
1239 002222          001  PADING::       .BYTE  1          ;ENABLE PAD BLOCKING
1240
1241 002223          000  PRNTPA::        .BYTE  0          ;CHANGE PRINT PARAMETERS
1242 002224          000  SOERRP::       .BYTE  0          ;ENABLE SOFT ERROR REPORT FLAG
1243 002225          001  RDSOER::       .BYTE  1          ;ENABLE READ SOFT ERRORS ONLY
1244 002226          000  CLRMED::       .BYTE  0          ;ENABLE CLEAR MEDIA TABLE EVERY PASS
1245 002227          000  MEDTBL::       .BYTE  0          ;ENABLE PRINT MEDIA TABLE CONTENTS
1246 002230          000  NOCLR::       .BYTE  0          ;ENABLE CLEAR STATS ON FATAL ERROR
1247 002231          000  DMPFLG::       .BYTE  0          ;ENABLE PROGRAM TABLE DUMP ON ERROR
1248
1249 002232          T7TBL::              ;COMMAND TABLE TOP -6
1250
1251 002232          000  TESTPA::        .BYTE  0          ;CHANGE TEST PARAMETERS
1252 002233          000  PATERN::       .BYTE  0          ;CHANGE DATA PATTERN
1253 002234          001  T3ONLY::       .BYTE  1          ;RUN TEST 3 ONLY
1254 002235          001  T5CMP::       .BYTE  1          ;ENABLE DATA COMPARES IN TEST 5
1255 002236          000  DMPBUF::       .BYTE  0          ;ENABLE READ BUFFER DUMP IN TEST 5
1256 002237          000  CHGFLG::       .BYTE  0          ;CHANGE CMD SEQ TABLE FLAG
1257
1258 002240          160  T7CMD1:         .BYTE  REW          ;REWIND
1259 002241          000          .BYTE  NULPAT
1260 002242          000000          .WORD  0
1261 002244          000001          .WORD  1
1262
1263 002246          020  T7CMD2:         .BYTE  WR          ;WRITE RECORDS
1264 002247          200          .BYTE  ALLPAT
1265 002250          004000          .WORD  2048.
1266 002252          000310          .WORD  200.
1267
1268 002254          100  T7CMD3:         .BYTE  WTM          ;WRITE TAPE MARK
1269 002255          000          .BYTE  NULPAT
1270 002256          000000          .WORD  0
1271 002260          000002          .WORD  2
1272
1273 002262          160  T7CMD4:         .BYTE  REW          ;REWIND
1274 002263          000          .BYTE  NULPAT

```

1275	002264	000000		.WORD	0	
1276	002266	000001		.WORD	1	
1277						
1278	002270	010	T7CMD5:	.BYTE	RD	;READ RECORDS
1279	002271	200		.BYTE	ALLPAT	
1280	002272	004000		.WORD	2048.	
1281	002274	000310		.WORD	200.	
1282						
1283	002276	060	T7CMD6:	.BYTE	SKP	;SKIP TAPE MARK
1284	002277	000		.BYTE	NULPAT	
1285	002300	000001		.WORD	1	
1286	002302	000002		.WORD	2	
1287						
1288	002304	160	T7CMD7:	.BYTE	REW	;REWIND
1289	002305	000		.BYTE	NULPAT	
1290	002306	000000		.WORD	0	
1291	002310	000001		.WORD	1	
1292						
1293	002312	177777	T7END:	.WORD	1	
1294				.EVEN		
1295						
1296	002314		ENDSW			
	002314		L10001:			
1297						
1298	002314		ENDMOD			

000034 EF.PWR== 28. ; (004000) A POWER-FAIL/POWER-UP OCCURRED

;
; PRIORITY LEVEL DEFINITIONS

000340 PRI07== 340
000300 PRI06== 300
000240 PRI05== 240
000200 PRI04== 200
000140 PRI03== 140
000100 PRI02== 100
000040 PRI01== 40
000000 PRI00== 0

;
; OPERATOR FLAG BITS

000004 EVL== 4
000010 LOT== 10
000020 ADR== 20
000040 IDU== 40
000100 ISR== 100
000200 UAM== 200
000400 BOE== 400
001000 PNT== 1000
002000 PRI== 2000
004000 IXE== 4000
010000 IBE== 10000
020000 IER== 20000
040000 LOE== 40000
100000 HOE== 100000

```

1405
1406
1407 ;/*/*/*/*/*/*/*/*/*/*/*/*/*/*/*/*/*/*/*/*/*/*/*/*/*/*/*/*/*/*
1408 ; 2.0 TMSCP COMMAND LITERALS
1409 ;/*/*/*/*/*/*/*/*/*/*/*/*/*/*/*/*/*/*/*/*/*/*/*/*/*/*/*/*/*
1410
1411 ; 2.1 COMMAND PACKET OPCODES
1412 000001 OP.ABO == 01 ; ABORT COMMAND
1413 000020 OP.ACC == 20 ; ACCESS COMMAND
1414 000010 OP.AVL == 10 ; AVAILABLE COMMAND
1415 000040 OP.CMP == 40 ; COMPARE HOST DATA COMMAND
1416 000013 OP.DAP == 13 ; DETERMINE ACCESS PATH COMMAND
1417 000022 OP.ERS == 22 ; ERASE COMMAND
1418 000026 OP.ERG == 26 ; ERASE GAP COMMAND
1419 000002 OP.GCS == 02 ; GET COMMAND STATUS COMMAND
1420 000003 OP.GUS == 03 ; GET UNIT STATUS COMMAND
1421 000011 OP.ONL == 11 ; ONLINE COMMAND
1422 000041 OP.RD == 41 ; READ COMMAND
1423 000045 OP.REP == 45 ; REPOSITION COMMAND
1424 000004 OP.SCC == 04 ; SET CONTROLLER CHARACTERISTICS COMMAND
1425 000012 OP.SUC == 12 ; SET UNIT CHARACTERISTICS COMMAND
1426 000042 OP.WR == 42 ; WRITE COMMAND
1427 000044 OP.WTM == 44 ; WRITE TAPE MARK COMMAND
1428 000200 OP.END == 200 ; END MESSAGE FLAG
1429 000100 OP.AVA == 100 ; AVAILABLE ATTENTION MESSAGE
1430 000102 OP.ACP == 102 ; ACCESS PATH ATTENTION MESSAGE
1431
1432 ; 2.2 COMMAND MODIFIERS
1433 040000 MD.CMP == 040000 ; COMPARE
1434 020000 MD.CSE == 020000 ; CLEAR SERIOUS EXCEPTION
1435 001000 MD.SEC == 001000 ; SUPPRESS ERROR CORRECTION
1436 000400 MD.SER == 000400 ; SUPPRESS ERROR RECOVERY
1437 000200 MD.DLE == 000200 ; DETECT LEOT
1438 000100 MD.IMM == 000100 ; IMMEDIATE
1439 000040 MD.EXC == 000040 ; EXCLUSIVE ACCESS
1440 000020 MD.UNL == 000020 ; UNLOAD
1441 000010 MD.REV == 000010 ; REVERSE
1442 000004 MD.OBC == 000004 ; OBJECT COUNT
1443 000004 MD.SWP == 000004 ; SET WRITE PROTECT
1444 000002 MD.RWD == 000002 ; REWIND
1445 000002 MD.ALL == 000002 ; ALL CLASS DRIVERS
1446 000001 MD.SPD == 000001 ; SPEED
1447 000001 MD.NXU == 000001 ; NEXT UNIT
1448
1449 ; 2.3 GENERIC COMMAND PACKET OFFSETS
1450 000000 P.CRF == 0 ; COMMAND REFERENCE NUMBER
1451 000004 P.UNIT == 4 ; UNIT NUMBER
1452 000010 P.OPCD == 10 ; OPCODE
1453 000012 P.MOD == 12 ; MODIFIERS
1454 000014 P.BCNT == 14 ; BYTE COUNT
1455 000020 P.BUFF == 20 ; BUFFER DESCRIPTOR
1456
1457 ; 2.4 ABORT AND GET COMMAND STATUS OFFSETS PACKET OFFSETS
1458 000014 P.OTRF == 14 ; OUTSTANDING COMMAND REFERENCE NUMBER
1459
1460 ; 2.5 ONLINE AND SET UNIT CHARACTERISTICS PACKET OFFSETS
1461 000014 P.UNFL == 14 ; UNIT FLAGS

```

```

1462      000034      P.DVPM ==      34      ;DEVICE DEPENDENT PARAMETERS
1463      000040      P.FORM ==      40      ;FORMAT
1464      000042      P.SPED ==      42      ;SPEED
1465
1466      ; 2.6 REPOSITION COMMAND PACKET OFFSETS
1467      000014      P.REDD ==      14      ;RECORD/OBJECT COUNT
1468      000020      P.TMGC ==      20      ;TAPE MARK COUNT
1469
1470      ; 2.7 SET CONTROLLER CHARACTERISTICS PACKET OFFSETS
1471      000014      P.VRSN ==      14      ;MSCP VERSION
1472      000016      P.CNTF ==      16      ;CONTROLLER FLAGS
1473      000020      P.HTMO ==      20      ;HOST TIMEOUT
1474      000024      P.TIME ==      24      ;QUAD-WORD TIME AND DATE
1475      000034      P.CTPM ==      34      ;CONTROLLER DEPENDENT PARAMETERS
1476

```



```

1546 ;/*/*/*/*/*/*/*/*/*/*/*/*/*/*/*/*/*/*/*/*/*/*/*/*/*/*/*/*/*\
1547 ; 4.0 ERROR LOG LITERALS
1548 ;/*/*/*/*/*/*/*/*/*/*/*/*/*/*/*/*/*/*/*/*/*/*/*/*/*/*/*/*/*\
1549
1550 ; 4.1 ERROR LOG MESSAGE FORMAT CODES
1551 000000 FM.CNT == 000000 ;CONTROLLER ERRORS
1552 000001 FM.BAD == 000001 ;HOST MEMORY ACCESS ERRORS WITH BUS ADDRESS
1553 000005 FM.TPE == 000005 ;TAPE TRANSFER ERRORS
1554
1555 ; 4.2 ERROR LOG MESSAGE FLAGS
1556 000200 LF.SUC == 000200 ;OPERATION SUCCESSFUL
1557 000100 LF.CON == 000100 ;OPERATION CONTINUING
1558 000001 LF.SNR == 000001 ;SEQUENCE NUMBER REQUEST
1559
1560 ; 4.3 TAPE FORMAT FLAG VALUES
1561 000001 TF.800 == 000001 ;NRZI 800 BPI
1562 000002 TF.PE == 000002 ;PHASE ENCODED 1600 BPI
1563 000004 TF.GCR == 000004 ;GROUP CODED RECORDING 6250 BPI
1564 000010 TF.BLK == 000010 ;6667 BPI
1565
1566 ; 4.4 ERROR LOG MESSAGE OFFSETS
1567 000000 L.CRF == 0 ;COMMAND REFERENCE NUMBER
1568 000004 L.UNIT == 4 ;UNIT NUMBER
1569 000006 L.SEQN == 6 ;SEQUENCE NUMBER
1570 000010 L.FMT == 10 ;FORMAT
1571 000011 L.FLGS == 11 ;ERROR LOG MESSAGE FLAGS
1572 000012 L.EVNT == 12 ;EVENT CODES
1573 000014 L.CNTI == 14 ;CONTROLLER ID
1574 000024 L.CSVR == 24 ;CONTRCROLLER SOFTWARE VERSION
1575 000025 L.CHVR == 25 ;CONTROLLER HARDWARE VERSION
1576 000026 L.MLUN == 26 ;MULTI UNIT CODE
1577 000030 L.UNTI == 30 ;UNIT ID
1578 000030 L.BADR == 30 ;BUS ADDRESS
1579 000040 L.USVR == 40 ;UNIT SOFTWARE VERSION
1580 000041 L.UHVR == 41 ;UNIT HARDWARE VERSION
1581 000042 L.LVL == 42 ;RETRY LEVEL
1582 000042 L.FMTD == 42 ;FORMAT DEPENDENT
1583 000043 L.RTRY == 43 ;RETRY COUNT FOR THE CURRENT LEVEL
1584 000044 L.GPCT == 44 ;GAP COUNT
1585 000044 L.VSER == 44 ;VOLUME SERIAL NUMBER
1586 000044 L.PSTN == 44 ;TAPE OBJECT COUNT
1587 000050 L.STI == 50 ;STI INFORMATION
1588 000050 L.FHVR == 50 ;FORMATTER HARDWARE VERSION
1589 000051 L.FSVR == 51 ;FORMATTER SOFTWARE VERSION
1590 000052 L.STS == 52 ;CONTROLLER INTERNAL STATUS
1591 000053 L.DRVC == 53 ;DRIVE ERROR CODE
1592 000054 L.DFLG == 54 ;DRIVE STATE FLAGS
1593 000055 L.TRK == 55 ;LOGICAL TRACK NUMBER
1594 000056 L.PBLK == 56 ;PHYSICAL BLOCK NUMBER
1595 000060 L.LBLK == 60 ;LOGICAL BLOCK NUMBER
1596 000061 L.CNT0 == 61 ;TAPE COUNT 0
1597 000062 L.CNT1 == 62 ;TAPE COUNT 1
1598 000063 L.CNT2 == 63 ;TAPE COUNT 2
1599 000064 L.DRVS == 64 ;DRIVE STATE
1600 000066 L.RWST == 66 ;READ/WRITE STATE
1601 000070 L.OPFL == 70 ;OPERATION FLAGS
1602

```

		;	4.5	STATUS AND EVENT CODES	
1603					
1604	000037	ST.MSK	==	37	;STATUS/EVENT CODE MASK
1605	000040	ST.SUB	==	40	;SUB-CODE MULTIPLIER
1606	000000	ST.SUC	==	0	;SUCCESS
1607	000001	ST.CMD	==	1	;INVALID COMMAND
1608	000002	ST.ABO	==	2	;COMMAND ABORTED
1609	000003	ST.OFL	==	3	;UNIT-OFFLINE
1610	000004	ST.AVL	==	4	;UNIT-AVAILABLE
1611	000005	ST.MFE	==	5	;MEDIA FORMAT ERROR
1612	000006	ST.WPR	==	6	;WRITE PROTECTED
1613	000007	ST.CMP	==	7	;COMPARE ERROR
1614	000010	ST.DAT	==	10	;DATA ERROR
1615	000011	ST.HST	==	11	;HOST BUFFER ACCESS ERROR
1616	000012	ST.CNT	==	12	;CONTROLLER ERROR
1617	000013	ST.DRV	==	13	;DRIVE ERROR
1618	000014	ST.FNT	==	14	;FORMATTER ERROR
1619	000015	ST.BOT	==	15	;BOT ENCOUNTERED
1620	000016	ST.TM	==	16	;TAPE MARK ENCOUNTERED
1621	000020	ST.RDT	==	20	;RECORD DATA TRUNCATED
1622	000021	ST.POL	==	21	;POSITION LOST
1623	000022	ST.SEX	==	22	;SERIOUS EXCEPTION
1624	000023	ST.LED	==	23	;LEOT DETECTED
1625	000037	ST.DIA	==	37	;INTERNAL DIAGNOSTIC MESSAGE
1626	000400	ST.ONL	==	400	;UNIT ALREADY ONLINE
1627					
1628	000010	EV.LGP	==	10	;LONG GAP ENCOUNTERED
1629	000050	EV.DST	==	50	;DATA SYNC TIMEOUT
1630	000052	EV.CTO	==	52	;COMM CHANNEL TIMEOUT
1631	000053	EV.SRT	==	53	;DRIVE COMMAND TIMEOUT
1632	000113	EV.SRI	==	113	;CONTROLLER DETECTED TRANSMISSION ERROR
1633	000150	EV.COR	==	150	;CORRECTABLE ERROR
1634	000152	EV.IDS	==	152	;INTERNAL INCONSISTENCY ERROR
1635	000153	EV.SER	==	153	;SOFT ERROR
1636	000213	EV.HER	==	213	;HARD ERROR
1637	000350	EV.URE	==	350	;UNRECOVERABLE DATA ERROR


```

1696      000006      PAT6      ==      6      ;1110 REPEATING PATTERN
1697      000007      PAT7      ==      7      ;COMBINATION PATTERN 3 AND 5
1698      000010      ENDPAT   ==      8      ;RANDOM PATTERN VALUE
1699      000200      ALLPAT   ==     200      ;CYCLE THROUGH ALL PATTERNS
1700      000002      UNTSTP   ==      2      ;STEP THROUGH UNITS
1701      000000      HSTIMO   ==      0      ;HOST TIMEOUT VALUE
1702      000000      MSCPVR   ==      0      ;MSCP VERSION NUMBER
1703      177776      LOBYTE   ==     -2      ;LOW BYTE OFFSET FOR COMPARE DATA
1704      177777      HIBYTE   ==     -1      ;HIGH BYTE OFFSET FOR COMPARE DATA
1705      004716      T2END    ==    2510.     ;RECORDS TO FILL 2 TRACKS
1706      000004      N        ==      4      ;VALUE USED IN SUBITR
1707      000001      ONE      ==      1      ;BYTE OFFSET
1708
1709      ; 5.7  ERROR MASKING LITERALS
1710      000001      LEDB     ==    000001     ;DETECT LOGICAL END OF TAPE
1711      000002      RDT8     ==    000002     ;RECORD DATA TRUNCATED
1712      000004      SEX8     ==    000004     ;SERIOUS EXCEPTION
1713      000010      TMB      ==    000010     ;ENCOUNTERED TAPE MARK
1714      000020      WPRB     ==    000020     ;DRIVE WRITE PROTECTED
1715      000040      AVLB     ==    000040     ;UNIT AVAILABLE
1716      000100      ONLB     ==    000100     ;UNIT ONLINE
1717
1718      ; 5.8  ERROR TYPE LITERALS
1719      000000      SYSFAT   ==      0      ;SYSTEM FATAL ERROR
1720      000001      DEVFAT   ==      1      ;DEVICE FATAL ERROR
1721      000002      HARD     ==      2      ;HARD DEVICE ERROR
1722      000003      SOFT     ==      3      ;SOFT DEVICE ERROR
1723      000004      STATUS   ==      4      ;STATUS MESSAGE
1724
1725      ; 5.9  BIT VALUES FOR LUN FLAG
1726      000001      INTDON   ==    000001     ;INITIALIZATION HAS BEEN DONE ON THIS UNIT
1727      000002      SEREXC   ==    000002     ;A SERIOUS EXCEPTION CONDITION EXISTS
1728      000004      NOTALY   ==    000004     ;DON'T TALLY BYTES FOR THIS COMMAND
1729      000010      EOTPR    ==    000010     ;EOT PRINTED FOR THIS UNIT
1730      000020      ODDFLG   ==    000020     ;ODD BYTE COUNT FLAG
1731      000040      MTBLOV   ==    000040     ;MEDIA STATS OVERFLOW FLAG
1732      000100      ECCFLG   ==    000100     ;DON'T DECREMENT ECC COUNT FLAG
1733
1734      ;PROGRAM CONTROL FLAG BIT VALUES
1735      000001      T7BRFL   ==    000001     ;BRANCH FLAG FOR TEST 7
1736      000002      NCLKFL   ==    000002     ;NO CLOCK PRESENT FLAG
1737      000004      TCNTFL   ==    000004     ;COUNT RECORDS AND TAPE MARKS FLAG
1738      000010      DRERFL   ==    000010     ;DRIVE ERROR FLAG
1739      000020      GCSCFL   ==    000020     ;GET COMMAND STATUS COMMAND FLAG
1740      000040      GCSRFL   ==    000040     ;GET COMMAND STATUS RESPONSE FLAG
1741      000100      NOMDTB   ==    000100     ;DON'T PRINT MEDIA DEFECTS FLAG
1742      000200      CMDONE   ==    000200     ;ALL COMMANDS ISSUED FLAG
1743      000400      DROPIT   ==    000400     ;DRIVE BEING DROPPED
    
```


1795				
1796		: 6.7	LUN TABLE OFFSETS	
1797	000000	TKIP	== 0	;IP REGISTER ADDRESS
1798	000002	TKSA	== 2	;SA REGISTER ADDRESS
1799	000004	TKUNIT	== 4	;TMSCP DEVICE UNIT NUMBER
1800	000006	CMDSEQ	== 6	;COMMAND REFERENCE NUMBER
1801	000010	SLTUSE	== 10	;BIT MAP OF RESPONSES RECEIVED
1802	000012	CMDSSV	== 12	;COMMAND DESCRIPTOR
1803	000014	CNUSAV	== 14	;NEW COMMAND BUFFER POINTER
1804	000016	COLSAV	== 16	;OLD COMMAND BUFFER POINTER
1805	000020	RNUSAV	== 20	;NEW RESPONSE BUFFER POINTER
1806	000022	ROLSAV	== 22	;OLD RESPONSE BUFFER POINTER
1807	000024	PATSAV	== 24	;DATA PATTERN
1808	000026	LUNFLG	== 26	;INITIALIZATION FLAG
1809	000030	LEOTFL	== 30	;UNIT LOGICAL END OF TAPE FLAG
1810	000032	UNDROP	== 32	;UNIT DROF COUNT
1811	000034	OBJFDL	== 34	;OBJECT COUNT LOW ORDER
1812	000036	OBJFDH	== 36	;OBJECT COUNT HIGH ORDER
1813	000040	S1READ	== 40	;SOFT READ ERROR CHANNEL 1
1814	000042	S2READ	== 42	;SOFT READ ERROR CHANNEL 2
1815	000044	S1WRIT	== 44	;SOFT WRITE ERROR CHANNEL 1
1816	000046	S2WRIT	== 46	;SOFT WRITE ERROR CHANNEL 2
1817	000050	EC1COR	== 50	;ECC CORRECTABLE CHANNEL 1
1818	000052	EC2COR	== 52	;ECC CORRECTABLE CHANNEL 2
1819	000054	H1READ	== 54	;HARD READ ERROR CHANNEL 1
1820	000056	H2READ	== 56	;HARD READ ERROR CHANNEL 2
1821	000060	H1WRIT	== 60	;HARD WRITE ERROR CHANNEL 1
1822	000062	H2WRIT	== 62	;HARD WRITE ERROR CHANNEL 2
1823	000064	RTYEC1	== 64	;CRC ERROR ON ECC BLOCK CHANNEL 1
1824	000066	RTYEC2	== 66	;CRC ERROR ON ECC BLOCK CHANNEL 2
1825	000070	DTCPER	== 70	;DATA COMPARE ERROR
1826	000072	UNDRUN	== 72	;DATA UNDERRUN
1827	000074	OVRRUN	== 74	;DATA OVERRUN
1828	000076	RMSPOS	== 76	;READ MISPOSITION ERROR
1829	000100	WMSPOS	== 100	;WRITE MISPOSITION ERROR
1830	000102	OTHER	== 102	;OTHER ERRORS
1831	000104	DROPPD	== 104	;TIMES UNIT WAS DROPPED
1832	000106	NOERR	== 106	;NO ERROR
1833	000110	RDBYT1	== 110	;HUNDREDS BYTES READ
1834	000112	RDBYT2	== 112	;THOUSANDS BYTES READ
1835	000114	RDBYT3	== 114	;MILLIONS BYTES READ
1836	000116	RDBYT4	== 116	;BILLIONS BYTES READ
1837	000120	WRBYT1	== 120	;HUNDREDS BYTES WRITTEN
1838	000122	WRBYT2	== 122	;THOUSANDS BYTES WRITTEN
1839	000124	WRBYT3	== 124	;MILLIONS BYTES WRITTEN
1840	000126	WRBYT4	== 126	;BILLIONS BYTES WRITTEN
1841	000130	TBLTOP	== 130	;TOP OF MEDIA STATS TABLE
1842	000132	TBLBTM	== 132	;BOTTOM OF MEDIA STATS TABLE
1843	000134	LSTENT	== 134	;FIRST FREE SLOT IN MEDIA STATS TABLE
1844	000136	SED1	== 136	;PRIME RANDOM GENERATOR SEED
1845	000140	SED2	== 140	;PRIME RANDOM GENERATOR SEED
1846	000142	SED3	== 142	;PRIME RANDOM GENERATOR SEED
1847	000144	SEED1	== 144	;RANDOM GENERATOR SEED
1848	000146	SEED2	== 146	;RANDOM GENERATOR SEED
1849	000150	SEED3	== 150	;RANDOM GENERATOR SEED
1850	000152	OLDBLK	== 152	;SAVE OF LAST BLOCK IN ERROR
1851	000154	OLDTRK	== 154	;SAVE OF LAST TRACK IN ERROR

1853	000156	URSPBF	==	156	;START OF THIS UNITS RESPONSE BUFFER
1854	000160	URBEND	==	160	;END OF THIS UNITS RESPONSE BUFFER
1855	000162	URDSRG	==	162	;START OF THIS UNITS RESPONSE DESCRIPTOR RING
1856	000164	URDEND	==	164	;END OF THIS UNITS RESPONSE DESCRIPTOR RING
1857	000166	UCDSRG	==	166	;START OF THIS UNITS COMMAND DESCRIPTOR RING
1858	000170	UCDEND	==	170	;END OF THIS UNITS COMMAND DESCRIPTOR RING
1859	000172	LUNSTP	==	172	;OFFSET TO NEXT LUN BLOCK

1964					
1965	003414	000000	RESP::	.WORD 0	;DRIVER RESPONSE COUNT
1966	003416	000000	BYTES::	.WORD 0	;BYTE COUNT
1967	003420	000000	ITERS::	.WORD 0	;ITERATION COUNT
1968	003422	000000	BUFADR::	.WORD 0	;COMMAND BUFFER ADDRESS
1969	003424	000000	SUBCNT::	.WORD 0	;SUB-ITERATION COUNT FOR DATA COMPARES
1970					
1971	003426	000000	RANWRD::	.WORD 0	;USED BY RANGEN
1972	003430	000000	RAN1::	.WORD 0	;SEED WORK LOCATION
1973	003432	000000	RAN2::	.WORD 0	;SEED WORK LOCATION
1974	003434	000000	RAN3::	.WORD 0	;SEED WORK LOCATION
1975					
1976	003436	000000	SAVDIF::	.WORD 0	;COMMAND AND RESPONSE COUNT DIFFERENCE
1977	003440	000000	TSTMSK::	.WORD 0	;TEST LOAD WITH ACCEPTABLE ERROR CODES
1978	003442	000000	WRKMSK::	.WORD 0	;USED BY ERROR DECODE
1979					
1980	003444	000000	CMPERR::	.WORD 0	;NUMBER OF BYTES IN ERROR
1981	003446		BYTADD::	.BLKW 10.	;SAVE TABLE FOR BYTE IN ERROR ADDRESS
1982		003472	TBLEND ==		;END OF BYTE ADDRESS TABLE
1983	003472		DATBL::	.BLKW 10.	;SAVE TABLE FOR BYTE IN ERROR DATA
1984	003516	000000	PCFLAG::	.WORD 0	;PROGRAM CONTROL FLAGS
1985					
1986	003520	000000	OBJECT::	.WORD 0	;OBJECT COUNTER FOR TEST 2
1987	003522	000000	PASCNT::	.WORD 0	;PASS COUNTER
1988	003524	000000	UDROP::	.WORD 0	;NUMBER OF DROPPED UNITS
1989	003526	000000	UEOT::	.WORD 0	;COUNT OF UNITS AT EOT
1990					
1991	003530	000000	R8::	.WORD 0	;USED FOR TEMP STORAGE
1992	003532	000000	R9::	.WORD 0	;USED FOR TEMP STORAGE
1993	003534	000000	R10::	.WORD 0	;USED FOR TEMP STORAGE
1994	003536	000000	R11::	.WORD 0	;USED FOR TEMP STORAGE
1995	003540	000000	R12::	.WORD 0	;USED FOR TEMP STORAGE
1996					
1997	003542	000000	SECRNS::	.WORD 0	;SERIOUS EXCEPTION CMD REF #
1998	003544	000000	TBLOFF::	.WORD 0	;MEDIA ERROR TABLE OFFSET
1999	003546	000000	RECCNT::	.WORD 0	;NUMBER OF RECORDS
2000	003550	000000	TMCNT::	.WORD 0	;NUMBER OF TAPE MARKS
2001	003552	000000	LOHEX::	.WORD 0	;LOW ORDER HEX DIGIT
2002					
2003	003554	000000	HIHEX::	.WORD 0	;HIGH ORDER HEX DIGIT
2004	003556	000000	EVENT::	.WORD 0	;EVENT CODE STORAGE
2005	003560	000000	R3SAVE::	.WORD 0	;SAVE LOCATION FOR R3
2006	003562	000000	R4SAVE::	.WORD 0	;SAVE LOCATION FOR R4
2007	003564	000	DAYS::	.BYTE 0	;NUMBER OF DAYS IN RUN
2008			.EVEN		

```

2010 ;/*/*/*/*/*/*/*/*/*/*/*/*/*/*/*/*/*/*/*/*/*/*/*/*/*/*/*/*/*
2011 ; 9.0 TMSCP CLASS AND PORT DRIVER DATA STRUCTURES
2012 ;/*/*/*/*/*/*/*/*/*/*/*/*/*/*/*/*/*/*/*/*/*/*/*/*/*/*/*/*
2013
2014 ; 9.1 CLASS DRIVER BUFFERS
2015 003566 CMDBF1:: .BLKW 2. ;DRIVE COMMAND BUFFER 1
2016 003572 DCMDBF:: .BLKW 18. ;DRIVER COMMAND BUFFER
2017 003636 CMDBF2:: .BLKW 20. ;DRIVE COMMAND BUFFER 2
2018 003706 CMDBF3:: .BLKW 20. ;DRIVE COMMAND BUFFER 3
2019 003756 CMDBF4:: .BLKW 20. ;DRIVE COMMAND BUFFER 4
2020
2021 004026 RSPBF0:: .BLKW 2. ;TOP 4 LOCATIONS OF RESPONSE BUFFER UNIT 0
2022 004032 DCBEND:: ;END OF COMMAND BUFFER
2023 004032 DRSPB0:: .BLKW 270. ;DRIVER RESPONSE BUFFER UNIT 0
2024
2025 005066 RSPBF1:: .BLKW 2. ;TOP 4 LOCATIONS OF RESPONSE BUFFER UNIT 1
2026 005072 DRBEN0:: ;END OF RESPONSE BUFFER UNIT 0
2027 005072 DRSPB1:: .BLKW 270. ;DRIVER RESPONSE BUFFER UNIT 1
2028
2029 006126 RSPBF2:: .BLKW 2. ;TOP 4 LOCATIONS OF RESPONSE BUFFER UNIT 2
2030 006132 DRBEN1:: ;END OF RESPONSE BUFFER UNIT 1
2031 006132 DRSPB2:: .BLKW 270. ;DRIVER RESPONSE BUFFER UNIT 2
2032
2033 007166 RSPBF3:: .BLKW 2. ;TOP 4 LOCATIONS OF RESPONSE BUFFER UNIT 3
2034 007172 DRBEN2:: ;END OF RESPONSE BUFFER UNIT 2
2035 007172 DRSPB3:: .BLKW 270. ;DRIVER RESPONSE BUFFER UNIT 3
2036

```

```

2038 ; 9.2 U/Q PORT DESCRIPTOR RINGS
2039 010226 DSRNG0:: .BLKW 2. ;DESCRIPTOR RING UNIT 0
2040 010232 DRBEN3:: ;END OF RESPONSE BUFFER UNIT 3
2041 010232 RDSRG0:: .BLKW 16. ;RESPONSE DESCRIPTOR RING UNIT 0
2042 010272 RDRENO:: ;END OF RESPONCE DESCRIPTOR RING UNIT 0
2043 010272 CDSRG0:: .BLKW 8. ;COMMAND DESCRIPTOR RING UNIT 0
2044 010312 CDRENO:: ;END OF COMMAND DESCRIPTOR RING UNIT 0
2045
2046 010312 DSRNG1:: .BLKW 2. ;DESCRIPTOR RING UNIT 1
2047 010316 RDSRG1:: .BLKW 16. ;RESPONSE DESCRIPTOR RING UNIT 1
2048 010356 RDREN1:: ;END OF RESPONCE DESCRIPTOR RING UNIT 1
2049 010356 CDSRG1:: .BLKW 8. ;COMMAND DESCRIPTOR RING UNIT 1
2050 010376 CDREN1:: ;END OF COMMAND DESCRIPTOR RING UNIT 1
2051
2052 010376 DSRNG2:: .BLKW 2. ;DESCRIPTOR RING UNIT 2
2053 010402 RDSRG2:: .BLKW 16. ;RESPONSE DESCRIPTOR RING UNIT 2
2054 010442 RDREN2:: ;END OF RESPONCE DESCRIPTOR RING UNIT 2
2055 010442 CDSRG2:: .BLKW 8. ;COMMAND DESCRIPTOR RING UNIT 2
2056 010462 CDREN2:: ;END OF COMMAND DESCRIPTOR RING UNIT 2
2057
2058 010462 DSRNG3:: .BLKW 2. ;DESCRIPTOR RING UNIT 3
2059 010466 RDSRG3:: .BLKW 16. ;RESPONSE DESCRIPTOR RING UNIT 3
2060 010526 RDREN3:: ;END OF RESPONCE DESCRIPTOR RING UNIT 3
2061 010526 CDSRG3:: .BLKW 8. ;COMMAND DESCRIPTOR RING UNIT 3
2062 010546 CDREN3:: ;END OF COMMAND DESCRIPTOR RING UNIT 3
2063
2064 ; 9.3 CLASS AND PORT DRIVER VARIABLES
2065 010546 000000 IOSTAT:: .WORD 0 ;I/O STATUS
2066 010550 177777 CMSTSV:: .WORD -1 ;COMMAND STATUS FROM GCS MODE
2067 010552 000000 GCSREF:: .WORD 0 ;GCS COMMAND REFERENCE NUMBER
2068 010554 000000 CNTHI:: .WORD 0 ;VALUE OF THE HIGH TIMEOUT
2069 010556 000000 TIMER:: .WORD 0 ;TIMER VALUE
2070 010560 000000 LOOPS:: .WORD 0 ;
2071 010562 000120 CNTFLG:: .WORD CF.THS!CF.MSC ;CONTROLLER FLAGS(ENABLE THIS HOSTS
;AND MISCELLANEOUS ERROR LOG MESSAGES)
2072
2073 010564 000000 PCKSIZ:: .WORD 0 ;PACKET SIZE IN BYTES
2074 010566 000000 SAERR:: .WORD 0 ;SA REGISTER SAVE ON ERROR
2075 010570 000 MINLIM:: .BYTE 0 ;MINIMUM REQUIRED CREDIT LIMIT
2076 010571 004 CRDLIM:: .BYTE 4 ;DRIVER CREDIT LIMIT
2077 .EVEN
2078

```

2080									
2081									
2082	010572	001	IOERTB::	.BYTE	DEVFAT				; GET COMMAND STATUS FAILED
2083	010573	102		.BYTE	OTHER				
2084	010574	000001		.WORD	1				
2085	010576	012213		.WORD	CMLSER				
2086	010600	013272		.WORD	DEVERR				
2087									
2088	010602	001		.BYTE	DEVFAT				; CONTROLLER HUNG
2089	010603	102		.BYTE	OTHER				
2090	010604	000002		.WORD	2				
2091	010606	012246		.WORD	HUNGER				
2092	010610	013272		.WORD	DEVERR				
2093									
2094	010612	001		.BYTE	DEVFAT				; PORT DETECTED ERROR
2095	010613	102		.BYTE	OTHER				
2096	010614	000003		.WORD	3				
2097	010616	012266		.WORD	PORTER				
2098	010620	013272		.WORD	DEVERR				
2099									
2100	010622	001		.BYTE	DEVFAT				; PROGRAM DETECTED COMMAND TIMEOUT
2101	010623	102		.BYTE	OTHER				
2102	010624	000004		.WORD	4				
2103	010626	012312		.WORD	TIMERR				
2104	010630	013272		.WORD	DEVERR				
2105									
2106	010632	001		.BYTE	DEVFAT				; COMMAND SEQUENCE ERROR
2107	010633	102		.BYTE	OTHER				
2108	010634	000005		.WORD	5				
2109	010636	012347		.WORD	SEQR				
2110	010640	013272		.WORD	DEVERR				
2111									
2112	010642	001		.BYTE	DEVFAT				; ERROR DETECTED DURING INIT
2113	010643	102		.BYTE	OTHER				
2114	010644	000006		.WORD	6				
2115	010646	012400		.WORD	INITER				
2116	010650	013272		.WORD	DEVERR				
2117									
2118									
2119									
2120	010652	001	CMDT::	.BYTE	DEVFAT				;INVALID COMMAND ISSUED
2121	010653	102		.BYTE	OTHER				
2122	010654	000007		.WORD	7.				
2123	010656	011502		.WORD	CMDR				
2124	010660	013272		.WORD	DEVERR				
2125									
2126	010662	001	ABOT::	.BYTE	DEVFAT				;COMMAND ABORTED
2127	010663	102		.BYTE	OTHER				
2128	010664	000010		.WORD	8.				
2129	010666	011525		.WORD	ABOER				
2130	010670	013272		.WORD	DEVERR				
2131									
2132	010672	001	OFLT::	.BYTE	DEVFAT				;UNIT OFFLINE
2133	010673	102		.BYTE	OTHER				
2134	010674	000011		.WORD	9.				
2135	010676	011541		.WORD	OFLER				
2136	010700	013272		.WORD	DEVERR				

2137					
2138	010702	001	AVLT::	.BYTE DEVFAT	;UNIT AVAILABLE ERROR
2139	010703	102		.BYTE OTHER	
2140	010704	000012		.WORD 10.	
2141	010706	011556		.WORD AVLER	
2142	010710	013272		.WORD DEVERR	
2143					
2144	010712	001	IVST1::	.BYTE DEVFAT	;INVALID STATUS RETURNED
2145	010713	102		.BYTE OTHER	
2146	010714	000013		.WORD 11.	
2147	010716	012135		.WORD IVSER	
2148	010720	013272		.WORD DEVERR	
2149					
2150	010722	001	WPRT::	.BYTE DEVFAT	;UNIT WRITE PROTECTED
2151	010723	102		.BYTE OTHER	
2152	010724	000014		.WORD 12.	
2153	010726	011603		.WORD WPRER	
2154	010730	013272		.WORD DEVERR	
2155					
2156	010732	002	CMPT::	.BYTE HARD	;DATA COMPARE ERROR
2157	010733	070		.BYTE DTCPER	
2158	010734	000015		.WORD 13.	
2159	010736	012464		.WORD CMPER	
2160	010740	013272		.WORD DEVERR	
2161					
2162	010742	002	DATT::	.BYTE HARD	;DATA ERROR
2163	010743	106		.BYTE NOERR	
2164	010744	000016		.WORD 14.	
2165	010746	011630		.WORD DATER	
2166	010750	013272		.WORD DEVERR	
2167					
2168	010752	001	HSTT::	.BYTE DEVFAT	;HOST DETECTED TIMEOUT
2169	010753	102		.BYTE OTHER	
2170	010754	000017		.WORD 15.	
2171	010756	012165		.WORD HSTER	
2172	010760	013272		.WORD DEVERR	
2173					
2174	010762	001	CNTT::	.BYTE DEVFAT	;CONTROLLER ERROR
2175	010763	102		.BYTE OTHER	
2176	010764	000020		.WORD 16.	
2177	010766	011674		.WORD CNTER	
2178	010770	013272		.WORD DEVERR	
2179					
2180	010772	001	DRVT::	.BYTE DEVFAT	;DRIVE ERROR
2181	010773	102		.BYTE OTHER	
2182	010774	000021		.WORD 17.	
2183	010776	011715		.WORD DRVER	
2184	011000	013272		.WORD DEVERR	
2185					
2186	011002	001	FMTT::	.BYTE DEVFAT	;FORMATTER ERROR
2187	011003	102		.BYTE OTHER	
2188	011004	000022		.WORD 18.	
2189	011006	011731		.WORD FMTER	
2190	011010	013272		.WORD DEVERR	
2191					
2192	011012	001	BOTT::	.BYTE DEVFAT	;UNEXPECTED BOT ENCOUNTERED
2193	011013	102		.BYTE OTHER	

2194	011014	000023		.WORD	19.	
2195	011016	011751		.WORD	BOTER	
2196	011020	013272		.WORD	DEVERR	
2197						
2198	011022	001	TMT::	.BYTE	DEVFAT	;UNEXPECTED TAPE MARK ENCOUNTERED
2199	011023	102		.BYTE	OTHER	
2200	011024	000024		.WORD	20.	
2201	011026	011771		.WORD	TMER	
2202	011030	013272		.WORD	DEVERR	
2203						
2204	011032	001	IVST2::	.BYTE	DEVFAT	;INVALID STATS RECEIVED
2205	011033	102		.BYTE	OTHER	
2206	011034	000025		.WORD	21.	
2207	011036	012135		.WORD	IVSER	
2208	011040	013272		.WORD	DEVERR	
2209						
2210	011042	001	RDTT::	.BYTE	DEVFAT	;DATA RECORD TRUNCATED
2211	011043	102		.BYTE	OTHER	
2212	011044	000026		.WORD	22.	
2213	011046	012017		.WORD	RDTER	
2214	011050	013272		.WORD	DEVERR	
2215						
2216	011052	001	POLT::	.BYTE	DEVFAT	;TAPE POSITION LOST
2217	011053	076		.BYTE	RMSPOS	
2218	011054	000027		.WORD	23.	
2219	011056	012045		.WORD	POLER	
2220	011060	013272		.WORD	DEVERR	
2221						
2222	011062	001	SEXT::	.BYTE	DEVFAT	;SERIOUS EXCEPTION
2223	011063	102		.BYTE	OTHER	
2224	011064	000030		.WORD	24.	
2225	011066	012063		.WORD	SEXER	
2226	011070	013272		.WORD	DEVERR	
2227						
2228	011072	001	LEDT::	.BYTE	DEVFAT	;LEOT ENCOUNTERED
2229	011073	102		.BYTE	OTHER	
2230	011074	000031		.WORD	25.	
2231	011076	012105		.WORD	LEDER	
2232	011100	013272		.WORD	DEVERR	
2233						
2234	011102	001	IVST3::	.BYTE	DEVFAT	;INVALID STATUS RETURNED
2235	011103	102		.BYTE	OTHER	
2236	011104	000032		.WORD	26.	
2237	011106	012135		.WORD	IVSER	
2238	011110	013272		.WORD	DEVERR	
2239						
2240	011112	002	DCMPT::	.BYTE	HARD	;DATA COMPARE ERROR
2241	011113	070		.BYTE	DTCPER	
2242	011114	000033		.WORD	27.	
2243	011116	012507		.WORD	DCMPER	
2244	011120	013272		.WORD	DEVERR	
2245						
2246	011122	002	RLST::	.BYTE	HARD	;RECORD LENGTH SHORT ERROR
2247	011123	102		.BYTE	OTHER	
2248	011124	000034		.WORD	28.	
2249	011126	012421		.WORD	RLSER	
2250	011130	013272		.WORD	DEVERR	

2251					
2252					;ERROR LOG ERROR TABLES
2253					
2254	011132				CNTLER:
2255	011132	002	.BYTE	HARD	;CONTROLLER (LAST FAIL) ERROR LOG
2256	011133	106	.BYTE	NOERR	
2257	011134	000035	.WORD	29.	
2258	011136	03172	.WORD	CNTLEL	
2259	011140	014502	.WORD	ERLGER	
2260					
2261	011142				BADERL:
2262	011142	002	.BYTE	HARD	;HOST MEMORY ACCSESS ERROR
2263	011143	106	.BYTE	NOERR	
2264	011144	000036	.WORD	30.	
2265	011146	013233	.WORD	BADEL	
2266	011150	014502	.WORD	ERLGER	
2267					
2268	011152				CMPEL:
2269	011152	002	.BYTE	HARD	;COMPARE ERROR
2270	011153	070	.BYTE	DTCPER	
2271	011154	000037	.WORD	31.	
2272	011156	013030	.WORD	CMPEL	
2273	011160	014502	.WORD	ERLGER	
2274					
2275	011162				DRVERL:
2276	011162	002	.BYTE	HARD	;DRIVE ERROR LOG
2277	011163	106	.BYTE	NOERR	
2278	011164	000040	.WORD	32.	
2279	011166	013125	.WORD	DRVEL	
2280	011170	014502	.WORD	ERLGER	
2281					
2282	011172				CNTERL:
2283	011172	002	.BYTE	HARD	;CONTROLLER ERROR
2284	011173	106	.BYTE	NOERR	
2285	011174	000041	.WORD	33.	
2286	011176	013145	.WORD	CNTEL	
2287	011200	014502	.WORD	ERLGER	
2288					
2289	011202				LGPERL:
2290	011202	002	.BYTE	HARD	;LONG GAP ENCOUNTERED
2291	011203	106	.BYTE	NOERR	
2292	011204	000042	.WORD	34.	
2293	011206	013057	.WORD	LGPEL	
2294	011210	014502	.WORD	ERLGER	
2295					
2296	011212				RDSTEL:
2297	011212	002	.BYTE	HARD	;READ DATA SYNC NOT FOUND
2298	011213	054	.BYTE	H1READ	
2299	011214	000043	.WORD	35.	
2300	011216	013103	.WORD	DSTEL	
2301	011220	014502	.WORD	ERLGER	
2302					
2303	011222				WDSTEL:
2304	011222	002	.BYTE	HARD	;WRITE DATA SYNC NOT FOUND
2305	011223	060	.BYTE	H1WRIT	
2306	011224	000043	.WORD	35.	
2307	011226	013103	.WORD	DS*EL	

2308	011230	014502	.WORD	ERLGER	
2309					
2310	011232				
2311	011232	003	.BYTE	SOFT	;CORRECTABLE WRITE DATA ERROR
2312	011233	044	.BYTE	S1WRIT	
2313	011234	000044	.WORD	36.	
2314	011236	012627	.WORD	COREL	
2315	011240	014502	.WORD	ERLGER	
2316					
2317	011242				
2318	011242	002	.BYTE	HARD	;UNRECOVERABLE WRITE DATA ERROR
2319	011243	060	.BYTE	H1WRIT	
2320	011244	000045	.WORD	37.	
2321	011246	012760	.WORD	UWEEL	
2322	011250	014502	.WORD	ERLGER	
2323					
2324	011252				
2325	011252	003	.BYTE	SOFT	;SOFT READ ERROR
2326	011253	040	.BYTE	S1READ	
2327	011254	000052	.WORD	42.	
2328	011256	012574	.WORD	RTYEL	
2329	011260	014502	.WORD	ERLGER	
2330					
2331	011262				
2332	011262	002	.BYTE	HARD	;UNRECOVERABLE READ DATA ERROR
2333	011263	054	.BYTE	H1READ	
2334	011264	000046	.WORD	38.	
2335	011266	012740	.WORD	UREEL	
2336	011270	014502	.WORD	ERLGER	
2337					
2338	011272				
2339	011272	003	.BYTE	SOFT	;CRC ERROR ON AN ECC BLOCK
2340	011273	064	.BYTE	RTYEC1	
2341	011274	000047	.WORD	39.	
2342	011276	013001	.WORD	ECBEL	
2343	011300	014502	.WORD	ERLGER	
2344					
2345	011302				
2346	011302	003	.BYTE	SOFT	;ECC CORRECTION ON A TAPE MARK
2347	011303	050	.BYTE	EC1COR	
2348	011304	000050	.WORD	40.	
2349	011306	012707	.WORD	ECTEL	
2350	011310	014502	.WORD	ERLGER	
2351					
2352	011312				
2353	011312	003	.BYTE	SOFT	;ECC CORRECTION ON A DATA BLOCK
2354	011313	050	.BYTE	EC1COR	
2355	011314	000051	.WORD	41.	
2356	011316	012663	.WORD	ECDEL	
2357	011320	014502	.WORD	ERLGER	
2358					

```

2360      .SBTTL GLOBAL TEXT SECTION
2361
2362      ; COMMAND PRIMITIVE ASCII
2363
2364 011322      CMDASC::
2365 011322      116      125      114      .ASCIZ ?NUL?      ;NULL
2366 011326      122      104      040      .ASCIZ ?RD?      ;READ
2367 011332      127      122      124      .ASCIZ ?WRT?      ;WRITE
2368 011336      103      115      120      .ASCIZ ?CMP?      ;COMPARE HOST DATA
2369 011342      101      103      103      .ASCIZ ?ACC?      ;ACCESS
2370 011346      123      120      103      SPCASC: .ASCIZ ?SPC?      ;SPACE RECORDS
2371 011352      123      113      120      .ASCIZ ?SKP?      ;SKIP TAPE MARKS
2372 011356      123      120      117      .ASCIZ ?SPO?      ;SPACE OBJECTS
2373 011362      127      124      115      .ASCIZ ?WTM?      ;WRITE TAPE MARK
2374 011366      105      122      123      .ASCIZ ?ERS?      ;ERASE
2375 011372      105      122      107      .ASCIZ ?ERG?      ;ERASE GAP
2376 011376      101      126      114      .ASCIZ ?AVL?      ;AVAILABLE
2377 011402      117      116      114      .ASCIZ ?ONL?      ;ONLINE
2378 011406      123      125      103      .ASCIZ ?SUC?      ;SET UNIT CHARACTERISTICS
2379 011412      122      105      127      .ASCIZ ?REW?      ;REWIND
2380 011416      111      116      124      .ASCIZ ?INT?      ;INITIALIZE
2381 011422      101      102      117      .ASCIZ ?ABO?      ;ABORT
2382 011426      107      103      123      .ASCIZ ?GCS?      ;GET COMMAND STATUS
2383 011432      107      125      123      .ASCIZ ?GUS?      ;GET UNIT STATUS
2384 011436      123      103      103      .ASCIZ ?SCC?      ;SET CONTROLLER CHARACTERISTICS
2385
2386
2387      ;ASCII FOR HEXADECIMAL PRINTOUTS
2388 011442      HEXTBL::
2389 011442      060      000      .ASCIZ ?0?      ;
2390 011444      061      000      .ASCIZ ?1?      ;
2391 011446      062      000      .ASCIZ ?2?      ;
2392 011450      063      000      .ASCIZ ?3?      ;
2393 011452      064      000      .ASCIZ ?4?      ;
2394 011454      065      000      .ASCIZ ?5?      ;
2395 011456      066      000      .ASCIZ ?6?      ;
2396 011460      067      000      .ASCIZ ?7?      ;
2397 011462      070      000      .ASCIZ ?8?      ;
2398 011464      071      000      .ASCIZ ?9?      ;
2399 011466      101      000      .ASCIZ ?A?      ;
2400 011470      102      000      .ASCIZ ?B?      ;
2401 011472      103      000      .ASCIZ ?C?      ;
2402 011474      104      000      .ASCIZ ?D?      ;
2403 011476      105      000      .ASCIZ ?E?      ;
2404 011500      106      000      .ASCIZ ?F?      ;
2405      .EVEN

```

```

2407
2408      ; FORMAT STATEMENTS USED IN PRINT CALLS
2409      ;
2410
2411 011502      111      116      126  CMDER: .ASCIZ /INVALID CMD ISSUED/
2412 011525      103      115      104  ABOER: .ASCIZ /CMD ABORTED/
2413 011541      125      116      111  OFLER: .ASCIZ /UNIT OFFLINE/
2414 011556      125      116      111  AVLER: .ASCIZ /UNIT AVAILABLE ERROR/
2415 011603      125      116      111  WPRER: .ASCIZ /UNIT WRITE PROTECTED/
2416 011630      104      101      124  DATER: .ASCIZ /DATA ERROR/
2417 011643      110      117      123  BADER: .ASCIZ /HOST BUFFER ACCESS ERROR/
2418 011674      103      117      116  CNTER: .ASCIZ /CONTROLLER ERROR/
2419 011715      104      122      111  DRVER: .ASCIZ /DRIVE ERROR/
2420 011731      106      117      122  FMTER: .ASCIZ /FORMATTER ERROR/
2421 011751      102      117      124  BOTER: .ASCIZ /BOT ENCOUNTERED/
2422 011771      124      101      120  TMER: .ASCIZ /TAPE MARK ENCOUNTERED/
2423 012017      104      101      124  RDTER: .ASCIZ /DATA RECORD TRUNCATED/
2424 012045      120      117      123  POLER: .ASCIZ /POSITION LOST/
2425 012063      123      105      122  SEXER: .ASCIZ /SERIOUS EXCEPTION/
2426 012105      114      117      107  LEDER: .ASCIZ /LOGICAL EOT ENCOUNTERED/
2427 012135      111      116      126  IVSER: .ASCIZ /INVALID STATUS RECEIVED/
2428 012165      110      117      123  HSTER: .ASCIZ /HOST DETECTED TIMEOUT/
2429 012213      116      117      040  CMLSER: .ASCIZ /NO RESPONSE TO GCS COMMAND/
2430 012246      103      117      116  HUNGER: .ASCIZ /CONTROLLER HUNG/
2431 012266      120      117      122  PORTER: .ASCIZ /PORT-DETECTED ERROR/
2432 012312      120      122      117  TIMERR: .ASCIZ /PROGRAM DETECTED CMD TIMEOUT/
2433 012347      122      105      123  SEQER: .ASCIZ /RESPONSE OUT OF SEQUENCE/
2434 012400      120      117      122  INITER: .ASCIZ /PORT INIT FAILED/
2435 012421      122      105      103  RLSER: .ASCIZ /RECORD LENGTH SHORT/
2436 012445      123      124      101  STATER: .ASCIZ /STATUS MESSAGE/
2437 012464      104      101      124  CMPER: .ASCIZ /DATA COMPARE ERROR/
2438 012507      123      057      127  DCMPE: .ASCIZ ?S/W DETECTED DATA COMPARE?
2439 012541      104      101      124  UDRNER: .ASCIZ /DATA UNDERRUN/
2440 012557      104      101      124  OVERUN: .ASCIZ /DATA OVERRUN/
2441 012574      122      105      124  RTYEL: .ASCIZ /RETRY RECOVERED READ ERROR/
2442 012627      122      105      124  COREL: .ASCIZ /RETRY RECOVERED WRITE ERROR/
2443 012663      105      103      103  ECDEL: .ASCIZ /ECC DATA CORRECTION/
2444 012707      105      103      103  ECTEL: .ASCIZ /ECC TAPE MARK CORRECTION/
2445 012740      110      101      122  UREEL: .ASCIZ /HARD READ ERROR/
2446 012760      110      101      122  UWEEL: .ASCIZ /HARD WRITE ERROR/
2447 013001      103      122      103  ECBEL: .ASCIZ /CRC ERROR ON ECC BLOCK/
2448 013030      104      101      124  CMPEL: .ASCIZ /DATA COMPARE ERROR LOG/
2449 013057      114      117      116  LGPEL: .ASCIZ /LONG GAP ENCOUNTERED/
2450 013103      104      101      124  DSTEL: .ASCIZ /DATA SYNC TIMEOUT/
2451 013125      104      122      111  DRVEL: .ASCIZ /DRIVE ERROR LOG/
2452 013145      103      117      116  CNTEL: .ASCIZ /CONTROLLER ERROR LOG/
2453 013172      103      117      116  CNTLEL: .ASCIZ /CONTROLLER (LAST FAIL) ERROR LOG/
2454 013233      110      117      123  BADEL: .ASCIZ /HOST MEMORY ERROR LOG/
2455
2456      .EVEN
2463
2464
    
```

```

2473 .SBTTL GLOBAL ERROR REPORT SECTION
2474
2475 ;**
2476 ; THE GLOBAL ERROR REPORT SECTION CONTAINS THE PRINTB AND PRINTX CALLS
2477 ; THAT ARE USED IN MORE THAN ONE TEST. IT ALSO INCLUDES THE ASCII MESSAGES
2478 ; THAT ARE USED BY THE PRINTB AND PRINTX CALLS..
2479 ;--
2480
2481 013262 ERRTBL ;GENERIC ERROR TABLE
    013262 L$ERRTBL::
    013262 000000 ERRTP:: .WORD 0
    013264 000000 ERRNBR:: .WORD 0
    013266 000000 ERRMSG:: .WORD 0
    013270 000000 ERRBLK:: .WORD 0
2482
2483 013272 BGNMSG DEVERR
    013272 DEVERR::
2484 013272 PUSH <R1,R5> ;SAVE R1 AND R5
    013272 010146 MOV R1,-(SP) ;;PUSH R1 ON STACK
    013274 010546 MOV R5,-(SP) ;;PUSH R5 ON STACK
2485 013276 013703 003560 MOV R3SAVE,R3 ;RESTORE R3
2486 013302 013704 003562 MOV R4SAVE,R4 ;RESTORE R4
2487 013306 116205 000000 MOVB CMD(R2),R5 ;GET THE COMMAND PRIMITIVE
2488 013312 042705 177407 BIC #177407,R5 ;CLEAR MODIFIERS
2489 013316 006205 ASR R5 ;THE PRIMITIVE
2490 013320 062705 011322 ADD #CMDASC,R5 ;PUT ADDRESS IN R5
2491 013324 016237 000000 003530 MOV CMD(R2),R8 ;GET THE PRIMITIVE AGAIN
2492 013332 042737 177770 003530 BIC #177770,R8 ;SAVE THE LAST 3 BITS
2493 013340 001014 BNE 1$ ;BRANCH IF NOT ZERO
2494 013342 PRINTB #ERR00,R5,TKUNIT(R4)
    013342 016446 000004 MOV TKUNIT(R4),-(SP)
    013346 010546 MOV R5,-(SP)
    013350 012746 015770 MOV #ERR00,-(SP)
    013354 012746 000003 MOV #3,-(SP)
    013360 010600 MOV SP,R0
    013362 104414 TRAP C$PNTB
    013364 062706 000010 ADD #10,SP
2495 013370 000513 BR 6$ ;GO PRINT THE REST OF THE MESSAGE
2496 013372 022737 000001 003530 1$: CMP #REVBIT,R8 ;IS IT A REVERSE ?
2497 013400 001014 BNE 2$ ;BRANCH IF NOT
2498 013402 PRINTB #ERR01,R5,TKUNIT(R4)
    013402 016446 000004 MOV TKUNIT(R4),-(SP)
    013406 010546 MOV R5,-(SP)
    013410 012746 016040 MOV #ERR01,-(SP)
    013414 012746 000003 MOV #3,-(SP)
    013420 010600 MOV SP,R0
    013422 104414 TRAP C$PNTB
    013424 062706 000010 ADD #10,SP
2499 013430 000473 BR 6$ ;GO PRINT THE REST OF THE MESSAGE
2500 013432 032737 000002 003530 2$: BIT #EOTBIT,R8 ;IS IT A DETECT LEOT ?
2501 013440 001414 BEQ 3$ ;BRANCH IF NOT
2502 013442 PRINTB #ERR02,R5,TKUNIT(R4)
    013442 016446 000004 MOV TKUNIT(R4),-(SP)
    013446 010546 MOV R5,-(SP)
    013450 012746 016116 MOV #ERR02,-(SP)
    013454 012746 000003 MOV #3,(SP)
    013460 010600 MOV SP,R0
    
```

	013462	104414			TRAP	C\$PNTB	
	013464	062706	000010		ADD	#10,SP	
2503	013470	000453			BR	6\$	
2504	013472	022737	000003	003530	3\$:	CMP	#IMMBIT,R8 ;GO PRINT THE REST OF THE MESSAGE
2505	013500	001014			BNE	4\$;IS IT A IMMEDIATE ?
2506	013502				PRINTB	#ERR03,R5,TKUNIT(R4)	;BRANCH IF NOT
	013502	016446	000004		MOV	TKUNIT(R4),-(SP)	
	013506	010546			MOV	R5,(SP)	
	013510	012746	016175		MOV	#ERR03,-(SP)	
	013514	012746	000003		MOV	#3,-(SP)	
	013520	010600			MOV	SP,RO	
	013522	104414			TRAP	C\$PNTB	
	013524	062706	000010		ADD	#10,SP	
2507	013530	000433			BR	6\$	
2508	013532	022737	000004	003530	4\$:	CMP	#UNLBIT,R8 ;GO PRINT THE REST OF THE MESSAGE
2509	013540	001014			BNE	5\$;IS IT A UNLOAD ?
2510	013542				PRINTB	#ERR04,R5,TKUNIT(R4)	;BRANCH IF NOT
	013542	016446	000004		MOV	TKUNIT(R4),-(SP)	
	013546	010546			MOV	R5,-(SP)	
	013550	012746	016253		MOV	#ERR04,-(SP)	
	013554	012746	000003		MOV	#3,-(SP)	
	013560	010600			MOV	SP,RO	
	013562	104414			TRAP	C\$PNTB	
	013564	062706	000010		ADD	#10,SP	
2511	013570	000413			BR	6\$	
2512	013572				PRINTB	#ERR05,R5,TKUNIT(R4)	;GO PRINT THE REST OF THE MESSAGE
	013572	016446	000004		MOV	TKUNIT(R4),-(SP)	
	013576	010546			MOV	R5,-(SP)	
	013600	012746	016332		MOV	#ERR05,-(SP)	
	013604	012746	000003		MOV	#3,-(SP)	
	013610	010600			MOV	SP,RO	
	013612	104414			TRAP	C\$PNTB	
	013614	062706	000010		ADD	#10,SP	
2513	013620				PRINTB	#ERR06,PASCNT,PATSAV(R4)	
	013620	016446	000024		MOV	PATSAV(R4),-(SP)	
	013624	013746	003522		MOV	PASCNT,-(SP)	
	013630	012746	016411		MOV	#ERR06,-(SP)	
	013634	012746	000003		MOV	#3,-(SP)	
	013640	010600			MOV	SP,RO	
	013642	104414			TRAP	C\$PNTB	
	013644	062706	000010		ADD	#10,SP	
2514	013650	022705	011346		CMP	#SPCASC,R5 ;IS IT A DATA TRANSFER ERROR ?	
2515	013654	101412			BLOS	7\$;NO, DON'T PRINT THE BYTE COUNT	
2516	013656				PRINTB	#ERR07,BYTES ;PRINT THE BYTE COUNT	
	013656	013746	003416		MOV	BYTES,(SP)	
	013662	012746	016461		MOV	#ERR07,-(SP)	
	013666	012746	000002		MOV	#2,-(SP)	
	013672	010600			MOV	SP,RO	
	013674	104414			TRAP	C\$PNTB	
	013676	062706	000006		ADD	#6,SP	
2517	013702				PRINTB	#ERR08,OBOFFH(R2),OBOFFL(R2)	
	013702	016246	000004		MOV	OBOFFL(R2),(SP)	
	013706	016246	000006		MOV	OBOFFH(R2),-(SP)	
	013712	012746	016521		MOV	#ERR08,(SP)	
	013716	012746	000003		MOV	#3,-(SP)	
	013722	010600			MOV	SP,RO	
	013724	104414			TRAP	C\$PNTB	

2518	013726	062706	000010			ADD	#10,SP	
2519	013732	032737	000010	003516		BIT	#DRERFL,PCFLAG	; WAS IT A DRIVE ERROR ?
2520	013740	001417				BEQ	8\$; NO, GO PRINT TIME
2521	013742	042737	000010	003516		BIC	#DRERFL,PCFLAG	; CLEAR THE DRIVE ERROR FLAG
	013750					PRINTB	#ERR09,HIHEX,LOHEX	; PRINT THE DRIVE ERROR CODE
	013754	013746	003552			MOV	LOHEX,-(SP)	
	013760	012746	003554			MOV	HIHEX,-(SP)	
	013764	012746	016556			MOV	#ERR09,-(SP)	
	013770	012746	000003			MOV	#3,-(SP)	
	013772	010600				MOV	SP,R0	
	013774	104414				TRAP	C\$PNTB	
2522	013774	062706	000010			ADD	#10,SP	
2523	014000	122737	000006	010546	8\$:	CMPB	#INTERR,IOSTAT	; IS IT A PORT INIT FAILURE ?
2524	014006	001001				BNE	9\$; KEEP GOING IF IT ISN'T
2525	014010	000404				BR	10\$; GO PRINT SA CONTENTS
2526	014012	122737	000003	010546	9\$:	CMPB	#IOPDRE,IOSTAT	; IS IT A PORT DETECTED FAILURE ?
2527	014020	001014				BNE	11\$; KEEP GOING IF IT ISN'T
	014022				10\$:	PRINTB	#ERR10,SAERR	; PRINT THE SA CONTENTS IF IT IS
	014022	013746	010566			MOV	SAERR,-(SP)	
	014026	012746	016617			MOV	#ERR10,-(SP)	
	014032	012746	000002			MOV	#2,-(SP)	
	014036	010600				MOV	SP,R0	
	014040	104414				TRAP	C\$PNTB	
	014042	062706	000006			ADD	#6,SP	
2528	014046	005037	010566			CLR	SAERR	; CLEAR THE ERROR OUT OF THE LOCATION
2529	014052	105737	010546		11\$:	TSTB	IOSTAT	; WAS IT AN I/O ERROR ?
2530	014056	001150				BNE	DEVEXT	; GET OUT IF IT WAS
2531	014060	005737	003444			TST	CMPIERR	; WAS IT A COMPARE ERROR ?
2532	014064	001051				BNE	CMPPRI	; GO PRINT THE ERROR DATA
2533	014066					PRINTX	#ERR11	
	014066	012746	016651			MOV	#ERR11,-(SP)	
	014072	012746	000001			MOV	#1,(SP)	
	014076	010600				MOV	SP,R0	
	014100	104415				TRAP	C\$PNTX	
2534	014102	062706	000004			ADD	#4,SP	
	014106					PRINTX	#ERR12	
	014106	012746	016675			MOV	#ERR12,(SP)	
	014112	012746	000001			MOV	#1,-(SP)	
	014116	010600				MOV	SP,R0	
	014120	104415				TRAP	C\$PNTX	
	014122	062706	000004			ADD	#4,SP	
2535	014126	010305				MOV	R3,R5	; GET POINTER TO RESPONSE PACKET
2536	014130	010301				MOV	R3,R1	; AND A SECOND COPY
2537	014132	062701	000002			ADD	#2,R1	; R1 POINT TO SECOND WORD OF PACKET
2538	014136	005763	177774		PRIPCK:	TST	MSGLEN(R3)	; CHECK THE MESSAGE LENGTH
2539	014142	100422				BMI	CMPPRI	; GET OUT IF IT WENT NEGATIVE
2540	014144					PRINTX	#ERR13,(R1),(R5)	
	014144	011546				MOV	(R5),-(SP)	
	014146	011146				MOV	(R1),-(SP)	
	014150	012746	016733			MOV	#ERR13,-(SP)	
	014154	012746	000003			MOV	#3,-(SP)	
	014160	010600				MOV	SP,R0	
	014162	104415				TRAP	C\$PNTX	
	014164	062706	000010			ADD	#10,SP	
2541	014170	062701	000004			ADD	#4,R1	; GET THE NEXT WORD
2542	014174	062705	000004			ADD	#4,R5	; AND AGAIN
2543	014200	162763	000004	177774		SUB	#4,MSGLEN(R3)	; ADJUST MESSAGE LENGTH DOWN 2 WORDS

```

2544 014206 001353
2545 014210 005737 003444
2546 014214 001471
2547 014216
2548 014220 012701 003446
2549 014224 012702 003472
2550 014230 013705 003444
2551 014234
    014234 012746 016764
    014240 012746 000001
    014244 010600
    014246 104415
    014250 062706 000004
2552 014254
    014254 012746 017012
    014260 012746 000001
    014264 010600
    014266 104415
    014270 062706 000004
2553 014274
    014274 005046
    014276 151216
    014300 005046
    014302 156216 000001
    014306 011146
    014310 012746 017062
    014314 012746 000004
    014320 010600
    014322 104415
    014324 062706 000012
2554 014330 005337 003444
2555 014334 001405
2556 014336 005721
2557 014340 005722
2558 014342 022701 003472
2559 014346 001352
2560 014350
    014350 010546
    014352 012746 017107
    014356 012746 000002
    014362 010600
    014364 104415
    014366 062706 000006
2561 014372 005037 003444
2562 014376
2563 014400
    014400 012746 020527
    014404 012746 000001
    014410 010600
    014412 104417
    014414 062706 000004
2564 014420 105737 002212
2565 014424 001421
2566 014426
    014426 005046
    014430 153716 002215
    014434 005046

CMPPRI: BNE PRIPCK ;KEEP PRINTING TILL ALL DONE
        TST CMPERR ;WAS THIS A COMPARE ERROR ?
        BEQ DEVEXT ;GET OUT IF IT WASN'T
        PUSH <R2> ;SAVE R2
        MOV #BYTADD,R1 ;POINT R1 TO THE BYTE ADDRESS TABLE
        MOV #DATBL,R2 ;POINT R2 TO THE WRITE DATA TABLE
        MOV CMPERR,R5 ;LET R5 = THE NUMBER OF BYTES IN ERROR
        PRINTX #ERR14
        MOV #ERR14,(SP)
        MOV #1,-(SP)
        MOV SP,R0
        TRAP C$PNTX
        ADD #4,SP
        PRINTX #ERR15
        MOV #ERR15,-(SP)
        MOV #1,-(SP)
        MOV SP,R0
        TRAP C$PNTX
        ADD #4,SP
1$: PRINTX #ERR16,(R1),<B,ONE(R2)>,<B,(R2)>
        CLR -(SP)
        BISB (R2),(SP)
        CLR -(SP)
        BISB ONE(R2),(SP)
        MOV (R1),-(SP)
        MOV #ERR16,-(SP)
        MOV #4,-(SP)
        MOV SP,R0
        TRAP C$PNTX
        ADD #12,SP
        DEC CMPERR
        BEQ CPRIEX ;SUBTRACT 1 FROM NUMBER OF ERRORS
        TST (R1)+ ;GO PRINT TOTAL NUMBER IN ERROR
        TST (R2)+ ;POINT R1 TO THE NEXT ADDRESS
        CMP #TBLEND,R1 ;POINT R2 TO THE NEXT DATA
        BNE 1$ ;HAVE WE PRINTED THE WHOLE TABLE ?
        PRINTX #ERR17,R5 ;NO CONTINUE
        MOV R5,-(SP)
        MOV #ERR17,-(SP)
        MOV #2,-(SP)
        MOV SP,R0
        TRAP C$PNTX
        ADD #6,SP
        CLR CMPERR ;CLEAR THE ERROR COUNTER
        POP <R2>
DEVEXT: PRINTF #LINE
        MOV #LINE,-(SP)
        MOV #1,-(SP)
        MOV SP,R0
        TRAP C$PNTF
        ADD #4,SP
        TSTB CLOCK ;IS THE CLOCK ENABLED
        BEQ 1$ ;NO, THEN CAN'T PRINT TIME
        PRINTF #TIME,<B,HOURS>,<B,MINUTE>,<B,SECOND>
        CLR -(SP)
        BISB SECOND,(SP)
        CLR -(SP)

```

	014436	153716	002214		BISB	MINUTE,(SP)
	014442	005046			CLR	-(SP)
	014444	153716	002213		BISB	HOURS,(SP)
	014450	012746	020044		MOV	#TIME,-(SP)
	014454	012746	000004		MOV	#4,-(SP)
	014460	010600			MOV	SP,RO
	014462	104417			TRAP	C\$PNTF
	014464	062706	000012		ADD	#12,SP
2567	014470			1\$:	POP	<R5,R1>
2568	014474				EXIT	MSG
	014474	000167			.WORD	J\$.JMP
	014476	000000			.WORD	L10002-2-
2569	014500				ENDMSG	
	014500			L10002:		
	014500	104423			TRAP	C\$MSG

2571	014502					BGNMSG	ERLGER	
	014502					ERLGER::		
2572	014502					PUSH	<R1,R5>	;SAVE R1 AND R5
	014502	010146				MOV	R1,-(SP)	::PUSH R1 ON STACK
	014504	010546				MOV	R5,-(SP)	::PUSH R5 ON STACK
2573	014506	013703	003560			MOV	R3SAVE,R3	;RESTORE R3
2574	014512	013704	003562			MOV	R4SAVE,R4	;RESTORE R4
2575	014516	113705	003536			MOVB	R11,R5	;GET THE COMMAND PRIMITIVE
2576	014522	042705	177407			BIC	#177407,R5	;CLEAR MODIFIERS
2577	014526	006205				ASR	R5	;THE PRIMITIVE
2578	014530	062705	011322			ADD	#CMDASC,R5	;PUT ADDRESS IN R5
2579	014534	013737	003536	003530		MOV	R11,R8	;GET THE PRIMITIVE AGAIN
2580	014542	042737	177770	003530		BIC	#177770,R8	;SAVE THE LAST 3 BITS
2581	014550	001014				BNE	5\$;BRANCH IF NOT ZERO
2582	014552					PRINTB	#ERR00,R5,TKUNIT(R4)	
	014552	016446	000004			MOV	TKUNIT(R4),-(SP)	
	014556	010546				MOV	R5,-(SP)	
	014560	012746	015770			MOV	#ERR00,-(SP)	
	014564	012746	000003			MOV	#3,-(SP)	
	014570	010600				MOV	SP,R0	
	014572	104414				TRAP	C\$PNTB	
	014574	062706	000010			ADD	#10,SP	
2583	014600	000512				BR	30\$;GO PRINT THE REST OF THE MESSAGE
2584	014602	022737	000001	003530	5\$:	CMP	#REVBIT,R8	;IS IT A REVERSE ?
2585	014610	001013				BNE	10\$;BRANCH IF NOT
2586	014612					PRINTB	#ERR01,R5,TKUNIT(R4)	
	014612	016446	000004			MOV	TKUNIT(R4),-(SP)	
	014616	010546				MOV	R5,-(SP)	
	014620	012746	016040			MOV	#ERR01,-(SP)	
	014624	012746	000003			MOV	#3,-(SP)	
	014630	010600				MOV	SP,R0	
	014632	104414				TRAP	C\$PNTB	
	014634	062706	000010			ADD	#10,SP	
2587	014640	032737	000002	003530	10\$:	BIT	#EOTBIT,R8	;IS IT A DETECT LEOT ?
2588	014646	001414				BEQ	15\$;BRANCH IF NOT
2589	014650					PRINTB	#ERR02,R5,TKUNIT(R4)	
	014650	016446	000004			MOV	TKUNIT(R4),-(SP)	
	014654	010546				MOV	R5,-(SP)	
	014656	012746	016116			MOV	#ERR02,-(SP)	
	014662	012746	000003			MOV	#3,-(SP)	
	014666	010600				MOV	SP,R0	
	014670	104414				TRAP	C\$PNTB	
	014672	062706	000010			ADD	#10,SP	
2590	014676	000453				BR	30\$;GO PRINT THE REST OF THE MESSAGE
2591	014700	022737	000003	003530	15\$:	CMP	#IMMBIT,R8	;IS IT A IMMEDIATE ?
2592	014706	001014				BNE	20\$;BRANCH IF NOT
2593	014710					PRINTB	#ERR03,R5,TKUNIT(R4)	
	014710	016446	000004			MOV	TKUNIT(R4),-(SP)	
	014714	010546				MOV	R5,-(SP)	
	014716	012746	016175			MOV	#ERR03,-(SP)	
	014722	012746	000003			MOV	#3,-(SP)	
	014726	010600				MOV	SP,R0	
	014730	104414				TRAP	C\$PNTB	
	014732	062706	000010			ADD	#10,SP	
2594	014736	000433				BR	30\$;GO PRINT THE REST OF THE MESSAGE
2595	014740	022737	000004	003530	20\$:	CMP	#UNLBIT,R8	;IS IT A UNLOAD ?
2596	014746	001014				BNE	25\$;BRANCH IF NOT

2597	014750			PRINTB	#ERR04,R5,TKUNIT(R4)		
	014750	016446	000004	MOV	TKUNIT(R4),-(SP)		
	014754	010546		MOV	R5,-(SP)		
	014756	012746	016253	MOV	#ERR04,-(SP)		
	014762	012746	000003	MOV	#3,-(SP)		
	014766	010600		MOV	SP,R0		
	014770	104414		TRAP	C\$PNTB		
	014772	062706	000010	ADD	#10,SP		
2598	014776	000413		BR	30\$;GO PRINT THE REST OF THE MESSAGE	
2599	015000			25\$: PRINTB	#ERR05,R5,TKUNIT(R4)		
	015000	016446	000004	MOV	TKUNIT(R4),-(SP)		
	015004	010546		MOV	R5,-(SP)		
	015006	012746	016332	MOV	#ERR05,-(SP)		
	015012	012746	000003	MOV	#3,-(SP)		
	015016	010600		MOV	SP,R0		
	015020	104414		TRAP	C\$PNTB		
	015022	062706	000010	ADD	#10,SP		
2600	015026			30\$: PRINTB	#ERR06,PASCNT,PATSAV(R4)		
	015026	016446	000024	MOV	PATSAV(R4),-(SP)		
	015032	013746	003522	MOV	PASCNT,-(SP)		
	015036	012746	016411	MOV	#ERR06,(SP)		
	015042	012746	000003	MOV	#3,-(SP)		
	015046	010600		MOV	SP,R0		
	015050	104414		TRAP	C\$PNTB		
	015052	062706	000010	ADD	#10,SP		
2601	015056	122763	000005	000010	CMPB	#FM.TPE,L.FMT(R3)	;IS IT A TAPE TRANSFER ERROR LOG ?
2602	015064	001402		BEQ	35\$;YES, GO PRINT IT	
2603	015066	000137	015476	JMP	PKPRNT	;NO, PRINT THE ERROR LOG PACKET	
2604	015072			35\$: PRINTB	#ERR07,BYTES	;PRINT THE BYTE COUNT	
	015072	013746	003416	MOV	BYTES,-(SP)		
	015076	012746	016461	MOV	#ERR07,-(SP)		
	015102	012746	000002	MOV	#2,-(SP)		
	015106	010600		MOV	SP,R0		
	015110	104414		TRAP	C\$PNTB		
	015112	062706	000006	ADD	#6,SP		
2605	015116			PRINTB	#ERR08,OBOFFH(R2),OBOFFL(R2)		
	015116	016246	000004	MOV	OBOFFL(R2),-(SP)		
	015122	016246	000006	MOV	OBOFFH(R2),-(SP)		
	015126	012746	016521	MOV	#ERR08,-(SP)		
	015132	012746	000003	MOV	#3,-(SP)		
	015136	010600		MOV	SP,R0		
	015140	104414		TRAP	C\$PNTB		
	015142	062706	000010	ADD	#10,SP		
2606	015146			PRINTX	#ERL00,L.PSTN+2(R3),L.PSTN(R3)		
	015146	016346	000044	MOV	L.PSTN(R3),-(SP)		
	015152	016346	000046	MOV	L.PSTN+2(R3),-(SP)		
	015156	012746	017146	MOV	#ERL00,-(SP)		
	015162	012746	000003	MOV	#3,-(SP)		
	015166	010600		MOV	SP,R0		
	015170	104415		TRAP	C\$PNTX		
	015172	062706	000010	ADD	#10,SP		
2607	015176			PRINTX	#ERL01,<B,L.TRK(R3)>,<B,L.LVL(R3)>,<B,L.RTRY(R3)>		
	015176	005046		CLR	-(SP)		
	015200	156316	000043	BISB	L.RTRY(R3),(SP)		
	015204	005046		CLR	-(SP)		
	015206	156316	000042	BISB	L.LVL(R3),(SP)		
	015212	005046		CLR	-(SP)		

	015214	156316	000055	BISB	L. TRK(R3), (SP)
	015220	012746	017203	MOV	#ERL01, -(SP)
	015224	012746	000004	MOV	#4, -(SP)
	015230	010600		MOV	SP, R0
	015232	104415		TRAP	C\$PNTX
	015234	062706	000012	ADD	#12, SP
2608	015240			PRINTX	#ERL02, <B, L. LBLK(R3)>, L. PBLK(R3)
	015240	016346	000056	MOV	L. PBLK(R3), -(SP)
	015244	005046		CLR	-(SP)
	015246	156316	000060	BISB	L. LBLK(R3), (SP)
	015252	012746	017301	MOV	#ERL02, -(SP)
	015256	012746	000003	MOV	#3, -(SP)
	015262	010600		MOV	SP, R0
	015264	104415		TRAP	C\$PNTX
	015266	062706	000010	ADD	#10, SP
2609	015272			PRINTX	#ERL03, <B, L. DRVC(R3)>, <B, L. DFLG(R3)>
	015272	005046		CLR	-(SP)
	015274	156316	000054	BISB	L. DFLG(R3), (SP)
	015300	005046		CLR	-(SP)
	015302	156316	000053	BISB	L. DRVC(R3), (SP)
	015306	012746	017365	MOV	#ERL03, -(SP)
	015312	012746	000003	MOV	#3, -(SP)
	015316	010600		MOV	SP, R0
	015320	104415		TRAP	C\$PNTX
	015322	062706	000010	ADD	#10, SP
2610	015326			PRINTX	#ERL04, L. DRVS(R3), <B, L. STS(R3)>
	015326	005046		CLR	-(SP)
	015330	156316	000052	BISB	L. STS(R3), (SP)
	015334	016346	000064	MOV	L. DRVS(R3), -(SP)
	015340	012746	017454	MOV	#ERL04, -(SP)
	015344	012746	000003	MOV	#3, -(SP)
	015350	010600		MOV	SP, R0
	015352	104415		TRAP	C\$PNTX
	015354	062706	000010	ADD	#10, SP
2611	015360			PRINTX	#ERL05, <B, L. CNT0(R3)>, <B, L. CNT1(R3)>
	015360	005046		CLR	-(SP)
	015362	156316	000062	BISB	L. CNT1(R3), (SP)
	015366	005046		CLR	-(SP)
	015370	156316	000061	BISB	L. CNT0(R3), (SP)
	015374	012746	017540	MOV	#ERL05, -(SP)
	015400	012746	000003	MOV	#3, -(SP)
	015404	010600		MOV	SP, R0
	015406	104415		TRAP	C\$PNTX
	015410	062706	000010	ADD	#10, SP
2612	015414			PRINTX	#ERL06, <B, L. CNT2(R3)>, L. RWST(R3)
	015414	016346	000066	MOV	L. RWST(R3), -(SP)
	015420	005046		CLR	-(SP)
	015422	156316	000063	BISB	L. CNT2(R3), (SP)
	015426	012746	017627	MOV	#ERL06, -(SP)
	015432	012746	000003	MOV	#3, -(SP)
	015436	010600		MOV	SP, R0
	015440	104415		TRAP	C\$PNTX
	015442	062706	000010	ADD	#10, SP
2613	015446			PRINTX	#ERL07, L. OPFL(R3)
	015446	016346	000070	MOV	L. OPFL(R3), -(SP)
	015452	012746	017713	MOV	#ERL07, (SP)
	015456	012746	000002	MOV	#2, -(SP)

015462	010600		MOV	SP,R0		
015464	104415		TRAP	C\$PNTX		
015466	062706	000006	ADD	#6,SP		
2614 015472	000137	015670	JMP	MSGEXT	;GET OUT	
2615 015476			PKPRNT: PRINTB	#ERR08,0BOFFH(R2),0BOFFL(R2)		
015476	016246	000004	MOV	0BOFFL(R2),-(SP)		
015502	016246	000006	MOV	0BOFFH(R2),-(SP)		
015506	012746	016521	MOV	#ERR08,-(SP)		
015512	012746	000003	MOV	#3,-(SP)		
015516	010600		MOV	SP,R0		
015520	104414		TRAP	C\$PNTB		
015522	062706	000010	ADD	#10,SP		
2616 015526			PRINTX	#ERL08		
015526	012746	017743	MOV	#ERL08,-(SP)		
015532	012746	000001	MOV	#1,-(SP)		
015536	010600		MOV	SP,R0		
015540	104415		TRAP	C\$PNTX		
015542	062706	000004	ADD	#4,SP		
2617 015546			PRINTX	#ERR11		
015546	012746	016651	MOV	#ERR11,-(SP)		
015552	012746	000001	MOV	#1,-(SP)		
015556	010600		MOV	SP,R0		
015560	104415		TRAP	C\$PNTX		
015562	062706	000004	ADD	#4,SP		
2618 015566			PRINTX	#ERR12		
015566	012746	016675	MOV	#ERR12,-(SP)		
015572	012746	000001	MOV	#1,-(SP)		
015576	010600		MOV	SP,R0		
015600	104415		TRAP	C\$PNTX		
015602	062706	000004	ADD	#4,SP		
2619 015606	010305		MOV	R3,R5	;GET POINTER TO RESPONSE PACKET	
2620 015610	010301		MOV	R3,R1	;AND A SECOND COPY	
2621 015612	062701	000002	ADD	#2,R1	;R1 POINT TO SECOND WORD OF PACKET	
2622 015616	005763	177774	1\$: TST	MSGLEN(R3)	;ARE WE STILL POSITIVE :	
2623 015622	100422		BMI	MSGEXT	;NO, GET OUT	
2624 015624			PRINTX	#ERR13,(R1),(R5)		
015624	011546		MOV	(R5),-(SP)		
015626	011146		MOV	(R1),-(SP)		
015630	012746	016733	MOV	#ERR13,-(SP)		
015634	012746	000003	MOV	#3,-(SP)		
015640	010600		MOV	SP,R0		
015642	104415		TRAP	C\$PNTX		
015644	062706	000010	ADD	#10,SP		
2625 015650	062701	000004	ADD	#4,R1	;GET THE NEXT WORD	
2626 015654	062705	000004	ADD	#4,R5	;AND AGAIN	
2627 015660	162763	000004	2627 015660	SUB	#4,MSGLEN(R3)	;ADJUST MESSAGE LENGTH DOWN 2 WORDS
2628 015666	001353		BNE	1\$;KEEP PRINTING TILL ALL DONE	
2629 015670			MSGEXT: PRINTF	#LINE		
015670	012746	020527	MOV	#LINE,-(SP)		
015674	012746	000001	MOV	#1,-(SP)		
015700	010600		MOV	SP,R0		
015702	104417		TRAP	C\$PNTF		
015704	062706	000004	ADD	#4,SP		
2630 015710	105737	002212	TSTB	CLOCK	;IS THE CLOCK ENABLED	
2631 015714	001421		BEQ	1\$;NO, THEN CAN T PRINT TIME	
2632 015716			PRINTF	#TIME,<B,HOURS>,<B,MINUTE>,<B,SECOND>		
015716	005046		CLR	-(SP)		

2678 021001 045 116 045 BYPASS:: .ASCIZ /#N#A TEST #Z3#A BYPASSED#N/
2679 .EVEN
2680 021034 ENDMSG
021034 L10003:
021034 104423 TRAP C#MSG

2682
2683
2684 021036
021036
2685 021036 000000
2686 021040 177777
2687 021042 177777
2688 021044
2689

;PROTECTION TABLE

BGNPROT
L\$PROT::
.WORD 0
.WORD -1
.WORD -1
ENDPROT

```

2691          .SBTTL  CLOCK HANDLER
2692
2693 021044    BGNSRV  NOCLK
      021044    NOCLK::
2694
2695 021044    105037  002212          CLRB   CLOCK          ;CLEAR THE CLOCK ENABLED BIT
2696 021050    052737  000002  003516    BIS    #NCLKFL,PCFLAG ;SET UP NO CLOCK PRESENT FLAG
2697 021056    012746  017770          PRINTF #NCLK           ;PRINT MESSAGE
      021056    012746  000001          MOV    #NCLK,-(SP)
      021062    012746  000001          MOV    #1,-(SP)
      021066    010600          MOV    SP,R0
      021070    104417          TRAP  C$PNTF
      021072    062706  000004          ADD   #4,SP
2698
2699 021076    ENDSRV
      021076    L10005:
      021076    000002          RTI
2700
2701 021100    BGNSRV  KWHDL
      C?1100    KWHDL::
2702
2703 021100    105237  002216          INCB  SUBSEC          ;INCREMENT THE SUB-SECOND COUNTER
2704 021104    122737  000074  002216    CMPB  #60.,SUBSEC    ;IS IT A SECOND YET ?
2705 021112    001051          BNE   HDLEXT         ;NO, GET OUT
2706 021114    105037  002216          CLRB  SUBSEC         ;CLEAR THE SUBSEC COUNTER
2707 021120    105237  002215          INCB  SECOND         ;INCREMENT THE SECONDS COUNTER
2708 021124    005237  010556          INC   TIMER          ;INCREMENT THE COMMAND TIMER
2709 021130    122737  000074  002215    CMPB  #60.,SECOND    ;IS IT A MINUTE YET ?
2710 021136    001037          BNE   HDLEXT         ;NO, GET OUT
2711 021140    105037  002215          CLRB  SECOND         ;CLEAR THE SECOND COUNTER
2712 021144    105237  002214          INCB  MINUTE         ;INCREMENT THE MINUTE COUNTER
2713 021150    122737  000074  002214    CMPB  #60.,MINUTE    ;IS IT AN HOUR YET ?
2714 021156    001027          BNE   HDLEXT         ;NO, GET OUT
2715 021160    105037  002214          CLRB  MINUTE         ;CLEAR THE MINUTE COUNTER
2716 021164    105237  002213          INCB  HOURS          ;INCREMENT THE HOUR COUNTER
2717 021170    122737  000030  002213    CMPB  #24.,HOURS     ;IS IT A DAY YET ?
2718 021176    001017          BNE   HDLEXT         ;NO, GET OUT
2719 021200    105037  002213          CLRB  HOURS          ;CLEAR THE HOURS COUNTER
2720 021204    105237  003564          INCB  DAYS           ;INCREMENT THE DAY COUNT
2721 021210    PRINTF  #DAY,<B,DAYS> ;PRINT END OF DAY STATMENT
      021210    005046          CLR   -(SP)
      021212    153716  003564          BISB  DAYS,(SP)
      021216    012746  020125          MOV   #DAY,-(SP)
      021222    012746  000002          MOV   #2,-(SP)
      021226    010600          MOV   SP,R0
      021230    104417          TRAP  C$PNTF
      021232    062706  000006          ADD   #6,SP
2722 021236    HDLEXT:
2723 021236    ENDSRV
      021236    L10006:
      021236    000002          RTI

```



```

2725 .SBTTL SCHEDULER
2726 ;*****
2727 ;
2728 ; SCHEDULER
2729 ;
2730 ;Called by      : Test N
2731 ;Calls to       : CMMDSQ
2732 ;Outputs        : EOT Flag, Dropped Flag
2733 ;Register Inputs: R5 (Pointer to command active in table - not used here)
2734 ;Registers Used : R4 (Pointer to LUN Block for use by called subs)
2735 ;
2736
2737 021240 SCHED::
2738 021240 005001 CLR R1 ;SET R1 TO FIST UNIT
2739 021242 005037 002074 CLR L$LUN ;SET L$LUN TO FIRST UNIT
2740 021246 012704 002314 MOV #LUN0,R4 ;SET R4 TO THE FIRST LUN BLOCK
2741 021252 022737 000003 002114 CMP #3,L$TEST ;ARE WE IN TEST 3 ?
2742 021260 001014 BNE 1$ ;YES, PRINT LINEFEED
2743 021262 122765 000020 000000 CMPB #WR,CMD(R5) ;IS IT A WRITE COMMAND ?
2744 021270 001010 BNE 1$ ;NO, GET OUT
2745 021272 PRINTF #LINE ;PRINT A LINE FEED
      021272 012746 020527 MOV #LINE,-(SP)
      021276 012746 000001 MOV #1,-(SP)
      021302 010600 MOV SP,R0
      021304 104417 TRAP C$PNTF
      021306 062706 000004 ADD #4,SP
2746 021312 032761 000001 003350 1$: BIT #AVB,DRINUS(R1) ;SEE IF DRIVE IS PRESENT AND AVAILABLE
2747 021320 001416 BEQ 2$ ;GET THE NEXT DRIVE IF IT ISN'T
2748 021322 032761 000004 003350 BIT #EOT,DRINUS(R1) ;CHECK IF THE DRIVE IS AT EOT
2749 021330 001012 BNE 2$ ;GET NEXT DRIVE IF IT IS
2750
2751 021332 012764 000377 000010 MOV #377,SLTUSE(R4) ;SET ALL RESPONSE SLOTS TO THE PORT
2752 021340 004737 026230 JSR PC,PRICLR ;GO DO IT
2753 021344 112737 000004 010571 MOVB #4,CRDLIM ;CREDITS START AT 4 FOR NEW LUN
2754 021352 004737 021406 JSR PC,CMMDSQ ;GO DO THE TEST ON THIS DRIVE
2755
2756 021356 022701 000006 2$: CMP #6.,R1 ;HAVE WE DONE THEM ALL ?
2757 021362 001410 BEQ 3$ ;GET OUT
2758 021364 062701 000002 ADD #UNTSTP,R1 ;GET NEXT UNIT
2759 021370 062704 000172 ADD #LUNSTP,R4 ;SET UP THE NEXT LUN BLOCK
2760 021374 005237 002074 INC L$LUN ;GET NEXT UNIT
2761 021400 BREAK
      021400 104422 TRAP C$BRK
2762 021402 000743 BR 1$ ;GO DO THE NEXT ONE
2763 021404 000207 3$: RTS PC ;RETURN

```

```

2765 .SBTTL COMMAND SEQUENCER
2766 ;*****
2767 ;
2768 ;COMMAND SEQUENCER
2769 ;
2770 ;Called by : SCHED
2771 ;Calls to : CMDBLD, QCMD, CLSDRV, RSPHDR, UNJAM
2772 ;Register Inputs: R5 - POINTER TO COMMAND ACTIVE IN TABLE
2773 ; R4 - POINTER TO LUN BLOCK
2774 ;
2775 ;
2776 021406 CMMDSQ::
2777 021406 PUSH <R1,R5> ;SAVE R1 AND R5
021406 010146 MOV R1,-(SP) ;;PUSH R1 ON STACK
021410 010546 MOV R5,-(SP) ;;PUSH R5 ON STACK
2778 021412 004737 022002 JSR PC,CMDBLD ;GO BUILD THE COMMAND
2779 021416 042737 000200 003516 BIC #CMDONE,PCFLAG ;GET SET TO START ISSUING COMMANDS
2780 021424 013737 003420 003400 MOV ITERS,CMDCNT ;GET THE COMMAND COUNT
2781 021432 013737 003420 003402 MOV ITERS,RSPCNT ;GET THE RESPONSE COUNT
2782 021440 012737 177777 010550 MOV #-1,CMSTSV ;RESET THE GCS PROGRESS COUNT
2783
2784 021446 032737 000040 003516 5$: BIT #GCSRFL,PCFLAG ;STILL LOOKING FOR A GCS RESPONSE ?
2785 021454 001023 BNE 15$ ;DON'T QUEUE UP THE NEXT COMMAND
2786 021456 005737 003400 TST CMDCNT ;DO WE STILL HAVE COMMANDS TO ISSUE ?
2787 021462 001004 BNE 10$ ;YES, KEEP GOING.
2788 021464 052737 000200 003516 BIS #CMDONE,PCFLAG ;SET THE ALL COMMANDS ISSUED FLAG
2789 021472 000414 BR 15$
2790
2791 021474 004737 023050 10$: JSR PC,QCMD ;GO QUEUE UP THE NEXT COMMAND
2792 021500 032737 000002 003516 BIT #NCLKFL,PCFLAG ;IS A CLOCK PRESENT ?
2793 021506 001003 BNE 13$ ;NO CLOCK, START REGULAR TIMER
2794 021510 005037 010556 CLR TIMER ;SET TIMER TO 0
2795 021514 000403 BR 15$ ;GO ISSUE COMMAND
2796 021516 012737 010000 010556 13$: MOV #10000,TIMER ;SET UP THE TIMER
2797
2798 021524 15$: BREAK
021524 104422 TRAP C$BRK
2799 021526 004737 023444 JSR PC,CLSDRV ;CALL THE CLASS DRIVER
2800 021532 032737 000002 003516 BIT #NCLKFL,PCFLAG ;IS A CLOCK PRESENT ?
2801 021540 001007 BNE 18$ ;NO CLOCK, START REGULAR TIMER
2802 021542 022737 000171 010556 CMP #121.,TIMER ;HAVE WE TIMED OUT ?
2803 021550 101012 BHI 20$ ;NO, KEEP GOING
2804 021552 005037 010556 CLR TIMER ;SET TIMER TO 0
2805 021556 000414 BR 25$ ;YES,SET UP FOR TIME OUT
2806
2807 021560 005337 010556 18$: DEC TIMER ;DECREMENT THE TIMER
2808 021564 001004 BNE 20$ ;BRANCH IF NOT 0
2809 021566 012737 010000 010556 MOV #10000,TIMER ;RESET THE TIMER
2810 021574 000405 BR 25$ ;SET TIME OUT ERROR
2811
2812 021576 022737 020000 010546 20$: CMP #IOICRD,IOSTAT ;INSUFFICIENT CREDITS ?
2813 021604 001747 BEQ 15$ ;YES, TRY AGAIN
2814 021606 000425 BR 35$ ;YES, CHECK IT OUT
2815
2816 021610 032737 000040 003516 25$: BIT #GCSRFL,PCFLAG ;WAITING FOR A GCS RESPONSE ?
2817 021616 001412 BEQ 30$ ;NO, SET UP TO DO A GCS
2818 021620 042737 000040 003516 BIC #GCSRFL,PCFLAG ;CLEAR THE GCS RESPONSE FLAG

```

```

2819 021626 012737 000004 010546      MOV      #IOTIME,IOSTAT      ;SET UP TIME OUT ERROR
2820 021634 004737 033734              JSR      PC,CORDMP          ;DO A VARIABLES DUMP
2821 021640 000240                      NOP
2822 021642 000407                      BR       35$                ;GO REPORT ERROR AND DROP UNIT
2823
2824 021644 052737 000020 003516 30$:   BIS      #GCSCFL,PCFLAG      ;SET THE GCS COMMAND FLAG
2825 021652 052737 000040 003516       BIS      #GCSRFL,PCFLAG      ;SET THE GCS RESPONSE FLAG
2826 021660 000721                      BR       15$                ;GO ISSUE THE GCS COMMAND
2827
2828 021662 022737 060000 010546 35$:   CMP      #ERRLOG!IOICRD,IOSTAT ;DID WE GET ERROR LOG PACKET ONLY?
2829 021670 001406                      BEQ      40$                ;YES - SO BRANCH AROUND NEXT INSTRUCTION
2830 021672 032737 000240 003516       BIT      #CMDONE!GCSRFL,PCFLAG ;HAVE ALL COMMANDS BEEN ISSUED ?
2831 021700 001002                      BNE      40$                ;YES, DON'T LET CMDCNT GO NEGATIVE
2832 021702 005337 003400              DEC      CMDCNT             ;DECREMENT THE COMMAND COUNT
2833 021706 022737 000000 010546 40$:   CMP      #IONORM,IOSTAT      ;WAS IT A NORMAL COMPLETION ?
2834 021714 001414                      BEQ      45$                ;YES , GET OUT
2835 021716 004737 027426              JSR      PC,RSPHDL          ;NO, LETS SEE WHAT IT WAS
2836
2837 021722 032737 020000 010546       BIT      #IOICRD,IOSTAT      ;WERE WE IN RSPHDL FOR ERROR LOG ONLY?
2838 021730 001275                      BNE      15$                ;YES - GO TRY TO POST SAME COMMAND.
2839 021732 022737 000002 003406       CMP      #SEREXC,RESPON      ;WAS IT A SERIOUS EXCEPTION ?
2840 021740 001002                      BNE      45$                ;NO, CONTINUE
2841 021742 004737 033242              JSR      PC,UNJAM           ;YES, GO UNJAM THE QUEUES
2842
2843 021746 032761 000001 003350 45$:   BIT      #AVB,DRINUS(R1)     ;HAS THE DRIVE BEEN DROPPED ?
2844 021754 001407                      BEQ      50$                ;YES, EXIT
2845 021756 005737 003402              TST      RSPCNT             ;HAVE WE GOTTEN ALL THE RESPONSE BACK ?
2846 021762 001231                      BNE      5$                 ;NO, GO BACK TO THE TOP
2847 021764 032737 000040 003516       BIT      #GCSRFL,PCFLAG      ;STILL LOOKING FOR A GCS RESPONSE ?
2848 021772 001225                      BNE      5$                 ;YES, DON'T GET OUT YET
2849
2850 021774 012605 50$:   POP      <R5,R1>            ;RESTORE REGISTERS R1 AND R5
      021774 012601          MOV      (SP)+,R5          ;;POP STACK INTO R5
      021776 012601          MOV      (SP)+,R1          ;;POP STACK INTO R1
2851 022000 000207          RTS      PC                ;RETURN

```

```

2853 .SBTTL COMMAND BUILDER
2854 ;*****
2855 ;
2856 ;COMMAND BUILDER
2857 ;
2858 ;Called by : CMMDSQ
2859 ;Calls to : BYTCNT, SELDAT, SELREC
2860 ;Register Inputs: R5 - pointer to test's command table
2861 ; R4 - pointer to LUN BLOCK
2862 ;Register Output: R3 - new pointer for command ring (set to start of ring)
2863 ; R2 - old pointer for command ring (set to start of ring)
2864 ;Registers Used : R3 Pointer to dummy packet before setting to command ring
2865 ;
2866 ;
2867 022002 CMDBLD::
2868 022002 012737 000004 003424 MOV #N,SUBCNT ;INITIALIZE THE SUB-ITERATION COUNTER
2869 022010 012703 003344 MOV #DUMPKT,R3 ;PUT THE DUMMY PACKET ADDRESS IN R3
2870 022014 116563 000000 000000 MOVB CMD(R5),CMD(R3) ;MOVE THE COMMAND PRIMITIVE TO THE PACKET
2871 ;
2872 022022 004737 022124 JSR PC,BYTCNT ;GO GET THE BYTE COUNT
2873 022026 013763 003416 000002 MOV BYTES,ITMOFF(R3) ;PUT THE BYTE COUNT IN THE DUMMY PACKET
2874 022034 004737 022212 JSR PC,SELDAT ;GO GET THE DATA
2875 022040 004737 022644 JSR PC,SELREC ;GO GET THE RECORD COUNT
2876 ;
2877 022044 022737 000003 002114 CMP #3,L$TEST ;ARE WE IN TEST 3 ?
2878 022052 001020 BNE 5$ ;YES, PRINT COUNTS
2879 022054 122765 000020 000000 CMPB #WR,CMD(R5) ;IS IT A WRITE COMMAND ?
2880 022062 001014 BNE 5$ ;NO, GET OUT
2881 022064 PRINTF #COUNTS,BYTES,ITERS ;YES, PRINT BYTE AND ITERATION COUNTS
2882 022064 013746 003420 MOV ITERS,-(SP)
2883 022070 013746 003416 MOV BYTES,-(SP)
2884 022074 012746 020202 MOV #COUNTS,-(SP)
2885 022100 012746 000003 MOV #3,-(SP)
2886 022104 010600 MOV SP,R0
2887 022106 104417 TRAP C$PNTF
2888 022110 062706 000010 ADD #10,SP
2889 ;
2890 5$: MOV #PCMDBF,R3 ;PUT THE PROGRAM COMMAND RING ADDRESS IN
2891 MOV R3,R2 ;R3 AND R2
2892 RTS PC ;RETURN

```

```

2889 .SBTTL BYTE COUNT
2890 ;*****
2891 ;
2892 ; BYTE COUNT
2893 ;
2894 ;Called by      : CMDBLD
2895 ;Calls to       : RANGEN
2896 ;Outputs        : BYTES (contains byte or item count to be used for this iteration set)
2897 ;Register Inputs: R5 - pointer to test command table
2898 ;               R4 - pointer to LUN BLOCK
2899 ;Register Output: None
2900 ;Registers Used : None
2901 ;
2902 ;
2903 022124 BYTCNT::
2904 022124 005037 003416 CLR      BYTES      ;CLEAR BYTES
2905 022130 005765 000002 TST     ITMCNT(R5) ;CHECK ITMCNT FOR 0
2906 022134 001404      BEQ     1$      ;CONTINUE IF IT IS 0
2907 022136 016537 000002 003416 MOV     ITMCNT(R5),BYTES ;PUT ITMCNT INTO BYTES
2908 022144 000421      BR      3$      ;EXIT
2909
2910 022146 122765 000020 000000 1$:  CMPB   #WR,CMD(R5) ;IS IT A READ OR WRITE
2911 022154 103415      BLO     3$      ;GET OUT IF IT ISN'T
2912
2913 022156 004737 022716      2$:  JSR     PC,RANGEN ;GO TO THE RANDOM GENERATOR
2914 022162 023727 003426 020000      CMP     RANWRD,#MAXBUF ;IS THE RESULT WITHIN THE LIMITS ?
2915 022170 101372      BHI     2$      ;BRANCH IF TOO HIGH
2916 022172 023727 003426 000024      CMP     RANWRD,#MINBUF ;IS IT TOO SMALL ?
2917 022200 103766      BLO     2$      ;BRANCH IF TOO SMALL
2918 022202 013737 003426 003416      MOV     RANWRD,BYTES ;PUT RANWRD INTO BYTES
2919 022210 000207      3$:  RTS     PC      ;RETURN

```

```

2921 .SBTTL SELECT DATA PATTERN
2922 ;*****
2923 ;
2924 ; SELECT DATA PATTERN
2925 ;
2926 ;Called by      : CMDBLD
2927 ;Calls to      : RANGEN
2928 ;Inputs       : Data Pattern in test command table
2929 ;              : PATSAV in LUN BLOCK if rotating pattern in use
2930 ;Outputs      : Write Buffer filled with appropriate data pattern
2931 ;              : PATSAV in LUN BLOCK updated to next pattern
2932 ;Register Inputs: R5 - pointer to test command table
2933 ;              : R4 - pointer to LUN BLOCK
2934 ;Registers Used : R3 - pointer to WRTBUF
2935 ;              : R2 - pointer to data pattern
2936 ;
2937 ;
2938 SELDAT::
2939     PUSH    <R1,R5>                ;SAVE R1 AND R5
2940     TSTB   DATPAT(R5)              ;TEST DATPAT FOR A TEST PATTERN
2941     BEQ    20$                     ;BRANCH IF WE DON'T NEED ONE
2942     TSTB   PATERN                  ;PATTERN SPECIFIED IN SOFTWARE P-TABLE ?
2943     BEQ    1$                      ;NO, KEEP GOING
2944     MOVB   PATERN,PATSAV(R4)       ;PUT THE PATTERN IN THE SAVE LOCATION
2945     BR     10$
2946
2947     1$:   TSTB   DATPAT(R5)          ;DO WE WANT ROTATING DATA PATTERNS ?
2948     BMI   5$                       ;IF NEGATIVE GO TO 5$
2949     MOVB   DATPAT(R5),PATSAV(R4)    ;LET PATSAV EQUAL DATPAT
2950     BR     10$                     ;BRANCH
2951
2952     5$:   INC    PATSAV(R4)          ;ADD 1 TO PATSAV
2953     CMP    PATSAV(R4),#ENDPAT       ;ARE WE AT THE END OF THE PATTERN TABLE ?
2954     BNE   10$                       ;NO, KEEP GOING
2955     MOV    #1.,PATSAV(R4)          ;AT THE END, LET PATSAV EQUAL 1
2956
2957     10$:  MOV    BYTES,R5            ;PUT THE BYTE COUNT IN R5
2958     BIT    #BIT0,R5                ;IS THE BYTE COUNT ODD ?
2959     BEQ    15$                      ;BRANCH IF NOT
2960     INC    R5                       ;MAKE BYTE COUNT EVEN FOR PATGEN
2961
2962     15$:  MOV    #WRTBUF,R3          ;POINT R3 TO THE WRITE BUFFER
2963     MOVB   PATSAV(R4),R1            ;SAVE PATSAV IN R1
2964     DEC    R1                       ;ADJUST FOR TABLE STEP
2965     ASL    R1                       ;MAKE IT MOD 2 OFFSET
2966
2967     20$:  JSR    PC,@PATTBL(R1)      ;GO FILL THE BUFFER
2968     POP    <R5,R1>                  ;RESTORE R5 AND R1
2969     RTS    PC                       ;RETURN
2970
2971 PATTBL::
2972     .WORD  PATGN1                    ;ALL 1'S
2973     .WORD  PATGN2                    ;ALL 0'S
2974     .WORD  PATGN3                    ;WORST CASE MFM
2975     .WORD  PATGN4                    ;ALTERNATE 1'S AND 0'S
2976     .WORD  PATGN5                    ;RANDOM DATA
2977

```

```

2978 022356 022504 .WORD PATGN6 ;1110 REPEATING PATTERN
2979 022360 022520 .WORD PATGN7 ;COMBINATION PAT 3 AND 5
2980
2981 022362 PATGN1:
2982 022362 012723 177777 MOV #-1,(R3)+ ;PUT ALL 1'S INTO THE BUFFER
2983 022366 162705 000002 SUB #2,R5 ;SUBTRACT TWO FROM R5
2984 022372 001373 BNE PATGN1 ;KEEP GOING IF WE AREN'T AT 0
2985 022374 000207 RTS PC ;RETURN
2986
2987 022376 PATGN2:
2988 022376 005023 CLR (R3)+ ;PUT ALL 0'S INTO THE BUFFER
2989 022400 162705 000002 SUB #2,R5 ;SUBTRACT TWO FROM R5
2990 022404 001374 BNE PATGN2 ;KEEP GOING IF WE AREN'T AT 0
2991 022406 000207 RTS PC ;RETURN
2992
2993 022410 PATGN3:
2994 022410 012723 133333 MOV #133333,(R3)+ ;PUT THE NUMBER INTO THE BUFFER
2995 022414 162705 000002 SUB #2,R5 ;SUBTRACT TWO FROM R5
2996 022420 001412 BEQ 1$ ;KEEP GOING IF WE AREN'T AT 0
2997 022422 012723 155555 MOV #155555,(R3)+ ;PUT THE NUMBER INTO THE BUFFER
2998 022426 162705 000002 SUB #2,R5 ;SUBTRACT TWO FROM R5
2999 022432 001405 BEQ 1$ ;KEEP GOING IF WE AREN'T AT 0
3000 022434 012723 066666 MOV #066666,(R3)+ ;PUT THE NUMBER INTO THE BUFFER
3001 022440 162705 000002 SUB #2,R5 ;SUBTRACT TWO FROM R5
3002 022444 001361 BNE PATGN3 ;KEEP GOING IF WE AREN'T AT 0
3003 022446 000207 1$: RTS PC ;RETURN
3004
3005 022450 PATGN4:
3006 022450 012723 125252 MOV #125252,(R3)+ ;PUT ALTERNATING 1 AND 0 INTO THE BUFFER
3007 022454 162705 000002 SUB #2,R5 ;SUBTRACT TWO FROM R5
3008 022460 001373 BNE PATGN4 ;KEEP GOING IF WE AREN'T AT 0
3009 022462 000207 RTS PC ;RETURN
3010
3011 022464 PATGN5:
3012 022464 004737 022716 JSR PC,RANGEN ;GO GENERATE RANDOM PATTERN
3013 022470 013723 003426 MOV RANWRD,(R3)+ ;PUT THE NUMBER INTO THE BUFFER
3014 022474 162705 000002 SUB #2,R5 ;SUBTRACT TWO FROM R5
3015 022500 001371 BNE PATGN5 ;KEEP GOING IF WE AREN'T AT 0
3016 022502 000207 RTS PC ;RETURN
3017
3018 022504 PATGN6:
3019 022504 012723 167356 MOV #167356,(R3)+ ;PUT 1110 REPEATING IN BUFFER
3020 022510 162705 000002 SUB #2,R5 ;SUBTRACT TWO FROM R5
3021 022514 001373 BNE PATGN6 ;KEEP GOING IF WE AREN'T AT 0
3022 022516 000207 RTS PC ;RETURN
3023
3024 022520 PATGN7:
3025 022520 PUSH <R2>
3026 022522 012702 001000 1$: MOV #512.,R2
3027 022526 012723 133333 3$: MOV #133333,(R3)+ ;PUT THE NUMBER INTO THE BUFFER
3028 022532 162705 000002 SUB #2,R5 ;SUBTRACT TWO FROM R5
3029 022536 001440 BEQ 10$ ;KEEP GOING IF WE AREN'T AT 0
3030 022540 162702 000002 SUB #2,R2 ;HAVE WE DONE A FULL BLOCK YET
3031 022544 001420 BEQ 5$ ;YES DO NEXT BLOCK IN PATTERN 5
3032 022546 012723 155555 MOV #155555,(R3)+ ;PUT THE NUMBER INTO THE BUFFER
3033 022552 162705 000002 SUB #2,R5 ;SUBTRACT TWO FROM R5
3034 022556 001430 BEQ 10$ ;KEEP GOING IF WE AREN'T AT 0
    
```



```

3055 .SBTTL SELECT RECORD
3056 ;*****
3057 ;
3058 ; SELECT RECORD
3059 ;
3060 ;Called by      : CMDBLD
3061 ;Calls to      : RANGEN
3062 ;Outputs       : ITERS (number of iterations for this set)
3063 ;Register Inputs: R5 - pointer to test command table
3064 ;              : R4 - pointer to LUN BLOCK
3065 ;
3066 ;
3067 022644 SELREC::
3068 022644 005765 000004      TST      ITRCNT(R5)      ;TEST THE ITERATION COUNT
3069 022650 001016          BNE      10$          ;IF IT ISN'T 0 THEN BRANCH
3070 022652 004737 022716    5$: JSR      PC,RANGEN      ;GO TO THE RANDOM GENERATOR
3071 022656 023727 003426 003720  CMP      RANWRD,#MAXITR    ;IS THE ITERATION COUNT TO HIGH ?
3072 022664 101372          BHI      5$          ;GO TRY AGAIN
3073 022666 023727 003426 000144  CMP      RANWRD,#MINITR    ;IS THE ITERATION SET TOO SMALL ?
3074 022674 103766          BLO      5$          ;GO TRY AGAIN
3075 022676 013737 003426 003420  MOV      RANWRD,ITERS     ;SAVE THE RANDOM NUMBER
3076 022704 000403          BR       15$          ;EXIT
3077 022706 016537 000004 003420 10$: MOV      ITRCNT(R5),ITERS ;SAVE THE ITERATION COUNT
3078 022714 000207          15$: RTS      PC          ;RETURN

```

```

3080 .SBTTL RANDOM NUMBER GENERATOR
3081 ;*****
3082 ;
3083 ;RANDOM NUMBER GENERATOR
3084 ;
3085 ;Called by      : BYTCNT, SELDAT, SELREC
3086 ;Inputs        : RAN1, RAN2, RAN3
3087 ;Outputs       : RANWRD
3088 ;Registers Used : R5
3089 ;
3090
3091 022716          RANGEN::
3092 022716          PUSH    <R5>                ;SAVE R5
3093 022720 016437 000144 003430          MOV    SEED1(R4),RAN1          ;PUT SEED1 INTO RAN1
3094 022726 016437 000146 003432          MOV    SEED2(R4),RAN2          ;PUT SEED2 INTO RAN2
3095 022734 016437 000150 003434          MOV    SEED3(R4),RAN3          ;PUT SEED3 INTO RAN3
3096 022742 013705 003130                MOV    RAN1,R5                ;MOVE THE FIRST SEED INTO R5
3097 022746 000241                CLC                          ;CLEAR THE CARRY FLAG
3098 022750 005337 003434          DEC    RAN3                    ;DECREMENT THE THIRD SEED
3099 022754 006105                ROL    R5
3100 022756 006105                ROL    R5
3101 022760 063705 003432          ADD    RAN2,R5                ;ADD THE SECOND SEED TO R5
3102 022764 010537 003430          MOV    R5,RAN1                ;PUT IT ALL IN THE FIRST SEED
3103 022770 063705 003434          ADD    RAN3,R5                ;PUT THE THIRD SEED INTO R5
3104 022774 006105                ROL    R5
3105 022776 006105                ROL    R5
3106 023000 063705 003432          ADD    RAN2,R5                ;ADD THE SECOND SEED TO R5
3107 023004 006105                ROL    R5
3108 023006 006105                ROL    R5
3109 023010 010537 003432          MOV    R5,RAN2                ;PUT IT IN THE SECOND SEED
3110 023014 013737 003430 003426          MOV    RAN1,RANWRD            ;PUT THE FIRST SEED INTO RANWORD
3111 023022 013764 003430 000144          MOV    RAN1,SEED1(R4)         ;PUT RAN1 INTO SEED1
3112 023030 013764 003432 000146          MOV    RAN2,SEED2(R4)         ;PUT RAN2 INTO SEED2
3113 023036 013764 003434 000150          MOV    RAN3,SEED3(R4)         ;PUT RAN3 INTO SEED3
3114 023044                POP    <R5>                    ;RESTORE R5
3115 023046 000207          RTS    PC                      ;EXIT

```

```

3117 .SBTTL QUEUE COMMANDS
3118 ;*****
3119 ;
3120 ; QUEUE COMMANDS
3121 ;
3122 ;Called by      : CMMDSQ
3123 ;Calls to      : SUBITR
3124 ;Register Inputs: R3 - pointer to next slot in ring
3125 ;               R4 - pointer to LUN BLOCK
3126 ;Register Output: R3 - updated to point to next available slot
3127 ;Registers Used : R5 - Points to dummy packet
3128 ;
3129 ;
3130 QCMD::
3131 023050 022703 003344      CMP      #PCBEND,R3      ;IS R3 POINTING AT THE END OF THE RING ?
3132 023054 001002          BNE      1$             ;NO, THEN KEEP GOING
3133 023056 012703 003264      MOV      #PCMDBF,R3     ;YES, SET IT TO THE RING BEGINNING
3134 023062 012705 003344      MOV      #DUMPK1,R5     ;POINT R5 TO THE DUMMY PACKET
3135 023066 116563 000000 000000 1$:  MOVB    CMD(R5),CMD(R3) ;PUT THE COMMAND PRIMITIVE INTO THE RING
3136 023074 016563 000002 000002  MOV      ITMOFF(R5),ITMOFF(R3) ;PUT THE ITEM OFFSET INTO THE RING
3137
3138 023102 004737 023276      JSR      PC,OBCTHD      ;GO GET THE OBJECT COUNT
3139 023106 016463 000034 000004  MOV      OBJFDL(R4),OBOFFL(R3) ;PUT THE LOW FIELD INTO THE RING
3140 023114 016463 000036 000006  MOV      OBJFDH(R4),OBOFFH(R3) ;PUT THE HIGH FIELD INTO THE RING
3141
3142 023122 004737 023142      JSR      PC,SUBITR      ;GO TO SUB-ITERS
3143 023126 013763 003422 000012  MOV      BUFADR,BUFOFF(R3) ;PUT THE BUFFER ADDRESS INTO THE RING
3144 023134 062703 000014      ADD      #PCBSTP,R3     ;MOVE R3 TO THE NEXT SLOT IN THE RING
3145 023140 000207          RTS      PC             ;RETURN

```

```

3147      .SBTTL  SUB-ITERATION
3148      ;*****
3149      ;
3150      ; SUB-ITERATION
3151      ;
3152      ;Called by      : QCMD
3153      ;Outputs       : BUFADR
3154      ;Register Inputs: R3 - pointer to command slot
3155      ;               : R4 - pointer to LUN BLOCK
3156      ;
3157      ;
3158      SUBITR::
3159      023142 105763 000000      TSTB  CMD(R3)      ;ARE WE ISSUING NULL COMMANDS ?
3160      023146 001452          BEQ    10$          ;BRANCH IF THE NULL COMMAND
3161      023150 122763 0J0020 000000  CMPB  #WR,CMD(R3)  ;IS IT GREATER THAN A WRITE
3162      023156 103446          BLO    10$          ;YES, BRANCH
3163      ;
3164      023160 005337 003424      1$:  DEC  SUBCNT      ;SUBTRACT 1 FROM SUBCNT
3165      023164 001025          BNE    5$          ;BRANCH IF NOT 0
3166      023166 016337 000004 070550  MOV  OBOFFL(R3),WRTBUF-2 ;PUT LOW ORDER OBJECT COUNT IN WRTBUF
3167      023174 012737 000004 003424  MOV  #N,SUBCNT      ;REINIT SUBCNT
3168      023202 012737 050552 003422  4$:  MOV  #RDBUF,BUFADR  ;PUT THE READ BUFFER ADDRESS IN BUFADR
3169      023210 122763 000020 000000  CMPB  #WR,CMD(R3)  ;IS IT A WRITE COMMAND
3170      023216 001026          BNE    10$         ;GET OUT IF IT'S NOT
3171      023220 022737 000003 002114  CMP  #3,L$TEST    ;ARE WE IN TEST 3 ?
3172      023226 001017          BNE    7$          ;NO, SET WRITE BUFFER IN BUFADR
3173      023230 012737 070550 003422  MOV  #WRTBUF-2,BUFADR ;SET MODIFIED WRITE BUFFER IN BUFADR
3174      023236 000416          BR    10$         ;EXIT
3175      ;
3176      023240 122763 000020 000000  5$:  CMPB  #WR,CMD(R3)  ;SEE IF ITS A WRITE
3177      023246 001407          BEQ    7$          ;YES, BRANCH
3178      023250 022737 000006 002114  CMP  #6,L$TEST    ;ARE WE IN TEST 6 ?
3179      023256 001751          BEQ    4$          ;YES, PUT READ BUFFER IN BUFADR
3180      023260 112763 000040 000000  MOVB  #ACC,CMD(R3) ;SET UP A COMPARE HOST DATA COMMAND
3181      023266 012737 070552 003422  7$:  MOV  #WRTBUF,BUFADR ;SET WRTBUF ADDRESS IN BUFADR
3182      023274 000207          10$:  RTS   PC          ;RETURN

```

```

3184 .SBTTL OBJET COUNT HANDLER
3185 ;*****
3186 ;
3187 ; OBJECT COUNT HANDLER
3188 ;
3189 ;Called by      : QCMD
3190 ;Inputs        : Current Object Count in LUN BLOCK
3191 ;Outputs       : Updated Object Count in LUN BLOCK
3192 ;Register Inputs: R3 - pointer to command slot
3193 ;              : R4 - pointer to LUN BLOCK
3194 ;
3195 ;
3196 023276 OBCTHD::
3197 023276          PUSH    <R1>                ;SAVE R1
3198 023300 116301 000000          MOVB   CMD(R3),R1          ;PUT THE COMMAND PRIMITIVE INTO R1
3199 023304 042701 000007          BIC   #7,R1             ;STRIP OFF THE MODIFIERS
3200 023310 005701                TST   R1                 ;IS IT THE NULL COMMAND ?
3201 023312 001452                BEQ   6$                 ;EXIT IF IT IS
3202 023314 022701 000160          CMP   #REW,R1           ;IS IT A REWIND ?
3203 023320 001005                BNE   1$                 ;BRANCH IF NOT
3204 023322 005064 000034          CLR  OBJFDL(R4)        ;CLEAR THE OBJECT
3205 023326 005064 000036          CLR  OBJFDH(R4)        ;COUNT FIELD AND
3206 023332 000442                BR    6$                 ;EXIT
3207
3208 023334 022701 000050          1$:  CMP   #SPC,R1        ;IS IT A NON-DATA TRANSFER COMMAND ?
3209 023340 101007                BHI   2$                 ;BRANCH IF IT IS
3210 023342 022701 000100          CMP   #WTM,R1          ;IS IT A WRITE TAPE MARK ?
3211 023346 001404                BEQ   2$                 ;BRANCH IF IT IS
3212 023350 016337 000002 003530  MOV   ITMOFF(R3),R8     ;PUT THE ITEM COUNT IN TEMP REGISTER
3213 023356 000403                BR    3$                 ;CONTINUE
3214
3215 023360 012737 000001 003530  2$:  MOV   #1,R8             ;PUT A 1 IN THE TEMP REGISTER
3216 023366 032763 000002 000000  3$:  BIT   #EOTBIT,CMD(R3)  ;IS IT AN LEOT COMMAND ?
3217 023374 001021                BNE   6$                 ;GET OUT IF IT IS
3218 023376 032763 000001 000000  BIT   #REVBIT,CMD(R3)  ;IS THE COMMAND REVERSE ?
3219 023404 001007                BNE   4$                 ;BRANCH IF REVERSE
3220 023406 063764 003530 000034  ADD   R8,OBJFDL(R4)    ;ADD TEMP TO THE OBJECT COUNT
3221 023414 103011                BCC   6$                 ;BRANCH IF NO CARRY
3222 023416 005264 000036          INC  OBJFDH(R4)        ;OTHERWISE ADD 1 TO THE HIGH OBJECT COUNT
3223 023422 000406                BR    6$                 ;EXIT
3224
3225 023424 163764 003530 000034  4$:  SUB   R8,OBJFDL(R4)    ;IF REVERSE, SUBTRACT TEMP FROM THE
3226 023432 103002                BCC   6$                 ;OBJECT COUNT AND BRANCH IF NO CARRY
3227 023434 005364 000036          DEC  OBJFDH(R4)        ;OTHERWISE SUBTRACT 1 FROM OBJECT COUNT HIGH
3228 023440                POP   <R1>              ;RESTORE R1
3229 023442 000207                RTS   PC                 ;EXIT

```

```

3231 .SBTTL CLASS DRIVER TRANSMIT
3232 ;*****
3233 ;
3234 ;Class Driver Transmit
3235 ;
3236 ;Called By      : CMMDSQ
3237 ;Calls To       : CDRECV, STFPCK, PRTRV
3238 ;Inputs         : CRDLIM - Command slots open in the drive.
3239                : COLSAV - Old driver command pointer.
3240 ;Outputs        : IOSTAT - Transfer status.
3241                : CMDSEQ - Number appended to each command packet.
3242                : GCSREF - Get Command Status reference number.
3243 ;Register Inputs: R2 - Old pointer to program command ring.
3244                : R3 - New pointer to program command ring.
3245                : R4 - Lun block pointer.
3246 ;Register Outputs: R5 - Old pointer to driver command ring.
3247 ;
3248 ;
3249 023444 CLSDRV::
3250 023444          PUSH      <R3,R5>          ;SAVE R3, R5
3251 023450 016405 000016          MOV      COLSAV(R4),R5          ;POINT R5 TO THE OLD DRIVER COMMAND
3252 023454 032737 000040 003516          BIT      #GCSRFL,PCFLAG        ;IS THIS A GCS COMMAND ?
3253 023462 001010                    BNE     10$                    ;YES, GO SETUP
3254 023464 022703 003264          CMP      #PCMDBF,R3           ;IS R3 AT THE BEGINNING OF THE PROGRAM RING ?
3255 023470 001403                    BEQ     5$                      ;YES, BRANCH
3256 023472 162703 000014          SUB      #PCBSTP,R3          ;NO, MOVE R3 ONE SLOT BACK
3257 023476 000402                    BR      10$                    ;CONTINUE
3258 ;
3259 023500 062703 000044          5$:    ADD      #PCB3SP,R3        ;YES, ADVANCE R3 TWO SLOTS
3260 023504 005037 010546          10$:   CLR      IOSTAT           ;CLEAR THE I/O STATUS WORD
3261 023510 122763 000170 000000          CMPB    #INT,CMD(R3)         ;IS THIS A INITIALIZATION COMMAND ?
3262 023516 001003                    BNE     15$                    ;CONTINUE IF IT ISN'T
3263 ;
3264 023520 004737 026306          JSR      PC,PRINT            ;CALL THE PORT INIT ROUTINE
3265 023524 000464                    BR      55$                    ;EXIT
3266 023526 005764 000010          15$:   TST      SLTUSE(R4)       ;DID WE HANDLE ANY RESPONSES LAST TIME ?
3267 023532 001402                    BEQ     20$                    ;BRANCH IF NOT
3268 023534 004737 026230          JSR      PC,PRTCLR          ;GO CLEAR THE OLD RESPONSES
3269 ;
3270 023540 004737 023710          20$:   JSR      PC,CDRECV         ;GO CHECK FOR ANY NEW RESPONSES
3271 023544 105737 010546          TSTB    IOSTAT              ;IS THE I/O STATUS O.K. ?
3272 023550 001052                    BNE     55$                    ;EXIT IF IT ISN'T
3273 ;
3274 023552 032737 000020 003516          25$:   BIT      #GCSCFL,PCFLAG    ;IS THIS A GCS COMMAND ?
3275 023560 001010                    BNE     30$                    ;YES, GO SETUP MINLIM
3276 023562 032737 000200 003516          BIT      #CMDONE,PCFLAG      ;IS THIS A NULL COMMAND ?
3277 023570 001042                    BNE     55$                    ;EXIT IF IT IS
3278 023572 032763 000200 000000          BIT      #IMM,CMD(R3)       ;IS THIS AN IMMEDIATE COMMAND ?
3279 023600 001404                    BEQ     35$                    ;NO, BRANCH
3280 023602 112737 000001 010570          30$:   MOVB    #1,MINLIM        ;YES, SET MINIMUM LIMIT TO 1
3281 023610 000403                    BR      40$                    ;BRANCH
3282 ;
3283 023612 112737 000002 010570          35$:   MOVB    #2,MINLIM        ;NO, SET MINIMUM LIMIT TO 2
3284 023620 123737 010571 010570          40$:   CMPB    CRDLIM,MINLIM    ;DO WE HAVE ENOUGH CREDITS ?
3285 023626 103004                    BHIS   45$                    ;YES, KEEP GOING
3286 023630 052737 020000 010546          BIS     #IOICRD,IOSTAT      ;SET INSUFFICIENT CREDIT IN I/O STATUS
3287 023636 000417                    BR      55$                    ;GET OUT

```

```
3288
3289 023640 005264 000006          45$:  INC  CMDSEQ(R4)          ;ADD 1 TO THE COMMAND SEQUENCE NUMBER
3290 023644 032737 000020 003516  BIT  #GCSCFL,PCFLAG      ;IS IT A GCS COMMAND ?
3291 023652 001403                BEQ  50$                ;NO, BRANCH
3292 023654 016437 000006 010552  MOV  CMDSEQ(R4),GCSREF  ;SAVE THE COMMAND REFERENCE NUMBER
3293
3294 023662 105337 010571          50$:  DECB CRDLIM          ;SUBTRACT 1 FROM THE CREDIT LIMIT
3295 023666 004737 024164          JSR  PC,STFPCK         ;GO FILL THE TMSCP PACKET
3296 023672 004737 025730          JSR  PC,PRTDRV        ;GO SEND THE COMMAND
3297
3298 023676 010564 000016          55$:  MOV  R5,COLSAV(R4)   ;SAVE R5 IN CMMAND OLD POINTER SAVE
3299 023702                POP  <R5,R3>          ;RESTORE R3, AND R5
3300 023706 000207                RTS  PC                ;RETURN
```


3359	024106	000403			BR	25\$;BRANCH TO THE END
3360								
3361	024110	052737	040000	010546	20\$:	BIS	#ERRLOG,IOSTAT	;SET ERROR LOG IN I/O STATUS
3362	024116	005337	003414		25\$:	DEC	RESP	;SUBTRACT 1 FROM THE RESPONSE COUNT
3363	024122	062701	000104			ADD	#DRBSTP,R1	;ADJUST R1
3364	024126	026401	000160			CMP	URBEND(R4),R1	;IS IT AT THE END OF THE RING ?
3365	024132	001002				BNE	30\$;NO, KEEP GOING
3366	024134	016401	000156			MOV	URSPBF(R4),R1	;YES, SET IT TO BEGINNING OF THE RING
3367								
3368	024140	005737	003414		30\$:	TST	RESP	;HAVE WE DONE ALL THE RESPONSES ?
3369	024144	001275				BNE	5\$;NO, DO IT AGAIN
3370								
3371	024146	005037	003414		35\$:	CLR	RESP	;CLEAR NOW IN CASE WE MADE ERROR EXIT
3372	024152	010164	000020			MOV	R1,RNUSAV(R4)	;SAVE THE NEW RESPONSE BUFFER POINTER
3373	024156					POP	<R2,R1>	;RESTORE R2,R1
3374	024162	000207				RTS	PC	;RETURN

```

3376 .SBTTL COMMAND STUFFER
3377 ;*****
3378 ;
3379 ; Stuff TMSCP Command Packet
3380 ;
3381 ;Called By      : CLSDRV
3382 ;Inputs        : CNUSAV - Points to next slot in the driver command ring.
3383 ;              : CMDSEQ - Number appended to each command packet.
3384 ;              : GCSREF - Get Command Status reference number.
3385 ;              : SEREXP - Flag set non-zero on occurrence of a serious exception.
3386 ;Outputs       : PCKSIZ - Length in bytes of the command packet.
3387 ;Register Inputs: R3 - New pointer to program command ring.
3388 ;              : R4 - Lun block pointer.
3389 ;              : R5 - Old pointer to driver command ring.
3390 ;Registers Used: R1 - New pointer to driver command ring.
3391
3392 024164 STFPCK::
3393 024164      PUSH    <R1,R2>                ;SAVE R1 AND R2
3394 024170      CLR     PCKSIZ                ;CLEAR PACKET SIZE
3395 024174      MOV     CNUSAV(R4),R1         ;LET R1 EQUAL THE NEW COMMAND POINTER
3396 024200      MOV     CMDSEQ(R4),P.CRF(R1) ;PUT COMMAND SEQUENCE NUMBER INTO PACKET
3397 024206      CLR     P.CRF+2(R1)         ;CLEAR THE UPPER WORD
3398 024212      MOV     TKUNIT(R4),P.UNIT(R1) ;PUT THE UNIT NUMBER INTO THE PACKET
3399 024220      CLR     P.UNIT+2(R1)        ;CLEAR THE UPPER WORD
3400 024224      CLR     P.MOD(R1)           ;CLEAR MODIFIERS FIELD
3401 024230      BIT     #GCSCFL,PCFLAG      ;ARE WE IN GCS COMMAND MODE ?
3402 024236      BEQ     5$                  ;NO, CONTINUE
3403 024240      JMP     GCMDST              ;YES, GO DO A GET COMMAND STATUS
3404
3405 024244      5$:  MOV     ITMOFF(R3),P.BCNT(R1) ;PUT THE BYTE COUNT INTO THE PACKET
3406 024252      CLR     P.BCNT+2(R1)        ;CLEAR THE UPPER WORD
3407 024256      MOV     CMD(R3),R8          ;PUT THE PRIMITIVE IN R8
3408 024264      BIC     #177770,R8         ;GET JUST THE MODIFIERS
3409 024272      CMP     #REVBIT,R8         ;IS THE COMMAND A REVERSE ?
3410 024300      BNE     10$                 ;NO, BRANCH
3411 024302      BIS     #MD.REV,P.MOD(R1)   ;YES, SET REVERSE IN THE MODIFIER FIELD
3412
3413 024310      10$: BIT     #SEREXC,LUNFLG(R4) ;IS IT A SERIOUS EXCEPTION CONDITION ?
3414 024316      BEQ     15$                 ;NO, BRANCH
3415 024320      BIS     #MD.CSE,P.MOD(R1)   ;YES, SET CLEAR SERIOUS EXCEPTION
3416 024326      BIC     #SEREXC,LUNFLG(R4)   ;CLEAR SERIOUS EXCEPTION FLAG
3417
3418 024334      15$: MOVVB  CMD(R3),R2        ;PUT THE COMMAND PRIMITIVE INTO R1
3419 024340      ASR     R2
3420 024342      ASR     R2
3421 024344      BIC     #1C76,R2           ;ADJUST FOR THE CASE STATEMENT
3422 024350      CMP     #46,R2             ;ARE WE IN THE RANGE ?
3423 024354      BHIS   20$                 ;YES, KEEP GOING
3424 024356      JMP     ILCMD              ;NO, HANDLE AN ILLEGAL COMMAND
3425 024362      20$: JMP     @CMDTBL(R2)     ;SELECT
3426
3427 024366      CMDTBL: .WORD  NULL
3428 024370      .WORD  READ
3429 024372      .WORD  WRITE
3430 024374      .WORD  CHODAT
3431 024376      .WORD  ACCESS
3432 024400      .WORD  SPCRF.C

```

```

3433 024402 024610 .WORD SKPTMK
3434 024404 024664 .WORD SPCOBJ
3435 024406 024722 .WORD WTAPMK
3436 024410 024750 .WORD ERASE
3437 024412 025006 .WORD ERASGP
3438 024414 025026 .WORD AVALAB
3439 024416 025064 .WORD ONLINE
3440 024420 025162 .WORD SUNCHR
3441 024422 025254 .WORD REWIND
3442 024424 025342 .WORD INIT
3443 024426 025346 .WORD ABOR
3444 024430 025400 .WORD GCMOST
3445 024432 025440 .WORD GUNSTA
3446 024434 025466 .WORD SCNTCH
3447
3448 024436 000137 025700 NULL: JMP COMEXI ;EXIT
3449
3450 024442 012761 000041 000010 READ: MOV #OP.RD,P.OPCD(R1) ;PUT THE READ OPCODE INTO THE PACKET
3451 024450 012737 000034 010564 MOV #34,PCKSIZ ;PUT THE PACKET SIZE INTO THE PACKET
3452 024456 000137 025616 JMP BUFDSC ;GOTO THE BUFFER DESCRIPTOR ROUTINE
3453
3454 024462 012761 000042 000010 WRITE: MOV #OP.WR,P.OPCD(R1) ;PUT THE WRITE OPCODE INTO THE PACKET
3455 024470 012737 000034 010564 MOV #34,PCKSIZ ;PUT THE PACKET SIZE INTO THE PACKET
3456 024476 000137 025616 JMP BUFDSC ;GOTO THE BUFFER DESCRIPTOR ROUTINE
3457
3458 024502 012761 000040 000010 CHODAT: MOV #OP.CMP,P.OPCD(R1) ;PUT COMPARE HOST DATA OPCODE IN PACKET
3459 024510 012737 000034 010564 MOV #34,PCKSIZ ;PUT THE PACKET SIZE INTO THE PACKET
3460 024516 000137 025616 JMP BUFDSC ;GOTO THE BUFFER DESCRIPTOR ROUTINE
3461
3462 024522 012761 000020 000010 ACCESS: MOV #OP.ACC,P.OPCD(R1) ;PUT THE ACCESS OPCODE INTO THE PACKET
3463 024530 012737 000020 010564 MOV #20,PCKSIZ ;PUT THE PACKET SIZE INTO THE PACKET
3464 024536 000137 025650 JMP SUPRES ;GOTO THE SUPPRESS ROUTINE
3465
3466 024542 012761 000045 000010 SPCREC: MOV #OP.REP,P.OPCD(R1) ;PUT REPOSITION OPCODE INTO THE PACKET
3467 024550 005061 000020 CLR P.TMGC(R1) ;CLEAR THE TAPE MARK COUNT
3468 024554 005061 000022 CLR P.TMGC+2(R1) ;CLEAR THE UPPER WORD
3469 024560 032737 000002 003530 BIT #EOTBIT,R8 ;IS THE DETECT LEOT BIT SET ?
3470 024566 001403 BEQ 70$ ;NO,CONTINUE
3471 024570 052761 000200 000012 BIS #MD.DLE,P.MOD(R1) ;YES, SET DETECT LEOT IN THE MODIFIER
3472 024576 012737 000024 010564 70$: MOV #24,PCKSIZ ;PUT THE PACKET SIZE INTO THE PACKET
3473 024604 000137 025650 JMP SUPRES ;GOTO THE SUPPRESS ROUTINE
3474
3475 024610 012761 000045 000010 SKPTMK: MOV #OP.REP,P.OPCD(R1) ;PUT THE REPOSITION OPCODE IN PACKET
3476 024616 016161 000014 000020 MOV P.BCNT(R1),P.TMGC(R1) ;PUT THE TAPE MARK COUNT IN PACKET
3477 024624 005061 000022 CLR P.TMGC+2(R1) ;CLEAR THE TAPE MARK FIELD
3478 024630 005061 000014 CLR P.BCNT(R1) ;CLEAR THE UPPER WORD
3479 024634 032737 000002 003530 BIT #EOTBIT,R8 ;IS THE DETECT LEOT BIT SET ?
3480 024642 001403 BEQ 100$ ;NO,CONTINUE
3481 024644 052761 000200 000012 BIS #MD.DLE,P.MOD(R1) ;YES, SET DETECT LEOT IN THE MODIFIER
3482 024652 012737 000024 010564 100$: MOV #24,PCKSIZ ;PUT THE PACKET SIZE INTO THE PACKET
3483 024660 000137 025650 JMP SUPRES ;GOTO THE SUPPRESS ROUTINE
3484
3485 024664 012761 000045 000010 SPCOBJ: MOV #OP.REP,P.OPCD(R1) ;PUT THE REPOSITION OPCODE IN PACKET
3486 024672 052761 000004 000012 BIS #MD.OBC,P.MOD(R1) ;SET THE OBJECT BIT IN THE MODIFIER
3487 024700 005061 000020 CLR P.TMGC(R1) ;CLEAR THE TAPE MARK FIELD
3488 024704 005061 000022 CLR P.TMGC+2(R1) ;CLEAR THE UPPER WORD
3489 024710 012737 000024 010564 MOV #24,PCKSIZ ;PUT THE PACKET SIZE INTO THE PACKET

```

```

3490 024716 000137 025650          JMP      SUPRES          ;GOTO THE SUPPRESS ROUTINE
3491
3492 024722 012761 000044 000010 WTAPMK: MOV      #OP.WTM,P.OPCD(R1)    ;PUT WRITE TAPE MARK OPCODE IN PACKET
3493 024730 052761 020000 000012      BIS      #MD.CSE,P.MOD(R1)    ;YES, SET CLEAR SERIOUS EXCEPTION
3494 024736 012737 000014 010564      MOV      #14,PCKSIZ          ;PUT THE PACKET SIZE INTO THE PACKET
3495 024744 000137 025700          JMP      COMEXI           ;GOTO THE EXIT
3496
3497 024750 012761 000022 000010 ERASE: MOV      #OP.ERS,P.OPCD(R1)    ;PUT THE ERASE OPCODE INTO THE PACKET
3498 024756 022737 000003 003530      CMP      #IMMBIT,R8        ;IS THE IMMEDIATE BIT SET ?
3499 024764 001403          BEQ      20$              ;NO,CONTINUE
3500 024766 052761 000100 000012      BIS      #MD.IMM,P.MOD(R1)    ;YES, SET IMMEDIATE IN THE MODIFIER
3501 024774 012737 000014 010564 20$:  MOV      #14,PCKSIZ          ;PUT THE PACKET SIZE INTO THE PACKET
3502 025002 000137 025700          JMP      COMEXI           ;GOTO THE EXIT
3503
3504 025006 012761 000026 000010 ERASGP: MOV      #OP.ERG,P.OPCD(R1)    ;PUT ERASE GAP OPCODE INTO THE PACKET
3505 025014 012737 000014 010564      MOV      #14,PCKSIZ          ;PUT THE PACKET SIZE INTO THE PACKET
3506 025022 000137 025700          JMP      COMEXI           ;GOTO THE EXIT
3507
3508 025026 012761 000010 000010 AVALAB: MOV      #OP.AVL,P.OPCD(R1)    ;PUT AVAILABLE OPCODE INTO THE PACKET
3509 025034 022737 000004 003530      CMP      #UNLBIT,R8        ;IS THE UNLOAD BIT SET ?
3510 025042 001403          BEQ      10$              ;NO,CONTINUE
3511 025044 052761 000020 000012      BIS      #MD.UNL,P.MOD(R1)    ;YES, SET UNLOAD IN THE MODIFIER FIELD
3512 025052 012737 000014 010564 10$:  MOV      #14,PCKSIZ          ;PUT THE PACKET SIZE INTO THE PACKET
3513 025060 000137 025700          JMP      COMEXI           ;GOTO THE EXIT
3514
3515 025064 012761 000011 000010 ONLINE: MOV      #OP.ONL,P.OPCD(R1)    ;PUT THE ONLINE OPCODE INTO THE PACKET
3516 025072 005061 000014          CLR      P.UNFL-2(R1)      ;CLEAR THE UNIT FLAG FIELD
3517 025076 005061 000016          CLR      P.UNFL(R1)
3518 025102 005061 000020          CLR      P.UNFL+2(R1)
3519 025106 005061 000022          CLR      P.UNFL+4(R1)
3520 025112 005061 000024          CLR      P.UNFL+6(R1)
3521 025116 005061 000026          CLR      P.UNFL+10(R1)
3522 025122 005061 000030          CLR      P.UNFL+12(R1)
3523 025126 005061 000032          CLR      P.UNFL+14(R1)
3524 025132 005061 000034          CLR      P.DVPM(R1)        ;CLEAR THE DEVICE PARAMETER FIELD
3525 025136 012761 000010 000040 MOV      #TF.BLK,P.FORM(R1)  ;PUT THE TAPE FORMAT INTO THE PACKET
3526 025144 005061 000042          CLR      P.SPED(R1)        ;CLEAR THE SPEED FIELD
3527 025150 012737 000044 010564 MOV      #44,PCKSIZ          ;PUT THE PACKET SIZE INTO THE PACKET
3528 025156 000137 025700          JMP      COMEXI           ;GOTO THE EXIT
3529
3530 025162 012761 000012 000010 SUNCHR: MOV      #OP.SUC,P.OPCD(R1)    ;SET UNIT CHARA. OPCODE INTO THE PACKET
3531 025170 005061 000014          CLR      P.UNFL-2(R1)      ;CLEAR THE UNIT FLAG FIELD
3532 025174 005061 000016          CLR      P.UNFL(R1)
3533 025200 005061 000020          CLR      P.UNFL+2(R1)
3534 025204 005061 000022          CLR      P.UNFL+4(R1)
3535 025210 005061 000024          CLR      P.UNFL+6(R1)
3536 025214 005061 000026          CLR      P.UNFL+10(R1)
3537 025220 005061 000030          CLR      P.UNFL+12(R1)
3538 025224 005061 000032          CLR      P.UNFL+14(R1)
3539 025230 005061 000034          CLR      P.DVPM(R1)        ;CLEAR THE DEVICE PARAMETERS FIELD
3540 025234 012761 000010 000040 MOV      #TF.BLK,P.FORM(R1)  ;PUT THE TAPE FORMAT INTO THE PACKET
3541 025242 005061 000042          CLR      P.SPED(R1)        ;CLEAR THE SPEED FIELD
3542 025246 012737 000044 010564 MOV      #44,PCKSIZ          ;PUT THE PACKET SIZE INTO THE PACKET
3543
3544 025254 012761 000045 000010 REWIND: MOV      #OP.REP,P.OPCD(R1)    ;PUT THE REPOSITION OPCODE INTO PACKET
3545 025262 052761 020002 000012      BIS      #MD.CSE!MD.RWD,P.MOD(R1) ;SET THE REWIND MODIFIER
3546 025270 022737 000003 003530      CMP      #IMMBIT,R8        ;IS THE IMMEDIATE BIT SET

```

```

3547 025276 001003          BNE      60$          ;NO,CONTINUE
3548 025300 052761 000100 000012          BIS      #MD.IMM,P.MOD(R1) ;YES, SET THE IMMEDIATE MODIFIER
3549 025306 005061 000020          60$:   CLR      P.TMGC(R1)      ;CLEAR THE TAPE MARK
3550 025312 005061 000022          CLR      P.TMGC+2(R1)     ;COUNT FIELD
3551 025316 012737 000024 010564          MOV      #24,PCKSIZ      ;PUT THE PACKET SIZE INTO THE PACKET
3552 025324 005064 000006          CLR      CMDSEQ(R4)      ;RESET THE COMMAND SEQUENCE NUMBER
3553 025330 042764 000010 000026          BIC      #EOTPR,LUNFLG(R4) ;CLEAT THE EOT PRINT FLAG
3554 025336 000137 025700          JMP      COMEXI          ;GOTO THE EXIT
3555
3556 025342 000137 025722          INIT:   JMP      EXIT          ;EXIT
3557
3558 025346 012761 000001 000010          ABOR:   MOV      #OP.ABO,P.OPCD(R1) ;PUT THE ABORT OPCODE INTO THE PACKET
3559 025354 016561 000000 000014          MOV      P.CRF(R5),P.OTRF(R1) ;PUT THE OLD CRN INTO THE PACKET
3560 025362 005061 000016          CLR      P.OTRF+2(R1)     ;CLEAR THE UPPER WORD
3561 025366 012737 000020 010564          MOV      #20,PCKSIZ      ;PUT THE PACKET SIZE INTO THE PACKET
3562 025374 000137 025700          JMP      COMEXI          ;GOTO THE EXIT
3563
3564 025400 012761 000002 000010          GCMDST: MOV      #OP.GCS,P.OPCD(R1) ;PUT GCS OPCODE INTO THE PACKET
3565 025406 016561 000000 000014          MOV      P.CRF(R5),P.OTRF(R1) ;PUT THE OLD CRN INTO THE PACKET
3566 025414 005061 000016          CLR      P.OTRF+2(R1)     ;CLEAR THE UPPER WORD
3567 025420 012737 000020 010564          MOV      #20,PCKSIZ      ;PUT THE PACKET SIZE INTO THE PACKET
3568 025426 042737 000020 003516          BIC      #GCSFL,PCFLAG   ;CLEAR GCS COMMAND MODE ?
3569 025434 000137 025700          JMP      COMEXI          ;GOTO THE EXIT
3570
3571 025440 012761 000003 000010          GUNSTA: MOV      #OP.GUS,P.OPCD(R1) ;PUT THE GUS OPCODE INTO THE PACKET
3572 025446 042761 020000 000012          BIC      #MD.CSE,P.MOD(R1) ;CLEAR CLEAR SERIOUS EXCEPTION MODIFIER
3573 025454 012737 000014 010564          MOV      #14,PCKSIZ      ;PUT THE PACKET SIZE INTO THE PACKET
3574 025462 000137 025700          JMP      COMEXI          ;GOTO THE EXIT
3575
3576 025466 012761 000004 000010          SCNTCH: MOV      #OP.SCC,P.OPCD(R1) ;PUT THE SCC OPCODE INTO THE PACKET
3577 025474 005061 000004          CLR      P.UNIT(R1)      ;CLEAR THE UNIT NUMBER
3578 025500 012761 000000 000014          MOV      #MSCPVR,P.VRSN(R1) ;PUT THE MSCP VERSION INTO THE PACKET
3579 025506 013761 010562 000016          MOV      CNTFLG,P.CNTF(R1) ;PUT CONTROLLER FLAGS INTO THE PACKET
3580 025514 012761 000000 000020          MOV      #HSTIMO,P.HTMO(R1) ;PUT THE HOST TIMEOUT INTO THE PACKET
3581 025522 005061 000022          CLR      P.HTMO+2(R1)     ;CLEAR THE TIME FIELD
3582 025526 005061 000024          CLR      P.TIME(R1)      ;
3583 025532 005061 000026          CLR      P.TIME+2(R1)     ;
3584 025536 005061 000030          CLR      P.TIME+4(R1)     ;
3585 025542 005061 000032          CLR      P.TIME+6(R1)     ;
3586 025546 005061 000034          CLR      P.CTPM(R1)      ;CLEAR THE FIRDT WORD
3587 025552 005061 000036          CLR      P.CTPM+2(R1)     ;CLEAR THE SECOND WORD
3588 025556 105737 002222          TSTB    PADDING          ;ARE WE ALLOWING WRITE PADDING ?
3589 025562 001003          BNE      5$            ;YES, DON'T SET THE BIT
3590 025564 052761 000001 000034          BIS      #BIT0,P.CTPM(R1) ;SET CNTRL PARAM TO DISABLE PADDING
3591 025572 012737 000040 010564          5$:   MOV      #40,PCKSIZ      ;PUT THE PACKET SIZE INTO THE PACKET
3592 025600 000137 025700          JMP      COMEXI          ;GOTO THE EXIT
3593
3594 025604 052737 000007 010546          ILCMD:  BIS      #ILLCMD,IOSTAT ;SET ILLCMD IN THE I/O STATUS
3595 025612 000137 025722          JMP      EXIT          ;GOTO THE ERROR EXIT
3596
3597 025616 016361 000012 000020          BUFDSC: MOV      BUFOFF(R3),P.BUFF(R1) ;PUT THE BUFFER ADDRESS INTO THE PACKET
3598 025624 005061 000022          CLR      P.BUFF+2(R1)     ;CLEAR THE REST OF THE BUFFER FIELD
3599 025630 005061 000024          CLR      P.BUFF+4(R1)     ;
3600 025634 005061 000026          CLR      P.BUFF+6(R1)     ;
3601 025640 005061 000030          CLR      P.BUFF+10(R1)    ;
3602 025644 005061 000032          CLR      P.BUFF+12(R1)    ;
3603

```

3604	025650	105737	002220		SUPRES:	TSTB	SERREC		;IS SUPPRESS ERROR CORRECTION ENABLED ?
3605	025654	001003				BNE	105\$;NO
3606	025656	052761	001000	000012		BIS	#MD.SEC,P.MOD(R1)		;YES, SET SEC MODIFIER
3607	025664	105737	002221		105\$:	TSTB	SERCOR		;IS SUPPRESS ERROR RECOVERY ENABLED ?
3608	025670	001003				BNE	COMEXI		;NO
3609	025672	052761	000400	000012		BIS	#MD.SER,P.MOD(R1)		;YES, SET THE SER MODIFIER
3610									
3611	025700	062701	000050		COMEXI:	ADD	#DCBSTP,R1		;SET THE POINTER TO THE NEXT SLOT
3612	025704	022701	004032			CMP	#DCBEND,R1		;ARE WE AT THE END OF THE RING ?
3613	025710	001002				BNE	110\$;NO, EXIT
3614	025712	012701	003572			MOV	#DCMDBF,R1		;YES, SET THE POINTER TO START OF RING
3615	025716	010164	000014		110\$:	MOV	R1,CNUSAV(R4)		;SAVE THE POINTER
3616									
3617	025722				EXIT:	POP	<R2,R1>		;RESTORE R1
3618	025726	000207				RTS	PC		;RETURN


```

3666 .SBTTL PORT DRIVER RECEIVE
3667 ;*****
3668 ;
3669 ;Port Driver Receive
3670 ;
3671 ;Called By : CORECV
3672 ;Inputs : URDSRG - RESPONSE descriptor ring
3673 ; UCDSRG - Command descriptor ring.
3674 ;Outputs : RESP - Number of new RESPONSEs.
3675 ;Registers Used : R1 - RESPONSE descriptor ring pointer.
3676 ;
3677 ;
3678 PDRECV:
3679 026106 PUSH <R1> ;SAVE R1
3680 026110 016401 000162 MOV URDSRG(R4),R1 ;SET R1 TO THE RESPONSE DESCRIPTOR
3681 026114 017437 000002 010566 MOV @TKSA(R4),SAERR ;SAVE THE SA FOR THE ERROR PRINTOUT
3682 026122 005737 010566 TST SAERR ;READ THE SA REGISTER
3683 026126 100003 BPL 1$ ;BRANCH IF NO ERRORS
3684 026130 052737 000003 010546 BIS #IOPDRE,IOSTAT ;SET PORT DETECTED ERROR IN I/O STATUS
3685 ;
3686 026136 006364 000010 1$: ASL SLTUSE(R4) ;SHIFT BITMAP
3687 026142 032737 000040 003516 BIT #GCSRFL,PCFLAG ;ARE WE IN GCS MODE ?
3688 026150 001403 BEQ 2$ ;NO, DO ALL RESPONSES
3689 026152 005737 003414 TST RESP ;HAVE WE GOTTEN A RESPONSE ?
3690 026156 001012 BNE 5$ ;YES, GCS MODE ALLOW ONLY 1 RESPONSE
3691 ;
3692 026160 032761 100000 000002 2$: BIT #OWN,HIADDR(R1) ;IS THE SLOT SET TO US ?
3693 026166 001006 BNE 5$ ;NO , BRANCH
3694 026170 005237 003414 INC RESP ;ADD 1 TO THE RESPONSE COUNT
3695 026174 052764 000001 000010 BIS #ITO,SLTUSE(R4) ;SET SLOT-IN-USE
3696 026202 000403 BR 10$
3697 ;
3698 026204 042764 000001 000010 5$: BIC #BIT0,SLTUSE(R4) ;ELSE CLEAR THIS SLOT-IN-USE
3699 026212 062701 000004 10$: ADD #DSPSTP,R1 ;SET THE POINTER TO THE NEXT SLOT
3700 026216 026401 000164 CMP URDEND(R4),R1 ;ARE WE AT THE END OF THE RING ?
3701 026222 001345 BNE 1$ ;NO, KEEP GOING TILL WE GET THEM ALL
3702 026224 POP <R1> ;RESTORE R1
3703 026226 000207 RTS PC ;RETURN
    
```



```

3705 .SBTTL PORT DRIVER CLEAR
3706 ;*****
3707 ;
3708 ;Port Driver Clear
3709 ;
3710 ;Called By : CDRECV
3711 ;Register Inputs: R4 - Lun block pointer.
3712 ;Registers Used : R1 - Current location in the RESPONSE descriptor ring.
3713 ;
3714 ;
3715 026230 PRTCLR::
3716 026230 PUSH <R1,R2> ;SAVE R1 AND R2
3717 026234 016401 000164 MOV URDEND(R4),R1 ;R1 = END OF RESPONSE DESCRIPTOR RING
3718 026240 016402 000162 MOV URDSRG(R4),R2 ;R2 = RESPONSE DESCRIPTOR RING
3719 026244 162702 000004 SUB #4,R2 ;BACK UP POINTER BY A LONGWORD
3720 ;
3721 026250 162701 000004 1$: SUB #4,R1 ;BACK UP POINTER BY A LONGWORD
3722 026254 020201 CMP R2,R1 ;BACKED UP PAST START OF RING?
3723 026256 001410 BEQ 20$ ;YES - SO GET OUT
3724 026260 000241 CLC ;CLEAR THE CARRY
3725 026262 006064 000010 ROR SLTUSE(R4) ;MOVE BIT0 TO CARRY BIT
3726 026266 103003 BCC 5$ ;BRANCH IF SLOT NOT USED
3727 026270 012761 100000 000002 MOV #OWN,HIADDR(R1) ;GIVE SLOT BACK TO PORT
3728 ;
3729 026276 000764 5$: BR 1$ ;LOOK FOR MORE
3730 026300 20$: POP <R2,R1> ;RESTORE R2 AND R1
3731 026304 000207 RTS PC ;RETURN
3732
    
```

```

3734 .SBTTL PORT DRIVER INITIALIZATION
3735 ;*****
3736 ;
3737 ;Port Driver Initialization
3738 ;
3739 ;Called By : CLSDRV
3740 ;Register Inputs: R4 - Lun block pointer.
3741 ;Registers Used : R1 - Current init step in process
3742 ;                 R2 - Used by the watchdog timer
3743 ;                 R3 - Initialization data table pointer
3744 ;
3745 ;
3746 026306 PRTINT::
3747 026306 PUSH <R1,R2,R3,R5> ;SAVE R1, R2, R3 AND R5
3748 026316 010174 000000 MOV R1,@TKIP(R4) ;INITIALIZE THE DRIVE
3749 026322 016437 000162 026522 MOV URDSRG(R4),INTTBL+2 ;PUT RESP DESCRIPTOR ADDRESS IN TABLE
3750 026330 012703 026520 MOV #INTTBL,R3 ;PUT THE TABLE ADDRESS INTO R3
3751 026334 012701 004000 MOV #S1,R1 ;SET UP TO BEGIN AT STEP 1
3752
3753 026340 012737 000050 010554 LOOP: MOV #40.,CNTHI ;SET UP THE TIME OUT COUNTER
3754 026346 005002 CLR R2 ;CLEAR R2
3755
3756 026350 ILOOP:
3757 026350 BREAK
026350 104422 TRAP C$BRK
3758
3759 026352 005202 INC R2 ;INCREMENT HI TIME OUT VALUE ?
3760 026354 001003 BNE 2$ ;IF NOT, BRANCH
3761 026356 005337 010554 DEC CNTHI ;ELSE, INCREMENT HI TIMEOUT
3762 026362 001444 BEQ TKERR ;GET OUT, WE'VE TIMED OUT
3763
3764 026364 037401 000002 2$: BIT @TKSA(R4),R1 ;TEST FOR STEP BIT FROM DRIVE
3765 026370 001767 BEQ ILOOP ;LOOP UNTIL SOMETHING SETS
026372 017437 000002 010566 MOV @TKSA(R4),SAERR ;SAVE THE SA FOR THE ERROR PRINTOUT
3766 026372 017437 000002 010566 TST SAERR ;CHECK FOR ERROR
3767 026400 005737 010566 BMI TKERR ;GET OUT ON ERROR
3768 026404 100433
3769 026406 012374 000002 3$: MOV (R3)+,@TKSA(R4) ;WRITE WORD FROM TABLE TO CONTROLLER
3770 026412 006301 ASL R1 ;SHIFT TO NEXT STEP
3771 026414 100351 BPL LOOP ;IF NOT A1 LAST STEP LOOP
3772
3773 026416 016402 000162 MOV URDSRG(R4),R2 ;PUT THE RESPONSE DESCRIPTOR ADD IN R2
3774 026422 016403 000156 MOV URSPBF(R4),R3 ;PUT THE RESPONSE BUFFER ADDRESS IN R3
3775 026426 010322 5$: MOV R3,(R2)+ ;PUT THE BUFF ADD IN THE DESCRIPTOR
3776 026430 005022 CLR (R2)+ ;CLEAR THE NEXT WORD
3777 026432 062703 000104 ADD #DRBSTP,R3 ;STEP TO THE NEXT BUFFER SLOT
3778 026436 026403 000160 CMP URBEND(R4),R3 ;ARE WE AT THE END OF THE BUFFER ?
3779 026442 001371 BNE 5$ ;NO, KEEP GOING
3780
3781 026444 016402 000166 MOV UCDSRG(R4),R2 ;PUT THE CMD DESCRIPTOR ADDRESS IN R2
3782 026450 012703 003572 MOV #DCMBF,R3 ;PUT THE CMD BUFFER ADDRESS IN R3
3783 026454 010322 10$: MOV R3,(R2)+ ;PUT THE BUFF ADD IN THE DESCRIPTOR
3784 026456 005022 CLR (R2)+ ;CLEAR THE NEXT WORD
3785 026460 062703 000050 ADD #DCBSTP,R3 ;STEP TO THE NEXT BUFFER SLOT
3786 026464 022703 004032 CMP #DCBEND,R3 ;ARE WE AT THE END OF THE BUFFER ?
3787 026470 001371 BNE 10$ ;NO, KEEP GOING
3788 026472 000403 BR IDONE ;ALL DONE
3789

```

```

3790 026474 012737 000006 010546 TKERR: MOV      INTERR,Iostat      ;SET UP FOR A FATAL ERROR
3791
3792 026502 005337 003402      IDONE: DEC      RSPCNT
3793 026506      POP      <R5,R3,R2,R1>      ;RESTORE THE REGISTERS
      026506 012605      MOV      (SP)+,R5      ;;POP STACK INTO R5
      026510 012603      MOV      (SP)+,R3      ;;POP STACK INTO R3
      026512 012602      MOV      (SP)+,R2      ;;POP STACK INTO R2
      026514 012601      MOV      (SP)+,R1      ;;POP STACK INTO R1
3794 026516 000207      RTS      PC      ;RETURN
3795
3796      ;INIT DATA TABLE
3797
3798 026520 111400      INTTBL: .WORD  TKINIT
3799 026522 000000      .WORD  0
3800 026524 000000      .WORD  0
3801 026526 000001      .WORD  GO

```

```

3803 .SBTTL GCS RESPONSE HANDLER
3804 ;*****
3805 ;
3806 ;GCS RESPONSE HANDLER
3807 ;
3808 ;Called By :
3809 ;Calls To :
3810 ;Register Inputs :
3811 ;
3812 ;Register Inputs :
3813 ;
3814 ;
3815 026530 GCSHDL::
3816 026530 023761 010550 000020 CMP CMSTSV,P.CMST(R1) ;ANY PROGRESS ?
3817 026536 101017 BHI 5$ ;YES, CLKEAN UP THE MESS
3818 026540 042737 000040 003516 BIC #GCSRFL,PCFLAG ;CLEAR THE GCS MODE FLAG
3819 026546 005037 003414 CLR RESP ;TAKE OFF THE RESPONSE
3820 026552 005037 003412 CLR HNDLRP ;TAKE OFF THE RESPONSE
3821 026556 112737 000002 010546 MOVB #IOHUNG,IOSTAT ;SET HUNG CONTROLLER BIT
3822 026564 004737 033734 JSR PC,CORDMP
3823 026570 000240 NOP
3824 026572 000137 027314 JMP GCSEXT ;GET OUT
3825
3826 026576 5$: PUSH <R1,R2> ;
026576 010146 MOV R1,-(SP) ;;PUSH R1 ON STACK
026600 010246 MOV R2,-(SP) ;;PUSH R2 ON STACK
3827 026602 016401 000016 MOV COLSAV(R4),R1 ;PUT THE OLD POINTER IN R1
3828 026606 162701 000004 SUB #4,R1 ;ADJUST TO INCLUDE DESCRIPTOR WORDS
3829 026612 016402 000014 MOV CNUSAV(R4),R2 ;PUT THE NEW POINTER IN R2
3830 026616 162702 000004 SUB #4,R2 ;ADJUST TO INCLUDE DESCRIPTOR WORDS
3831 026622 022701 003566 CMP #CMDBF1,R1 ;OLD POINTER AT BF1 ?
3832 026626 001407 BEQ OLD1 ;YES, GO HANDLE IT
3833 026630 022701 003636 CMP #CMDBF2,R1 ;OLD POINTER AT BF2 ?
3834 026634 001434 BEQ OLD2 ;YES, GO HANDLE IT
3835 026636 022701 003706 CMP #CMDBF3,R1 ;OLD POINTER AT BF3 ?
3836 026642 001461 BEQ OLD3 ;YES, GO HANDLE IT
3837 026644 000510 BR OLD4 ;NO, GO HANDLE BF4
3838
3839 026646 022702 003706 OLD1: CMP #CMDBF3,R2 ;NEW POINTER AT BF3 ?
3840 026652 001004 BNE 5$ ;NO, TRY AGAIN
3841 026654 004737 027316 JSR PC,EXC1A2 ;GO MOVE COMMAND 1 TO 2
3842 026660 000137 027142 JMP ADJUST ;GO ADJUST THE OLD POINTER
3843 026664 022702 003756 5$: CMP #CMDBF4,R2 ;NEW POINTER AT BF4 ?
3844 026670 001006 BNE 10$ ;NO, TRY AGAIN
3845 026672 004737 027340 JSR PC,EXC2A3 ;GO MOVE COMMAND 2 TO 3
3846 026676 004737 027316 JSR PC,EXC1A2 ;GO MOVE COMMAND 1 TO 2
3847 026702 000137 027142 JMP ADJUST ;GO ADJUST THE OLD POINTER
3848 026706 004737 027362 10$: JSR PC,EXC3A4 ;GO MOVE COMMAND 3 TO 4
3849 026712 004737 027340 JSR PC,EXC2A3 ;GO MOVE COMMAND 2 TO 3
3850 026716 004737 027316 JSR PC,EXC1A2 ;GO MOVE COMMAND 1 TO 2
3851 026722 000137 027142 JMP ADJUST ;GO ADJUST THE OLD POINTER
3852
3853 026726 022702 003756 OLD2: CMP #CMDBF4,R2 ;NEW POINTER AT BF4 ?
3854 026732 001004 BNE 5$ ;NO, TRY AGAIN
3855 026734 004737 027340 JSR PC,EXC2A3 ;GO MOVE COMMAND 2 TO 3
3856 026740 000137 027142 JMP ADJUST ;GO ADJUST THE OLD POINTER
3857 026744 022702 003566 5$: CMP #CMDBF1,R2 ;NEW POINTER AT BF1 ?

```

```

3858 026750 001006          BNE      10$          ;NO, TRY AGAIN
3859 026752 004737 027362   JSR      PC,EXC3A4    ;GO MOVE COMMAND 3 TO 4
3860 026756 004737 027340   JSR      PC,EXC2A3    ;GO MOVE COMMAND 2 TO 3
3861 026762 000137 027142   JMP      ADJUST       ;GO ADJUST THE OLD POINTER
3862 026766 004737 027404   10$:   JSR      PC,EXC4A1    ;GO MOVE COMMAND 4 TO 1
3863 026772 004737 027362   JSR      PC,EXC3A4    ;GO MOVE COMMAND 3 TO 4
3864 026776 004737 027340   JSR      PC,EXC2A3    ;GO MOVE COMMAND 2 TO 3
3865 027002 000137 027142   JMP      ADJUST       ;GO ADJUST THE OLD POINTER
3866
3867 027006 022702 003566   OLD3:  CMP      #CMDBF1,R2 ;NEW POINTER AT BF1 ?
3868 027012 001004          BNE      5$          ;NO, TRY AGAIN
3869 027014 004737 027362   JSR      PC,EXC3A4    ;GO MOVE COMMAND 3 TO 4
3870 027020 000137 027142   JMP      ADJUST       ;GO ADJUST THE OLD POINTER
3871 027024 022702 003636   5$:   CMP      #CMDBF2,R2 ;NEW POINTER AT BF2 ?
3872 027030 001006          BNE      10$          ;NO, TRY AGAIN
3873 027032 004737 027404   JSR      PC,EXC4A1    ;GO MOVE COMMAND 4 TO 1
3874 027036 004737 027362   JSR      PC,EXC3A4    ;GO MOVE COMMAND 3 TO 4
3875 027042 000137 027142   JMP      ADJUST       ;GO ADJUST THE OLD POINTER
3876 027046 004737 027316   10$:  JSR      PC,EXC1A2    ;GO MOVE COMMAND 1 TO 2
3877 027052 004737 027404   JSR      PC,EXC4A1    ;GO MOVE COMMAND 4 TO 1
3878 027056 004737 027362   JSR      PC,EXC3A4    ;GO MOVE COMMAND 3 TO 4
3879 027062 000137 027142   JMP      ADJUST       ;GO ADJUST THE OLD POINTER
3880
3881 027066 022702 003636   OLD4:  CMP      #CMDBF2,R2 ;NEW POINTER AT BF2 ?
3882 027072 001004          BNE      5$          ;NO, TRY AGAIN
3883 027074 004737 027404   JSR      PC,EXC4A1    ;GO MOVE COMMAND 4 TO 1
3884 027100 000137 027142   JMP      ADJUST       ;GO ADJUST THE OLD POINTER
3885 027104 022702 003706   5$:   CMP      #CMDBF3,R2 ;NEW POINTER AT BF3 ?
3886 027110 001006          BNE      10$          ;NO, TRY AGAIN
3887 027112 004737 027316   JSR      PC,EXC1A2    ;GO MOVE COMMAND 1 TO 2
3888 027116 004737 027404   JSR      PC,EXC4A1    ;GO MOVE COMMAND 4 TO 1
3889 027122 000137 027142   JMP      ADJUST       ;GO ADJUST THE OLD POINTER
3890 027126 004737 027340   10$:  JSR      PC,EXC2A3    ;GO MOVE COMMAND 2 TO 3
3891 027132 004737 027316   JSR      PC,EXC1A2    ;GO MOVE COMMAND 1 TO 2
3892 027136 004737 027404   JSR      PC,EXC4A1    ;GO MOVE COMMAND 4 TO 1
3893
3894 027142          ADJUST: POP      <R2,R1> ;
      027142 012602      MOV      (SP)+,R2      ;;POP STACK INTO R2
      027144 012601      MOV      (SP)+,R1      ;;POP STACK INTO R1
3895 027146 042737 000040 003516   BIC      #GCSRFL,PCFLAG ;CLEAR THE GCS MODE FLAG
3896 027154 016137 000020 010550   MOV      P,CMST(R1),CMSTSV ;PUT THE CMD STATUS INTO THE SAVE LOC
3897 027162 005037 003414          CLR      RESP         ;TAKE OFF THE RESPONSE
3898 027166 005037 003412          CLR      HNDLRP       ;TAKE OFF THE RESPONSE
3899 027172 005364 000006          DEC      CMDSEQ(R4)    ;ADJUST THE CMDSEQ NUMBER BACK 1
3900 027176 042737 100000 010546   BIC      #NURESP,IOSTAT ;CLEAR THE NEW RESPONSE FLAG IN IOSTAT
3901 027204 105237 010571          INCB     CRDLIM        ;ADD 1 TO THE CREDIT LIMIT
3902
3903 027210 062764 000050 000016   ADD      #DCBSTP,COLSAV(R4) ;ADJUST THE OLD COMMAND POINTER
3904 027216 022764 004032 000016   CMP      #DCBEND,COLSAV(R4) ;IS IT AT THE END OF THE RING ?
3905 027224 001003          BNE      5$          ;NO, BRANCH
3906 027226 012764 003572 000016   MOV      #DCMDBF,COLSAV(R4) ;YES, SET IT TO THE TOP OF THE RING
3907
3908 027234 062705 000050          5$:   ADD      #DCBSTP,R5     ;ADJUST THE OLD COMMAND POINTER
3909 027240 022705 004032          CMP      #DCBEND,R5     ;IS IT AT THE END OF THE RING ?
3910 027244 001002          BNE      10$          ;NO, BRANCH
3911 027246 012705 003572          MOV      #DCMDBF,R5     ;YES, SET IT TO THE TOP OF THE RING
3912

```

```

3913 027252 062764 000104 000022 10$: ADD #DRBSTP,ROLSAV(R4) ;ADJUST THE OLD RESPONSE POINTER
3914 027260 026464 000160 000022 CMP URBEND(R4),ROLSAV(R4) ;IS IT AT THE END OF THE BUFFER ?
3915 027266 001003 BNE 15$ ;NO, KEEP GOING
3916 027270 016464 000156 000022 MOV URSPBF(R4),ROLSAV(R4) ;YES, SET IT TO BEGINNING OF THE BUFFER
3917
3918 027276 062701 000104 15$: ADD #DRBSTP,R1 ;ADJUST R1
3919 027302 026401 000160 CMP URBEND(R4),R1 ;IS IT AT THE END OF THE BUFFER ?
3920 027306 001002 BNE GCSEXT ;NO, GET OUT
3921 027310 016401 000156 MOV URSPBF(R4),R1 ;YES, SET IT TO BEGINNING OF THE BUFFER
3922
3923 027314 GCSEXT:
3924 027314 000207 RTS PC ;RETURN
3925
3926
3927 027316 012701 003566 EXC1A2: MOV #CMDBF1,R1 ;SET R1 TO BF1
3928 027322 012702 003636 MOV #CMDBF2,R2 ;SET R2 TO BF2
3929 027326 012122 5$: MOV (R1)+,(R2)+ ;MOV BF1 CONTENTS TO BF2
3930 027330 022701 003636 CMP #CMDBF2,R1 ;HAVE WE MOVED THEM ALL
3931 027334 001374 BNE 5$ ;NO, KEEP MOVING IT
3932 027336 000207 RTS PC ;YES, GET OUT
3933
3934 027340 012701 003636 EXC2A3: MOV #CMDBF2,R1 ;SET R1 TO BF2
3935 027344 012702 003706 MOV #CMDBF3,R2 ;SET R2 TO BF3
3936 027350 012122 5$: MOV (R1)+,(R2)+ ;MOV BF2 CONTENTS TO BF3
3937 027352 022701 003706 CMP #CMDBF3,R1 ;HAVE WE MOVED THEM ALL
3938 027356 001374 BNE 5$ ;NO, KEEP MOVING IT
3939 027360 000207 RTS PC ;YES, GET OUT
3940
3941 027362 012701 003706 EXC3A4: MOV #CMDBF3,R1 ;SET R1 TO BF3
3942 027366 012702 003756 MOV #CMDBF4,R2 ;SET R2 TO BF4
3943 027372 012122 5$: MOV (R1)+,(R2)+ ;MOV BF3 CONTENTS TO BF4
3944 027374 022701 003756 CMP #CMDBF4,R1 ;HAVE WE MOVED THEM ALL
3945 027400 001374 BNE 5$ ;NO, KEEP MOVING IT
3946 027402 000207 RTS PC ;YES, GET OUT
3947
3948 027404 012701 003756 EXC4A1: MOV #CMDBF4,R1 ;SET R1 TO BF4
3949 027410 012702 003566 MOV #CMDBF1,R2 ;SET R2 TO BF1
3950 027414 012122 5$: MOV (R1)+,(R2)+ ;MOV BF4 CONTENTS TO BF1
3951 027416 022702 003636 CMP #CMDBF2,R2 ;HAVE WE MOVED THEM ALL
3952 027422 001374 BNE 5$ ;NO, KEEP MOVING IT
3953 027424 000207 RTS PC ;YES, GET OUT

```

```

3955 .SBTTL RESPONSE HANDLER
3956 ;*****
3957 ;
3958 ;RESPONSE HANDLER
3959 ;
3960 ;Called By      : CMDSEQ
3961 ;Calls To       : ERRDEI, ERRDEL, ERRDEC, CMPDAT, DQCMD
3962 ;Register Inputs: R1 - UNIT NUMBER
3963 ;               : R4 - LUN BLOCK POINTER
3964 ;Register Inputs: R3 - POINTER TO CURRENT RESPONSE PACKET
3965 ;
3966 ;
3967 027426 RSPHDL::
3968 027426      PUSH    <R3>
          027426 010346      MOV     R3,-(SP)      ;;PUSH R3 ON STACK
3969 027430 005037 003406      CLR     RESPON      ;0 HERE TELLS CMDSEQ ALL'S OKAY
3970 027434 105737 010546      TSTB   IOSTAT      ;DID WE HAVE I/O TYPE FAILURE?
3971 027440 001403              BEQ     5$          ;BRANCH AROUND IF NOT
3972 027442 004737 030774      JSR    PC,ERRDEI   ;ELSE DECODE AND PRINT IT
3973 027446 000501              BR     60$         ;GET OUT NOW
3974
3975 027450 016403 000022      5$:    MOV     ROLSAV(R4),R3      ;GET OLD RESPONSE BUFFER POINTER
3976 027454 005737 003412      TST    HNDLRP      ;DID WE HAVE ANY RESPONSES ?
3977 027460 001474              BEQ     60$         ;NO, GET OUT OF HERE
3978 027462 032763 000200 000010 10$:    BIT     #OP.END,P.OPCD(R3)  ;IS IT AN END PACKET?
3979 027470 001003              BNE    15$         ;YES, BRANCH
3980 027472 004737 031236      JSR    PC,ERRDEL   ;GO HANDLE ERROR LOG PACKET
3981 027476 000453              BR     50$         ;SEE IF THERE'S MORE RESPONSES
3982
3983 027500 005763 000012      15$:   TST    P.STS(R3)    ;WAS STATUS "NORMAL"?
3984 027504 001413              BEQ     25$         ;YES - BRANCH
3985 027506 022763 000022 000012      CMP     #ST.SEX,P.STS(R3)  ;IS IT SERIOUS EXCEPTION STATUS?
3986 027514 001004              BNE    20$         ;NO - GO HANDLE THE ERROR
3987 027516 052764 000004 000026      BIS     #NOTALY,LUNFLG(R4) ;SET THE NO-TALLY FLAG
3988 027524 000436              BR     45$         ;YES, GO DE-QUE THE COMMANDS
3989 027526 004737 030304      20$:   JSR    PC,ERRDEC   ;GO HANDLE ERROR STATUS
3990 027532 000403              BR     30$
3991 027534 000241      25$:   CLC
3992 027536 006164 000030      ROL    LEOTFL(R4)  ;ROTATE THE CARRY INTO THE LEOT FLAG
3993
3994 027542 032761 000001 003350 30$:   BIT     #AVB,DRINUS(R1)  ;HAVE WE DROPPED THE UNIT?
3995 027550 001440              BEQ     60$         ;YES - GET OUT
3996 027552 022763 000241 000010      CMP     #OP.END!OP.RD,P.OPCD(R3) ;DID WE READ THIS TIME?
3997 027560 001020              BNE    45$         ;NO - SKIP DATA COMPARE
3998 027562 022737 000005 002114      CMP     #5,L$TEST      ;ARE WE IN TEST 5?
3999 027570 001003              BNE    35$         ;YES - SKIP DATA COMPARE
4000 027572 105737 002235      TSTB   TSCMP      ;DO DATA COMPARES IN TEST 5 ?
4001 027576 001403              BEQ     40$         ;NO, SKIP DATA COMPARE
4002 027600 004737 032552      35$:   JSR    PC,CMPDAT   ;DO COMPARE DATA
4003 027604 000406              BR     45$
4004 027606 132737 000001 002236 40$:   BITB   #BITO,DMPBUF   ;SHOULD WE DUMP RDBUF?
4005 027614 001402              BEQ     45$         ;NO - BRANCH
4006 027616 004737 034066      JSR    PC,BUFDMP
4007
4008 027622 004737 027662      45$:   JSR    PC,DQCMD    ;DEQUEUE THE COMMAND
4009 027626 062703 000104      50$:   ADD     #DRBSTP,R3    ;ADJUST POINTER TO NEXT PACKET
4010 027632 026403 000160      CMP    URBEND(R4),R3 ;END OF RESPONSE BUFFER?

```

```

4011 027636 001002          BNE 55$          ;NO - BRANCH AROUND
4012 027640 016403 000156  MOV URSPBF(R4),R3 ;PUT POINTER AT BEGINNING OF BUFFER
4013
4014 027644 005337 003412 55$: DEC HNDLRP      ;DECREMENT RESPONSE COUNTER
4015 027650 001304          BNE 10$          ;GO HANDLE ANOTHER ONE
4016
4017 027652 010364 000022 60$: MOV R3,ROLSAV(R4) ;SAVE OLD RESPONSE BUFFER POINTER
4018 027656          POP <R3>
      027656 012603      MOV (SP)+,R3      ;;POP STACK INTO R3
4019 027660 000207          RTS PC

```



```

4021 .SBTTL DE-QUEUE COMMAND
4022 ;*****
4023 ;
4024 ; DE-QUEUE COMMAND
4025 ;
4026 ;Called By      : RSPHDL
4027 ;Calls To       : LGSTAT
4028 ;Register Inputs : R2 - OLD POINTER TO PROGRAM COMMAND RING
4029 ;Register Outputs: R2 - UPDATED
4030 ;
4031 ;
4032 027662 DQCMD::
4033 027662 004737 027720 JSR PC, LGSTAT ;CALL LOG STATS
4034 027666 062702 000014 ADD #PCBSTP, R2 ;ADJUST THE OLD COMMAND POINTER
4035 027672 022702 003344 CMP #PCBEND, R2 ;ARE WE AT THE END OF THE BUFFER ?
4036 027676 001002 BNE 5$ ;NO, KEEP GOING
4037 027700 012702 003264 MOV #PCMDBF, R2 ;YES, SET IT BACK TO THE TOP
4038 ;
4039 027704 005337 003402 5$: DEC RSPCNT ;DECREMENT THE RESPONSE COUNTER
4040 027710 012737 177777 010550 MOV # 1, CMSTSV ;RESET THE GCS PROGRFSS COUNT
4041 027716 000207 RTS PC ;RETURN
4042 ;

```

```

4044 .SBTTL LOG STATISTICS
4045 ;*****
4046 ;
4047 ; LOG STATISTICS
4048 ;
4049 ;Called By      : DQCMD
4050 ;Register Inputs : R2 - OLD PROGRAM COMMAND POINTER
4051 ;               : R4 - LUN BLOCK POINTER
4052 ;
4053 ;
4054 LGSTAT::
4055 027720 032764 000004 000026 BIT #NOTALY,LUNFLG(R4) ;IS THIS NOT TO BE TALLIED ?
4056 027726 001162 BNE 10$ ;YES, GET OUT
4057 027730 122762 000040 000000 CMPB #ACC,CMD(R2) ;SEE IF COMMAND A READ OR WRITE
4058 027736 103556 BLO 10$ ;NO, EXIT SUBROUTINE
4059 027740 105762 000000 TSTB CMD(R2) ;IS IT A NULL ?
4060 027744 001553 BEQ 10$ ;YES, EXIT SUBROUTINE
4061 027746 122762 000020 000000 CMPB #WR,CMD(R2) ;IS IT A WRITE ?
4062 027754 001056 BNE 5$ ;NO, HANDLE READ
4063 027756 066264 000002 000120 ADD ITMOFF(R2),WRBYT1(R4) ;YES, ADD THE BYTES WRITTEN TO TOTAL
4064 027764 026427 000120 001747 1$: CMP WRBYT1(R4),#999. ;IS IT HIGER THAN 999. ?
4065 027772 003406 BLE 2$ ;BRANCH IF IT'S NOT
4066 027774 162764 001750 000120 SUB #1000.,WRBYT1(R4) ;SUBTRACT 1000. FROM THE LOWER ORDER WORD
4067 030002 005264 000122 INC WRBYT2(R4) ;INCREMENT THE SECOND WORD
4068 030006 000766 BR 1$
4069 030010 026427 000122 001747 2$: CMP WRBYT2(R4),#999. ;IS IT HIGER THAN 999. ?
4070 030016 003406 BLE 3$ ;BRANCH IF IT'S NOT
4071 030020 162764 001750 000122 SUB #1000.,WRBYT2(R4) ;SUBTRACT 1000. FROM THE LOWER ORDER WORD
4072 030026 005264 000124 INC WRBYT3(R4) ;INCREMENT THE SECOND WORD
4073 030032 000766 BR 2$
4074 030034 026427 000124 001747 3$: CMP WRBYT3(R4),#999. ;IS IT HIGER THAN 999. ?
4075 030042 003406 BLE 4$ ;BRANCH IF IT'S NOT
4076 030044 162764 001750 000124 SUB #1000.,WRBYT3(R4) ;SUBTRACT 1000. FROM THE LOWER ORDER WORD
4077 030052 005264 000126 INC WRBYT4(R4) ;INCREMENT THE SECOND WORD
4078 030056 000766 BR 3$
4079 030060 026427 000126 001747 4$: CMP WRBYT4(R4),#999. ;IS IT HIGER THAN 999. ?
4080 030066 003502 BLE 10$ ;BRANCH IF IT'S NOT
4081 030070 005064 000120 CLR WRBYT1(R4) ;
4082 030074 005064 000122 CLR WRBYT2(R4) ;
4083 030100 005064 000124 CLR WRBYT3(R4) ;
4084 030104 005064 000126 CLR WRBYT4(R4) ;
4085 030110 000471 BR 10$ ;EXIT
4086 ;
4087 030112 022763 000016 000012 5$: CMP #ST.TM,P.STS(R3) ;WAS THIS A TAPE MARK DURING READ
4088 030120 001465 BEQ 10$ ;YES, GET OUT
4089 030122 022763 000010 000012 CMP #10,P.STS(R3) ;WAS THIS A DATA ERROR DURING READ
4090 030130 001461 BEQ 10$ ;YES, GET OUT
4091 030132 022763 000350 000012 CMP #350,P.STS(R3) ;WAS THIS A DATA ERROR DURING READ
4092 030140 001455 BEQ 10$ ;YES, GET OUT
4093 030142 066364 000040 000110 ADD P.TRBC(R3),RDBYT1(R4) ;YES, ADD THE BYTES WRITTEN TO TOTAL
4094 030150 026427 000110 001747 6$: CMP RDBYT1(R4),#999. ;IS IT HIGER THAN 999. ?
4095 030156 003406 BLE 7$ ;BRANCH IF IT'S NOT
4096 030160 162764 001750 000110 SUB #1000.,RDBYT1(R4) ;SUBTRACT 1000. FROM THE LOWER ORDER WORD
4097 030166 005264 000112 INC RDBYT2(R4) ;INCREMENT THE SECOND WORD
4098 030172 000766 BR 6$
4099 030174 026427 000112 001747 7$: CMP RDBYT2(R4),#999. ;IS IT HIGER THAN 999. ?
4100 030202 003406 BLE 8$ ;BRANCH IF IT'S NOT

```

4101	030204	162764	001750	000112		SUB	#1000.,RDBYT2(R4)		;SUBTRACT 1000. FROM THE LOWER ORDER WORD
4102	030212	005264	000114			INC	RDBYT3(R4)		;INCREMENT THE SECOND WORD
4103	030216	000766				BR	7\$		
4104	030220	026427	000114	001747	8\$:	CMP	RDBYT3(R4),#999.		;IS IT HIGER THAN 999. ?
4105	030226	003406				BLE	9\$;BRANCH IF IT'S NOT
4106	030230	162764	001750	000114		SUB	#1000.,RDBYT3(R4)		;SUBTRACT 1000. FROM THE LOWER ORDER WORD
4107	030236	005264	000116			INC	RDBYT4(R4)		;INCREMENT THE SECOND WORD
4108	030242	000766				BR	8\$		
4109	030244	026427	000116	001747	9\$:	CMP	RDBYT4(R4),#999.		;IS IT HIGER THAN 999. ?
4110	030252	003410				BLE	10\$;BRANCH IF IT'S NOT
4111	030254	005064	000110			CLR	RDBYT1(R4)		:
4112	030260	005064	000112			CLR	RDBYT2(R4)		:
4113	030264	005064	000114			CLR	RDBYT3(R4)		:
4114	030270	005064	000116			CLR	RDBYT4(R4)		:
4115	030274	042764	000004	000026	10\$:	BIC	#NOTALY,LUNFLG(R4)		;CLEAR THE NO-TALLY FLAG BEFORE EXITING
4116	030302	000207				RTS	PC		;RETURN


```

4167
4168 030540 022705 000016      10$:  CMP      #ST.TM,R5      ;IS IT A TAPE MARK ERROR ?
4169 030544 001061                BNE      15$          ;BRANCH IF IT ISN'T
4170 030546 052764 000002 000026  BIS      #SEREXC,LUNFLG(R4) ;SET SERIOUS EXCEPTION
4171 030554 000261                SEC
4172 030556 006164 000030                ROL      LEOTFL(R4)
4173 030562 042764 177774 000030  BIC      #177774,LEOTFL(R4)
4174 030570 022764 000003 000030  CMP      #3,LEOTFL(R4)
4175 030576 001032                BNE      13$          ;
4176 030600 052761 000004 003350  BIS      #EOT,DRINUS(R1)
4177 030606 005237 003526                INC      UEOT          ;INC THE EOT FLAG
4178 030612                PUSH     <R1>         ;SAVE R1
4179 030614 006001                ROR      R1           ;DIVIDE R1 BY 2
4180 030616                PRINTF  #UNTLOT,R1    ;PRINT UNIT AT EOT MESSAGE
      030616 010146                MOV      R1,-(SP)
      030620 012746 020312                MOV      #UNTLOT,-(SP)
      030624 012746 000002                MOV      #2,-(SP)
      030630 010600                MOV      SP,R0
      030632 104417                TRAP    C$PNTF
      030634 062706 000006                ADD      #6,SP
4181 030640                POP      <R1>         ;RESTORE R1
4182 030642 163737 003400 003402  SUB      CMDCNT,RSPCNT ;SET RESPONSE COUNT TO NUMBER OUT
4183 030650 005037 003400                CLR      CMDCNT      ;ISSUE NO MORE COMMANDS
4184 030654 005037 003406                CLR      RESPON      ;MAKE SURE WE
4185 030660 000137 031232                JMP      EDCEXT       ;GET OUT
4186
4187 030664 132763 000010 000011 13$:  BITB    #EF.EOT,P.FLGS(R3) ;IS THE TAPE MARK AT EOT ?
4188 030672 001402                BEQ      14$          ;NO, KEEP ON GOING
4189 030674 000137 031232                JMP      EDCEXT       ;YES, GET OUT
4190 030700 012737 000010 003442 14$:  MOV      #TMB,WRKMSK  ;SET THE ERROR BIT IN THE MASK
4191 030706 000425                BR       MSKTST       ;GO TEST IF IT'S O.K.
4192
4193 030710 022705 000020      15$:  CMP      #ST.RDT,R5    ;IS IT A RECORD DATA TRUNCATED ERROR ?
4194 030714 001007                BNE      20$          ;BRANCH IF IT ISN'T
4195 030716 000241                CLC
4196 030720 006164 000030                ROL      LEOTFL(R4)  ;ROTATE THE CARRY INTO THE LEOT FLAG
4197 030724 012737 000002 003442  MOV      #RDTB,WRKMSK ;SET THE ERROR BIT IN THE MASK
4198 030732 000413                BR       MSKTST       ;GO TEST IF IT'S O.K.
4199
4200 030734 022705 000023      20$:  CMP      #ST.LED,R5    ;IS IT A LOGICAL END OF TAPE ERROR ?
4201 030740 001002                BNE      25$          ;BRANCH IF IT ISN'T
4202 030742 000137 031232                JMP      EDCEXT       ;GET OUT IF LEOT DETECTED
4203
4204 030746 022705 000004      25$:  CMP      #ST.AVL,R5    ;IS IT A UNIT AVAILABLE ERROR ?
4205 030752 001020                BNE      ERREXT       ;BRANCH IF IT ISN'T
4206 030754 012737 000040 003442  MOV      #AVLB,WRKMSK ;SET THE ERROR BIT IN THE MASK
4207
4208 030762 033737 003442 003440  MSKTST: BIT      WRKMSK,TSTMSK ;IS IT AN ACCEPTABLE ERROR ?
4209 030770 001120                BNE      EDCEXT       ;GET OUT IF IT IS
4210 030772 000410                BR       ERREXT       ;OTHERWISE PRINT THE ERROR
4211
4212 030774                ERRDEI::
4213 030774                PUSH     <R5>         ;SAVE R5
4214 030776 113705 010546                MOV      IOSTAT,R5    ;PUT THE I/O ERROR CODE INTO R5
4215 031002 012737 010572 003530  MOV      #IOERTB,R8   ;SET THE ERROR TABLE ADDRESS IN R8
4216 031010 042705 177770                BIC      #177770,R5   ;CLEAR OFF ALL UNWANTED BITS
4217

```

4218	031014	005305				ERREXT: DEC	R5	;SUBTRACT 1 FROM R5
4219	031016	006305				ASL	R5	;MULTIPLY R5 BY 10(8)
4220	031020	006305				ASL	R5	:
4221	031022	006305				ASL	R5	:
4222	031024	063705	003530			ADD	R8,R5	;ADD THE TABLE ADDRESS TO R5
4223	031030					PUSH	<R3>	
4224	031032	012703	013262			MOV	#L\$ERRTBL,R3	;SET R3 TO THE GENERIC ERROR TABLE
4225	031036	012523				MOV	(R5)+,(R3)+	;MOVE ERROR TABLE CONTENTS
4226	031040	012523				MOV	(R5)+,(R3)+	;MOVE ERROR TABLE CONTENTS
4227	031042	012523				MOV	(R5)+,(R3)+	;MOVE ERROR TABLE CONTENTS
4228	031044	011513				MOV	(R5),(R3)	;MOVE ERROR TABLE CONTENTS
4229	031046					POP	<R3>	
4230								
4231	031050	022762	000010	000010		CMP	#EV.LGP,XFERST(R2)	;IS IS A LONG GAP ENCOUNTERED ?
4232	031056	001031				BNE	ERTLY	;NO, KEEP GOING
4233	031060	112737	000001	013262		MOVB	#DEVFAT,ERRTYP	;YES, DROP THE UNIT
4234	031066	004737	033734			JSR	PC,CORDMP	;;;GO DO IT
4235	031072	000240				NOP		;;;
4236	031074	122762	000020	000000		CMPB	#WR,CMD(R2)	;IS IT A WRITE ?
4237	031102	001004				BNE	1\$;NO TRY AGAIN
4238	031104	112737	000060	013263		MOVB	#H1WRIT,ERRTYP+1	;AND TALLY THE HARD WRITE ERROR
4239	031112	000413				BR	ERTLY	;GO TALLY THE ERROR
4240	031114	122762	000100	000000	1\$:	CMPB	#WTM,CMD(R2)	;IS IT A WRITE TAPE MARK ?
4241	031122	001004				BNE	5\$;NO, MUST BE A READ
4242	031124	112737	000060	013263		MOVB	#H1WRIT,ERRTYP+1	;AND TALLY THE HARD WRITE ERROR
4243	031132	000403				BR	ERTLY	;TALLY THE ERROR
4244	031134	112737	000054	013263	5\$:	MOVB	#H1READ,ERRTYP+1	;TALLY THE HARD READ ERROR
4245	031142	004737	032036		ERTLY:	JSR	PC,ERTLY	;TALLY THE ERROR
4246	031146	105037	013263			CLRB	ERRTYP+1	;CLEAR UPPER BYTE
4247	031152	105737	002224			TSTB	SOERRP	;ARE SOFT ERRORS ENABLED ?
4248	031156	001004				BNE	6\$;YES, GO PRINT THE ERROR
4249	031160	122737	000003	013262		CMPB	#SOFT,L\$ERRTBL	;IS IT A SOFT ERROR ?
4250	031166	001402				BEQ	8\$;YES, DON'T PRINT IT
4251	031170	004737	032536		6\$:	JSR	PC,PRIERR	;GO PRINT THE ERROR
4252								
4253	031174	132737	000001	002231	8\$:	BITB	#BITO,DMPFLG	;SHOULD WE DUMP PROGRAM TABLES?
4254	031202	001403				BEQ	10\$;NO - BRANCH
4255	031204	004737	033734			JSR	PC,CORDMP	;GO DO IT
4256	031210	000240				NOP		
4257	031212	022737	000001	013262	10\$:	CMP	#DEVFAT,ERRTYP	;IS IT A FATAL ERROR ?
4258	031220	001004				BNE	EDCEXT	;NO EXIT
4259	031222	010100				MOV	R1,R0	;MOVE UNIT # * 2 TO R0
4260	031224	006000				ROR	R0	;DIVIDE BY 2
4261	031226	004737	040442			JSR	PC,DROPUN	;DROP DRIVE FROM TESTING
4262	031232				EDCEXT:	POP	<R5>	;RESTORE REGISTERS
4263	031234	000207				RTS	PC	;RETURN
4264								

```

4266 .SBTTL ERROR LOG DECODE
4267 ;*****
4268 ;
4269 ; ERROR LOG DECODE
4270 ;
4271 ;Called By :
4272 ;Calls To :
4273 ;Inputs :
4274 ;Outputs :
4275 ;Register Inputs :
4276 ;Register Outputs:
4277
4278 031236 ERRDEL::
4279 031236 PUSH <R3,R5>
4280 031242 116237 000000 003536 MOVB CMD(R2),R11 ;GET THE COMMAND PRIMITIVE FOR LATER USE
4281 031250 042737 177407 003536 BIC #177407,R11 ;GET JUST THE ROOT PRIMITIVE
4282 031256 PUSH <R1> ;SAVE R1
4283 031260 122763 000005 000010 CMPB #FM.TPE,L.FMT(R3) ;TAPE TRANSFER ERROR LOG?
4284 031266 001420 BEQ 5$ ;YES, DECODE IT
4285 031270 122763 000000 000010 CMPB #FM.CNT,L.FMT(R3) ;CONTROLLER ERROR LOG?
4286 031276 001004 BNE 1$ ;NO, SEE WHAT IT IS
4287 031300 012705 011132 MOV #CNTLR,R5 ;PRINT PACKET
4288 031304 000137 031724 JMP PRTEXT ;GET OUT
4289 031310 122763 000001 000010 1$: CMPB #FM.BAD,L.FMT(R3) ;HOST MEMORY ACCESS ERROR LOG?
4290 031316 001004 BNE 5$ ;NO, GET OUT
4291 031320 012705 011142 MOV #BADERL,R5 ;SET UP TO PRINT HOST MEM ACC ERL
4292 031324 000137 031724 JMP PRTEXT ;GO PRINT IT
4293 031330 016337 000012 003556 5$: MOV L.EVNT(R3),EVENT ;GET A COPY OF THE EVENT CODE
4294 031336 042737 177740 003556 BIC #177740,EVENT ;GET RID OF THE SUBCODES
4295 031344 022737 000007 003556 CMP #ST.CMP,EVENT ;IS IT A COMPARE ERROR ?
4296 031352 001004 BNE 10$ ;NO, KEEP GOING
4297 031354 012705 011152 MOV #CMPERL,R5 ;SET UP TO PRINT ERROR
4298 031360 000137 031724 JMP PRTEXT ;GO PRINT THE ERROR
4299 031364 022737 000013 003556 10$: CMP #ST.DRV,EVENT ;IS IT A DRIVE ERROR ?
4300 031372 001004 BNE 15$ ;NO, KEEP GOING
4301 031374 012705 011162 MOV #DRVERL,R5 ;SET UP TO PRINT ERROR
4302 031400 000137 031724 JMP PRTEXT ;GO PRINT THE ERROR
4303 031404 022737 000012 003556 15$: CMP #ST.CNT,EVENT ;IS IT A CONTROLLER ERROR ?
4304 031412 001004 BNE DATAEL ;NO, MUST BE A DATA ERROR
4305 031414 012705 011172 MOV #CNTERL,R5 ;SET UP TO PRINT ERROR
4306 031420 000137 031724 JMP PRTEXT ;GO PRINT THE ERROR
4307 031424 022763 000010 000012 DATAEL: CMP #EV.LGP,L.EVNT(R3) ;IS IT A LONG GAP ENCOUNTERED ?
4308 031432 001004 BNE 1$ ;NO, KEEP TRYING
4309 031434 012705 011202 MOV #LGPERL,R5 ;SET UP TO PRINT LGE ERROR
4310 031440 000137 031724 JMP PRTEXT ;GO PRINT IT
4311 031444 012701 003360 1$: MOV #UNITBL,R1 ;PUT THE TEMP TABLE ADDRESS IN R1
4312 031450 116361 000055 000000 MOVB L.TRK(R3),TRK(R1) ;PUT THE TRACK NUMBER IN TEMP
4313 031456 016361 000056 000002 MOV L.PBLK(R3),PHB(R1) ;PHYSICAL BLOCK NUMBER TO TEMP
4314 031464 022737 000020 003536 CMP #WR,R11 ;WAS IT A WRITE COMMAND ?
4315 031472 001404 BEQ WRERL ;GO HANDLE WRITE ERROR
4316 031474 022737 000100 003536 CMP #WTM,R11 ;IS IT A WRITE TAPE MARK COMMAND ?
4317 031502 001026 BNE RDERL ;GO HANDLE READ ERROR LOG
4318 031504 022763 000150 000012 WRERL: CMP #EV.COR,L.EVNT(R3) ;IS IT A SOFT ERROR ?
4319 031512 001005 BNE 1$ ;NO, INCREMENT HARD ERROR
4320 031514 105261 000004 INCB SWR(R1) ;YES, INCREMENT SOFT WRITE ERROR
4321 031520 012705 011232 MOV #CORERL,R5 ;SET UP TO PRINT ERROR
4322 031524 000413 BR 10$ ;KEEP GOING
    
```

4323	031526	105261	000006		1\$:	INCB	HWR(R1)		;INCREMENT HARD WRITE ERROR
4324	031532	022763	000050	000012		CMP	#EV.DST,L.EVNT(R3)		;IS IT A DATA SYNC TIMEOUT ?
4325	031540	001003				BNE	5\$;NO, KEEP TRYING
4326	031542	012705	011222			MOV	#WDSTEL,R5		;SET UP TO PRINT DST ERROR
4327	031546	000402				BR	10\$;GO PRINT IT
4328	031550	012705	011242		5\$:	MOV	#UWEERL,R5		;SET UP TO PRINT ERROR
4329	031554	000137	031720		10\$:	JMP	SRTBL		;GO SORT THE ERROR AND PRINT
4330	031560	022763	000350	000012	RDERL:	CMP	#EV.URE,L.EVNT(R3)		;IS IT A HARD READ ERROR ?
4331	031566	001006				BNE	1\$;NO, HANDLE SOFT READ
4332	031570	105261	000007			INCB	HRD(R1)		;INC THE HARD READ ERROR
4333	031574	012705	011262			MOV	#UREERL,R5		;SET UP TO PRINT ERROR
4334	031600	000137	031720			JMP	SRTBL		;GO SORT AND PRINT IT
4335	031604	022763	000050	000012	1\$:	CMP	#EV.DST,L.EVNT(R3)		;IS IT A DATA SYNC TIMEOUT ?
4336	031612	001006				BNE	5\$;NO, KEEP TRYING
4337	031614	105261	000007			INCB	HRD(R1)		;INC THE HARD READ ERROR
4338	031620	012705	011212			MOV	#RDSTEL,R5		;SET UP TO PRINT DST ERROR
4339	031624	000137	031720			JMP	SRTBL		;GO SORT AND PRINT IT
4340	031630	105261	000005		5\$:	INCB	SRD(R1)		;INC THE SOFT READ ERROR
4341	031634	105763	000042			TSTB	L.LVL(R3)		;WAS IT RECOVERED BY ECC ?
4342	031640	001025				BNE	20\$;NO, KEEP TRYING
4343	031642	122763	000036	000052		CMPB	#ECCBC,L.STS(R3)		;WAS THE ERROR ON AN ECC BLOCK ?
4344	031650	001003				BNE	10\$;NO TRY AGAIN
4345	031652	012705	011272			MOV	#ECBERL,R5		;SET UP TO PRINT ERROR
4346	031656	000420				BR	SRTBL		;GO SORT AND PRINT ERROR
4347	031660	122763	000030	000052	10\$:	CMPB	#ECCTC,L.STS(R3)		;WAS ERROR ON A TAPE MARK ?
4348	031666	001003				BNE	15\$;TRY AGAIN
4349	031670	012705	011302			MOV	#ECTERL,R5		;SET UP TO PRINT ERROR
4350	031674	000411				BR	SRTBL		;GO SORT AND PRINT ERROR
4351	031676	122763	000026	000052	15\$:	CMPB	#ECCDC,L.STS(R3)		;WAS ERROR ON DATA BLOCK ?
4352	031704	001003				BNE	20\$;NO, TRY AGAIN
4353	031706	012705	011312			MOV	#ECDERL,R5		;SET UP TO PRINT ERROR
4354	031712	000402				BR	SRTBL		;GO SORT AND PRINT ERROR
4355	031714	012705	011252		20\$:	MOV	#RTYERL,R5		;SET UP RETRY RECOVERED
4356	031720	004737	032162		SRTBL:	JSR	PC,TBLSRT		;GO TALLY ERROR IN MEDIA TABLE
4357	031724	116237	000000	003536	PRTEXT:	MOVB	CMD(R2),R11		;GET THE COMMAND PRIMITIVE FOR LATER USE
4358	031732	012701	013262			MOV	#L\$ERRTBL,R1		;R1 = SUPERVISORS ERROR TABLE
4359	031736	012521				MOV	(R5)+,(R1)+		;COPY PROGRAM'S ERROR TABLE
4360	031740	012521				MOV	(R5)+,(R1)+		; TO SUPERVISOR'S
4361	031742	012521				MOV	(R5)+,(R1)+		; ERROR
4362	031744	011511				MOV	(R5),(R1)		; TABLE
4363	031746					POP	<R1>		;RESTORE R1
4364	031750	004737	032120			JSR	PC,ETLLY		;TALLY ERROR FIRST
4365	031754	105037	013263			CLRB	ERRTYP+1		;DISCARD INFO USED BY ERRTLLY
4366	031760	105737	002224			TSTB	SOERRP		;ARE SOFT ERRORS ENABLED ?
4367	031764	001017				BNE	1\$;YES, GO PRINT THE ERROR
4368	031766	122737	000003	013262		CMPB	#SOFT,L\$ERRTBL		;IS IT A SOFT ERROR ?
4369	031774	001013				BNE	1\$;NO, PRINT IT
4370	031776	022737	000020	003536		CMP	#WR,R11		;IS IT A WRITE ?
4371	032004	001411				BEQ	EDLEXT		;DON'T PRINT IT
4372	032006	022737	000100	003536		CMP	#WTM,R11		;IS IT A WRITE TAPE MARK ?
4373	032014	001405				BEQ	EDLEXT		;DON'T PRINT IT
4374	032016	105737	002225			TSTB	RDSOER		;ARE WE PRINTING SOFT READ ERRORS ?
4375	032022	001402				BEQ	EDLEXT		;NO, DON'T PRINT IT
4376	032024	004737	032536		1\$:	JSR	PC,PRIERR		;GO PRINT IT
4377	032030				EDLEXT:	POP	<R5,R3>		
4378	032034	000207				RTS	PC		


```

4380 .SBTTL ERROR TALLY
4381 ;*****
4382 ;
4383 ; ERROR TALLY
4384 ;
4385 ;Called By      : ERRDEC, ERRDEI, ERDEL
4386 ;
4387
4388 ERRPLY::
4389 032036          PUSH    <R1>                ;SAVE R1
4390 032040 122737 000106 013263          CMPB   #NOERR,ERRTYP+1          ;IS IT A STATUS ERROR ?
4391 032046 001422                                BEQ    15$                      ;YES, GET OUT
4392 032050 113701 013263          MOVB   ERRTYP+1,R1             ;PUT THE ERROR TYPE IN R1
4393 032054 122737 000076 013263          CMPB   #RMSPOS,ERRTYP+1       ;IS IT A MISPOSITION ERROR
4394 032062 001012                                BNE    10$                      ;NO KEEP GOING
4395 032064 122762 000020 000000          CMPB   #WR,CMD(P2)           ;IS IT A WRITE ?
4396 032072 001404                                BEQ    5$                        ;NO TRY AGAIN
4397 032074 122762 000100 000000          CMPB   #WTM,CMD(R2)          ;IS IT A WRITE TAPE MARK ?
4398 032102 001002                                BNE    10$                      ;NO, MUST BE A READ
4399 032104 112701 000100          5$:  MOVB   #WMSPO,R1          ;SET IT TO A WRITE MISPOSITON
4400 032110 060401          10$: ADD    R4,R1              ;ADD THE OFFSET TO THE LUN POINTER
4401 032112 005211                                INC    (R1)                    ;INC THE ERROR COUNT
4402 032114          15$: POP    <R1>              ;RESTORE R1 AND R4
4403 032116 000207          RTS    PC                    ;EXIT
4404

```

```

4406 .SBTTL ERROR LOG TALLY
4407 ;*****
4408 ;
4409 ; ERROR LOG TALLY
4410 ;
4411 ;Called By :
4412 ;Calls To :
4413 ;Inputs :
4414 ;Outputs :
4415 ;Register Inputs :
4416 ;Register Outputs:
4417

```

```

4418 032120 ETLTY::
4419 032120          PUSH    <R1>                ;SAVE R1
4420 032122 122737 000106 013263          CMPB    #NOERR,ERRTYP+1          ;IS IT A STATUS ERROR ?
4421 032130 001412                                BEQ    ELTEXT                    ;YES, GET OUT
4422 032132 113701 013263          MOVB   ERRTYP+1,R1              ;LET R1 = THE ERROR TYPE
4423 032136 060401                                ADD    R4,R1                    ;ADD THE LUN BLOCK POINTER TO R1
4424 032140 132763 000001 000055          BITB   #BIT0,L.TRK(R3)          ;IS IT AN ODD TRACK NUMBER ?
4425 032146 001402                                BEQ    1$                       ;YES, GO TALLY THE ERROR
4426 032150 062701 000002          ADD    #2,R1                   ;NO, TALLY THE EVEN TRACK
4427 032154 005211          1$: INC    (R1)                 ;INC THE ERROR COUNT
4428 032156          ELTEXT: POP   <R1>            ;RESTORE R1
4429 032160 000207          RTS    PC                      ;RETURN

```

```

4431      .SBTTL  TABLE SORTER
4432      ;*****
4433      ;
4434      ; TABLE SORTER
4435      ;
4436      ;Called By      : ERRDEL
4437      ;Register Inputs :
4438      ;Register Outputs:
4439
4440 032162      TBLSORT: PUSH      <R1,R2,R3,R5>      ;SAVE R1,R2,R3,R5 ON STACK
4441 032172      012701 003360      MOV      #UNTTBL,R1      ;INITIALIZE R1,R2,R3,R5
4442 032176      016402 000130      MOV      TBLTOP(R4),R2      ;FOR LOCAL USAGE
4443 032202      016403 000134      MOV      LSTENT(R4),R3
4444 032206      010205      MOV      R2,R5
4445 032210      020203      CMP      R2,R3      ;FIRST ENTRY?
4446 032212      001524      BEQ      30$      ;YES, PUT IT IN THE TABLE
4447
4448 032214      126162 000000 000000 1$:  CMPB     TRK(R1),TRK(R2)      ;GET TO THE RIGHT TRACK
4449 032222      101405      BLOS    5$
4450 032224      062702 000010      ADD     #8.,R2
4451 032230      020203      CMP     R2,R3      ;NEXT UNUSED SLOT ?
4452 032232      001514      BEQ     30$      ;YES, PUT IT IN THE TABLE
4453 032234      000767      BR      1$
4454
4455 032236      126162 000000 000000 5$:  CMPB     TRK(R1),TRK(R2)      ;MAKE SURE ITS THE SAME TRACK
4456 032244      001024      BNE     15$
4457 032246      026162 000002 000002      CMP     PHB(R1),PHB(R2)      ;IF OUR OBJ CNT HIGHER
4458 032254      101405      BLOS    10$
4459 032256      062702 000010      ADD     #8.,R2      ;GO TO NEXT TABLE LOCATION
4460 032262      020203      CMP     R2,R3      ;NEXT UNUSED SLOT ?
4461 032264      001477      BEQ     30$      ;YES, PUT IT IN THE TABLE
4462 032266      000763      BR      5$
4463
4464 032270      026162 000002 000002 10$:  CMP     PHB(R1),PHB(R2)      ;IF THE ENTRY ALREADY EXIST
4465 032276      001007      BNE     15$
4466 032300      066162 000004 000004      ADD     SWR(R1),SWR(R2)      ;THEN ADD THE ERROR COUNT
4467 032306      066162 000006 000006      ADD     HWR(R1),HWR(R2)
4468 032314      000477      BR      EXTSRT
4469
4470 032316      026464 000134 000132 15$:  CMP     LSTENT(R4),TBLBTM(R4) ;IF THE TABLES FULL, GET OUT
4471 032324      001022      BNE     20$
4472 032326      032764 000040 000026 17$:  BIT     #MTBLOV,LUNFLG(R4)
4473 032334      001067      BNE     EXTSRT
4474 032336      052764 000040 000026      BIS     #MTBLOV,LUNFLG(R4)
4475 032344      PRINTF  #TBLFUL,L$LUN
      032344      013746 002074      MOV     L$LUN,-(SP)
      032350      012746 036716      MOV     #TBLFUL,-(SP)
      032354      012746 000002      MOV     #2,-(SP)
      032360      010600      MOV     SP,R0
      032362      104417      TRAP   C$PNTF
      032364      062706 000006      ADD     #6,SP
4476 032370      000451      BR      EXTSRT
4477
4478 032372      016405 000134      20$:  MOV     LSTENT(R4),R5      ;MOVE THE TABLE CONTENTS
4479 032376      010503      MOV     R5,R3      ;DOWN TO PUT IN THE ENTRY
4480 032400      162703 000010      SUB     #8.,R3
4481

```

```

4482 032404 116365 000000 000000 25$:  MOVB  TRK(R3),TRK(R5)
4483 032412 016365 000002 000002      MOV   PHB(R3),PHB(R5)
4484 032420 116365 000004 000004      MOVB  SWR(R3),SWR(R5)
4485 032426 116365 000005 000005      MOVB  SRD(R3),SRD(R5)
4486 032434 116365 000006 000006      MOVB  HWR(R3),HWR(R5)
4487 032442 116365 000007 000007      MOVB  HRD(R3),HRD(R5)
4488 032450 162703 000010                SUB   #8.,R3
4489 032454 162705 000010                SUB   #8.,R5
4490 032460 020502                CMP   R5,R2                ;KEEP GOING TILL THE RIGHT SLOT
4491 032462 101350                BHI   25$                ;IS OPEN
4492
4493 032464 026464 000134 000132 30$:  CMP   LSTENT(R4),TBLBTM(R4) ;IF THE TABLES FULL, GET OUT
4494 032472 001715                BEQ   17$
4495 032474 020301                CMP   R3,R1
4496 032476 103403                BLO   35$
4497 032500 010103                MOV   R1,R3
4498 032502 010205                MOV   R2,R5
4499 032504 000737                BR    25$
450  032506 062764 000010 000134 35$:  ADD   #8.,LSTENT(R4)        ;ADJUST THE LAST ENTRY POINTER
4501 032514 005061 000004                EXTSRT: CLR  SWR(R1)         ;CLEAR SOFT ERROR COUNTERS
4502 032520 005061 000006                CLR  HWR(R1)         ;CLEAR HARD ERROR COUNTERS
4503 032524                POP   <R5,R3,R2,R1>      ;RESTORE REGISTERS
4504 032534 000207                RTS   PC

```

```

4506      .SBTTL PRINT ERROR
4507      ;*****
4508      ;
4509      ; PRINT ERROR
4510      ;
4511      ;Called By      : ERRDEC, ERRDEI, ERRDEL
4512      ;Calls To     : ERROR
4513      ;Register Inputs :
4514      ;Register Outputs:
4515
4516 032536      PRIERR::
4517 032536 010337 003560      MOV      R3,R3SAVE      ;SAVE R3
4518 032542 010437 003562      MOV      R4,R4SAVE      ;SAVE R4
4519 032546      ERROR      ;ERROR MACRO
      032546 104460      TRAP     C$ERROR
4520 032550 000207      RTS      PC      ;RETURN
4521

```

```

4523      .SBTTL COMPARE DATA
4524      ;*****
4525      ;
4526      ; COMPARE DATA
4527      ;
4528      ;Called By      :
4529      ;Calls To      :
4530      ;Inputs        :
4531      ;Outputs       :
4532      ;Register Inputs :
4533      ;Register Outputs:
4534      ;
4535
4536 032552 CMPDAT::
4537 032552      PUSH      <R1,R2,R3,R5>          ;SAVE R1,R2,R3,R5
          032552 010146      MOV      R1,-(SP)          ;;PUSH R1 ON STACK
          032554 010246      MOV      R2,-(SP)          ;;PUSH R2 ON STACK
          032556 010346      MOV      R3,-(SP)          ;;PUSH R3 ON STACK
          032560 010546      MOV      R5,-(SP)          ;;PUSH R5 ON STACK
4538 032562 026363 000040 000014      CMP      P.TRBC(R3),P.BCNT(R3) ;AS MANY BYTES READ AS WRITTEN ?
4539 032570 001426      BEQ      5$          ;BRANCH IF YES
4540 032572 012705 011122      MOV      #RLST,R5          ;PUT THE RLS TABLE ADDRESS IN R5
4541 032576 012702 013262      MOV      #L$ERRTBL,R2     ;PUT THE ERROR TABLE ADDRESS IN R2
4542 032602 012522      MOV      (R5)+,(R2)+      ;MOV THE RLS TABLE TO THE ERROR TABLE
4543 032604 012522      MOV      (R5)+,(R2)+
4544 032606 012522      MOV      (R5)+,(R2)+
4545 032610 011512      MOV      (R5),(R2)
4546 032612      POP      <R5,R3,R2,R1>      ;RESTORE REGISTERS
          032612 012605      MOV      (SP)+,R5          ;;POP STACK INTO R5
          032614 012603      MOV      (SP)+,R3          ;;POP STACK INTO R3
          032616 012602      MOV      (SP)+,R2          ;;POP STACK INTO R2
          032620 012601      MOV      (SP)+,R1          ;;POP STACK INTO R1
4547 032622 004737 032036      JSR      PC,ERRTLY        ;GO TALLY THE ERROR
4548 032626 105037 013263      CLR      ERRTYP+1        ;CLEAR THE LUN POINTER
4549 032632 004737 032536      JSR      PC,PRIERR        ;GO PRINT THE ERROR
4550 032636      PUSH      <R1,R2,R3,R5>      ;SAVE R1,R2,R3,R5
          032636 010146      MOV      R1,-(SP)          ;;PUSH R1 ON STACK
          032640 010246      MOV      R2,-(SP)          ;;PUSH R2 ON STACK
          032642 010346      MOV      R3,-(SP)          ;;PUSH R3 ON STACK
          032644 010546      MOV      R5,-(SP)          ;;PUSH R5 ON STACK
4551 032646 005037 003444 5$:      CLR      CMPERR          ;CLEAR LOCATION CMPERR
4552 032652 042764 000020 000026      BIC      #ODDFLG,LUNFLG(R4) ;CLEAR THE ODD BYTE COUNT FLAG
4553 032660 016337 000014 003530      MOV      P.BCNT(R3),R8    ;PUT THE TAPE RECORD BYTE COUNT IN R8
4554 032666 005037 003532      CLR      R9              ;CLEAR THE BYTE ADDRESS COUNTER
4555 032672 032737 000001 003530      BIT      #BIT0,R8        ;IS THE BYTE COUNT ODD
4556 032700 001406      BEQ      10$            ;BRANCH IF NOT
4557 032702 042737 000001 003530      BIC      #BIT0,R8        ;MAKE THE COUNT EVEN
4558 032710 052764 000020 000026      BIS      #ODDFLG,LUNFLG(R4) ;SET THE ODD BYTE FLAG
4559 032716 012701 003446 10$:      MOV      #BYTADD,R1      ;LET R1 POINT TO THE ADDRESS TABLE
4560 032722 022737 000003 002114      CMP      #3,L$TEST        ;ARE WE IN TEST 3 ?
4561 032730 001003      BNE      11$            ;NO, SO JUST SET RDBUF IN BUFADR
4562 032732 012702 070550      MOV      #WRTBUF-2,R2    ;LET R2 POINT TO THE WRITE BUFFER
4563 032736 000402      BR      12$
4564 032740 012702 070552 11$:      MOV      #WRTBUF,R2      ;LET R2 POINT TO THE WRITE BUFFER
4565 032744 012703 050552 12$:      MOV      #RDBUF,R3      ;LET R3 POINT TO THE READ BUFFER
4566 032750 012705 003472      MOV      #DATBL,R5      ;LET R5 POINT TO THE ERROR DATA TABLE
4567 032754 022322 14$:      CMP      (R3)+,(R2)+      ;COMPARE THE FIRST WORD OF DATA

```


GLOBAL AREAS
COMPARE DATA

MACRO Y05.02 Thursday 18-Apr-85 13:37 Page 59-2

L9

SEQ 115

4617 033240 000207

454: RTS PC

;RETURN


```

4619      .SBTTL UNJAM
4620      ;*****
4621      ;
4622      ; UNJAM
4623      ;
4624      ;Called By      :
4625      ;Calls To      :
4626      ;Inputs        :
4627      ;Outputs       :
4628      ;Register Inputs:
4629      ;Register Outputs:
4630      ;
4631
4632      UNJAM::
4633      033242 005737 003344      TST      DUMPKT      ;ARE WE ISSUEING NULL COMMANDS
4634      033246 001444              BEQ      15$        ;YES , THEN EXIT
4635      033250 023764 003542 000006  CMP      SECURNS,CMDSEQ(R4) ;IS IT THE ONLY COMMAND OUT ?
4636      033256 001440              BEQ      15$        ;YES , THEN EXIT
4637      033260 016437 000006 003436  MOV      CMDSEQ(R4),SAVDIF ;SET UP TO UNJAM THE QUEUES
4638      033266 163737 003542 003436  SUB      SECURNS,SAVDIF ;SUBTRACT CURRENT FROM THE HIGHEST
4639      033274 063737 003436 003400  ADD      SAVDIF,CMDCNT ;ADJUST THE COMMAND COUNT
4640      033302 063737 003436 003402  ADD      SAVDIF,RSPCNT ;ADJUST THE RESPONSE COUNT
4641      033310 042737 000200 003516  BIC      #CMDONE,PCFLAG ;CLEAR THE ALL COMMANDS ISSUED FLAG
4642      033316 163764 003436 000034  SUB      SAVDIF,OBJFDL(R4) ;ADJUST THE OBJECT COUNT
4643      033324 103002              BCC      5$        ;GET OUT IF NO CARRY
4644      033326 005364 000036              DEC      OBJFDH(R4) ;OTHERWISE, ADJUST THE HIGH WORD
4645      033332 022737 000004 003424 5$:  CMP      #N,SUBCNT
4646      033340 001002              BNE      10$
4647      033342 005037 003424              CLR      SUBCNT
4648      033346 005237 003424 10$:  INC      SUBCNT
4649      033352 005337 003436              DEC      SAVDIF
4650      033356 001365              BNE      5$
4651      033360 052764 000002 000026 15$:  BIS      #SEREXC,LUNFLG(R4) ;SET THE SERIOUS EXCEPTION FLAG
4652      033366 000207              RTS      PC        ;RETURN
4653      033370
4654      ENDMOD

```

```

4656 .SBTTL OCTAL -> HEX CONVERTER
4657 ;*****
4658 ;
4659 ; OCTAL -> HEX CONVERTER
4660 ;
4661
4662 OCTHEX::
4663 033370 116337 000013 003552      MOVB   P.STS+1(R3),LOHEX      ;GET THE UPPER STATUS BYTE
4664 033376 013737 003552 003554      MOV    LOHEX,HIHEX          ;GET A SECOND COPY
4665 033404 042737 177760 003552      BIC    #177760,LOHEX        ;CLEAR THE UNWANTED BITS
4666 033412 042737 177417 003554      BIC    #177417,HIHEX        ;CLEAR THE UNWANTED BITS
4667 033420 006037 003554              ROR    HIHEX                ;SHIFT THE
4668 033424 006037 003554              ROR    HIHEX                ;      UPPER NIBBLE
4669 033430 006037 003554              ROR    HIHEX                ;      OVER 3 PLACES
4670 033434 006137 003552              ROL    LOHEX                ;SHIFT THE LOWER NIBBLE OVER 1
4671 033440 062737 011442 003552      ADD    #HEXTBL,LOHEX        ;ADD ASCII TABLE BASE ADDRESS
4672 033446 062737 011442 003554      ADD    #HEXTBL,HIHEX        ;ADD ASCII TABLE BASE ADDRESS
4673 033454 000207              RTS    PC                   ;RETURN

```

```

4675      .SBTTL  CLEAR EOT
4676      ;*****
4677      ;
4678      ; CLEAR EOT
4679      ;
4680
4681 033456      CLREOT::
4682 033456      PUSH    <R1,R2,R4>                ;SAVE R1, R2, AND R4
                MOV     R1,-(SP)                ;;PUSH R1 ON STACK
                MOV     R2,-(SP)                ;;PUSH R2 ON STACK
                MOV     R4,-(SP)                ;;PUSH R4 ON STACK
                CLR     R2                        ;CLEAR OUT R2
4683 033464 005002      CLR     R2                        ;CLEAR OUT R2
4684 033466 012701 177776      MOV     #-2.,R1                ;SET R1 TO THE FIRST UNIT
4685 033472 012704 002314      MOV     #LUN0,R4                ;LET R4 EQUAL THE FIRDT LUN
4686 033476 062701 000002      5$:  ADD     #2.,R1                ;ADD 2 TO THE UNIT POINTER
4687 033502 062702 000001      ADD     #1.,R2                ;ADD 1 TO R2
4688 033506 112764 177777 000154      MOV     #-1,OLDTRK(R4)        ;INITIALIZE OLD TRACK
4689 033514 012764 177777 000152      MOV     #-1,OLDBLK(R4)       ;INITIALIZE OLD BLOCK
4690 033522 062704 000172      ADD     #LUNSTP,R4           ;SET R4 TO THE NEXT LUN
4691 033526 042761 000004 003350      BIC     #EOT,DRINUS(R1)     ;CLEAR THE EOT BIT IN DRINUS
4692 033534 005037 003526      CLR     UEOT                ;CLEAR THE EOT FLAG
4693 033540 023702 002012      10$:  CMP     L'UNIT,R2        ;HAVE WE DONE THEM ALL
4694 033544 001354      BNE     5$                    ;NO, KEEP GOING TILL ALL DONE
4695 033546      POP     <R4,R2,R1>                ;RESTORE R4, R2, AND R1
                MOV     (SP)+,R4                ;;POP STACK INTO R4
                MOV     (SP)+,R2                ;;POP STACK INTO R2
                MOV     (SP)+,R1                ;;POP STACK INTO R1
4696 033554 000207      RTS     PC                    ;RETURN

```

```

4698          .SBTTL SEED SETUP AND SAVE
4699          ;*****
4700          ;
4701          ; SEED SETUP
4702          ;
4703          ;
4704 033556    SDSTUP::
4705 033556    PUSH    <R1,R4>
          033556 010146    MOV    R1,-(SP)          ;;PUSH R1 ON STACK
          033560 010446    MOV    R4,-(SP)          ;;PUSH R4 ON STACK
4706 033562 013701 002012    MOV    L$UNIT,R1
4707 033566 012704 002314    MOV    #LUNO,R4
4708 033572 016464 000136 000144 5$: MOV    SED1(R4),SEED1(R4)
4709 033600 016464 000140 000146    MOV    SED2(R4),SEED2(R4)
4710 033606 016464 000142 000150    MOV    SED3(R4),SEED3(R4)
4711 033614 062704 000172    ADD    #LUNSTP,R4
4712 033620 005301    DEC    R1
4713 033622 001363    BNE    5$
4714 033624    POP    <R4,R1>
          033624 012604    MOV    (SP)+,R4          ;;POP STACK INTO R4
          033626 012601    MOV    (SP)+,R1          ;;POP STACK INTO R1
4715 033630 000207    RTS    PC
4716          ;
4717          ;*****
4718          ;
4719          ; SEED SAVE
4720          ;
4721          ;
4722 033632    SDSAVE::
4723 033632    PUSH    <R1,R4>
          033632 010146    MOV    R1,-(SP)          ;;PUSH R1 ON STACK
          033634 010446    MOV    R4,-(SP)          ;;PUSH R4 ON STACK
4724 033636 013701 002012    MOV    L$UNIT,R1
4725 033642 012704 002314    MOV    #LUNO,R4
4726 033646 016464 000144 000136 5$: MOV    SEED1(R4),SED1(R4)
4727 033654 016464 000146 000140    MOV    SEED2(R4),SED2(R4)
4728 033662 016464 000150 000142    MOV    SEED3(R4),SED3(R4)
4729 033670 062704 000172    ADD    #LUNSTP,R4
4730 033674 005301    DEC    R1
4731 033676 001363    BNE    5$
4732 033700    POP    <R4,R1>
          033700 012604    MOV    (SP)+,R4          ;;POP STACK INTO R4
          033702 012601    MOV    (SP)+,R1          ;;POP STACK INTO R1
4733 033704 000207    RTS    PC
4734          ;

```

```

4736
4737
4738
4739
4740
4741
4742
4743
4744 033706
4745 033706 012704 002314
4746 033712 005064 000024
4747 033716 022704 003072
4748 033722 001403
4749 033724 062704 000172
4750 033730 000770
4751 033732 000207
4752
4753
4754

```

```

.SBTTL PATTERN CLEAR
;*****
;
; PATTERN CLEAR
;
; THIS ROUTINE DOES NOT SAVE R4 AND THEREFORE SHOULD NOT BE CALLED FROM ANY
; PLACE OTHER THAN A TEST.

PATCLR:
1$:  MOV    #LUN0,R4
      CLR   PATSAV(R4)
      CMP   #LUN3,R4
      BEQ   2$
      ADD   #LUNSTP,R4
      BR    1$
2$:  RTS   PC

```

```

4756
4757
4758
4759
4760
4761
4762
4763
4764
4765
4766
4767 033734
4768 033734
      033734 010146
      033736 010246
4769 033740
      033740 010546
      033742 010446
      033744 010346
      033746 010246
      033750 010146
      033752 012746 020347
      033756 012746 000006
      033762 010600
      033764 104417
      033766 062706 000016
4770 033772 012701 002314
4771 033776 012702 002314
4772 034002
      034002 016246 000006
      034006 016246 000004
      034012 016246 000002
      034016 011246
      034020 010146
      034022 012746 020471
      034026 012746 000006
      034032 010600
      034034 104417
      034036 062706 000016
4773
4774 034042 062701 000010
4775 034046 062702 000010
4776 034052 022701 010572
4777 034056 101351
4778 034060
4779 034064 000207
4780

```

```

.SBTTL CORE DUMP
;*****
;
; CORE DUMP
;
; THIS ROUTINE IS DESIGNED TO DUMP ALL CRITICAL MEMORY LOCATIONS ON
; OCCURRENCE OF ERRORS, WHEN ENEABLED BY THE OPERATOR VIA THE SOFTWARE
; QUESTIONS. IT IS INTENDED PRIMARILY AS AN AID TO DEBUGGING THE
; PROGRAM, BUT MAY PROVE USEFUL IN ANALYZING CERTAIN DEVICE ERRORS
; AS WELL.
CORDMP:
      PUSH    <R1,R2>
      MOV     R1,-(SP)          ;; PUSH R1 ON STACK
      MOV     R2,-(SP)          ;; PUSH R2 ON STACK
      PRINTF  #DUMP,R1,R2,R3,R4,R5
      MOV     R5,-(SP)
      MOV     R4,-(SP)
      MOV     R3,-(SP)
      MOV     R2,-(SP)
      MOV     R1,-(SP)
      MOV     #DUMP,-(SP)
      MOV     #6,-(SP)
      MOV     SP,R0
      TRAP    C$PNTF
      ADD     #16,SP
      MOV     #LUN0,R1          ;PUT STARING ADDRESS IN R1
      MOV     #LUN0,R2          ;AND ANOTHER COPY IN R2
1$:   PRINTF  #DUMP2,R1,(R2),2(R2),4(R2),6(R2)
      MOV     6(R2),-(SP)
      MOV     4(R2),-(SP)
      MOV     2(R2),-(SP)
      MOV     (R2),-(SP)
      MOV     R1,-(SP)
      MOV     #DUMP2,-(SP)
      MOV     #6,-(SP)
      MOV     SP,R0
      TRAP    C$PNTF
      ADD     #16,SP
      ADD     #10,R1            ;UPDATE R1
      ADD     #10,R2            ;UPDATE R2
      CMP     #IOERTB,R1       ;ARE WE AT THE END OF DUMP AREA
      BHI    1$                ;KEEP GOING IF NOT
      POP     <R2,R1>
      RTS     PC

```

```

4782      .SBTTL BUFFER DUMP
4783      ;*****
4784      ;
4785      ; BUFFER DUMP
4786      ;
4787      ; THIS ROUTINE WILL PRINT THE READ BUFFER FOR EVERY RECORD READ IN
4788      ; TEST 5.
4789
4790 034066      BUFDMP:
4791 034066      PUSH      <R1,R2>
4792 034072      016301      000014      MOV      P.BCNT(R3),R1      ;GET NUMBER OF BYTES X-FERRED
4793 034076      012702      050552      MOV      #RDBUF,R2      ;GET BUFFER ADDRESS
4794 034102      034102      010146      PRINTF   #DUMP1,R1      ;PRINT NUMBER OF BYTES
4795      034104      012746      020430      MOV      R1,-(SP)
4796      034110      012746      000002      MOV      #DUMP1,-(SP)
4797      034114      010600      MOV      #2,-(SP)
4798      034116      104417      MOV      SP,R0
4799      034120      062706      000006      TRAP    C$PNTF
4800      1$:      PRINTF   #DUMP2,(R2),2(R2),4(R2),6(R2),10(R2)
4801      MOV      10(R2),(SP)
4802      MOV      6(R2),-(SP)
4803      MOV      4(R2),-(SP)
4804      MOV      2(R2),-(SP)
4805      MOV      (R2),-(SP)
4806      MOV      #DUMP2,-(SP)
4807      MOV      #6,-(SP)
4808      MOV      SP,R0
4809      TRAP    C$PNTF
4810      ADD      #16,SP
4811      ADD      #12,R2      ;ADJUST BUFFER POINTER
4812      SUB      #12,R1      ;ADJUST BYTE COUNT
4813      BEQ      5$      ;IF ZERO GET OUT
4814      BPL      1$      ;KEEP GOING IF POSITIVE
4815
4816      5$:      POP      <R2,R1>
4817      RTS      PC
4818
4819
4820
4821
4822
4823
4824
4825
4826
4827
4828
4829
4830
4831
4832
4833
4834
4835
4836
4837
4838
4839
4840

```

```

4816          .TITLE MISCELLANEOUS SECTIONS
4817          .SBTTL  REPORT CODING SECTION
4845
4846 034210          BGNMOD
4847
4848          ;**
4849          ; THE REPORT CODING SECTION CONTAINS THE
4850          ; "PRINTS" CALLS THAT GENERATE STATISTICAL REPORTS.
4851          ;--
4852
4853 034210          BGNRPT
         034210          L$RPT::
4854
4860 034210          PUSH    <R1,R4,R5>
         034210 010146          MOV     R1,-(SP)          ;;PUSH R1 ON STACK
         034212 010446          MOV     R4,-(SP)          ;;PUSH R4 ON STACK
         034214 010546          MOV     R5,-(SP)          ;;PUSH R5 ON STACK
4861 034216 032737 000400 003516          BIT     #DROPT,PCFLAG          ;ARE WE DROPPING A UNIT
4862 034224 001015          BNE     5$          ;YES, ONLY PRINT STATS FOR THIS UNIT
4863 034226 005001          CLR     R1          ;SET R1 TO FIRST UNIT
4864 034230 005037 002074          CLR     L$LUN          ;START WITH UNIT 0
4865 034234 012704 002314          MOV     #LUNO,R4          ;START WITH LUN BLOCK FOP UNIT 0
4866 034240 013705 002012          MOV     L$UNIT,R5          ;INIT UNIT COUNTER
4867 034244 032761 000010 003350 1$:          BIT     #DROP,DRINUS(R1)          ;HAS THE DRIVE BEEN DROPPED ?
4868 034252 001402          BEQ     5$          ;NO, PRINT ITS STATS
4869 034254 000137 035346          JMP     15$          ;OTHERWISE GET THE NEXT DRIVE
4870
4871 034260          5$:          PRINTS  #STAT0
         034260 012746 035464          MOV     #STAT0,-(SP)
         034264 012746 000001          MOV     #1,-(SP)
         034270 010600          MOV     SP,R0
         034272 104416          TRAP   C$PNTS
         034274 062706 000004          ADD     #4,SP
4872
4873 034300          PRINTS  #STAT1,L$LUN
         034300 013746 002074          MOV     L$LUN,-(SP)
         034304 012746 035467          MOV     #STAT1,(SP)
         034310 012746 000002          MOV     #2,(SP)
         034314 010600          MOV     SP,R0
         034316 104416          TRAP   C$PNTS
         034320 062706 000006          ADD     #6,SP
4874
4875 034324          PRINTS  #STAT2
         034324 012746 035533          MOV     #STAT2,-(SP)
         034330 012746 000001          MOV     #1,(SP)
         034334 010600          MOV     SP,R0
         034336 104416          TRAP   C$PNTS
         034340 062706 000004          ADD     #4,SP
4876
4877 034344          PRINTS  #STAT3
         034344 012746 035600          MOV     #STAT3,-(SP)
         034350 012746 000001          MOV     #1,(SP)
         034354 010600          MOV     SP,R0
         034356 104416          TRAP   C$PNTS
         034360 062706 000004          ADD     #4,SP
4878
4879 034364          PRINTS  #STAT4

```


	034364	012746	035660	MOV	#STAT4,-(SP)
	034370	012746	000001	MOV	#1,-(SP)
	034374	010600		MOV	SP,R0
	034376	104416		TRAP	C\$PNTS
	034400	062706	000004	ADD	#4,SP
4880					
4881	034404			PRINTS	#STAT5,S1READ(R4),S2READ(R4),S1WRIT(R4),S2WRIT(R4)
	034404	016446	000046	MOV	S2WRIT(R4),-(SP)
	034410	016446	000044	MOV	S1WRIT(R4),-(SP)
	034414	016446	000042	MOV	S2READ(R4),-(SP)
	034420	016446	000040	MOV	S1READ(R4),-(SP)
	034424	012746	035705	MOV	#STAT5,-(SP)
	034430	012746	000005	MOV	#5,-(SP)
	034434	010600		MOV	SP,R0
	034436	104416		TRAP	C\$PNTS
	034440	062706	000014	ADD	#14,SP
4882					
4883	034444			PRINTS	#STAT6,EC1COR(R4),EC2COR(R4)
	034444	016446	000052	MOV	EC2COR(R4),-(SP)
	034450	016446	000050	MOV	EC1COR(R4),-(SP)
	034454	012746	035752	MOV	#STAT6,-(SP)
	034460	012746	000003	MOV	#3,-(SP)
	034464	010600		MOV	SP,R0
	034466	104416		TRAP	C\$PNTS
	034470	062706	000010	ADD	#10,SP
4884					
4885	034474			PRINTS	#STAT7,H1READ(R4),H2READ(R4),H1WRIT(R4),H2WRIT(R4)
	034474	016446	000062	MOV	H2WRIT(R4),-(SP)
	034500	016446	000060	MOV	H1WRIT(R4),-(SP)
	034504	016446	000056	MOV	H2READ(R4),-(SP)
	034510	016446	000054	MOV	H1READ(R4),-(SP)
	034514	012746	036024	MOV	#STAT7,-(SP)
	034520	012746	000005	MOV	#5,-(SP)
	034524	010600		MOV	SP,R0
	034526	104416		TRAP	C\$PNTS
	034530	062706	000014	ADD	#14,SP
4886					
4887	034534			PRINTS	#STAT8,RTYEC1(R4),RTYEC2(R4)
	034534	016446	000066	MOV	RTYEC2(R4),-(SP)
	034540	016446	000064	MOV	RTYEC1(R4),-(SP)
	034544	012746	036071	MOV	#STAT8,-(SP)
	034550	012746	000003	MOV	#3,-(SP)
	034554	010600		MOV	SP,R0
	034556	104416		TRAP	C\$PNTS
	034560	062706	000010	ADD	#10,SP
4888					
4889	034564			PRINTS	#STAT9,DTCPER(R4)
	034564	016446	000070	MOV	DTCPER(R4),-(SP)
	034570	012746	036143	MOV	#STAT9,-(SP)
	034574	012746	000002	MOV	#2,-(SP)
	034600	010600		MOV	SP,R0
	034602	104416		TRAP	C\$PNTS
	034604	062706	000006	ADD	#6,SP
4890					
4891	034610			PRINTS	#STAT10,UNDRUN(R4)
	034610	016446	000072	MOV	UNDRUN(R4),-(SP)
	034614	012746	036222	MOV	#STAT10,-(SP)

	034620	012746	000002	MOV	#2,-(SP)
	034624	010600		MOV	SP,R0
	034626	104416		TRAP	C\$PNTS
	034630	062706	000006	ADD	#6,SP
4892					
4893	034634			PRINTS	#STAT11,OVRUN(R4)
	034634	016446	000074	MOV	OVRUN(R4),-(SP)
	034640	012746	036272	MOV	#STAT11,-(SP)
	034644	012746	000002	MOV	#2,-(SP)
	034650	010600		MOV	SP,R0
	034652	104416		TRAP	C\$PNTS
	034654	062706	000006	ADD	#6,SP
4894					
4895	034660			PRINTS	#STAT12,RMSPOS(R4),WMSPOS(R4)
	034660	016446	000100	MOV	WMSPOS(R4),-(SP)
	034664	016446	000076	MOV	RMSPOS(R4),-(SP)
	034670	012746	036344	MOV	#STAT12,-(SP)
	034674	012746	000003	MOV	#3,-(SP)
	034700	010600		MOV	SP,R0
	034702	104416		TRAP	C\$PNTS
	034704	062706	000010	ADD	#10,SP
4896					
4897	034710			PRINTS	#STAT13,OTHER(R4)
	034710	016446	000102	MOV	OTHER(R4),-(SP)
	034714	012746	036405	MOV	#STAT13,-(SP)
	034720	012746	000002	MOV	#2,-(SP)
	034724	010600		MOV	SP,R0
	034726	104416		TRAP	C\$PNTS
	034730	062706	000006	ADD	#6,SP
4898					
4899	034734			PRINTS	#STAT14,DROPPD(R4)
	034734	016446	000104	MOV	DROPPD(R4),-(SP)
	034740	012746	036430	MOV	#STAT14,-(SP)
	034744	012746	000002	MOV	#2,-(SP)
	034750	010600		MOV	SP,R0
	034752	104416		TRAP	C\$PNTS
	034754	062706	000006	ADD	#6,SP
4900					
4901	034760			PRINTS	#STAT15,WRBYT4(R4),WRBYT3(R4),WRBYT2(R4),WRBYT1(R4)
	034760	016446	000120	MOV	WRBYT1(R4),-(SP)
	034764	016446	000122	MOV	WRBYT2(R4),-(SP)
	034770	016446	000124	MOV	WRBYT3(R4),-(SP)
	034774	016446	000126	MOV	WRBYT4(R4),-(SP)
	035000	012746	036453	MOV	#STAT15,-(SP)
	035004	012746	000005	MOV	#5,-(SP)
	035010	010600		MOV	SP,R0
	035012	104416		TRAP	C\$PNTS
	035014	062706	000014	ADD	#14,SP
4902					
4903	035020			PRINTS	#STAT16,RDBYT4(R4),RDBYT3(R4),RDBYT2(R4),RDBYT1(R4)
	035020	016446	000110	MOV	RDBYT1(R4),-(SP)
	035024	016446	000112	MOV	RDBYT2(R4),-(SP)
	035030	016446	000114	MOV	RDBYT3(R4),-(SP)
	035034	016446	000116	MOV	RDBYT4(R4),-(SP)
	035040	012746	036525	MOV	#STAT16,-(SP)
	035044	012746	000005	MOV	#5,-(SP)
	035050	010600		MOV	SP,R0

4904	035052	104416		TRAP	C\$PNTS	
	035054	062706	000014	ADD	#14,SP	
4905	035060			PRINTS	#STAT0	
	035060	012746	035464	MOV	#STAT0,-(SP)	
	035064	012746	000001	MOV	#1,-(SP)	
	035070	010600		MOV	SP,R0	
	035072	104416		TRAP	C\$PNTS	
	035074	062706	000004	ADD	#4,SP	
4906						
4907	035100	032737	000100	003516	BIT	#NOMDTB,PCFLAG
4908	035106	001117		BNE	15\$;IS THE TABLE TO BE PRINTED
4909	035110			PUSH	<R1,R2,R3>	;NO, GET OUT
4910	035116	016401	000130	MOV	TBLTOP(R4),R1	;POINT R1 TO THE TOP OF THE TABLE
4911	035122	016402	000134	MOV	LSTENT(R4),R2	;POINT R2 TO THE LAST ENTRY IN TABLE
4912						
4913	035126			PRINTS	#STAT17,L\$LUN	
	035126	013746	002074	MOV	L\$LUN,-(SP)	
	035132	012746	036577	MOV	#STAT17,-(SP)	
	035136	012746	000002	MOV	#2,-(SP)	
	035142	010600		MOV	SP,R0	
	035144	104416		TRAP	C\$PNTS	
	035146	062706	000006	ADD	#6,SP	
4914						
4915	035152	016103	000000	MOV	TRK(R1),R3	;MOV THE TRACK # TO TEMP STORAGE
4916	035156	020102		10\$: CMP	R1,R2	;HAVE WE PRINTED ALL ENTRIES ?
4917	035160	001451		BEQ	12\$;YES, GET OUT
4918	035162	026103	000000	CMP	TRK(R1),R3	;ARE WE STILL ON THE SAME TRACK ?
4919	035166	001412		BEQ	11\$;YES, KEEP GOING
4920	035170	016103	000000	MOV	TRK(R1),R3	;MOV THE TRACK # TO TEMP STORAGE
4921						
4922	035174			PRINTS	#STAT0	
	035174	012746	035464	MOV	#STAT0,-(SP)	
	035200	012746	000001	MOV	#1,-(SP)	
	035204	010600		MOV	SP,R0	
	035206	104416		TRAP	C\$PNTS	
	035210	062706	000004	ADD	#4,SP	
4923						
4924	035214			11\$: PRINTS	#STAT18,<B,TRK(R1)>,PHB(R1),<B,HWR(R1)>,<B,HRD(R1)>,<B,SWR(R1)>,<B,SRD(R1)>	
	035214	005046		CLR	-(SP)	
	035216	156116	000005	BISB	SRD(R1),(SP)	
	035222	005046		CLR	-(SP)	
	035224	156116	000004	BISB	SWR(R1),(SP)	
	035230	005046		CLR	-(SP)	
	035232	156116	000007	BISB	HRD(R1),(SP)	
	035236	005046		CLR	-(SP)	
	035240	156116	000006	BISB	HWR(R1),(SP)	
	035244	016146	000002	MOV	PHB(R1),-(SP)	
	035250	005046		CLR	-(SP)	
	035252	156116	000000	BISB	TRK(R1),(SP)	
	035256	012746	036647	MOV	#STAT18,-(SP)	
	035262	012746	000007	MOV	#7,-(SP)	
	035266	010600		MOV	SP,R0	
	035270	104416		TRAP	C\$PNTS	
	035272	062706	000020	ADD	#20,SP	
4925						
4926	035276	062701	000010	ADD	#8.,R1	;STEP R1 TO THE NEXT TABLE LOCATION

```

4927 035302 000725          BR      10$          ;NO, KEEP PRINTING
4928 035304          POP      <R3,R2,R1>
4929 035312 032764 000040 000026 12$:  BIT      #MTBLOV, LUNFLG(R4)  ;IS THE TABLE FULL ?
4930 035320 001412          BEQ      15$          ;NO, GET NEXT DRIVE
4931 035322          PRINTF   #TBLFUL, L$LUN  ;PRINT TABLE FULL MESSAGE
      035322 013746 002074  MOV      L$LUN, -(SP)
      035326 012746 036716  MOV      #TBLFUL, -(SP)
      035332 012746 000002  MOV      #2, -(SP)
      035336 010600          MOV      SP, R0
      035340 104417          TRAP    C$PNTF
      035342 062706 000006  ADD      #6, SP

4932
4933 035346 032737 000400 003516 15$:  BIT      #DROPT, PCFLAG  ;ARE WE DROPPING A UNIT
4934 035354 001036          BNE     25$          ;YES, ONLY PRINT STATS FOR THIS UNIT
4935 035356 062704 000172  ADD      #LUNSTP, R4  ;R4 POINTS TO NEXT LUN BLOCK
4936 035362 062701 000002  ADD      #2, R1       ;POINT R1 TO THE NEXT UNIT
4937 035366 005237 002074  INC      L$LUN       ;POINTS TO NEXT UNIT NUMBER
4938 035372 005305          DEC     R5           ;ANY UNITS LEFT TO REPORT?
4939 035374 001402          BEQ     20$          ;BRANCH IF NOT
4940 035376 000137 034244  JMP      1$          ;ELSE, DO IT AGAIN
4941 035402 105737 002212 20$:  TSTB    CLOCK        ;IS THE CLOCK ENABLED
4942 035406 001421          BEQ     25$          ;NO, THEN CAN'T PRINT TIME
4943 035410          PRINTF   #TIME, <B,HOURS>, <B,MINUTE>, <B,SECOND>,
      035410 005046          CLR     -(SP)
      035412 153716 002215  BISB    SECOND, (SP)
      035416 005046          CLR     -(SP)
      035420 153716 002214  BISB    MINUTE, (SP)
      035424 005046          CLR     -(SP)
      035426 153716 002213  BISB    HOURS, (SP)
      035432 012746 020044  MOV     #TIME, -(SP)
      035436 012746 000004  MOV     #4, -(SP)
      035442 010600          MOV     SP, R0
      035444 104417          TRAP    C$PNTF
      035446 062706 000012  ADD     #12, SP
4944 035452          POP     <R5,R4,R1>  ;RESTORE REGS
4945 035460          EXIT    RPT
      035460 000167          .WORD  J$JMP
      035462 001316          .WORD  L10007-2-.

4946
4958

```

;FORMAT STATEMENTS FOR PRINT CALLS

```

4960
4961
4962 035464 045 116 000 STAT0: .ASCIZ ?N?
4963 035467 045 116 045 STAT1: .ASCIZ ?N*ASTATISTICAL REPORT FOR UNIT *D1?
4964 035533 045 116 045 STAT2: .ASCIZ ?N*S8*S8*S8*S8*S2*AREAD*S4*S8*AWRITE?
4965 035600 045 116 045 STAT3: .ASCIZ ?N*S8*S8*S8*S6*ACH 1*S4*ACH 2*S4*ACH 1*S4*ACH 2?
4966 035660 045 116 045 STAT4: .ASCIZ ?N*ASOFT DATA ERRORS?
4967 035705 045 116 045 STAT5: .ASCIZ ?N*A RETRY RECOVERED*S8*D8*D8*D8*D8?
4968 035752 045 116 045 STAT6: .ASCIZ ?N*A ECC CORRECTED *S8*D8*D8*S8*A N.A.?
4969 036024 045 116 045 STAT7: .ASCIZ ?N*AHARD DATA ERRORS *S8*D8*D8*D8*D8?
4970 036071 045 116 045 STAT8: .ASCIZ ?N*ACRC ON ECC BLOCKS*S8*D8*D8*S8*A N.A.?
4971 036143 045 116 045 STAT9: .ASCIZ ?N*ADATA COMPARE ERRORS *S8*D8*S8*A N.A.?
4972 036222 045 116 045 STAT10: .ASCIZ ?N*ADATA UNDERRUNS*S7*S7*S7*AN.A.*S6*D8?
4973 036272 045 116 045 STAT11: .ASCIZ ?N*ADATA OVERRUNS*S8*S8*D8*S8*A N.A.?
4974 036344 045 116 045 STAT12: .ASCIZ ?N*AMISPOSITIONS *S8*S8*D8*S8*D8?
4975 036405 045 116 045 STAT13: .ASCIZ ?N*AOTHERS *S8*D8?
4976 036430 045 116 045 STAT14: .ASCIZ ?N*ADROPS *S8*D8?
4977 036453 045 116 045 STAT15: .ASCIZ ?N*ABYTES WRITTEN *D3*A,*Z3*A,*Z3*A,*Z3?
4978 036525 045 116 045 STAT16: .ASCIZ ?N*ABYTES READ *D3*A,*Z3*A,*Z3*A,*Z3?
4979 036577 045 116 045 STAT17: .ASCIZ ?N*A TRK PHY BLK HWR HRD SWR SRD*N?
4980 036647 045 116 045 STAT18: .ASCIZ ?N*S2*D2*S3*D5*S3*D3*S2*D3*S2*D3?
4981 036716 045 116 045 TBLFUL: .ASCIZ ?N*AUNIT *D1*A MEDIA DATA TABLE HAS OVERFLOWED*N*N?
4982 .EVEN
4983
4984
4985 037002 ENDRPT
037002 L10007:
037002 104425 TRAP C$RPT

```

```

4987          .SBTTL  INITIALIZE SECTION
4988
4989
4990          ;**
4991          ; THE INITIALIZE SECTION CONTAINS THE CODING THAT IS PERFORMED
4992          ; AT THE BEGINNING OF EACH PASS.
4993          ;--
4994 037004          BGNINIT
037004          L$INIT::
4995
4996 037004          STINIT::
4997 037004          READEF  #EF.START
037004 012700 000040  MOV      #EF.START,R0
037010 104447      TRAP    C$REFG
4998 037012          BCOMPLETE      START
037012 103414      BCS     START
4999
5000 037014          READEF  #EF.RESTART
037014 012700 000037  MOV      #EF.RESTART,R0
037020 104447      TRAP    C$REFG
5001 037022          BCOMPLETE      START
037022 103410      BCS     START
5002
5003 037024          READEF  #EF.NEW
037024 012700 000035  MOV      #EF.NEW,R0
037030 104447      TRAP    C$REFG
5004 037032          BCOMPLETE      NUPASS
037032 103463      BCS     NUPASS
5005
5006 037034          READEF  #EF.CONTINUE
037034 012700 000036  MOV      #EF.CONTINUE,R0
037040 104447      TRAP    C$REFG
5007 037042          BCOMPLETE      NUPASS
037042 103457      BCS     NUPASS
5008
5009 037044 112737 000001 003564  START:  MOVB    #1,DAYS          ;SET TO FIRST DAY
5010 037052 005037 003522          CLR     PASCNT          ;CLEAR THE PASS COUNTER
5011 037056 005037 003444          CLR     CMPERR          ;CLEAR THE COMPARE ERROR COUNTER
5012
5013 037062 012701 002314          MOV     #LUN0,R1          ;SET R1 TO THE FIRST LUN
5014 037066 010102          5$:  MOV     R1,R2          ;LET R2 = R1
5015 037070 062702 000156          ADD     #URSPBF,R2       ;LET R2 = THE END OF THE CLEAR AREA
5016 037074 005021          10$:  CLR     (R1)+          ;CLEAR THE LUN LOCATION
5017 037076 020201          CMP     R2,R1          ;ARE WE AT THE END OF THE CLEAR AREA
5018 037100 001375          BNE    10$          ;NO, KEEP CLEARING
5019 037102 062701 000014          ADD     #14,R1          ;SET R1 TO THE NEXT LUN BLOCK
5020 037106 022701 003264          CMP     #PCMDBF,R1      ;HAVE WE DONE THEM ALL
5021 037112 001365          BNE    5$          ;GO CLEAR THE NEXT LUN BLOCK
5022
5023 037114 005021          15$:  CLR     (R1)+          ;CLEAR THE LOCATION AND GET THE NEXT
5024 037116 022701 003400          CMP     #CMDCNT,R1      ;HAVE WE CLEARED THEM ALL ?
5025 037122 001374          BNE    15$          ;NO, KEEP GOING
5026
5027 037124 012701 110552          20$:  MOV     #MEDIAS,R1      ;PUT THE STATS TABLE ADDRESS INTO R1
5028 037130 005021          CLR     (R1)+          ;CLEAR THE LOCATION AND GET THE NEXT
5029 037132 022701 115612          CMP     #PATCH,R1      ;HAVE WE CLEARED THEM ALL ?
5030 037136 001374          BNE    20$          ;NO, KEEP GOING

```

```

5031
5032 037140 012704 002314      MOV      #LUN0,R4      ;SET R4 TO THE FIRST LUN
5033 037144 013702 002012      MOV      L$UNIT,R2    ;SET UP R2
5034 037150 012764 001233 000136 25$:  MOV      #RS1,SED1(P4) ;SET UP THE SEED IN THE LUN BLOCK
5035 037156 012764 007622 000140      MOV      #RS2,SED2(R4) ;SET UP THE SEED IN THE LUN BLOCK
5036 037164 012764 000000 000142      MOV      #RS3,SED3(R4) ;SET UP THE SEED IN THE LUN BLOCK
5037
5038 037172 062704 000172      30$:    ADD      #LUNSTP,R4    ;SET UP THE NEXT LUN BLOCK
5039 037176 005302              DEC      R2            ;DECREMENT R2
5040 037200 001363              BNE     25$           ;DID YOU DO THEM ALL
5041
5042 037202              NUPASS: BRESET
          037202 104433          TRAP    C$RESET
5043 037204 005737 003522          TST     PASCNT        ;IS THIS THE FIRST PASS ?
5044 037210 001403              BEQ     1$            ;YES GO CLEAR THE MEDIA TABLE
5045 037212 105737 002226          TSTB   CLRMED        ;ARE WE CLEARING THE MEDIA TABLE ?
5046 037216 001457              BEQ     25$          ;NO GET OUT
5047
5048 037220              1$:
5049 037220 013702 002012      MOV      L$UNIT,R2    ;SET UP R2
5050 037224 022702 000001      CMP     #1,R2        ;IS THERE 1 UNIT ?
5051 037230 001004              BNE     3$            ;NO, CHECK AGAIN
5052 037232 012737 005040 003544      MOV      #U1OFF,TBLOFF ;SET OFFSET FOR 1 UNIT IN TBLOFF
5053 037240 000421              BR      15$          ;GO GET STARTED
5054
5055 037242 022702 000002      3$:    CMP     #2,R2        ;IS THERE 2 UNITS ?
5056 037246 001004              BNE     5$            ;NO, CHECK AGAIN
5057 037250 012737 002420 003544      MOV      #U2OFF,TBLOFF ;SET OFFSET FOR 2 UNITS IN TBLOFF
5058 037256 000412              BR      15$          ;GO GET STARTED
5059
5060 037260 022702 000003      5$:    CMP     #3,R2        ;IS THERE 3 UNITS ?
5061 037264 001004              BNE     10$           ;NO, CHECK AGAIN
5062 037266 012737 001540 003544      MOV      #U3OFF,TBLOFF ;SET OFFSET FOR 3 UNITS IN TBLOFF
5063 037274 000403              BR      15$          ;GO GET STARTED
5064
5065 037276 012737 001210 003544 10$:    MOV      #U4OFF,TBLOFF ;SET OFFSET FOR 4 UNITS IN TBLOFF
5066
5067 037304 012704 002314      15$:    MOV      #LUN0,R4    ;SET R4 TO THE FIRST LUN
5068 037310 012737 110552 003530      MOV      #MEDIAS,R8   ;PUT THE STAT TABLE ADDRESS IN R8
5069
5070 037316 013764 003530 000130 20$:    MOV      R8,TBLTOP(R4) ;PUT THE TOP TABLE ADDR IN LUN BLOCK
5071 037324 013764 003530 000134      MOV      R8,LSTENT(R4) ;PUT THE TABLE ENTRY ADDR IN LUN BLOCK
5072 037332 063737 003544 003530      ADD     TBLOFF,R8     ;ADD THE LENGTH FOR EACH UNIT
5073 037340 013764 003530 000132      MOV      R8,TBLBTM(R4) ;PUT THE TABLE BOTTOM POINTER IN LUNBLK
5074 037346 062704 000172      ADD     #LUNSTP,R4    ;SET UP THE NEXT LUN BLOCK
5075 037352 005302              DEC     R2            ;DECREMENT R2
5076 037354 001360              BNE     20$          ;DID YOU DO THEM ALL
5077
5078 037356 005037 003516      25$:    CLR     PCFLAG       ;CLEAR THE PROGRAM CONTROL FLAG.
5079 037362 005037 003526              CLR     UEOT         ;CLEAR THE EOT FLAG
5080 037366 013737 002012 003524      MOV     L$UNIT,UDROP  ;SET UP THE DROP UNIT FLAG
5081 037374 005237 003522              INC     PASCNT       ;ADD 1 TO PASS COUNTER
5082 037400 105737 002227              TSTB   MEDTBL       ;IS THE MEDIA TABLE TO BE PRINTED
5083 037404 001003              BNE     30$          ;YES, DON'T DO ANYTHING
5084 037406 052737 000100 003516      BIS     #NOMDTB,PCFLAG ;NO, SET THE PROGRAM CONTROL FLAG BIT
5085
5086 037414 012702 003566      30$:    MOV     #CMDBF1,R2   ;PUT COMMAND BUFFER ADDRESS IN R2

```

INITIALIZE SECTION

```

5087 037420 005022          35$: CLR      (R2)+           ;CLEAR THE BUFFERS
5088 037422 022702 010226  CMP      #DSRNGO,R2      ;ARE WE AT THE END OF THE BUFFER ?
5089 037426 001374          BNE      35$           ;KEPP GOING TILL WE ARE
5090
5091 037430 012704 002314          MOV      #LUNO,R4       ;SET R4 TO THE FIRST LUN
5092 037434 013702 ^02012  MOV      L$UNIT,R2     ;SET UP R2
5093 037440 005001          CLR      R1           ;CLEAR R1
5094 037442 005003          CLR      R3           ;CLEAR R3
5095
5096 037444 032761 000020 003350 40$: BIT      #FAIL,DRINUS(R1) ;HAS THIS DRIVE FAILED ?
5097 037452 001054          BNE      60$           ;YES, GET NEXT UNIT
5098
5099 037454 032761 000010 003350 45$: BIT      #DROP,DRINUS(R1) ;DID THIS DRIVE DROP LAST TIME
5100 037462 001002          BNE      50$           ;YES, KEEP GOING
5101 037464 005064 000032          CLR      UNDROP(R4)   ;OTHERWISE CLEAR THE DROP COUNTER
5102 037470 012761 000001 003350 50$: MOV      #AVB,DRINUS(R1) ;SET UP ALL DRIVES TO AVAILABLE
5103
5104 037476 012764 003572 000014 55$: MOV      #DCMDBF,CNUSAV(R4) ;SET UP NEW COMMAND BUFFER SAVE
5105 037504 012764 003572 000016  MOV      #DCMDBF,COLSAV(R4) ;SET UP OLD COMMAND BUFFER SAVE
5106 037512 016464 000166 000012  MOV      UCDSRG(R4),CMDSSV(R4) ;SET UP COMMAND DESCRIPTOR SAVE
5107 037520 016464 000156 000020  MOV      URSPBF(R4),RNUSAV(R4) ;SET UP NEW RESPONSE BUFFER SAVE
5108 037526 016464 000156 000022  MOV      URSPBF(R4),ROLSAV(R4) ;SET UP OLD RESPONSE BUFFER SAVE
5109
5110 037534 005064 000006          CLR      CMDSEQ(R4)   ;CLEAR THE COMMAND REFERENCE NUMBER
5111 037540 005064 000034          CLR      OBJFDL(R4)  ;CLEAR THE LOW OBJECT FIELD
5112 037544 005064 000036          CLR      OBJFDH(R4)  ;CLEAR THE HIGH OBJECT FIELD
5113 037550 005064 000010          CLR      SLTUSE(R4)  ;CLEAR THE SLCT IN USE FLAG
5114 037554 010300          MOV      R3,R0
5115
5116 037556          GPWARD R0,R0
5117 037560 104442          TRAP    C$GPHRD
5118 037560 103011          BNCOMPLETE 60$
5119 037562 011064 000000          BCC    60$
5120 037566 012064 000002          MOV    (R0),TKIP(R4)
5121 037572 062764 000002 000002  MOV    (R0)+,TKSA(R4)
5122 037600 011064 000004          ADD    #2,TKSA(R4)
5123 037600 011064 000004          MOV    (R0),TKUNIT(R4)
5124 037604 062701 000002          60$: ADD    #2,R1
5125 037610 062703 000001          ADD    #1,R3
5126 037614 062704 000172          ADD    #LUNSTP,R4
5127 037620 005302          DEC    R2
5128 037622 001310          BNE    40$
5129
5130 037624 042737 000002 003516  BIC    #NCLKFL,PCFLAG ;GET READY TO TEST FOR CLOCK PRESENT
5131 037632          SETVEC #4,#NOCLK,#PRI00 ;SET VECTOR 4 IN CASE NO CLOCK
5132 037632 012746 000000          MOV    #PRI00,-(SP)
5133 037636 012746 021044          MOV    #NOCLK,-(SP)
5134 037642 012746 000004          MOV    #4,-(SP)
5135 037646 012746 000003          MOV    #3,-(SP)
5136 037652 104437          TRAP  C$SVEC
5137 037654 062706 000010          ADD    #10,SP
5138 037660 005737 177546          TST    KWCSR
5139 037664 000240          NOP
5140 037666 000240          NOP
5141

```



```

5136 037670          CLRVEC #4          ;RETURN VECTOR TO TRAP CATCHER
      037670 012700 000004  MOV #4,R0
      037674 104436  TRAP C$CVEC
5137 037676 032737 000002 003516  BIT #NCLKFL,PCFLAG ;WAS A CLOCK PRESENT ?
5138 037704 001016  BNE ISTART ;NO CLOCK, START REGULAR INIT
5139 037706          SETVEC #100,#KWHDL,#PRI00 ;SET UP THE CLOCK VECTOR
      037706 012746 000000  MOV #PRI00,-(SP)
      037712 012746 021100  MOV #KWHDL,-(SP)
      037716 012746 000100  MOV #100,-(SP)
      037722 012746 000003  MOV #3,-(SP)
      037726 104437  TRAP C$SVEC
      037730 062706 000010  ADD #10,SP
5140 037734 012737 000100 177546  MOV #100,KWCSR ;ENABLE THE CLOCK INTERRUPTS
5141
5142 037742 005001          ISTART: CLR R1 ;SET R1 TO FIST UNIT
5143 037744 005037 002074  CLR L$LUN ;SET L$LUN TO FIRST UNIT
5144 037750 012704 002314  MOV #LUN0,R4 ;SET R4 TO THE FIRST LUN BLOCK
5145
5146 037754 032761 000001 003350 1$: BIT #AVB,DRINUS(R1) ;SEE IF DRIVE IS PRESENT AND AVAILABLE
5147 037762 001547  BEQ 20$ ;GET THE NEXT DRIVE IF IT ISN'T
5148 037764 032761 000004 003350  BIT #EOT,DRINUS(R1) ;CHECK IF THE DRIVE IS AT EOT
5149 037772 001143  BNE 20$ ;GET NEXT DRIVE IF IT IS
5150
5151 037774 012764 000377 000010  MOV #377,SLTUSE(R4) ;SET ALL RESPONSE SLOTS TO THE PORT
5152 040002 004737 026230  JSR PC,PRTCLR ;GO DO IT
5153 040006 112737 000004 010571  MOVB #4,CRDLIM ;CREDITS START AT 4 FOR NEW LUN
5154
5155 040014 012705 040336  MOV #INITIT,R5 ;PUT INIT TEST TABLE ADDRESS IN R5
5156 040020 004737 021406  JSR PC,CMMDSQ ;GO DO INIT ON THIS DRIVE
5157 040024 032761 000001 003350  BIT #AVB,DRINUS(R1) ;SEE IF DRIVE IS PRESENT AND AVAILABLE
5158 040032 001523  BEQ 20$ ;GET THE NEXT DRIVE IF IT ISN'T
5159
5160 040034 012764 000377 000010  MOV #377,SLTUSE(R4) ;SET ALL RESPONSE SLOTS TO THE PORT
5161 040042 004737 026230  JSR PC,PRTCLR ;GO DO IT
5162 040046 112737 000004 010571  MOVB #4,CRDLIM ;CREDITS START AT 4 FOR NEW LUN
5163 040054 062705 000006  ADD #TSTSTP,R5 ;POINT R5 TO THE SCC COMMAND
5164 040060 004737 021406  JSR PC,CMMDSQ ;GO DO SCC ON THIS DRIVE
5165 040064 032761 000001 003350  BIT #AVB,DRINUS(R1) ;SEE IF DRIVE IS PRESENT AND AVAILABLE
5166 040072 001503  BEQ 20$ ;GET THE NEXT DRIVE IF IT ISN'T
5167
5168 040074 012764 000377 000010 5$: MOV #377,SLTUSE(R4) ;SET ALL RESPONSE SLOTS TO THE PORT
5169 040102 004737 026230  JSR PC,PRTCLR ;GO DO IT
5170 040106 112737 000004 010571  MOVB #4,CRDLIM ;CREDITS START AT 4 FOR NEW LUN
5171 040114 062705 000006  ADD #TSTSTP,R5 ;POINT R5 TO THE ONL COMMAND
5172 040120 004737 021406  JSR PC,CMMDSQ ;GO DO ONLINE ON THIS DRIVE
5173
5174 040124 012764 000377 000010 10$: MOV #377,SLTUSE(R4) ;SET ALL RESPONSE SLOTS TO THE PORT
5175 040132 004737 026230  JSR PC,PRTCLR ;GO DO IT
5176 040136 112737 000004 010571  MOVB #4,CRDLIM ;CREDITS START AT 4 FOR NEW LUN
5177 040144 062705 000006  ADD #TSTSTP,R5 ;POINT R5 TO THE GUS COMMAND
5178 040150 012737 000040 003440  MOV #AVLB,TSTMSK ;ALLOW UNIT AVAILABLE ERRORS
5179 040156 004737 021406  JSR PC,CMMDSQ ;GO DO GUS ON THIS DRIVE
5180 040162 005037 003440  CLR TSTMSK ;ALLOW NO ERRORS
5181 040166 032761 000001 003350  BIT #AVB,DRINUS(R1) ;SEE IF DRIVE IS PRESENT AND AVAILABLE
5182 040174 001442  BEQ 20$ ;GET THE NEXT DRIVE IF IT ISN'T
5183
5184 040176 016403 000022          15$: MOV ROLSAV(R4),R3 ;LET R3 = OLD RESPONSE BUFFER POINTER

```

```

5185 040202 162703 000104      SUB      #DRBSTP,R3          ;ADJUST R3 BACK TO THE LAST RESPONCE
5186 040206      PRINTF    #CONTRV,L$LUN,<B,P.CSVR(R3)>,<B,P.CHVR(R3)>
      040206 005046      CLR      -(SP)
      040210 156316 000051      BISB    P.CHVR(R3),(SP)
      040214 005046      CLR      -(SP)
      040216 156316 000050      BISB    P.CSVR(R3),(SP)
      040222 013746 002074      MOV     L$LUN,-(SP)
      040226 012746 020532      MOV     #CONTRV,-(SP)
      040232 012746 000004      MOV     #4,-(SP)
      040236 010600      MOV     SP,R0
      040240 104417      TRAP    C$PNTF
      040242 062706 000012      ADD     #12,SP
5187 040246      PRINTF    #UNITRV,<B,P.USVR(R3)>,<B,P.UHVR(R3)>
      040246 005046      CLR      -(SP)
      040250 156316 000053      BISB    P.UHVR(R3),(SP)
      040254 005046      CLR      -(SP)
      040256 156316 000052      BISB    P.USVR(R3),(SP)
      040262 012746 020662      MOV     #UNITRV,-(SP)
      040266 012746 000003      MOV     #3,-(SP)
      040272 010600      MOV     SP,R0
      040274 104417      TRAP    C$PNTF
      040276 062706 000010      ADD     #10,SP
5188
5189 040302 022701 000006      20$:    CMP     #6.,R1          ;HAVE WE DONE THEM ALL ?
5190 040306 001411      BEQ     EXTINT          ;GET OUT
5191 040310 062701 000002      ADD     #UNTSTP,R1      ;GET NEXT UNIT
5192 040314 062704 000172      ADD     #LUNSTP,R4      ;SET UP THE NEXT LUN BLOCK
5193 040320 005237 002074      INC     L$LUN          ;GET NEXT UNIT
5194 040324      BREAK
      040324 104422      TRAP    C$BRK
5195 040326 000137 037754      JMP     1$              ;GO DO THE NEXT ONE
5196
5197 040332      EXTINT:  EXIT    INIT
      040332 104432      TRAP    C$EXIT
      040334 000032      .WORD   L10010-.
5198
5199      ;INIT TEST TABLE
5200 040336      170      INITIT::  .BYTE   INT      ;INITIALIZATION TABLE
5201 040337      000      .BYTE   NULPAT
5202 040340 000000      .WORD   0
5203 040342 000001      .WORD   1
5204 040344      230      .BYTE   SCC      ;SET CONTROLLER CHARACTERISTICS TABLE
5205 040345      000      .BYTE   NULPAT
5206 040346 000000      .WORD   0
5207 040350 000001      .WORD   1
5208 040352      140      .BYTE   ONL      ;ONLINE TABLE
5209 040353      000      .BYTE   NULPAT
5210 040354 000000      .WORD   0
5211 040356 000001      .WORD   1
5212 040360      220      .BYTE   GUS      ;GET UNIT STATUS TABLE
5213 040361      000      .BYTE   NULPAT
5214 040362 000000      .WORD   0
5215 040364 000001      .WORD   1
5216
5217      .EVEN
5218 040366      ENDINIT
      040366      L10010:

```

040366 104411

TRAP C\$INIT

5220 040370
040370
5221 040370
040370
040370 104461

BGNAUTO
L\$AUTO::
ENDAUTO
L10011: TRAP C\$AUTO

```

5223      .SBTTL  CLEANUP CODING SECTION
5224      ; THE CLEANUP CODING SECTION CONTAINS THE CODING THAT IS PERFORMED
5225      ; AFTER THE HARDWARE TESTS HAVE BEEN PERFORMED.
5226      ;--
5227
5228 040372      BGNCLN
      040372      L$CLEAN::
5229
5236 040372 032737 000002 003516      BIT      #NCLKFL,PCFLAG      ;WAS A CLOCK PRESENT ?
5237 040400 001005      BNE      5$      ;NO CLOCK, DO REPORT
5238 040402 005037 177546      CLR      KWCSR
5239 040406      CLRVEC  #100
      040406 012700 000100      MOV      #100,R0
      040412 104436      TRAP     C$CVEC
5240 040414      5$:      DORPT
      040414 104424      TRAP     C$DRPT
5241 040416 032737 000400 003516      BIT      #DROPIT,PCFLAG
5242 040424 001003      BNE      EXTCLN
5243 040426 042737 000100 003516      BIC      #NOMDTB,PCFLAG
5244 040434      EXTCLN: EXIT      CLN
      040434 104432      TRAP     C$EXIT
      040436 000002      .WORD    L10012 .
5245
5257
5258      .EVEN
5259
5260 040440      ENDCLN
      040440      L10012:
      040440 104412      TRAP     C$CLEAN

```

```

5262          .SBTTL  DROP UNIT SECTION
5263
5264          ;++
5265          ; THE DROP-UNIT SECTION CONTAINS THE CODING THAT CAUSES A DEVICE
5266          ; TO NO LONGER BE TESTED.
5267          ;--
5268
5269 040442     DROPUN:
5270 040442     PUSH    <R1>                ;SAVE R1
                    MOV    R1,-(SP)        ;;PUSH R1 ON STACK
                    MOV    R0,R1          ;POINT R1 TO THE DRINUS TABLE
                    ROL    R1             ;MULTIPLY BY 2
5271 040444     010146     INC    DROPPD(R4)    ;INC DROPPED FLAG FOR THIS UNIT
5272 040446     006101     BIS    #DROPI,PCFLAG ;SET THE UNIT DROP FLAG
5273 040450     005264     000104     PUSH    <R0>                ;SAVE R0
5274 040454     052737     000400     003516   MOV    R0,-(SP)        ;;PUSH R0 ON STACK
5275 040462     010046     DORPT          ;GO PRINT UNIT STATS
                    TRAP   C$DRPT
5276 040464     104424     POP    <R0>                ;RESTORE R0
                    MOV    (SP)+,R0      ;;POP STACK INTO R0
5277 040466     012600     BIC    #DROPI,PCFLAG    ;CLEAR THE DROP FLAG
5278 040470     042737     000400     003516   TSTB  NOCLR           ;DO WE WANT TO CLEAR STATS ON ERROR ?
5279 040476     105737     002230     BEQ    3$             ;NO, DON'T CLEAR THE STATS
5280 040502     001415     PUSH    <R2,R3>
5281 040504     010246     MOV    R2,-(SP)        ;;PUSH R2 ON STACK
                    010346     MOV    R3,-(SP)        ;;PUSH R3 ON STACK
5282 040510     012702     000040     MOV    #S1READ,R2    ;STARTING ADDRESS OF STATS IN R2
5283 040514     012703     000130     MOV    #TBLTOP,R3    ;END ADDRESS OF STATS IN R3
5284 040520     060402     ADD    R4,R2          ;ADD THE LUN BLOCK ADDRESS TO R2
5285 040522     060403     ADD    R4,R3          ;ADD THE LUN BLOCK ADDRESS TO R3
5286 040524     005022     2$:    CLR    (R2)+          ;CLEAR THE LOCATION
5287 040526     020203     CMP    R2,R3          ;ARE WE AT THE END OF THE STATS
5288 040530     001375     BNE    2$            ;NO, KEEP CLEARING
5289 040532     012603     POP    <R3,R2>
                    012602     MOV    (SP)+,R3        ;;POP STACK INTO R3
                    040534     MOV    (SP)+,R2        ;;POP STACK INTO R2
5290 040536     042761     000001     003350   3$:    BIC    #AVB,DRINUS(R1) ;CLEAR THE AVB BIT IN DRIVE IN USE TABLE
5291 040544     032761     000004     003350   BIT    #EOT,DRINUS(R1) ;IS THE DRIVE AT EOT ?
5292 040552     001404     BEQ    5$            ;BRANCH IF NOT
5293 040554     042761     000004     003350   BIC    #EOT,DRINUS(R1) ;CLEAR THE EOT BIT IN DRIVE IN USE TABLE
5294 040562     000402     BR     10$           ;GET OUT
5295 040564     005337     003524     5$:    DEC    UDROP          ;SUBTRACT 1 TO DROPPED FLAG
5296 040570     052761     000010     003350   10$:   BIS    #DROP,DRINUS(R1) ;SET DRIVE IN USE TABLE TO DROPPED
5297 040576     005264     000032     INC    UNDROP(R4)    ;ADD 1 TO THE UNIT DROP COUNT
5298 040602     022764     000012     000032   CMP    #10.,UNDROP(R4) ;DO WE HAVE 10. ERRORS ?
5299 040610     001004     BNE    15$           ;NO, GET OUT
5300 040612     052761     000020     003350   BIS    #FAIL,DRINUS(R1) ;SET THE DRIVE TO FAIL
5301 040620     104451     DODU    R0
                    TRAP   C$DODU
5302 040622     005037     003406     15$:   CLR    RESPON        ;CLEAR THE RESPONSE STATUS
5303 040626     012601     POP    <R1>          ;RESTORE R1
                    MOV    (SP)+,R1     ;;POP STACK INTO R1
5304 040630     012727     000024     DELAY  20.          ;DELAY FOR AWHILE
                    MOV    #20.,(PC)+
                    .WORD  0
                    MOV    L$DLY,(PC)+
                    .WORD  0

```

040644	005367	177772		DEC	-6(PC)	
040650	001375			BNE	.-4	
040652	005367	177756		DEC	-22(PC)	
040656	001367			BNE	.-20	
5305	040660	010174	000000	MOV	R1,@TKIP(R4)	;FLUSH THE DRIVE
5306	040664	012764	000377	MOV	#377,SLTUSE(R4)	;SET ALL RESPONCE SLOTS TO PORT
5307	040672	004737	026230	JSR	PC,PRTCLR	;GO CLEAR THE PORT
5308	040676	000207		RTS	PC	;RETURN
5309						
5310						
5311	040700			BGNDU		
	040700			L\$DU::		
5312	040700			EXIT	DU	
	040700	000167		.WORD	J\$JMP	
	040702	000000		.WORD	L10013-2-	
5313						
5325						
5326				.EVEN		
5327						
5328	040704			ENDDU		
	040704			L10013:		
	040704	104453		TRAP	C\$DU	

```

5330          .SBTTL  ADD UNIT SECTION
5331
5332          ;**
5333          ; THE ADD-UNIT SECTION CONTAINS ANY CODE THE PROGRAMMER WISHES
5334          ; TO BE EXECUTED IN CONJUNCTION WITH THE ADDING OF A UNIT BACK
5335          ; TO THE TEST CYCLE.
5336          ;--
5337
5338 040706          BGNAU
5339 040706
5339
5345
5346 040706          EXIT      AU
5346 040706 000167    .WORD    J$JMP
5346 040710 000000    .WORD    L10014-2-.
5347
5359
5360          .EVEN
5361
5362 040712          ENDAU
5362 040712          L10014:
5362 040712 104452    TRAP      C$AU
5363
5364 040714          ENDMOD
5365

```


5368
 5369
 5380
 5381
 5417
 5418 040714
 5419
 5420
 5421
 5422
 5423
 5424
 5425
 5426
 5433
 5439
 5440 040714
 5441 040714 105737 002234
 5442 040720 001414
 5443 040722
 040722 013746 002114
 040726 012746 021001
 040732 012746 000002
 040736 010600
 040740 104417
 040742 062706 000006
 5444 040746 000137 042624
 5445
 5446 040752 005737 003524
 5447 040756 001014
 5448 040760
 040760 013746 002114
 040764 012746 021001
 040770 012746 000002
 040774 010600
 040776 104417
 041000 062706 000006
 5449 041004 000137 042624
 5450
 5451 041010 105737 002212
 5452 041014 001421
 5453 041016
 041016 005046
 041020 153716 002215
 041024 005046
 041026 153716 002214
 041032 005046
 041034 153716 002213
 041040 012746 020044
 041044 012746 000004
 041050 010600
 041052 104417
 041054 062706 000012
 5454
 5455 041060 004737 033456
 5456 041064 012737 000100 003440

.TITLE HARDWARE TESTS

.SBTTL TEST 1: Basic Function Test

BGNMOD

```

; **
; This test will execute a subset of the legal commands on the unit
; under test. It serves as a quick verify test to ascertain that the
; unit can move tape and write/read predictably, without error. The
; subset of legal commands will be issued in a coherent manner.
; --
  
```

```

T1::      BGNTST
          TSTB   T3ONLY           ;CHECK THE TEST 3 ONLY FLAG
          BEQ    START1          ;BRANCH IF IT IS CLEAR
          PRINTF #BYPASS,L$TEST  ;PRINT THE TEST BYPASSED MESSAGE
          MOV    L$TEST,-(SP)
          MOV    #BYPASS,-(SP)
          MOV    #2,-(SP)
          MOV    SP,R0
          TRAP  C$PNTF
          ADD    #6,SP
          JMP    T1EXIT          ;JUMP IF IT IS NOT CLEAR

START1:   TST    UDROP           ;HAVE ALL UNITS BEEN DROPPED ?
          BNE    5$             ;NO, CONTINUE
          PRINTF #BYPASS,L$TEST  ;PRINT THE TEST BYPASSED MESSAGE
          MOV    L$TEST,-(SP)
          MOV    #BYPASS,(SP)
          MOV    #2,-(SP)
          MOV    SP,R0
          TRAP  C$PNTF
          ADD    #6,SP
          JMP    T1EXIT          ;GET OUT IF NONE LEFT TO TEST

5$:       TSTB   CLOCK           ;IS THE CLOCK ENABLED
          BEQ    GO1            ;NO, THEN CAN'T PRINT TIME
          PRINTF #TIME,<B,HOURS>,<B,MINUTE>,<B,SECOND>,
          CLR    -(SP)
          BISB  SECOND,(SP)
          CLR    -(SP)
          BISB  MIUTE,(SP)
          CLR    -(SP)
          BISB  HOURS,(SP)
          MOV   #TIME,-(SP)
          MOV   #4,-(SP)
          MOV   SP,R0
          TRAP  C$PNTF
          ADD   #12,SP

GO1:     JSR    PC,CLREOT        ;MAKE SURE EOT STATUS IS CLEAR
          MOV   #ONLB,TSTMSK    ;ALLOW ALREADY ONLINE STATUS
  
```

5457									
5458	041072	012705	042630		MOV	#T1ONL,R5			;SET UP TO DO AN ONLINE
5459	041076	004737	021240		JSR	PC,SCHED			;GO ISSUE THE COMMAND
5460	041102	005737	003524		TST	UDROP			;HAVE ALL UNITS BEEN DROPPED ?
5461	041106	001002			BNE	5\$;NO, CONTINUE
5462	041110	000137	042624		JMP	T1EXIT			;GET OUT IF NONE LEFT TO TEST
5463									
5464	041114	005037	003440	5\$:	CLR	TSTMSK			;ALLOW NO ERRORS
5465	041120	012705	042636		MOV	#T1REW,R5			;SET UP TO DO A REWIND
5466	041124	004737	021240		JSR	PC,SCHED			;GO ISSUE THE COMMAND
5467	041130	005737	003524		TST	UDROP			;HAVE ALL UNITS BEEN DROPPED ?
5468	041134	001002			BNE	10\$;NO, CONTINUE
5469	041136	000137	042624		JMP	T1EXIT			;GET OUT IF NONE LEFT TO TEST
5470									
5471	041142	012705	042644	10\$:	MOV	#T1LEOT,R5			;SET UP TO DO 2 TAPE MARK COMMANDS
5472	041146	004737	021240		JSR	PC,SCHED			;GO ISSUE THE COMMAND
5473	041152	005737	003524		TST	UDROP			;HAVE ALL UNITS BEEN DROPPED ?
5474	041156	001002			BNE	15\$;NO, CONTINUE
5475	041160	000137	042624		JMP	T1EXIT			;GET OUT IF NONE LEFT TO TEST
5476									
5477	041164	012705	042666	15\$:	MOV	#T1SKR,R5			;SET UP TO SKIP REVERSE 2 TAPE MARKS
5478	041170	004737	021240		JSR	PC,SCHED			;GO ISSUE THE COMMAND
5479	041174	005737	003524		TST	UDROP			;HAVE ALL UNITS BEEN DROPPED ?
5480	041200	001002			BNE	20\$;NO, CONTINUE
5481	041202	000137	042624		JMP	T1EXIT			;GET OUT IF NONE LEFT TO TEST
5482									
5483	041206	052737	000001	003440	20\$:	BIS	#LEDB,TSTMSK		;SET UP TO ALLOW LEOT DETECTED
5484	041214	012705	042776		MOV	#T1SKD,R5			;SET UP TO DO A SPACE TO LEOT
5485	041220	004737	021240		JSR	PC,SCHED			;GO ISSUE THE COMMAND
5486	041224	042737	000001	003440	BIC	#LEDB,TSTMSK			;DISALLOW LEOT DETECTED
5487	041232	005737	003524		TST	UDROP			;HAVE ALL UNITS BEEN DROPPED ?
5488	041236	001002			BNE	25\$;NO, CONTINUE
5489	041240	000137	042624		JMP	T1EXIT			;GET OUT IF NONE LEFT TO TEST
5490									
5491	041244	012705	042636	25\$:	MOV	#T1REW,R5			;SET UP TO DO A REWIND
5492	041250	004737	021240		JSR	PC,SCHED			;GO ISSUE THE COMMAND
5493	041254	005737	003524		TST	UDROP			;HAVE ALL UNITS BEEN DROPPED ?
5494	041260	001002			BNE	30\$;NO, CONTINUE
5495	041262	000137	042624		JMP	T1EXIT			;GET OUT IF NONE LEFT TO TEST
5496									
5497	041266	012705	043004	30\$:	MOV	#T1WR1,R5			;WRITE 99, 512 BYTE RECORDS
5498	041272	004737	033556		JSR	PC,SDSTUP			;RESET THE RANDOM SEEDS
5499	041276	004737	021240		JSR	PC,SCHED			;GO ISSUE THE COMMAND
5500	041302	005737	003524		TST	UDROP			;HAVE ALL UNITS BEEN DROPPED ?
5501	041306	001002			BNE	35\$;NO, CONTINUE
5502	041310	000137	042624		JMP	T1EXIT			;GET OUT IF NONE LEFT TO TEST
5503									
5504	041314	012705	042652	35\$:	MOV	#T1WTM,R5			;SET UP TO WRITE A TAPE MARK
5505	041320	004737	021240		JSR	PC,SCHED			;GO ISSUE THE COMMAND
5506	041324	005737	003524		TST	UDROP			;HAVE ALL UNITS BEEN DROPPED ?
5507	041330	001002			BNE	40\$;NO, CONTINUE
5508	041332	000137	042624		JMP	T1EXIT			;GET OUT IF NONE LEFT TO TEST
5509									
5510	041336	012705	043012	40\$:	MOV	#T1WR2,R5			;WRITE 84, 525 BYTE RECORDS
5511	041342	004737	033556		JSR	PC,SDSTUP			;RESET THE RANDOM SEEDS
5512	041346	004737	021240		JSR	PC,SCHED			;GO ISSUE THE COMMAND
5513	041352	005737	003524		TST	UDROP			;HAVE ALL UNITS BEEN DROPPED ?

5514	041356	001002		BNE	45\$;NO, CONTINUE
5515	041360	000137	042624	JMP	T1EXIT		;GET OUT IF NONE LEFT TO TEST
5516							
5517	041364	012705	042652	45\$:	MOV	#T1WTM,R5	;SET UP TO WRITE A TAPE MARK
5518	041370	004737	021240		JSR	PC,SCHED	;GO ISSUE THE COMMAND
5519	041374	005737	003524		TST	UDROP	;HAVE ALL UNITS BEEN DROPPED ?
5520	041400	001002		BNE	50\$;NO, CONTINUE
5521	041402	000137	042624	JMP	T1EXIT		;GET OUT IF NONE LEFT TO TEST
5522							
5523	041406	012705	043020	50\$:	MOV	#T1WR3,R5	;WRITE 69, 1038 BYTE RFCORDS
5524	041412	004737	033556		JSR	PC,SDSTUP	;RESET THE RANDOM SEEDS
5525	041416	004737	021240		JSR	PC,SCHED	;GO ISSUE THE COMMAND
5526	041422	005737	003524		TST	UDROP	;HAVE ALL UNITS BEEN DROPPED ?
5527	041426	001002		BNE	55\$;NO, CONTINUE
5528	041430	000137	042624	JMP	T1EXIT		;GET OUT IF NONE LEFT TO TEST
5529							
5530	041434	012705	042652	55\$:	MOV	#T1WTM,R5	;SET UP TO WRITE A TAPE MARK
5531	041440	004737	021240		JSR	PC,SCHED	;GO ISSUE THE COMMAND
5532	041444	005737	003524		TST	UDROP	;HAVE ALL UNITS BEEN DROPPED ?
5533	041450	001002		BNE	60\$;NO, CONTINUE
5534	041452	000137	042624	JMP	T1EXIT		;GET OUT IF NONE LEFT TO TEST
5535							
5536	041456	012705	043026	60\$:	MOV	#T1WR4,R5	;WRITE 54, 1551 BYTE RECORDS
5537	041462	004737	033556		JSR	PC,SDSTUP	;RESET THE RANDOM SEEDS
5538	041466	004737	021240		JSR	PC,SCHED	;GO ISSUE THE COMMAND
5539	041472	005737	003524		TST	UDROP	;HAVE ALL UNITS BEEN DROPPED ?
5540	041476	001002		BNE	65\$;NO, CONTINUE
5541	041500	000137	042624	JMP	T1EXIT		;GET OUT IF NONE LEFT TO TEST
5542							
5543	041504	012705	042652	65\$:	MOV	#T1WTM,R5	;SET UP TO WRITE A TAPE MARK
5544	041510	004737	021240		JSR	PC,SCHED	;GO ISSUE THE COMMAND
5545	041514	005737	003524		TST	UDROP	;HAVE ALL UNITS BEEN DROPPED ?
5546	041520	001002		BNE	70\$;NO, CONTINUE
5547	041522	000137	042624	JMP	T1EXIT		;GET OUT IF NONE LEFT TO TEST
5548							
5549	041526	012705	043034	70\$:	MOV	#T1WR5,R5	;WRITE 39, 2064 BYTE RECORDS
5550	041532	004737	033556		JSR	PC,SDSTUP	;RESET THE RANDOM SEEDS
5551	041536	004737	021240		JSR	PC,SCHED	;GO ISSUE THE COMMAND
5552	041542	005737	003524		TST	UDROP	;HAVE ALL UNITS BEEN DROPPED ?
5553	041546	001002		BNE	75\$;NO, CONTINUE
5554	041550	000137	042624	JMP	T1EXIT		;GET OUT IF NONE LEFT TO TEST
5555							
5556	041554	012705	042652	75\$:	MOV	#T1WTM,R5	;SET UP TO WRITE A TAPE MARK
5557	041560	004737	021240		JSR	PC,SCHED	;GO ISSUE THE COMMAND
5558	041564	005737	003524		TST	UDROP	;HAVE ALL UNITS BEEN DROPPED ?
5559	041570	001002		BNE	80\$;NO, CONTINUE
5560	041572	000137	042624	JMP	T1EXIT		;GET OUT IF NONE LEFT TO TEST
5561							
5562	041576	012705	043042	80\$:	MOV	#T1WR6,R5	;WRITE 24, 2577 BYTE RECORDS
5563	041602	004737	033556		JSR	PC,SDSTUP	;RESET THE RANDOM SEEDS
5564	041606	004737	021240		JSR	PC,SCHED	;GO ISSUE THE COMMAND
5565	041612	005737	003524		TST	UDROP	;HAVE ALL UNITS BEEN DROPPED ?
5566	041616	001002		BNE	85\$;NO, CONTINUE
5567	041620	000137	042624	JMP	T1EXIT		;GET OUT IF NONE LEFT TO TEST
5568							
5569	041624	012705	042644	85\$:	MOV	#T1LEOT,R5	;SET UP TO WRITE LOGICAL END OF TAPE
5570	041630	004737	021240		JSR	PC,SCHED	;GO ISSUE THE COMMAND

5571	041634	005737	003524	TST	UDROP	;HAVE ALL UNITS BEEN DROPPED ?	
5572	041640	001002		BNE	90\$;NO, CONTINUE	
5573	041642	000137	042624	JMP	T1EXIT	;GET OUT IF NONE LEFT TO TEST	
5574							
5575	041646	012705	042636	90\$:	MOV	#T1REW,R5	;SET UP TO REWIND
5576	041652	004737	021240		JSR	PC,SCHED	;GO ISSUE THE COMMAND
5577	041656	005737	003524		TST	UDROP	;HAVE ALL UNITS BEEN DROPPED ?
5578	041662	001002			BNE	95\$;NO, CONTINUE
5579	041664	000137	042624		JMP	T1EXIT	;GET OUT IF NONE LEFT TO TEST
5580							
5581	041670	012705	043050	95\$:	MOV	#T1RD1,R5	;SET UP TO READ 100 RECORDS
5582	041674	004737	033556		JSR	PC,SDSTUP	;RESET THE RANDOM SEEDS
5583	041700	004737	021240		JSR	PC,SCHED	;GO ISSUE THE COMMAND
5584	041704	005737	003524		TST	UDROP	;HAVE ALL UNITS BEEN DROPPED ?
5585	041710	001002			BNE	100\$;NO, CONTINUE
5586	041712	000137	042624		JMP	T1EXIT	;GET OUT IF NONE LEFT TO TEST
5587							
5588	041716	012705	042660	100\$:	MOV	#T1SKP,R5	;SET UP TO SKIP A TAPE MARK
5589	041722	004737	021240		JSR	PC,SCHED	;GO ISSUE THE COMMAND
5590	041726	005737	003524		TST	UDROP	;HAVE ALL UNITS BEEN DROPPED ?
5591	041732	001002			BNE	105\$;NO, CONTINUE
5592	041734	000137	042624		JMP	T1EXIT	;GET OUT IF NONE LEFT TO TEST
5593							
5594	041740	012705	042674	105\$:	MOV	#T1SPC1,R5	;SET UP TO SPACE 84 RECORDS
5595	041744	004737	021240		JSR	PC,SCHED	;GO ISSUE THE COMMAND
5596	041750	005737	003524		TST	UDROP	;HAVE ALL UNITS BEEN DROPPED ?
5597	041754	001002			BNE	110\$;NO, CONTINUE
5598	041756	000137	042624		JMP	T1EXIT	;GET OUT IF NONE LEFT TO TEST
5599							
5600	041762	012705	042660	110\$:	MOV	#T1SKP,R5	;SET UP TO SKIP A TAPE MARK
5601	041766	004737	021240		JSR	PC,SCHED	;GO ISSUE THE COMMAND
5602	041772	005737	003524		TST	UDROP	;HAVE ALL UNITS BEEN DROPPED ?
5603	041776	001002			BNE	115\$;NO, CONTINUE
5604	042000	000137	042624		JMP	T1EXIT	;GET OUT IF NONE LEFT TO TEST
5605							
5606	042004	012705	042702	115\$:	MOV	#T1SPC2,R5	;SET UP TO SPACE 69 RECORDS
5607	042010	004737	021240		JSR	PC,SCHED	;GO ISSUE THE COMMAND
5608	042014	005737	003524		TST	UDROP	;HAVE ALL UNITS BEEN DROPPED ?
5609	042020	001002			BNE	120\$;NO, CONTINUE
5610	042022	000137	042624		JMP	T1EXIT	;GET OUT IF NONE LEFT TO TEST
5611							
5612	042026	012705	042716	120\$:	MOV	#T1SP01,R5	;SET UP TO SPACE 56 OBJECTS
5613	042032	004737	021240		JSR	PC,SCHED	;GO ISSUE THE COMMAND
5614	042036	005737	003524		TST	UDROP	;HAVE ALL UNITS BEEN DROPPED ?
5615	042042	001002			BNE	125\$;NO, CONTINUE
5616	042044	000137	042624		JMP	T1EXIT	;GET OUT IF NONE LEFT TO TEST
5617							
5618	042050	012705	043100	125\$:	MOV	#T1RD5,R5	;SET UP TO READ 39 RECORDS
5619	042054	004737	033556		JSR	PC,SDSTUP	;RESET THE RANDOM SEEDS
5620	042060	004737	021240		JSR	PC,SCHED	;GO ISSUE THE COMMAND
5621	042064	005737	003524		TST	UDROP	;HAVE ALL UNITS BEEN DROPPED ?
5622	042070	001002			BNE	130\$;NO, CONTINUE
5623	042072	000137	042624		JMP	T1EXIT	;GET OUT IF NONE LEFT TO TEST
5624							
5625	042076	012705	042724	130\$:	MOV	#T1SKR1,R5	;SET UP TO SKIP REVERSE 4 TAPE MARKS
5626	042102	004737	021240		JSR	PC,SCHED	;GO ISSUE THE COMMAND
5627	042106	005737	003524		TST	UDROP	;HAVE ALL UNITS BEEN DROPPED ?

5628	042112	001002		BNE	135\$;NO, CONTINUE
5629	042114	000137	042624	JMP	T1EXIT		;GET OUT IF NONE LEFT TO TEST
5630							
5631	042120	012705	042660	135\$:	MOV	#T1SKP,R5	;SET UP TO SKIP TAPE MARK FORWARD
5632	042124	004737	021240		JSR	PC,SCHED	;GO ISSUE THE COMMAND
5633	042130	005737	003524		TST	UDROP	;HAVE ALL UNITS BEEN DROPPED ?
5634	042134	001002		BNE	140\$;NO, CONTINUE
5635	042136	000137	042624	JMP	T1EXIT		;GET OUT IF NONE LEFT TO TEST
5636							
5637	042142	012705	043056	140\$:	MOV	#T1RD2,R5	;SET UP TO READ 84 RECORDS
5638	042146	004737	033556		JSR	PC,SDSTUP	;RESET THE RANDOM SEEDS
5639	042152	004737	021240		JSR	PC,SCHED	;GO ISSUE THE COMMAND
5640	042156	005737	003524		TST	UDROP	;HAVE ALL UNITS BEEN DROPPED ?
5641	042162	001002		BNE	145\$;NO, CONTINUE
5642	042164	000137	042624	JMP	T1EXIT		;GET OUT IF NONE LEFT TO TEST
5643							
5644	042170	012705	042732	145\$:	MOV	#T1SP02,R5	;SET UP TO SPACE 71 OBJECTS
5645	042174	004737	021240		JSR	PC,SCHED	;GO ISSUE THE COMMAND
5646	042200	005737	003524		TST	UDROP	;HAVE ALL UNITS BEEN DROPPED ?
5647	042204	001002		BNE	150\$;NO, CONTINUE
5648	042206	000137	042624	JMP	T1EXIT		;GET OUT IF NONE LEFT TO TEST
5649							
5650	042212	012705	043072	150\$:	MOV	#T1RD4,R5	;SET UP TO READ 54 RECORDS
5651	042216	004737	033556		JSR	PC,SDSTUP	;RESET THE RANDOM SEEDS
5652	042222	004737	021240		JSR	PC,SCHED	;GO ISSUE THE COMMAND
5653	042226	005737	003524		TST	UDROP	;HAVE ALL UNITS BEEN DROPPED ?
5654	042232	001002		BNE	155\$;NO, CONTINUE
5655	042234	000137	042624	JMP	T1EXIT		;GET OUT IF NONE LEFT TO TEST
5656							
5657	042240	012705	042740	155\$:	MOV	#T1SP03,R5	;SET UP TO SPACE 66 OBJECTS
5658	042244	004737	021240		JSR	PC,SCHED	;GO ISSUE THE COMMAND
5659	042250	005737	003524		TST	UDROP	;HAVE ALL UNITS BEEN DROPPED ?
5660	042254	001002		BNE	160\$;NO, CONTINUE
5661	042256	000137	042624	JMP	T1EXIT		;GET OUT IF NONE LEFT TO TEST
5662							
5663	042262	012705	042746	160\$:	MOV	#T1SPR1,R5	;SET UP TO SPACE REVERSE 375 OBJECTS
5664	042266	004737	021240		JSR	PC,SCHED	;GO ISSUE THE COMMAND
5665	042272	005737	003524		TST	UDROP	;HAVE ALL UNITS BEEN DROPPED ?
5666	042276	001002		BNE	165\$;NO, CONTINUE
5667	042300	000137	042624	JMP	T1EXIT		;GET OUT IF NONE LEFT TO TEST
5668							
5669	042304	012705	042754	165\$:	MOV	#T1SKP1,R5	;SET UP TO SKIP FORWARD 4 TAPE MARKS
5670	042310	004737	021240		JSR	PC,SCHED	;GO ISSUE THE COMMAND
5671	042314	005737	003524		TST	UDROP	;HAVE ALL UNITS BEEN DROPPED ?
5672	042320	001002		BNE	170\$;NO, CONTINUE
5673	042322	000137	042624	JMP	T1EXIT		;GET OUT IF NONE LEFT TO TEST
5674							
5675	042326	012705	042710	170\$:	MOV	#T1SPC3,R5	;SET UP TO SPACE 39 RECORDS
5676	042332	004737	021240		JSR	PC,SCHED	;GO ISSUE THE COMMAND
5677	042336	005737	003524		TST	UDROP	;HAVE ALL UNITS BEEN DROPPED ?
5678	042342	001002		BNE	175\$;NO, CONTINUE
5679	042344	000137	042624	JMP	T1EXIT		;GET OUT IF NONE LEFT TO TEST
5680							
5681	042350	012705	042660	175\$:	MOV	#T1SKP,R5	;SET UP TO SKIP A TAPE MARK
5682	042354	004737	021240		JSR	PC,SCHED	;GO ISSUE THE COMMAND
5683	042360	005737	003524		TST	UDROP	;HAVE ALL UNITS BEEN DROPPED ?
5684	042364	001002		BNE	180\$;NO, CONTINUE

```

5685 042366 000137 042624      JMP      T1EXIT      ;GET OUT IF NONE LEFT TO TEST
5686
5687 042372 012705 043106      180$:   MOV      #T1RD6,R5      ;SET UP TO READ 824 RECORDS
5688 042376 004737 033556      JSR      PC,SDSTUP    ;RESET THE RANDOM SEEDS
5689 042402 004737 021240      JSR      PC,SCHED    ;GO ISSUE THE COMMAND
5690 042406 005737 003524      TST      UDROP      ;HAVE ALL UNITS BEEN DROPPED ?
5691 042412 001002      BNE      185$      ;NO, CONTINUE
5692 042414 000137 042624      JMP      T1EXIT      ;GET OUT IF NONE LEFT TO TEST
5693
5694 042420 012705 042762      185$:   MOV      #T1SKP2,R5    ;SET UP TO SKIP 2 TAPE MARKS
5695 042424 004737 021240      JSR      PC,SCHED    ;GO ISSUE THE COMMAND
5696 042430 005737 003524      TST      UDROP      ;HAVE ALL UNITS BEEN DROPPED ?
5697 042434 001002      BNE      190$      ;NO, CONTINUE
5698 042436 000137 042624      JMP      T1EXIT      ;GET OUT IF NONE LEFT TO TEST
5699
5700 042442 012705 042770      190$:   MOV      #T1SPR2,R5    ;SET UP TO SPACE REVERSE 192 OBJECTS
5701 042446 004737 021240      JSR      PC,SCHED    ;GO ISSUE THE COMMAND
5702 042452 005737 003524      TST      UDROP      ;HAVE ALL UNITS BEEN DROPPED ?
5703 042456 001002      BNE      195$      ;NO, CONTINUE
5704 042460 000137 042624      JMP      T1EXIT      ;GET OUT IF NONE LEFT TO TEST
5705
5706 042464 012705 042660      195$:   MOV      #T1SKP,R5    ;SET UP TO SKIP A TAPE MARK
5707 042470 004737 021240      JSR      PC,SCHED    ;GO ISSUE THE COMMAND
5708 042474 005737 003524      TST      UDROP      ;HAVE ALL UNITS BEEN DROPPED ?
5709 042500 001002      BNE      200$      ;NO, CONTINUE
5710 042502 000137 042624      JMP      T1EXIT      ;GET OUT IF NONE LEFT TO TEST
5711
5712 042506 012705 043064      200$:   MOV      #T1RD3,R5    ;SET UP TO READ 69 RECORDS
5713 042512 004737 033556      JSR      PC,SDSTUP    ;RESET THE RANDOM SEEDS
5714 042516 004737 021240      JSR      PC,SCHED    ;GO ISSUE THE COMMAND
5715 042522 005737 003524      TST      UDROP      ;HAVE ALL UNITS BEEN DROPPED ?
5716 042526 001002      BNE      205$      ;NO, CONTINUE
5717 042530 000137 042624      JMP      T1EXIT      ;GET OUT IF NONE LEFT TO TEST
5718
5719 042534 012705 042636      205$:   MOV      #T1REW,R5    ;SET UP TO REWIND
5720 042540 004737 021240      JSR      PC,SCHED    ;GO ISSUE THE COMMAND
5721 042544 005737 003524      TST      UDROP      ;HAVE ALL UNITS BEEN DROPPED ?
5722 042550 001002      BNE      210$      ;NO, CONTINUE
5723 042552 000137 042624      JMP      T1EXIT      ;GET OUT IF NONE LEFT TO TEST
5724
5725 042556 052737 000001 003440 210$:   BIS      #LEDB,TSTMSK  ;SET UP TO ALLOW LEOT DETECTED
5726 042564 012705 042776      MOV      #T1SKD,R5    ;SET UP TO SKIP TO LEOT
5727 042570 004737 021240      JSR      PC,SCHED    ;GO ISSUE THE COMMAND
5728 042574 042737 000001 003440  BIC      #LEDB,TSTMSK  ;DISALLOW LEOT DETECTED
5729 042602 005737 003524      TST      UDROP      ;HAVE ALL UNITS BEEN DROPPED ?
5730 042606 001002      BNE      215$      ;NO, CONTINUE
5731 042610 000137 042624      JMP      T1EXIT      ;GET OUT IF NONE LEFT TO TEST
5732
5733 042614 012705 042636      215$:   MOV      #T1REW,R5    ;SET UP TO REWIND
5734 042620 004737 021240      JSR      PC,SCHED    ;GO ISSUE THE COMMAND
5735
5736 042624      T1EXIT:  EXIT      TST
          042624 104432      TRAP      C$EXIT
          042626 000266      .WORD     L10015-.

```

5738					
5739	042630	140	T1ONL:	.BYTE ONL	; ONLINE COMMAND
5740	042631	000		.BYTE NULPAT	; NO DATA NEEDED
5741	042632	000000		.WORD 0	; NO ITEM COUNT
5742	042634	000001		.WORD 1	; DO IT ONE TIME
5743					
5744	042636	160	T1REW:	.BYTE REW	; REWIND COMMAND
5745	042637	000		.BYTE NULPAT	; NO DATA NEEDED
5746	042640	000000		.WORD 0	; NO ITEM COUNT
5747	042642	000001		.WORD 1	; DO IT ONE TIME
5748					
5749	042644	100	T1LEOT:	.BYTE WTM	; WRITE TAPE MARK
5750	042645	000		.BYTE NULPAT	; NO DATA NEEDED
5751	042646	000000		.WORD 0	; NO ITEM COUNT
5752	042650	000002		.WORD 2	; DO IT TWICE
5753					
5754	042652	100	T1WTM:	.BYTE WTM	; WRITE TAPE MARK
5755	042653	000		.BYTE NULPAT	; NO DATA NEEDED
5756	042654	000000		.WORD 0	; NO ITEM COUNT
5757	042656	000001		.WORD 1	; DO IT ONE TIME
5758					
5759	042660	060	T1SKP:	.BYTE SKP	; SKIP TAPE MARK
5760	042661	000		.BYTE NULPAT	; NO DATA NEEDED
5761	042662	000001		.WORD 1	; SKIP 1 TAPE MARK
5762	042664	000001		.WORD 1	; DO IT ONE TIME
5763					
5764	042666	061	T1SKR:	.BYTE SKR	; SKIP TAPE MARK REVERSE
5765	042667	000		.BYTE NULPAT	; NO DATA NEEDED
5766	042670	000002		.WORD 2	; SKIP REVERSE 2 TAPE MARKS
5767	042672	000001		.WORD 1	; DO IT ONCE
5768					
5769	042674	050	T1SPC1:	.BYTE SPC	; SPACE RECORDS
5770	042675	000		.BYTE NULPAT	; NO DATA NEEDED
5771	042676	000124		.WORD 84.	; SPACE 84 RECORDS
5772	042700	000001		.WORD 1	; DO IT ONE TIME
5773					
5774	042702	050	T1SPC2:	.BYTE SPC	; SPACE RECORDS
5775	042703	000		.BYTE NULPAT	; NO DATA NEEDED
5776	042704	000105		.WORD 69.	; SPACE 69 RECORDS
5777	042706	000001		.WORD 1	; DO IT ONE TIME
5778					
5779	042710	050	T1SPC3:	.BYTE SPC	; SPACE RECORDS
5780	042711	000		.BYTE NULPAT	; NO DATA NEEDED
5781	042712	000047		.WORD 39.	; SPACE 39 RECORDS
5782	042714	000001		.WORD 1	; DO IT ONE TIME
5783					
5784	042716	070	T1SP01:	.BYTE SPO	; SPACE OBJECTS
5785	042717	000		.BYTE NULPAT	; NO DATA NEEDED
5786	042720	000070		.WORD 50.	; SPACE 56 OBJECTS
5787	042722	000001		.WORD 1	; DO IT ONE TIME
5788					
5789	042724	061	T1SKR1:	.BYTE SKR	; SKIP TAPE MARK REVERSE
5790	042725	000		.BYTE NULPAT	; NO DATA NEEDED
5791	042726	000004		.WORD 4	; 4 TAPE MARKS
5792	042730	000001		.WORD 1	; DO IT ONCE
5793					
5794	042732	070	T1SP02:	.BYTE SPO	; SPACE OBJECTS

5795	042733	000	.BYTE	NULPAT	;NO DATA NEEDED
5796	042734	000107	.WORD	71.	;SPACE 71 OBJECTS
5797	042736	000001	.WORD	1	;DO IT ONE TIME
5798					
5799	042740	070	T1SP03: .BYTE	SPO	;SPACE OBJECTS
5800	042741	000	.BYTE	NULPAT	;NO DATA NEEDED
5801	042742	000102	.WORD	66.	;SPACE 66 OBJECTS
5802	042744	000001	.WORD	1	;DO IT ONE TIME
5803					
5804	042746	071	T1SPR1: .BYTE	SPR	;SPACE OBJECTS REVERSE
5805	042747	000	.BYTE	NULPAT	;NO DATA NEEDED
5806	042750	000567	.WORD	375.	;SPACE 375 OBJECTS
5807	042752	000001	.WORD	1	;DO IT ONE TIME
5808					
5809	042754	060	T1SKP1: .BYTE	SKP	;SKIP TAPE MARKS
5810	042755	000	.BYTE	NULPAT	;NO DATA NEEDED
5811	042756	000004	.WORD	4.	;SKIP 4 TAPE MARKS
5812	042760	000001	.WORD	1	;DO IT ONE TIME
5813					
5814	042762	060	T1SKP2: .BYTE	SKP	;SKIP TAPE MARKS
5815	042763	000	.BYTE	NULPAT	;NO DATA NEEDED
5816	042764	000002	.WORD	2.	;SKIP 2 TAPE MARKS
5817	042766	000001	.WORD	1	;DO IT ONE TIME
5818					
5819	042770	071	T1SPR2: .BYTE	SPR	;SPACE OBJECTS REVERSE
5820	042771	000	.BYTE	NULPAT	;NO DATA NEEDED
5821	042772	000300	.WORD	192.	;SPACE 192 OBJECTS
5822	042774	000001	.WORD	1	;DO IT ONE TIME
5823					
5824	042776	062	T1SKD: .BYTE	SKD	;SKIP TO LEOT
5825	042777	000	.BYTE	NULPAT	;NO DATA NEEDED
5826	043000	000004	.WORD	4	;NO ITEM COUNT
5827	043002	000001	.WORD	1	;DO IT ONE TIME
5828					
5829	043004	020	T1WR1: .BYTE	WR	;WRITE RECORD
5830	043005	001	.BYTE	PAT1	;DATA PATERN 1 (ALL 1'S)
5831	043006	000026	.WORD	22.	;BYTE COUNT OF 512.
5832	043010	000143	.WORD	99.	;DO IT 99 TIMES
5833					
5834	043012	020	T1WR2: .BYTE	WR	;WRITE RECORD
5835	043013	002	.BYTE	PAT2	;DATA PATERN 2 (ALL 0'S)
5836	043014	001015	.WORD	525.	;BYTE COUNT OF 525
5837	043016	000124	.WORD	84.	;DO IT 84 TIMES
5838					
5839	043020	020	T1WR3: .BYTE	WR	;WRITE RECORD
5840	043021	003	.BYTE	PAT3	;DATA PATERN 3 (WORST MFM)
5841	043022	002016	.WORD	1038.	;BYTE COUNT OF 1038
5842	043024	000105	.WORD	69.	;DO IT 69 TIMES
5843					
5844	043026	020	T1WR4: .BYTE	WR	;WRITE RECORD
5845	043027	004	.BYTE	PAT4	;DATA PATERN 4 (ALTERNATE 1'S AND 0'S)
5846	043030	003017	.WORD	1551.	;BYTE COUNT OF 1551
5847	043032	000066	.WORD	54.	;DO IT 54 TIMES
5848					
5849	043034	020	T1WR5: .BYTE	WR	;WRITE RECORD
5850	043035	003	.BYTE	PAT3	;DATA PATERN 3 (WORST MFM)
5851	043036	004020	.WORD	2064.	;BYTE COUNT OF 2064

TEST 1: Basic Function Test

```

5852 043040 000047          .WORD 39.          ;DO IT 39 TIMES
5853
5854 043042 020          T1WR6: .BYTE WR          ;WRITE RECORD
5855 043043 001          .BYTE PAT1         ;DATA PATERN 1 (ALL 1'S)
5856 043044 005021       .WORD 2577.        ;BYTE COUNT OF 2577
5857 043046 000030       .WORD 24.          ;DO IT 24 TIMES
5858
5859 043050 010          T1RD1: .BYTE RD          ;READ RECORD
5860 043051 001          .BYTE PAT1         ;DATA PATERN 1 (ALL 1'S)
5861 043052 000026       .WORD 22.          ;BYTE COUNT OF 512.
5862 043054 000143       .WORD 99.          ;DO IT 99 TIMES
5863
5864 043056 010          T1RD2: .BYTE RD          ;READ RECORD
5865 043057 002          .BYTE PAT2         ;DATA PATERN 2 (ALL 0'S)
5866 043060 001015       .WORD 525.         ;BYTE COUNT OF 525
5867 043062 000124       .WORD 84.          ;DO IT 84 TIMES
5868
5869 043064 010          T1RD3: .BYTE RD          ;READ RECORD
5870 043065 003          .BYTE PAT3         ;DATA PATERN 3 (WORST MFM)
5871 043066 002016       .WORD 1038.        ;BYTE COUNT OF 1038
5872 043070 000105       .WORD 69.          ;DO IT 69 TIMES
5873
5874 043072 010          T1RD4: .BYTE RD          ;READ RECORD
5875 043073 004          .BYTE PAT4         ;DATA PATERN 4 (ALTERNATE 1'S AND 0'S)
5876 043074 003017       .WORD 1551.        ;BYTE COUNT OF 1551
5877 043076 000066       .WORD 54.          ;DO IT 54 TIMES
5878
5879 043100 010          T1RD5: .BYTE RD          ;READ RECORD
5880 043101 003          .BYTE PAT3         ;DATA PATERN 3 (WORST MFM)
5881 043102 004020       .WORD 2064.        ;BYTE COUNT OF 2064
5882 043104 000047       .WORD 39.          ;DO IT 39 TIMES
5883
5884 043106 010          T1RD6: .BYTE RD          ;READ RECORD
5885 043107 001          .BYTE PAT1         ;DATA PATERN 1 (ALL 1'S)
5886 043110 005021       .WORD 2577.        ;BYTE COUNT OF 2577
5887 043112 000030       .WORD 24.          ;DO IT 24 TIMES
5888
5889          .EVEN
5890
5891 043114          ENDTST
          043114          L10015:
          043114 104401       TRAP C$ETST

```

```

5898          .SBTTL TEST 2: Quick Verify Write/Read Test
5899
5900          ;**
5901          ;This test rewinds the tape, then executes the following sequence:
5902          ;
5903          ;     1. Write record set,
5904          ;     2. Reposition over just written record set,
5905          ;     3. Then read the current record set,
5906          ;
5907          ;for 5 iterations or until fatal error is encountered. This test
5908          ;permits retries, fixed record length (2048 bytes), fixed number of
5909          ;records/set (400), and predetermined data patterns. This test will
5910          ;execute in a round-robin manner.
5911          ;--
5912          043116          BGNTST
5913          043116          T2::
5914          043116 105737 002234          TSTB      T3ONLY          ;CHECK THE TEST 3 ONLY FLAG
5915          043122 001414          BEQ        START2          ;BRANCH IF IT IS CLEAR
5916          043124 013746 002114          PRINTF   #BYPASS,L$TEST ;PRINT THE TEST BYPASSED MESSAGE
5917          043124 012746 021001          MOV      L$TEST,-(SP)
5918          043130 012746 000002          MOV      #BYPASS,-(SP)
5919          043134 012746 000000          MOV      #2,-(SP)
5920          043140 010600          MOV      SP,R0
5921          043142 104417          TRAP    C$PNTF
5922          043144 062706 000006          ADD     #6,SP
5923          043150 000137 043534          JMP     T2EXIT          ;GET OUT
5924          043154 005737 003524          START2: TST      UDROP          ;HAVE ALL UNITS BEEN DROPPED ?
5925          043160 001014          BNE     5$              ;GO START THE TEST
5926          043162 013746 002114          PRINTF   #BYPASS,L$TEST ;PRINT THE TEST BYPASSED MESSAGE
5927          043162 012746 021001          MOV      L$TEST,-(SP)
5928          043166 012746 000002          MOV      #BYPASS,(SP)
5929          043172 012746 000000          MOV      #2,-(SP)
5930          043176 010600          MOV      SP,R0
5931          043200 104417          TRAP    C$PNTF
5932          043202 062706 000006          ADD     #6,SP
5933          043206 000137 043534          JMP     T2EXIT          ;GET OUT IF NONE LEFT TO TEST
5934          043212 105737 002212          5$:      TSTB     CLOCK          ;IS THE CLOCK ENABLED
5935          043216 001421          BEQ     G02            ;NO, THEN CAN'T PRINT TIME
5936          043220 005046          PRINTF   #TIME,<B,HOURS>,<B,MINUTE>,<B,SECOND>,
5937          043220 153716 002215          CLR     -(SP)
5938          043222 005046          BISB    SECOND,(SP)
5939          043226 153716 002214          CLR     -(SP)
5940          043230 005046          BISB    MINUTE,(SP)
5941          043234 153716 002213          CLR     -(SP)
5942          043242 012746 020044          BISB    HOURS,(SP)
5943          043246 012746 000004          MOV     #TIME,-(SP)
5944          043252 010600          MOV     #4,-(SP)
5945          043254 104417          MOV     SP,R0
5946          043256 062706 000012          TRAP    C$PNTF
5947          043262 004737 033456          ADD     #12,SP
5948          043266 005037 003520          G02:    JSR     PC,CLREOT ;MAKE SURE EOT STATUS IS CLEAR
5949          043272 005037 003440          CLR     OBJECT        ;CLEAR THE OBJECT COUNTER
5950          043272 005037 003440          CLR     TSTMSK        ;ALLOW NO ERRORS

```

```

5931 043276 012705 043544      MOV      #T2REW,R5      ;SET UP TO DO A REWIND
5932 043302 004737 021240      JSR      PC,SCHED      ;GO ISSUE A REWIND TO ALL DRIVES
5933 043306 005737 003524      TST      UDROP         ;HAVE ALL UNITS BEEN DROPPED ?
5934 043312 001510      BEQ      T2EXIT        ;GET OUT IF NONE LEFT TO TEST
5935
5936
5937 043314 004737 033556      5$:     JSR      PC,SDSTUP   ;RESET THE RANDOM SEEDS
5938 043320 012705 043552      MOV      #T2WRT,R5    ;SET UP TO DO A WRITE ITERATION
5939 043324 004737 021240      JSR      PC,SCHED      ;GO ISSUE WRITES TO ALL DRIVES
5940 043330 005737 003524      TST      UDROP         ;HAVE ALL UNITS BEEN DROPPED ?
5941 043334 001477      BEQ      T2EXIT        ;GET OUT IF NONE LEFT TO TEST
5942
5943 043336 012705 043560      MOV      #T2LEOT,R5   ;SET UP TO DO A WRITE LEOT
5944 043342 004737 021240      JSR      PC,SCHED      ;GO ISSUE WRITES TO ALL DRIVES
5945 043346 005737 003524      TST      UDROP         ;HAVE ALL UNITS BEEN DROPPED ?
5946 043352 001470      BEQ      T2EXIT        ;GET OUT IF NONE LEFT TO TEST
5947
5948 043354 012705 043544      MOV      #T2REW,R5    ;SET UP TO DO A REWIND
5949 043360 004737 021240      JSR      PC,SCHED      ;GO ISSUE A REWIND TO ALL DRIVES
5950 043364 005737 003524      TST      UDROP         ;HAVE ALL UNITS BEEN DROPPED ?
5951 043370 001461      BEQ      T2EXIT        ;GET OUT IF NONE LEFT TO TEST
5952
5953 043372 005737 003520      TST      OBJECT        ;IS THIS THE FIRST TIME THROUGH ?
5954 043376 001412      BEQ      10$           ;YES, DON'T DO THE SPACE FORWARD
5955 043400 012705 043602      MOV      #T2SPO,R5    ;SET UP TO SPACE OBJECTS
5956 043404 013765 003520 000002  MOV      OBJECTS,ITMCNT(R5) ;SET UP # OF OBJECTS TO SPACE FORWARD
5957 043412 004737 021240      JSR      PC,SCHED      ;GO ISSUE A REWIND TO ALL DRIVES
5958 043416 005737 003524      TST      UDROP         ;HAVE ALL UNITS BEEN DROPPED ?
5959 043422 001444      BEQ      T2EXIT        ;GET OUT IF NONE LEFT TO TEST
5960
5961 043424 004737 033556      10$:    JSR      PC,SDSTUP   ;RESET THE RANDOM SEEDS
5962 043430 012705 043566      MOV      #T2RD,R5     ;SET UP TO DO A READITERATION
5963 043434 004737 021240      JSR      PC,SCHED      ;GO ISSUE READS TO ALL DRIVES
5964 043440 005737 003524      TST      UDROP         ;HAVE ALL UNITS BEEN DROPPED ?
5965 043444 001433      BEQ      T2EXIT        ;GET OUT IF NONE LEFT TO TEST
5966 043446 066537 000004 003520  ADD      ITRCNT(R5),OBJECTS ;ADD THE # OF RECORDS TO OBJECTS
5967
5968 043454 052737 000001 003440  BIS      #LEDB,TSTMSK  ;SET UP TO ALLOW LEOT DETECTED
5969 043462 012705 043574      MOV      #T2SKD,R5    ;SET UP TO DO A SKIP TO LEOT
5970 043466 004737 021240      JSR      PC,SCHED      ;GO ISSUE READS TO ALL DRIVES
5971 043472 042737 000001 003440  BIC      #LEDB,TSTMSK  ;DISALLOW LEOT DETECTED
5972 043500 005737 003524      TST      UDROP         ;HAVE ALL UNITS BEEN DROPPED ?
5973 043504 001413      BEQ      T2EXIT        ;GET OUT IF NONE LEFT TO TEST
5974 043506 066537 000004 003520  ADD      ITRCNT(R5),OBJECTS ;ADD THE # OF RECORDS TO OBJECTS
5975
5976 043514 022737 004716 003520  CMP      #T2END,OBJECTS ;HAVE WE DONE 2 TRACKS ?
5977 043522 001274      BNE      5$           ;NO, KEEP GOING
5978 043524 012705 043544      MOV      #T2REW,R5    ;SET UP TO DO A REWIND
5979 043530 004737 021240      JSR      PC,SCHED      ;GO ISSUE A REWIND TO ALL DRIVES
5980 043534 004737 033632      T2EXIT: JSR      PC,SDSAVE  ;RESET THE RANDOM SEEDS
5981 043540      EXIT      TST
          043540 104432      TRAP     C$EXIT
          043542 000046      .WORD   L10016-.
5982
5983
5984 043544      160      T2REW:  .BYTE   REW
5985 043545      000      .BYTE   NULPAT      ;REWIND
    
```

5986	043546	000000		.WORD	0	
5987	043550	000001		.WORD	1	
5988						
5989	043552	020	T2WRT:	.BYTE	WR	;WRITE RECORDS
5990	043553	003		.BYTE	PAT3	
5991	043554	010000		.WORD	4096.	
5992	043556	000372		.WORD	250.	
5993						
5994	043560	100	T2LEOT:	.BYTE	WTM	;WRITE TAPE MARK
5995	043561	000		.BYTE	NULPAT	;NO DATA NEEDED
5996	043562	000000		.WORD	0	;NO ITEM COUNT
5997	043564	000002		.WORD	2	;DO IT TWICE
5998						
5999	043566	010	T2RD:	.BYTE	RD	;READ RECORDS
6000	043567	003		.BYTE	PAT3	
6001	043570	010000		.WORD	4096.	
6002	043572	000372		.WORD	250.	
6003						
6004	043574	062	T2SKD:	.BYTE	SKD	;SKIP TAPE MARK TO LEOT
6005	043575	000		.BYTE	NULPAT	;NO DATA NEEDED
6006	043576	000062		.WORD	50.	;SKIP 50 TAPE MARKS
6007	043600	000001		.WORD	1	;DO IT ONE TIME
6008						
6009	043602	070	T2SPO:	.BYTE	SPO	;SPACE OBJECTS
6010	043603	000		.BYTE	NULPAT	
6011	043604	000001		.WORD	1	
6012	043606	000001		.WORD	1	
6013						
6014				.EVEN		
6015	043610			ENDTST		
	043610		L10016:			
	043610	104401		TRAP	C\$ETST	
6016						



```

6018 .SBTTL TEST 3: Complex Write/Read Test
6019
6020 ;**
6021 ;This test rewinds the tape, and executes the following sequence:
6022 ;
6023 ; 1. Write 1000 records,
6024 ; 2. Write a file mark,
6025 ; 3. Repeat 1 and 2 until EOT is reached,
6026 ; 4. Write 2 file marks (LEOT),
6027 ; 5. Rewind,
6028 ; 6. Read 1000 records,
6029 ; 7. Read 1 record (should see unexpected tape mark)
6030 ; 8. Repeat 6 and 7 until LEOT.
6031 ;
6032 ;# of records (N), and record size will be randomly selected. This
6033 ;sequence will permit hardware retries, if not user disabled. This
6034 ;test will run until EOT, LEOT or fatal error is detected. All data
6035 ;patterns including random data will be used in this test.
6036 ;--
6037 043612 BGNTST
        043612
6038 T3::
6039 043612 005737 003524 START3: TST UDROP ;HAVE ALL UNITS BEEN DROPPED ?
6040 043616 001014 BNE S$ ;GO START THE TEST
6041 043620 PRINTF #BYPASS,L$TEST ;PRINT THE TEST BYPASSED MESSAGE
        043620 013746 002114 MOV L$TEST,-(SP)
        043624 012746 021001 MOV #BYPASS,-(SP)
        043630 012746 000002 MOV #2,(SP)
        043634 010600 MOV SP,R0
        043636 104417 TRAP C$PNTF
        043640 062706 000006 ADD #6,SP
6042 043644 000137 044220 JMP T3EXIT ;GET OUT IF NONE LEFT TO TEST
6043
6044 043650 105737 002212 S$: TSTB CLOCK ;IS THE CLOCK ENABLED
6045 043654 001421 BEQ G03 ;NO, THEN CAN'T PRINT TIME
6046 043656 PRINTF #TIME,<B,HOURS>,<B,MINUTE>,<B,SECOND>,
        043656 005046 CLR -(SP)
        043660 153716 002215 BISB SECOND,(SP)
        043664 005046 CLR -(SP)
        043666 153716 002214 BISB MINUTE,(SP)
        043672 005046 CLR -(SP)
        043674 153716 002213 BISB HOURS,(SP)
        043700 012746 020044 MOV #TIME,-(SP)
        043704 012746 000004 MOV #4,-(SP)
        043710 010600 MOV SP,R0
        043712 104417 TRAP C$PNTF
        043714 062706 000012 ADD #12,SP
6047
6048 043720 004737 033706 G03: JSR PC,PATCLR ;MAKE SURE WE START WITH PATTERN 1
6049 043724 004737 033556 JSR PC,SDSTUP ;RESET THE RANDOM SEEDS
6050 043730 004737 033456 JSR PC,CLREOT ;CLEAR ALL EOT INDICATIONS
6051
6052 043734 005037 003440 CLR TSTMSK ;ALLOW NO ERRORS
6053 043740 005737 003524 TST UDROP ;HAVE ALL UNITS BEEN DROPPED ?
6054 043744 001525 BEQ T3EXIT ;GET OUT IF NONE LEFT TO TEST
6055
6056 043746 012705 044230 MOV #T3REW,R5 ;SET UP TO DO REWIND
    
```

6057	043752	004737	021240			JSR	PC,SCHED		;GO ISSUE TO ALL DRIVES
6058	043756	005737	003524			TST	UDROP		;HAVE ALL UNITS BEEN DROPPED ?
6059	043762	001516				BEQ	T3EXIT		;GET OUT IF NONE LEFT TO TEST
6060									
6061	043764	012705	044236	5\$:		MOV	#T3WRT,R5		;SET UP A WRITE ITERATION
6062	043770	004737	021240			JSR	PC,SCHED		;GO DO IT ON ALL DRIVES
6063	043774	005737	003524			TST	UDROP		;HAVE ALL UNITS BEEN DROPPED ?
6064	044000	001507				BEQ	T3EXIT		;GET OUT IF NONE LEFT TO TEST
6065	044002	023737	003526	003524		CMP	UEOT,UDROP		;ARE ALL UNITS AT EOT ?
6066	044010	001413				BEQ	10\$;YES, WRITE ONE MORE REC AND LEOT
6067									
6068	044012	012705	044244			MOV	#T3WTM,R5		;SET UP TO WRITE A TAPE MARK
6069	044016	004737	021240			JSR	PC,SCHED		;GO DO IT ON ALL DRIVES
6070	044022	005737	003524			TST	UDROP		;HAVE ALL UNITS BEEN DROPPED ?
6071	044026	001474				BEQ	T3EXIT		;GET OUT IF NONE LEFT TO TEST
6072	044030	023737	003526	003524		CMP	UEOT,UDROP		;ARE ALL UNITS AT EOT ?
6073	044036	001352				BNE	5\$;NO, KEEP WRITING
6074									
6075	044040	012737	000004	010560	10\$:	MOV	#4,LOOPS		;SET UP TO DO 4 TAPE MARKS
6076	044046	004737	033456		15\$:	JSR	PC,CLREOT		;CLEAR THE EOT INDICATORS
6077	044052	012705	044244			MOV	#T3WTM,R5		;SET UP TO WRITE A TAPE MARK
6078	044056	004737	021240			JSR	PC,SCHED		;GO DO IT ON ALL DRIVES
6079	044062	005737	003524			TST	UDROP		;HAVE ALL UNITS BEEN DROPPED ?
6080	044066	001454				BEQ	T3EXIT		;GET OUT IF NONE LEFT TO TEST
6081	044070	005337	010560			DEC	LOOPS		;SUBTRACT 1 FROM THE TAPE MARK COUNT
6082	044074	001364				BNE	15\$;KEEP GOING TIL THEY'RE ALL WRITTEN
6083									
6084	044076	004737	033456			JSR	PC,CLREOT		;CLEAR THE EOT INDICATORS
6085	044102	012705	044230			MOV	#T3REW,R5		;SET UP TO REWIND ALL DRIVES
6086	044106	004737	021240			JSR	PC,SCHED		;GO DO IT
6087	044112	005737	003524			TST	UDROP		;HAVE ALL UNITS BEEN DROPPED ?
6088	044116	001440				BEQ	T3EXIT		;GET OUT IF NONE LEFT TO TEST
6089	044120	004737	033706			JSR	PC,PATCLR		;START AT PATTERN 1
6090	044124	004737	033556			JSR	PC,SDSTUP		;RESET THE RANDOM SEEDS
6091									
6092	044130	012705	044252	20\$:		MOV	#T3RD,R5		;SET UP TO READ AN ITERATION SET
6093	044134	004737	021240			JSR	PC,SCHED		;GO ISSUE TO ALL DRIVES
6094	044140	005737	003524			TST	UDROP		;HAVE ALL UNITS BEEN DROPPED ?
6095	044144	001425				BEQ	T3EXIT		;GET OUT IF NONE LEFT TO TEST
6096	044146	023737	003526	003524		CMP	UEOT,UDROP		;ALL UNITS AT EOT ?
6097	044154	001413				BEQ	EX3REW		;YES, GO REWIND ALL DRIVES
6098									
6099	044156	012705	044260			MOV	#T3SPO,R5		;SPACE 1 OBJECT (TAPE MARK)
6100	044162	004737	021240			JSR	PC,SCHED		;GO DO IT AN ALL DRIVES
6101	044166	005737	003524			TST	UDROP		;HAVE ALL UNITS BEEN DROPPED ?
6102	044172	001412				BEQ	T3EXIT		;GET OUT IF NONE LEFT TO TEST
6103	044174	023737	003526	003524		CMP	UEOT,UDROP		;ALL UNITS AT EOT ?
6104	044202	001352				BNE	20\$;NO, KEEP READING
6105									
6106	044204	004737	033456	EX3REW:		JSR	PC,CLREOT		;CLEAR THE EOT INDICATORS
6107	044210	012705	044230			MOV	#T3REW,R5		;SET UP TO REWIND ALL DRIVES
6108	044214	004737	021240			JSR	PC,SCHED		;GO DO IT
6109									
6110	044220	004737	033632	T3EXIT:		JSR	PC,SDSAVE		;RESET THE RANDOM SEEDS
6111	044224					EXIT	TST		
	044224	104432				TRAP	C\$EXIT		
	044226	000040				.WORD	L10017-		

6112

6114	044230	160	T3REW:	.BYTE	REW	:REWIND
6115	044231	000		.BYTE	NULPAT	
6116	044232	000000		.WORD	0	
6117	044234	000001		.WORD	1	
6118						
6119	044236	020	T3WRT:	.BYTE	WR	:WRITE RECORDS
6120	044237	200		.BYTE	ALLPAT	
6121	044240	000000		.WORD	RNDBYT	
6122	044242	000000		.WORD	RNDITR	
6123						
6124	044244	100	T3WTM:	.BYTE	WTM	:WRITE TAPE MARK
6125	044245	000		.BYTE	NULPAT	
6126	044246	000000		.WORD	0	
6127	044250	000001		.WORD	1	
6128						
6129	044252	010	T3RD:	.BYTE	RD	:READ RECORDS
6130	044253	200		.BYTE	ALLPAT	
6131	044254	000000		.WORD	RNDBYT	
6132	044256	000000		.WORD	RNDITR	
6133						
6134	044260	070	T3SPO:	.BYTE	SPO	:SPACE OBJECT (TAPE MARK)
6135	044261	000		.BYTE	NULPAT	
6136	044262	000001		.WORD	1	
6137	044264	000001		.WORD	1	
6138						
6139				.EVEN		
6140	044266			ENDTST		
	044266		L10017:			
	044266	104401		TRAP	C\$ETST	


```

6142          .SBTTL TEST 4: Write Interchange Tape
6143
6144          ;**
6145          ;This test will rewind the tape, then write until EOT or a fatal error is
6146          ;encountered This test will keep track of the number of records and tape
6147          ;marks written. If a fatal error is encountered, a message will report
6148          ;it, and the unit prevented from executing further write operations.
6149          ;--
6150 044270      BGNTST
6151 044270      T4::
6152 044270      TSTB      T3ONLY          ;CHECK THE TEST 3 ONLY FLAG
6153 044270      BEQ        START4         ;BRANCH IF IT IS CLEAR
6154 044270      PRINTF     #BYPASS,L$TEST ;PRINT THE TEST BYPASSED MESSAGE
6155 044270      MOV        L$TEST,-(SP)
6156 044302      MOV        #BYPASS,-(SP)
6157 044306      MOV        #2,-(SP)
6158 044312      MOV        SP,R0
6159 044314      TRAP      C$PNTF
6160 044316      ADD        #6,SP
6161 044322      JMP        T4EXIT         ;GET OUT
6162
6163 044326      START4: TST      UDROP          ;HAVE ALL UNITS BEEN DROPPED ?
6164 044332      BNE        5$              ;GO START THE TEST
6165 044334      PRINTF     #BYPASS,L$TEST ;PRINT THE TEST BYPASSED MESSAGE
6166 044334      MOV        L$TEST,-(SP)
6167 044340      MOV        #BYPASS,-(SP)
6168 044344      MOV        #2,-(SP)
6169 044350      MOV        SP,R0
6170 044352      TRAP      C$PNTF
6171 044354      ADD        #6,SP
6172 044360      JMP        T4EXIT         ;GET OUT IF NONE LEFT TO TEST
6173
6174 044364      5$:   TSTB      CLOCK          ;IS THE CLOCK ENABLED
6175 044370      BEQ        G04             ;NO, THEN CAN'T PRINT TIME
6176 044372      PRINTF     #TIME,<B,HOURS>,<B,MINUTE>,<B,SECOND>,
6177 044372      CLR        -(SP)
6178 044374      BISB      SECOND,(SP)
6179 044400      CLR        -(SP)
6180 044402      BISB      MINUTE,(SP)
6181 044406      CLR        -(SP)
6182 044410      BISB      HOURS,(SP)
6183 044414      MOV        #TIME,-(SP)
6184 044420      MOV        #4,-(SP)
6185 044424      MOV        SP,R0
6186 044426      TRAP      C$PNTF
6187 044430      ADD        #12,SP
6188
6189 044434      G04:  JSR      PC,PATCLR
6190 044440      JSR      PC,SDSTUP        ;SET UP THE RANDOM SEEDS
6191 044444      JSR      PC,CLREOT       ;CLEAR THE EOT FLAG
6192 044450      CLR      TSTMSK         ;NO ALLOWABLE ERRORS
6193
6194 044454      MOV      #T4REW,R5      ;POINT R5 TO THE REWIND TABLE
6195 044460      JSR      PC,SCHED        ;GO START THE TEST
6196 044464      TST      UDROP          ;HAVE ALL UNITS BEEN DROPPED ?
6197 044470      BEQ      T4EXIT         ;GET OUT IF NONE LEFT TO TEST
6198

```



```

6221 .SBTTL TEST 5: Read Unknown Tape
6222
6223 ;**
6224 ;This test will rewind a tape, then read until EOT, LEOT or fatal error
6225 ;is encountered. This test will keep track of the number of records
6226 ;and files read. If a fatal error is encountered, a message will
6227 ;report it, the tape on the unit will be rewound, and the unit
6228 ;prevented from executing further read operations.
6229 ;--
6230 044650          RGNTST
6231 044650 105737 002234
6232 044654 001414
6233 044656          T5::
        044656 013746 002114          TSTB    T3ONLY          ;CHECK THE TEST 3 ONLY FLAG
        044662 012746 021001          BEQ     START5          ;BRANCH IF IT IS CLEAR
        044666 012746 000002          PRINTF #BYPASS,L$TEST ;PRINT THE TEST BYPASSED MESSAGE
        044672 010600          MOV     L$TEST,-(SP)
        044674 104417          MOV     #BYPASS,-(SP)
        044676 062706 000006          MOV     #2,-(SP)
6234 044702 000137 045146          MOV     SP,R0
        044702 000137 045146          TRAP   C$PNTF
        044702 000137 045146          ADD    #6,SP
        044702 000137 045146          JMP    T5EXIT          ;GET OUT
6235
6236 044706 005737 003524          START5: TST    UDROP          ;HAVE ALL UNITS BEEN DROPPED ?
6237 044712 001014          BNE    5$                ;GO START THE TEST
6238 044714          PRINTF #BYPASS,L$TEST ;PRINT THE TEST BYPASSED MESSAGE
        044714 013746 002114          MOV     L$TEST,-(SP)
        044720 012746 021001          MOV     #BYPASS,-(SP)
        044724 012746 000002          MOV     #2,-(SP)
        044730 010600          MOV     SP,R0
        044732 104417          TRAP   C$PNTF
        044734 062706 000006          ADD    #6,SP
6239 044740 000137 045146          JMP    T5EXIT          ;GET OUT IF NONE LEFT TO TEST
6240
6241 044744 105737 002212          5$:   TSTB    CLOCK          ;IS THE CLOCK ENABLED
6242 044750 001421          BEQ    G05                ;NO, THEN CAN'T PRINT TIME
6243 044752          PRINTF #TIME,<B,HOURS>,<B,MINUTE>,<B,SECOND>,
        044752 005046          CLR    -(SP)
        044754 153716 002215          BISB   SECOND,(SP)
        044760 005046          CLR    -(SP)
        044762 153716 002214          BISB   MINUTE,(SP)
        044766 005046          CLR    -(SP)
        044770 153716 002213          BISB   HOURS,(SP)
        044774 012746 020044          MOV    #TIME,-(SP)
        045000 012746 000004          MOV    #4,-(SP)
        045004 010600          MOV    SP,R0
        045006 104417          TRAP   C$PNTF
        045010 062706 000012          ADD    #12,SP
6244
6245 045014 004737 033706          G05:  JSR    PC,PATCLR          ;MAKE SURE WE START WITH PATTERN 1
6246 045020 004737 033556          JSR    PC,SDSTUP          ;SET UP THE RANDOM SEEDS
6247 045024 004737 033456          JSR    PC,CLREOT          ;CLEAR THE EOT FLAG
6248
6249 045030 132737 000001 002236          BITB   #BIT0,DMPBUF          ;ARE WE DUMPING ALL RECORDS?
6250 045036 001403          BEQ    1$                ;NO - BRANCH
6251 045040 012737 000001 045164          MOV    #1,T5RD+4          ;CHANGE ITER COUNT TO 1
6252
6253 045046 005037 003440          1$:   CLR    T5MSK          ;NO ALLOWABLE ERRORS
    
```

```

6254 045052 012705 045152          MOV    #TSREW,R5          ;POINT R5 TO THE REWIND TABLE
6255 045056 004737 021240          JSR    PC,SCHED          ;GO START THE TEST
6256 045062 005737 003524          TST    UDROP             ;HAVE ALL UNITS BEEN DROPPED ?
6257 045066 001427                   BEQ    TSEXIT            ;GET OUT IF NONE LEFT TO TEST
6258
6259 045070 012737 000012 003440      MOV    #TMB!RDTB,TSTMASK ;TAPE MARKS AND TRUNC. RECORDS OK
6260 045076 012705 045160          5$:   MOV    #TSRD,R5        ;POINT R5 TO THE TEST TABLE
6261 045102 004737 021240          JSR    PC,SCHED          ;GO START THE TEST
6262 045106 005737 003524          TST    UDROP             ;HAVE ALL UNITS BEEN DROPPED ?
6263 045112 001415                   BEQ    TSEXIT            ;GET OUT IF NONE LEFT TO TEST
6264 045114 023737 003526 003524      CMP    UEOT,UDROP        ;ARE THEY ALL AT EOT ?
6265 045122 001365                   BNE    5$                ;BRANCH IF THEY ARE NOT
6266
6267 045124 004737 033456          10$:  JSR    PC,CLREOT        ;
6268 045130 012705 045152          MOV    #TSREW,R5        ;POINT R5 TO THE REWIND TABLE
6269 045134 004737 021240          JSR    PC,SCHED          ;GO REWIND ALL UNITS
6270 045140 012737 001751 045164      MOV    #1001.,TSRD+4    ;RESTORE ITER COUNT
6271 045146          TSEXIT: EXIT           ;
        045146 104432      TRAP   C$EXIT          ;
        045150 000016      .WORD L10021- .
6272
6273 045152          160          TSREW: .BYTE  REW          ;REWIND
6274 045153          000          .BYTE  NULPAT
6275 045154          000000     .WORD  0
6276 045156          000001     .WORD  1
6277
6278 045160          010          TSRD:  .BYTE  RD          ;READ RECORDS
6279 045161          200          .BYTE  ALLPAT
6280 045162          010000     .WORD  4096.
6281 045164          001751     .WORD  1001.
6282
6283          .EVEN
6284 045166          ENDTST
        045166          L10021:
        045166 104401      TRAP   C$ETST

```

```

6286
6287
6288
6289
6290
6291
6292
6293
6294
6295
6296
6297
6298
6299
6300
6301
6302
6303 045170
      045170
6304 045170 105737 002234
6305 045174 001414
6306 045176
      045176 013746 002114
      045202 012746 021001
      045206 012746 000002
      045212 010600
      045214 104417
      045216 062706 000006
6307 045222 000137 045576
6308
6309 045226 005737 003524
6310 045232 001014
6311 045234
      045234 013746 002114
      045240 012746 021001
      045244 012746 000002
      045250 010600
      045252 104417
      045254 062706 000006
6312 045260 000137 045576
6313
6314 045264 105737 002212
6315 045270 001421
6316 045272
      045272 005046
      045274 153716 002215
      045300 005046
      045302 153716 002214
      045306 005046
      045310 153716 002213
      045314 012746 020044
      045320 012746 000004
      045324 010600
      045326 104417
      045330 062706 000012
6317
6318 045334 004737 033706

```

```

.SBTTL TEST 6: Start/Stop Write/Read Test
; **
; This test rewinds the tape, and executes the following sequence:
;
; 1. Write 1300 records one at a time,
; 2. Write 2 file marks (LEOT),
; 3. Rewind,
; 4. Read 1300 records one at a time,
; 5. Skip to LEOT.
; 6. Rewind,
;
; This sequence will permit hardware retries, if not user disabled.
; This test will run until exhaustion of the command count or fatal error
; is detected. All data patterns including random data will be used
; in this test.
; -
      BGNTST
T6::
      TSTB    T3ONLY
      BEQ     START6
      PRINTF  #BYPASS,L$TEST
      MOV     L$TEST,-(SP)
      MOV     #BYPASS,-(SP)
      MOV     #2,-(SP)
      MOV     SP,R0
      TRAP   C$PNTF
      ADD     #6,SP
      JMP     T6EXIT
; CHECK THE TEST 3 ONLY FLAG
; BRANCH IF IT IS CLEAR
; PRINT THE TEST BYPASSED MESSAGE
; GET OUT

START6:
      TST     UDROP
      BNE     5$
      PRINTF  #BYPASS,L$TEST
      MOV     L$TEST,-(SP)
      MOV     #BYPASS,(SP)
      MOV     #2,-(SP)
      MOV     SP,R0
      TRAP   C$PNTF
      ADD     #6,SP
      JMP     T6EXIT
; HAVE ALL UNITS BEEN DROPPED ?
; GO START THE TEST
; PRINT THE TEST BYPASSED MESSAGE
; GET OUT IF NONE LEFT TO TEST

5$:
      TSTB    CLOCK
      BEQ     G06
      PRINTF  #TIME,<B,HOURS>,<B,MINUTE>,<B,SECOND>,
      CLR     -(SP)
      BISB    SECOND,(SP)
      CLR     (SP)
      BISB    MINUTE,(SP)
      CLR     -(SP)
      BISB    HOURS,(SP)
      MOV     #TIME,-(SP)
      MOV     #4,-(SP)
      MOV     SP,R0
      TRAP   C$PNTF
      ADD     #12,SP

G06:
      JSR     PC,PATCLR
; MAKE SURE WE START WITH PATTERN 1

```

```

6319 045340 004737 033556      JSR    PC,SDSTUP      ;RESET THE RANDOM SEEDS
6320 045344 004737 033456      JSR    PC,CLREOT     ;CLEAR ALL EOT INDICATIONS
6321
6322 045350 005037 003440      CLR    TSTMSK        ;ALLOW NO ERRORS
6323 045354 005737 003524      TST    UDROP         ;HAVE ALL UNITS BEEN DROPPED ?
6324 045360 001506                BEQ    T6EXIT        ;GET OUT IF NONE LEFT TO TEST
6325
6326 045362 012705 045606      MOV    #T6REW,R5     ;SET UP TO DO REWIND
6327 045366 004737 021240      JSR    PC,SCHED      ;GO ISSUE TO ALL DRIVES
6328 045372 005737 003524      TST    UDROP         ;HAVE ALL UNITS BEEN DROPPED ?
6329 045376 001477                BEQ    T6EXIT        ;GET OUT IF NONE LEFT TO TEST
6330 045400 012737 002424 003404  MOV    #1300.,COUNT ;STEP UP THE WRITE ITERATION COUNT
6331
6332 045406 012705 045614      5$:  MOV    #T6WRT,R5   ;SET UP A WRITE ITERATION
6333 045412 004737 021240      JSR    PC,SCHED      ;GO DO IT ON ALL DRIVES
6334 045416 005737 003524      TST    UDROP         ;HAVE ALL UNITS BEEN DROPPED ?
6335 045422 001465                BEQ    T6EXIT        ;GET OUT IF NONE LEFT TO TEST
6336 045424 005337 003404      DEC    COUNT         ;SUBTRACT 1 FROM THE ITERATION COUNT
6337 045430 001366                BNE    5$           ;MORE COMMANDS TO DO, KEEP GOING
6338
6339 045432 012705 045622      MOV    #T6WTM,R5     ;SET UP TO WRITE A TAPE MARK
6340 045436 004737 021240      JSR    PC,SCHED      ;GO DO IT ON ALL DRIVES
6341 045442 005737 003524      TST    UDROP         ;HAVE ALL UNITS BEEN DROPPED ?
6342 045446 001453                BEQ    T6EXIT        ;GET OUT IF NONE LEFT TO TEST
6343
6344 045450 012705 045606      MOV    #T6REW,R5     ;SET UP TO REWIND ALL DRIVES
6345 045454 004737 021240      JSR    PC,SCHED      ;GO DO IT
6346 045460 005737 003524      TST    UDROP         ;HAVE ALL UNITS BEEN DROPPED ?
6347 045464 001444                BEQ    T6EXIT        ;GET OUT IF NONE LEFT TO TEST
6348
6349 045466 004737 033706      JSR    PC,PATCLR     ;START AT PATTERN 1
6350 045472 004737 033556      JSR    PC,SDSTUP     ;RESET THE RANDOM SEEDS
6351 045476 012737 002424 003404  MOV    #1300.,COUNT ;STEP UP THE WRITE ITERATION COUNT
6352
6353 045504 012705 045630      10$: MOV    #T6RD,R5    ;SET UP TO READ AN ITERATION SET
6354 045510 004737 021240      JSR    PC,SCHED      ;GO DO IT ON ALL DRIVES
6355 045514 005737 003524      TST    UDROP         ;HAVE ALL UNITS BEEN DROPPED ?
6356 045520 001426                BEQ    T6EXIT        ;GET OUT IF NONE LEFT TO TEST
6357 045522 005337 003404      DEC    COUNT         ;SUBTRACT 1 FROM THE ITERATION COUNT
6358 045526 001366                BNE    10$          ;MORE COMMANDS TO DO, KEEP GOING
6359
6360 045530 052737 000001 003440  BIS    #LEDB,TSTMSK  ;SET UP TO ALLOW LEOT DETECTED
6361 045536 012705 045636      MOV    #T6SKD,R5     ;SKIP TO LEOT
6362 045542 004737 021240      JSR    PC,SCHED      ;GO DO IT AN ALL DRIVES
6363 045546 042737 000001 003440  BIC    #LEDB,TSTMSK  ;DISALLOW LEOT DETECTED
6364 045554 005737 003524      TST    UDROP         ;HAVE ALL UNITS BEEN DROPPED ?
6365 045560 001406                BEQ    T6EXIT        ;GET OUT IF NONE LEFT TO TEST
6366
6367 045562 004737 033456      JSR    PC,CLREOT     ;CLEAR THE EOT INDICATORS
6368 045566 012705 045606      MOV    #T6REW,R5     ;SET UP TO REWIND ALL DRIVES
6369 045572 004737 021240      JSR    PC,SCHED      ;GO DO IT
6370
6371 045576 004737 033632      T6EXIT: JSR    PC,SDSAVE ;RESET THE RANDOM SEEDS
6372 045602                EXIT    TST
        045602 104432        TRAP   C$EXIT
        045604 000040        .WORD  L10022-.
6373

```

6375	045606	160	T6REW:	.BYTE	REW		
6376	045607	000		.BYTE	NULPAT		;REWIND
6377	045610	000000		.WORD	0		
6378	045612	000001		.WORD	1.		
6379							
6380	045614	020	T6WRT:	.BYTE	WR		
6381	045615	200		.BYTE	ALLPAT		;WRITE RECORDS
6382	045616	020000		.WORD	8192.		
6383	045620	000001		.WORD	1.		
6384							
6385	045622	100	T6WTM:	.BYTE	WTM		
6386	045623	000		.BYTE	NULPAT		;WRITE TAPE MARK
6387	045624	000000		.WORD	0		
6388	045626	000002		.WORD	2.		
6389							
6390	045630	010	T6RD:	.BYTE	RD		
6391	045631	200		.BYTE	ALLPAT		;READ RECORDS
6392	045632	020000		.WORD	8192.		
6393	045634	000001		.WORD	1.		
6394							
6395	045636	062	T6SKD:	.BYTE	SKD		
6396	045637	000		.BYTE	NULPAT		;SKIP TO LEOT
6397	045640	000001		.WORD	1		
6398	045642	000001		.WORD	1		
6399							
6400				.EVEN			
6401	045644			ENDTST			
	045644		L10022:				
	045644	104401		TRAP	C\$ETST		

```

6403          .SBTTL TEST 7: Conversation Test
6404
6405          ;++
6406          ;Conversation mode will run with or without error reports. The user
6407          ;can select, from a list of commands, a sequence which can be used to
6408          ;emulate a known failure mode. Between commands, the user can specify
6409          ;unique delays, ranging from 10 to 250 ms. The user can follow each
6410          ;tape command with integer values, the first indicating the
6411          ;byte/record/file count and the second indicating the # of repetitions
6412          ;necessary for that command.
6413          ;--
6414          BGNTST
6415          T7::
6416          TSTB      T3ONLY          ;CHECK THE TEST 3 ONLY FLAG
6417          BEQ       START7         ;BRANCH IF IT IS CLEAR
6418          PRINTF   #BYPASS,L$TEST  ;PRINT THE TEST BYPASSED MESSAGE
6419          MOV       L$TEST,-(SP)
6420          MOV       #BYPASS,-(SP)
6421          MOV       #2,-(SP)
6422          MOV       SP,R0
6423          TRAP    C$PNTF
6424          ADD      #6,SP
6425          JMP     T7EXIT          ;GET OUT
6426
6427          START7: TST      UDROP          ;HAVE ALL UNITS BEEN DROPPED ?
6428          BNE      5$              ;GO START THE TEST
6429          PRINTF   #BYPASS,L$TEST  ;PRINT THE TEST BYPASSED MESSAGE
6430          MOV       L$TEST,-(SP)
6431          MOV       #BYPASS,-(SP)
6432          MOV       #2,-(SP)
6433          MOV       SP,R0
6434          TRAP    C$PNTF
6435          ADD      #6,SP
6436          JMP     T7EXIT          ;GET OUT IF NONE LEFT TO TEST
6437
6438          5$:   TSTB      CLOCK          ;IS THE CLOCK ENABLED
6439          BEQ      G07              ;NO, THEN CAN'T PRINT TIME
6440          PRINTF   #TIME,<B,HOURS>,<B,MINUTE>,<B,SECOND>,
6441          CLR      -(SP)
6442          BISB     SECOND,(SP)
6443          CLR      -(SP)
6444          BISB     MINUTE,(SP)
6445          CLR      -(SP)
6446          BISB     HOURS,(SP)
6447          MOV      #TIME,-(SP)
6448          MOV      #4,-(SP)
6449          MOV      SP,R0
6450          TRAP    C$PNTF
6451          ADD      #12,SP
6452
6453          G07:  CLR      BRCNT          ; CLEAR THE BRANCH COUNTER
6454          MOV      #T7TBL,R5         ;POINT R5 TO TEST 7 TABLE
6455          CLR      TSTMSK           ;NO ALLOWABLE ERRORS
6456          BIS      #LEDB,TSTMSK     ;SET UP TO ALLOW LEOT DETECTED
6457
6458          10$:  JSR      PC,SDSTUP     ;SET UP THE RANDOM SEEDS
6459          JSR      PC,PATCLR         ;USE THE SAME PATTERN
    
```



```

6484
6486 .TITLE PARAMETER CODING
6497
6498 .SBTTL HARDWARE PARAMETER CODING SECTION
6526
6527 046234 BGNMOD
6528
6529
6530 ;**
6531 ; THE HARDWARE PARAMETER CODING SECTION CONTAINS MACROS
6532 ; THAT ARE USED BY THE SUPERVISOR TO BUILD P-TABLES. THE
6533 ; MACROS ARE NOT EXECUTED AS MACHINE INSTRUCTIONS BUT ARE
6534 ; INTERPRETED BY THE SUPERVISOR AS DATA STRUCTURES. THE
6535 ; MACROS ALLOW THE SUPERVISOR TO ESTABLISH COMMUNICATIONS
6536 ; WITH THE OPERATOR.
6537 ;--
6538 046234 BGNHRD
        046234 000032 .WORD L10024-L$HARD/2
        046236 L$HARD::
6539
6540 046236 GPRMA TKIPAD,0,0,160002,177564,YES
        046236 C00031 .WORD T$CODE
        046240 046262 .WORD TKIPAD
        046242 160002 .WORD T$LLOLIM
        046244 177564 .WORD T$HILIM
6541 046246 GPRMD TKUNT,2,0,777,0,251,YES
        046246 001032 .WORD T$CODE
        046250 046277 .WORD TKUNT
        046252 000777 .WORD 777
        046254 000000 .WORD T$LLOLIM
        046256 000251 .WORD T$HILIM
6542
6543 046260 EXIT HRD
        046260 021004 .WORD T$CODE
6544
6545
6546 046262 124 113 111 TKIPAD: .ASCIZ ?TKIP ADDRESS?
6547 046277 124 057 115 TKUNT: .ASCIZ ?T/MSCP UNIT NUMBER?
6548 .EVEN
6549 046322 ENDHRD
        046322 L10024: .EVEN

```

```

6551          .SBTTL  SOFTWARE PARAMETER CODING SECTION
6552
6553          ;**
6554          ; THE SOFTWARE PARAMETER CODING SECTION CONTAINS MACROS
6555          ; THAT ARE USED BY THE SUPERVISOR TO BUILD P-TABLES.  THE
6556          ; MACROS ARE NOT EXECUTED AS MACHINE INSTRUCTIONS BUT ARE
6557          ; INTERPRETED BY THE SUPERVISOR AS DATA STRUCTURES.  THE
6558          ; MACROS ALLOW THE SUPERVISOR TO ESTABLISH COMMUNICATIONS
6559          ; WITH THE OPERATOR.
6560          ;--
6561
6562 046322      BGNSFT
           046322 001107      .WORD L10025-L$SOFT/2
           046324      L$SOFT:
6563
6564 046324      GPRML  ECLK,0,1,YES
           046324 000130      .WORD  T$CODE
           046326 047210      .WORD  ECLK
           046330 000001      .WORD  1
6565 046332      XFERF  5$
           046332 013044      .WORD  T$CODE
6566 046334      GPRMD  HOUR,0,D,177400,0,24.,YES
           046334 000052      .WORD  T$CODE
           046336 047241      .WORD  HOUR
           046340 177400      .WORD  177400
           046342 000000      .WORD  T$LOLIM
           046344 000030      .WORD  T$HILIM
6567 046346      GPRMD  MINT,2,D,000377,0,60.,YES
           046346 001052      .WORD  T$CODE
           046350 047323      .WORD  MINT
           046352 000377      .WORD  000377
           046354 000000      .WORD  T$LOLIM
           046356 000074      .WORD  T$HILIM
6568
6569 046360      5$:  GPRML  CTPA,4,400,YES
           046360 002130      .WORD  T$CODE
           046362 047370      .WORD  CTPA
           046364 000400      .WORD  400
6570 046366      XFERF  10$
           046366 012044      .WORD  T$CODE
6571 046370      GPRML  SERR,6,1,YES
           046370 003130      .WORD  T$CODE
           046372 047425      .WORD  SERR
           046374 000001      .WORD  1
6572 046376      GPRML  SERC,6,400,YES
           046376 003130      .WORD  T$CODE
           046400 047471      .WORD  SERC
           046402 000400      .WORD  400
6573 046404      GPRML  PADD,10,1,YES
           046404 004130      .WORD  T$CODE
           046406 047533      .WORD  PADD
           046410 000001      .WORD  1
6574
6575 046412      10$: GPRML  PRPA,10,400,YES
           046412 004130      .WORD  T$CODE
           046414 047562      .WORD  PRPA
           046416 000400      .WORD  400

```

6576	046420		XFERF	15\$
	046420	024044	.WORD	T\$CODE
6577	046422		GPRML	SOER,12,1,YES
	046422	005130	.WORD	T\$CODE
	046424	047615	.WORD	SOER
	046426	000001	.WORD	1
6578	046430		XFERT	11\$
	046430	004024	.WORD	T\$CODE
6579	046432		GPRML	SRER,12,400,YES
	046432	005130	.WORD	T\$CODE
	046434	047660	.WORD	SRER
	046436	000400	.WORD	400
6580	046440		11\$: GPRML	CLMD,14,1,YES
	046440	006130	.WORD	T\$CODE
	046442	047717	.WORD	CLMD
	046444	000001	.WORD	1
6581	046446		GPRML	MDTB,14,400,YES
	046446	006130	.WORD	T\$CODE
	046450	047766	.WORD	MDTB
	046452	000400	.WORD	400
6582	046454		GPRML	NOCL,16,1,YES
	046454	007130	.WORD	T\$CODE
	046456	050026	.WORD	NOCL
	046460	000001	.WORD	1
6583	046462		GPRML	PDMP,16,400,YES
	046462	007130	.WORD	T\$CODE
	046464	050071	.WORD	PDMP
	046466	000400	.WORD	400
6584				
6585	046470		15\$: GPRML	TSPA,20,1,YES
	046470	010130	.WORD	T\$CODE
	046472	050134	.WORD	TSPA
	046474	000001	.WORD	1
6586	046476		XFERF	20\$
	046476	023044	.WORD	T\$CODE
6587	046500		GPRMD	PATE,20,0,177400,0,7,YES
	046500	010032	.WORD	T\$CODE
	046502	050163	.WORD	PATE
	046504	177400	.WORD	177400
	046506	000000	.WORD	T\$LOLIM
	046510	000007	.WORD	T\$HILIM
6588	046512		GPRML	T3ON,22,1,YES
	046512	011130	.WORD	T\$CODE
	046514	050201	.WORD	T3ON
	046516	000001	.WORD	1
6589	046520		GPRML	TSCP,22,400,YES
	046520	011130	.WORD	T\$CODE
	046522	050222	.WORD	TSCP
	046524	000400	.WORD	400
6590	046526		GPRML	BDMP,24,1,YES
	046526	012130	.WORD	T\$CODE
	046530	050262	.WORD	BDMP
	046532	000001	.WORD	1
6591	046534		GPRML	CHGF,24,400,YES
	046534	012130	.WORD	T\$CODE
	046536	050326	.WORD	CHGF
	046540	000400	.WORD	400

6592	046542		XFERT	25\$
	046542	002024	.WORD	T\$CODE
6593	046544		20\$: XFER	SFTEX1
	046544	076004	.WORD	T\$CODE
6594				
6595	046546		25\$: GPRMD	CMD1,26,0,000377,0,377,YES
	046546	013032	.WORD	T\$CODE
	046550	050470	.WORD	CMD1
	046552	000377	.WORD	000377
	046554	000000	.WORD	T\$LLOLIM
	046556	000377	.WORD	T\$HILIM
6596	046560		GPRMD	DPAT,26,0,177400,0,7,YES
	046560	013032	.WORD	T\$CODE
	046562	050362	.WORD	DPAT
	046564	177400	.WORD	177400
	046566	000000	.WORD	T\$LLOLIM
	046570	000007	.WORD	T\$HILIM
6597	046572		GPRMD	ICNT,30,D,177777,0,MAXBUF,YES
	046572	014052	.WORD	T\$CODE
	046574	050401	.WORD	ICNT
	046576	177777	.WORD	177777
	046600	000000	.WORD	T\$LLOLIM
	046602	020000	.WORD	T\$HILIM
6598	046604		GPRMD	ITER,32,D,177777,0,65000,YES
	046604	015052	.WORD	T\$CODE
	046606	050443	.WORD	ITER
	046610	177777	.WORD	177777
	046612	000000	.WORD	T\$LLOLIM
	046614	065000	.WORD	T\$HILIM
6599				
6600	046616		GPRMD	CMD2,34,0,000377,0,377,YES
	046616	016032	.WORD	T\$CODE
	046620	050476	.WORD	CMD2
	046622	000377	.WORD	000377
	046624	000000	.WORD	T\$LLOLIM
	046626	000377	.WORD	T\$HILIM
6601	046630		GPRMD	DPAT,34,0,177400,0,7,YES
	046630	016032	.WORD	T\$CODE
	046632	050362	.WORD	DPAT
	046634	177400	.WORD	177400
	046636	000000	.WORD	T\$LLOLIM
	046640	000007	.WORD	T\$HILIM
6602	046642		GPRMD	ICNT,36,D,177777,0,MAXBUF,YES
	046642	017052	.WORD	T\$CODE
	046644	050401	.WORD	ICNT
	046646	177777	.WORD	177777
	046650	000000	.WORD	T\$LLOLIM
	046652	020000	.WORD	T\$HILIM
6603	046654		GPRMD	ITER,40,D,177777,0,65000,YES
	046654	020052	.WORD	T\$CODE
	046656	050443	.WORD	ITER
	046660	177777	.WORD	177777
	046662	000000	.WORD	T\$LLOLIM
	046664	065000	.WORD	T\$HILIM
6604				
6605	046666		GPRMD	CMD3,42,0,000377,0,377,YES
	046666	021032	.WORD	T\$CODE

046670	050504	.WORD	CMD3
046672	000377	.WORD	000377
046674	000000	.WORD	T\$LOLIM
046676	000377	.WORD	T\$HILIM
6606	046700	GPRMD	DPAT,42,0,177400,0,7,YES
	046700	.WORD	T\$CODE
	046702	.WORD	DPAT
	046704	.WORD	177400
	046706	.WORD	T\$LOLIM
	046710	.WORD	T\$HILIM
6607	046712	GPRMD	ICNT,44,D,177777,0,MAXBUF,YES
	046712	.WORD	T\$CODE
	046714	.WORD	ICNT
	046716	.WORD	177777
	046720	.WORD	T\$LOLIM
	046722	.WORD	T\$HILIM
6608	046724	GPRMD	ITER,46,D,177777,0,65000,YES
	046724	.WORD	T\$CODE
	046726	.WORD	ITER
	046730	.WORD	177777
	046732	.WORD	T\$LOLIM
	046734	.WORD	T\$HILIM
6609	046736	XFER	CONT1
	046736	.WORD	T\$CODE
6610			
6611	046740	SFTEX1: XFER	SFTEX2
	046740	.WORD	T\$CODE
6612			
6613	046742	CONT1: GPRMD	CMD4,50,0,000377,0,377,YES
	046742	.WORD	T\$CODE
	046744	.WORD	CMD4
	046746	.WORD	000377
	046750	.WORD	T\$LOLIM
	046752	.WORD	T\$HILIM
6614	046754	GPRMD	DPAT,50,0,177400,0,7,YES
	046754	.WORD	T\$CODE
	046756	.WORD	DPAT
	046760	.WORD	177400
	046762	.WORD	T\$LOLIM
	046764	.WORD	T\$HILIM
6615	046766	GPRMD	ICNT,52,D,177777,0,MAXBUF,YES
	046766	.WORD	T\$CODE
	046770	.WORD	ICNT
	046772	.WORD	177777
	046774	.WORD	T\$LOLIM
	046776	.WORD	T\$HILIM
6616	047000	GPRMD	ITER,54,D,177777,0,65000,YES
	047000	.WORD	T\$CODE
	047002	.WORD	ITER
	047004	.WORD	177777
	047006	.WORD	T\$LOLIM
	047010	.WORD	T\$HILIM
6617			
6618	047012	GPRMD	CMD5,56,0,000377,0,377,YES
	047012	.WORD	T\$CODE
	047014	.WORD	CMD5
	047016	.WORD	000377

	047020	000000		.WORD	T\$LOLIM
	047022	000377		.WORD	T\$HILIM
6619	047024			GPRMD	DPAT,56,0,177400,0,7,YES
	047024	027032		.WORD	T\$CODE
	047026	050362		.WORD	DPAT
	047030	177400		.WORD	177400
	047032	000000		.WORD	T\$LOLIM
	047034	000007		.WORD	T\$HILIM
6620	047036			GPRMD	ICNT,60,D,177777,0,MAXBUF,YES
	047036	030052		.WORD	T\$CODE
	047040	050401		.WORD	ICNT
	047042	177777		.WORD	177777
	047044	000000		.WORD	T\$LOLIM
	047046	020000		.WORD	T\$HILIM
6621	047050			GPRMD	ITER,62,D,177777,0,65000,YES
	047050	031052		.WORD	T\$CODE
	047052	050443		.WORD	ITER
	047054	177777		.WORD	177777
	047056	000000		.WORD	T\$LOLIM
	047060	065000		.WORD	T\$HILIM
6622					
6623	047062			GPRMD	CMD6,64,0,000377,0,377,YES
	047062	032032		.WORD	T\$CODE
	047064	050526		.WORD	CMD6
	047066	000377		.WORD	000377
	047070	000000		.WORD	T\$LOLIM
	047072	000377		.WORD	T\$HILIM
6624	047074			GPRMD	DPAT,64,0,177400,0,7,YES
	047074	032032		.WORD	T\$CODE
	047076	050362		.WORD	DPAT
	047100	177400		.WORD	177400
	047102	000000		.WORD	T\$LOLIM
	047104	000007		.WORD	T\$HILIM
6625	047106			GPRMD	ICNT,66,D,177777,0,MAXBUF,YES
	047106	033052		.WORD	T\$CODE
	047110	050401		.WORD	ICNT
	047112	177777		.WORD	177777
	047114	000000		.WORD	T\$LOLIM
	047116	020000		.WORD	T\$HILIM
6626	047120			GPRMD	ITER,70,D,177777,0,65000,YES
	047120	034052		.WORD	T\$CODE
	047122	050443		.WORD	ITER
	047124	177777		.WORD	177777
	047126	000000		.WORD	T\$LOLIM
	047130	065000		.WORD	T\$HILIM
6627	047132			XFER	CONT2
	047132	002004		.WORD	T\$CODE
6628					
6629	047134		SFTEX2:	XFER	SFTEX3
	047134	025004		.WORD	T\$CODE
6630					
6631	047136		CONT2:	GPRMD	CMD7,72,0,000377,0,377,YES
	047136	035032		.WORD	T\$CODE
	047140	050534		.WORD	CMD7
	047142	000377		.WORD	000377
	047144	000000		.WORD	T\$LOLIM
	047146	000377		.WORD	T\$HILIM

6632	047150				GPRMD	DPAT,72,0,177400,0,7,YES
	047150	035032			.WORD	T\$CODE
	047152	050362			.WORD	DPAT
	047154	177400			.WORD	177400
	047156	000000			.WORD	T\$LOLIM
	047160	000007			.WORD	T\$HILIM
6633	047162				GPRMD	ICNT,74,D,177777,0,MAXBUF,YES
	047162	036052			.WORD	T\$CODE
	047164	050401			.WORD	ICNT
	047166	177777			.WORD	177777
	047170	000000			.WORD	T\$LOLIM
	047172	020000			.WORD	T\$HILIM
6634	047174				GPRMD	ITER,76,D,177777,0,65000,YES
	047174	037052			.WORD	T\$CODE
	047176	050443			.WORD	ITER
	047200	177777			.WORD	177777
	047202	000000			.WORD	T\$LOLIM
	047204	065000			.WORD	T\$HILIM
6635						
6636	047206				SFTEX3: XFER	SFTEX4
	047206	070004			.WORD	T\$CODE
6637						
6638	047210	105	116	101	ECLK: .ASCIZ	/ENABLE TIME OF DAY CLOCK/
6639	047241	040	111	116	HOUR: .ASCIZ	/ INPUT HOUR IN 24 HOUR FORMAT (OMIT LEADING ZERO)/
6640	047323	040	111	116	MINT: .ASCIZ	/ INPUT MINUTES (OMIT LEADING ZERO)/
6641					.EVEN	
6642						
6643	047366				SFTEX4: XFER	SFTEX5
	047366	075004			.WORD	T\$CODE
6644						
6645	047370	103	110	101	CTPA: .ASCIZ	/CHANGE CONTROLLER PARAMETERS/
6646	047425	040	105	116	SERR: .ASCIZ	/ ENABLE CONTROLLER ERROR CORRECTION/
6647	047471	040	105	116	SERC: .ASCIZ	/ ENABLE CONTROLLER ERROR RECOVERY/
6648	047533	040	105	116	PADD: .ASCIZ	/ ENABLE PAD BLOCKING/
6649					.EVEN	
6650						
6651	047560				SFTEX5: XFER	SFTEX6
	047560	102004			.WORD	T\$CODE
6652						
6653	047562	103	110	101	PRPA: .ASCIZ	/CHANGE PRINTING PARAMETERS/
6654	047615	040	105	116	SOER: .ASCIZ	/ ENABLE SOFT ERROR REPORT PRINTING/
6655	047660	040	040	105	SRER: .ASCIZ	/ ENABLE READ SOFT ERRORS ONLY/
6656	047717	040	105	116	CLMD: .ASCIZ	/ ENABLE CLEAR MEDIA TABLE EVERY PASS/
6657					.EVEN	
6658						
6659	047764				SFTEX6: XFER	SFTEX7
	047764	063004			.WORD	T\$CODE
6660						
6661	047766	040	105	116	MDTB: .ASCIZ	/ ENABLE PRINTING OF MEDIA TABLE/
6662	050026	040	105	116	NOCL: .ASCIZ	/ ENABLE CLEAR STATS ON FATAL ERROR/
6663	050071	040	105	116	PDMP: .ASCIZ	/ ENABLE VARIABLES DUMP ON ERROR/
6664					.EVEN	
6665						
6666	050132				SFTEX7: XFER	SFTEX8
	050132	113004			.WORD	T\$CODE
6667						
6668	050134	103	110	101	TSPA: .ASCIZ	/CHANGE TEST PARAMETERS/


```

6669 050163      040      104      101  PATE:  .ASCIZ  / DATA PATTERN/
6670 050201      040      122      125  T3ON:  .ASCIZ  / RUN TEST 3 ONLY/
6671 050222      040      105      116  T5CP:  .ASCIZ  / ENABLE DATA COMPARES IN TEST 5/
6672 050262      040      105      116  BDMP:  .ASCIZ  / ENABLE PRINT READ BUFFER IN TEST 5/
6673 050326      040      103      110  CHGF:  .ASCIZ  / CHANGE COMMAND SEQUENCE/
6674
6675
6676 050360      043004      SFTEX8: XFER      SFTEX9
        050360      .WORD      T$CODE
6677
6678 050362      040      040      104  DPAT:  .ASCIZ  / DATA PATTERN/
6679 050401      040      040      111  ICNT:  .ASCIZ  / ITEM COUNT (BYTE,RECORD,OBJECT)/
6680 050443      040      040      111  ITER:  .ASCIZ  / ITERATION COUNT/
6681
6682
6683 050466      026004      SFTEX9: XFER      SFTEXT
        050466      .WORD      T$CODE
6684
6685 050470      103      115      104  CMD1:  .ASCIZ  "CMD/1"
6686 050476      103      115      104  CMD2:  .ASCIZ  "CMD/2"
6687 050504      103      115      104  CMD3:  .ASCIZ  "CMD/3"
6688 050512      103      115      104  CMD4:  .ASCIZ  "CMD/4"
6689 050520      103      115      104  CMD5:  .ASCIZ  "CMD/5"
6690 050526      103      115      104  CMD6:  .ASCIZ  "CMD/6"
6691 050534      103      115      104  CMD7:  .ASCIZ  "CMD/7"
6692
6693
6694 050542      SFTEXT:
6695 050542      ENDSFT
        .EVEN
        050542      L10025:
6696
6703
6704 050542      ENDMOD
6705
6706 050542      .BLKW      4
6707 050552      RDBUF::    .BLKW      10000 ;READ BUFFER
6708 070552      WRTBUF::   .BLKW      10000 ;WRITE BUFFER
6709
6710 110552      MEDIAS::   .BLKW      1296. ;MEDIA STATISTICS TABLE
6711
6712 115612      PATCH::    .BLKW      50 ;PATCH SPACE
6713
6714
6715 115732      LASTAD
        .EVEN
        115732 115746 .WORD T$FREE
        115734 000004 .WORD T$SIZE
        115736 L$LAST::
6716
6717 115736      BGNSETUP      1 ;NUMBER OF P-TABLES
6718 115736      BGNPTAB
        115736 000000 .WORD 0
        115740 000002 .WORD L10030-./2-1
        115742 L10026:
6719 115742      174500 .WORD 174500 ;IP ADDRESS
6720 115744      000000 .WORD 0 ;UNIT NUMBER
    
```

```
6721 115746                    ENDPTAB  
      115746                    L10030:  
6722 115746                    ENDSETUP  
6723                    000001    .END
```

PARAMETER CODING
Symbol table

ABO = 000200 G	BYPASS 021001 G	CMR = 000031 G	C\$GETW= 000027	DRBSTP= 000104 G
ABOER 011525	BYTADD 003446 G	CMSTSV 010550 G	C\$GMAN= 000043	DRERFL= 000010 G
ABOR 025346	BYTCNT 022124 G	CNTEL 013145	C\$GPHR= 000042	DRINUS 003350 G
ABORT = 000004 G	BYTES 003416 G	CNTER 011674	C\$GPRI= 000040	DROP = 000010 G
ABOT 010662 G	CDRECV 023710 G	CNTERL 011172	C\$INIT= 000011	DROPIT= 000400 G
ACC = 000040 G	CDRENO 010312 G	CNTFLG 010562 G	C\$INLP= 000020	DROPPD= 000104 G
ACCESS 024522	CDREN1 010376 G	CNTHI 010554 G	C\$MANI= 000050	DROPUN 040442 G
ACR = 000041 G	CDREN2 010462 G	CNTLEL 013172	C\$MAP = 000102	DRSPB0 004032 G
ADJUST 027142	CDREN3 010546 G	CNTLER 011132	C\$MEM = 000031	DRSPB1 005072 G
ADR = 000020 G	CDSRGO 010272 G	CNTT 010762 G	C\$MMU = 000103	DRSPB2 006132 G
ALLPAT= 000200 G	CDSRG1 010356 G	CNUSAV= 000014 G	C\$MSG = 000023	DRSPB3 007172 G
ASSEMB= 000010	CDSRG2 010442 G	COLSAV= 000016 G	C\$OPNR= 000034	DRVEL 013125
AVALAB 025026	CDSRG3 010526 G	COMEXI 025700	C\$OPNW= 000104	DRVER 011715
AVB = 000001 G	CF.ATN= 000200 G	CONID = 177777 G	C\$PNTB= 000014	DRVERL 011162
AVL = 000130 G	CF.MSC= 000100 G	CONTPA 002217 G	C\$PNTF= 000017	DRVT 010772 G
AVLB = 000040 G	CF.OTH= 000040 G	CONTRV 020532 G	C\$PNTS= 000016	DSPSTP= 000004 G
AVLER 011556	CF.THS= 000020 G	CONT1 046742	C\$PNTX= 000015	DSRNG0 010226 G
AVLT 010702 G	CHGF 050326	CONT2 047136	C\$PUTB= 000072	DSRNG1 010312 G
AVU = 000134 G	CHGFLG 002237 G	CORDMP 033734	C\$PUTW= 000073	DSRNG2 010376 G
BADEL 013233	CHODAT 024502	CORREL 012627	C\$QIO = 000377	DSRNG3 010462 G
BADER 011643	CLMD 047717	CORERL 011232	C\$RDBU= 000007	DSTEL 013103
BADERL 011142	CLOCK 002212 G	COUNT 003404 G	C\$REFG= 000047	DTCPER= 000070 G
BDMP 050262	CLREOT 033456 G	COUNTS 020202 G	C\$REL = 000077	DUMP 020347 G
BIT0 = 000001 G	CLRMED 002226 G	CPRIEX 014350	C\$RESE= 000033	DUMPKT 003344 G
BIT00 = 000001 G	CLSDRV 023444 G	CRD = 177776 G	C\$REVI= 000004	DUMP1 020430 G
BIT01 = 000002 G	CMD = 000000 G	CRDLIM 010571 G	C\$RFLA= 000021	DUMP2 020471 G
BIT02 = 000004 G	CMDASC 011322 G	CTPA 047370	C\$RPT = 000025	ECBEL 013001
BIT03 = 000010 G	CMDBF1 003566 G	C\$AU = 000052	C\$SEFG= 000046	ECBERL 011272
BIT04 = 000020 G	CMDBF2 003636 G	C\$AUTO= 000061	C\$SPRI= 000041	ECCBC = 000036 G
BIT05 = 000040 G	CMDBF3 003706 G	C\$BRK = 000022	C\$SVEC= 000037	ECCDC = 000026 G
BIT06 = 000100 G	CMDBF4 003756 G	C\$BSEG= 000004	C\$TOME= 000076	ECCFLG= 000100 G
BIT07 = 000200 G	CMDBLD 022002 G	C\$BSUB= 000002	DATAEL 031424	ECCTC = 000030 G
BIT08 = 000400 G	CMDCNT 003400 G	C\$CLCK= 000062	DATBL 003472 G	ECDEL 012663
BIT09 = 001000 G	CMDER 011502	C\$CLEA= 000012	DATER 011630	ECDERL 011312
BIT1 = 000002 G	CMDLST= 000001 G	C\$CLOS= 000035	DATPAT= 000001 G	ECLK 047210
BIT10 = 002000 G	CMDONE= 000200 G	C\$CLP1= 000006	DATT 010742 G	ECTEL 012707
BIT11 = 004000 G	CMDSEQ= 000006 G	C\$CPBF= 000074	DAY 020125 G	ECTERL 011302
BIT12 = 010000 G	CMDSSV= 000012 G	C\$CPME= 000075	DAYS 003564 G	EC1COR= 000050 G
BIT13 = 020000 G	CMDT 010652 G	C\$CVEC= 000036	DCBEND 004032 G	EC2COR= 000052 G
BIT14 = 040000 G	CMDTBL 024366	C\$DCLN= 000044	DCBSTP= 000050 G	EDCEXT 031232
BIT15 = 100000 G	CMD1 050470	C\$DODU= 000051	DCB3SP= 000170 G	EDLEXT 032030
BIT2 = 000004 G	CMD2 050476	C\$DRPT= 000024	DCMDBF 003572 G	EF.CON= 000036 G
BIT3 = 000010 G	CMD3 050504	C\$DU = 000053	DCMPER 012507	EF.EOT= 000010 G
BIT4 = 000020 G	CMD4 050512	C\$EDIT= 000000	DCMPT 011112 G	EF.LOG= 000040 G
BIT5 = 000040 G	CMD5 050520	C\$ERDF= 000055	DEVERR 013272 G	EF.NEW= 000035 G
BIT6 = 000100 G	CMD6 050526	C\$ERHR= 000056	DEVEXT 014400	EF.PWR= 000034 G
BIT7 = 000200 G	CMD7 050534	C\$ERRO= 000060	DEVFAT= 000001 G	EF.RES= 000037 G
BIT8 = 000400 G	CMLSER 012213	C\$ERSF= 000054	DFPTBL 002204 G	EF.SEX= 000020 G
BIT9 = 001000 G	CMMSDQ 021406 G	C\$ERSO= 000057	DIAGMC= 000000	EF.STA= 000040 G
BOE = 000400 G	CMP = 000030 G	C\$ESCA= 000010	DMPBUF 002236 G	ELTEXT 032156
BOTER 011751	CMPDAT 032552 G	C\$ESEG= 000005	DMPFLG 002231 G	ENDPAT= 000010 G
BOTT 011012 G	CMPEL 013030	C\$ESUB= 000003	DPAT 050362	EOT = 000004 G
BRCNT 003410 G	CMPER 012464	C\$ETST= 000001	DQCMD 027662 G	EOTBIT= 000002 G
BUFADR 003422 G	CMPERL 011152	C\$EXIT= 000032	DRBENO 005072 G	EOTPR = 000010 G
BUFDMP 034066	CMPIER 003444 G	C\$FREQ= 000101	DRBEN1 006132 G	ERASE 024750
BUFDSC 025616	CMPPRI 014210	C\$FRME= 000100	DRBEN2 007172 G	ERASGP 025006
BUFOFF= 000012 G	CMPT 010732 G	C\$GETB= 000026	DRBEN3 010232 G	ERG = 000120 G

ERI = 000113 G	EXC3A4 027362	G\$EXCP= 000400	IONORM= 000000 G	LUN0 002314 G
ERLGER 014502 G	EXC4A1 027404	G\$HILI= 000002	IOPDRE= 000003 G	LUN1 002506 G
ERL00 017146 G	EXIT 025722	G\$LOLI= 000001	IOSTAT 010546 G	LUN2 002700 G
ERL01 017203 G	EXTCLN 040434	G\$NO = 000000	IOTIME= 000004 G	LUN3 003072 G
ERL02 017301 G	EXTINT 040332	G\$OFFS= 000400	ISR = 000100 G	L\$ACP 002110 G
ERL03 017365 G	EXTSRT 032514	G\$OFFSI= 000376	ISTART 037742	L\$APT 002036 G
ERL04 017454 G	EX3REW 044204	G\$PRMA= 000001	ITER 050443	L\$AU 040706 G
ERL05 017540 G	E\$END = 002100	G\$PRMD= 000002	ITERS 003420 G	L\$AUT 002070 G
ERL06 017627 G	E\$LOAD= 000035	G\$PRML= 000000	ITMCNT= 000002 G	L\$AUTO 040370 G
ERL07 017713 G	FAIL = 000020 G	G\$RADA= 000140	ITMOFF= 000002 G	L\$CCP 002106 G
ERL08 017743 G	FLAG = 040000 G	G\$RADB= 000000	ITRCNT= 000004 G	L\$CLEA 040372 G
ERR = 100000 G	FMTER 011731	G\$RADD= 000040	IVSER 012135	L\$CO 002032 G
ERRBLK 013270 G	FMTT 011002 G	G\$RADL= 000120	IVST1 010712 G	L\$DEPO 002011 G
ERRDEC 030304 G	FM.BAD= 000001 G	G\$RADO= 000020	IVST2 011032 G	L\$DESC 002142 G
ERRDEI 030774 G	FM.CNT= 000000 G	G\$XFER= 000004	IVST3 011102 G	L\$DESP 002076 G
ERRDEL 031236 G	FM.TPE= 000005 G	G\$YES = 000010	IXE = 004000 G	L\$DEVP 002060 G
ERREXT 031014	F\$AU = 000015	HARD = 000002 G	I\$AU = 000041	L\$DISP 002124 G
ERRLOG= 040000 G	F\$AUTO= 000020	HDLEXT 021236	I\$AUTO= 000041	L\$DLY 002116 G
ERRMSG 013266 G	F\$BGN = 000040	HELP = 000000	I\$CLN = 000041	L\$DTP 002040 G
ERRNBR 013264 G	F\$CLEA= 000007	HEXTBL 011442 G	I\$DU = 000041	L\$DTYP 002034 G
ERRTLY 032036 G	F\$DU = 000016	HIADDR= 000002 G	I\$HRD = 000041	L\$DU 040700 G
ERRTYP 013262 G	F\$END = 000041	HIBYTE= 177777 G	I\$INIT= 000041	L\$DUT 002072 G
ERR00 015770 G	F\$HARD= 000004	H:HEX 003554 G	I\$MOD = 000041	L\$DVTY 002174 G
ERR01 016040 G	F\$HW = 000013	HNDLRP 003412 G	I\$MSG = 000041	L\$EF 002052 G
ERR02 016116 G	F\$INIT= 000006	HCE = 100000 G	I\$PROT= 000040	L\$ENVI 002044 G
ERR03 016175 G	F\$JMP = 000050	HOUR 047241	I\$PTAB= 000041	L\$ERRT 013262 G
ERR04 016253 G	F\$MOD = 000000	HOURS 002213 G	I\$PWR = 000041	L\$ETP 002102 G
ERR05 016332 G	F\$MSG = 000011	HRD = 000007 G	I\$RPT = 000041	L\$EXP1 002046 G
ERR06 016411 G	F\$PROT= 000021	HSTER 012165	I\$SEG = 000041	L\$EXP4 002064 G
ERR07 016461 G	F\$PWR = 000017	HSTIMO= 000000 G	I\$SETU= 000041	L\$EXP5 002066 G
ERR08 016521 G	F\$RPT = 000012	HSTT 010752 G	I\$SFT = 000041	L\$HARD 046236 G
ERR09 016556 G	F\$SEG = 000003	HUNGER 012246	I\$SRV = 000041	L\$HIME 002120 G
ERR10 016617 G	F\$SOFT= 000005	HWR = 000006 G	I\$SUB = 000041	L\$HPCP 002016 G
ERR11 016651 G	F\$SRV = 000010	H1READ= 000054 G	I\$TST = 000041	L\$HPTP 002022 G
ERR12 016675 G	F\$SUB = 000002	H1WRIT= 000060 G	J\$JMP = 000167	L\$HW 002204 G
ERR13 016733 G	F\$SW = 000014	H2READ= 000056 G	KWCSR = 177546 G	L\$ICP 002104 G
ERR14 016764 G	F\$TEST= 000001	H2WRIT= 000062 G	KWHDL 021100 G	L\$INIT 002004 G
ERR15 017012 G	GCMDST 025400	IBE = 010000 G	LEDB = 000001 G	L\$LADP 002026 G
ERR16 017062 G	GCS = 000210 G	ICNT 050401	LEDER 012105	L\$LAST 0025736 G
ERR17 017107 G	GCSCFL= 000020 G	IDONE 026502	LEDT 011072 G	L\$LOAD 002100 G
ERS = 000110 G	GCSEXT 027314	IDU = 000040 G	LEOTFL= 000030 G	L\$LUN 002074 G
ERTLY 031142	GCSHDL 026530 G	IER = 020000 G	LF.CON= 000100 G	L\$MREV 002050 G
ETLLY 032120 G	GCSREF 010552 G	ILCMD 025604	LF.SNR= 000001 G	L\$NAME 002000 G
EVENT 003556 G	GCSRFL= 000040 G	ILLCMD= 000007 G	LF.SUC= 000200 G	L\$PRIO 002042 G
EVL = 000004 G	GO = 000001 G	ILOOP 026350	LGPEL 013057	L\$PROT 021036 G
EV.COR= 000150 G	GO1 041060	IMM = 000200 G	LGPERL 011202	L\$PRT 002112 G
EV.CTO= 000052 G	GO2 043262	IMMBIT= 000003 G	LGSTAT 027720 G	L\$REPP 002062 G
EV.DST= 000050 G	GO3 043720	INIT 025342	LINE 020527 G	L\$REV 002010 G
EV.HER= 000213 G	GO4 044434	INITER 012400	LOBYTE= 177776 G	L\$RPT 034210 G
EV.IDS= 000152 G	GO5 045014	INITIT 040336 G	LOE = 040000 G	L\$SOFT 046324 G
EV.LGP= 000010 G	GO6 045334	INT = 000170 G	LOHEX 003552 G	L\$SPC 002056 G
EV.SER= 000153 G	GO7 046012	INTDON= 000001 G	LOOP 026340	L\$SPCP 002020 G
EV.SRI= 000113 G	GUNSTA 025440	INTERR= 000006 G	LOOPS 010560 G	L\$SPTP 002024 G
EV.SRT= 000053 G	GUS = 000220 G	INTTBL 026520	LOT = 000010 G	L\$STA 002030 G
EV.URE= 000350 G	G\$CNTO= 000200	IOERTB 010572 G	LSTENT= 000134 G	L\$SW 002212 G
EXC1A2 027316	G\$DELM= 000372	IOHUNG= 000002 G	LUNFLG= 000026 G	L\$TEST 002114 G
EXC2A3 027340	G\$DISP= 000003	IOICRD= 020000 G	LUNSTP= 000172 G	L\$TIML 002014 G

RNDITR= 000000 G
 RNUSAV= 000020 G
 ROLSAV= 000022 G
 RSPBF0 004026 G
 RSPBF1 005066 G
 RSPBF2 006126 G
 RSPBF3 007166 G
 RSPCNT 003402 G
 RSPHDL 027426 G
 RS1 = 001233 G
 RS2 = 007622 G
 RS3 = 000000 G
 RTYEC1= 000064 G
 RTYEC2= 000066 G
 RTYEL 012574
 RTYERL 011252
 RWI = 000163 G
 R10 003534 G
 R11 003536 G
 R12 003540 G
 R3SAVE 003560 G
 R4SAVE 003562 G
 R8 003530 G
 R9 003532 G
 SAERR 010566 G
 SAVDIF 003436 G
 SCC = 000230 G
 SCD = 000052 G
 SCHED 021240 G
 SCNTCH 025466
 SCR = 000051 G
 SDSAVE 033632 G
 SDSTUP 033556 G
 SECOND 002215 G
 SECRNS 003542 G
 SED1 = 000136 G
 SED2 = 000140 G
 SED3 = 000142 G
 SEED1 = 000144 G
 SEED2 = 000146 G
 SEED3 = 000150 G
 SELDAT 022212 G
 SELREC 022644 G
 SEQER 012347 G
 SERC 047471
 SERCOR 002221 G
 SEREXC= 000002 G
 SERR 047425
 SERREC 002220 G
 SEXB = 000004 G
 SEXER 012063
 SEXT 011062 G
 SFPTBL 002212 G
 SFTEXT 050542
 SFTEX1 046740
 SFTEX2 047134
 SFTEX3 047206

SFTEX4 047366
 SFTEX5 047560
 SFTEX6 047764
 SFTEX7 050132
 SFTEX8 050360
 SFTEX9 050466
 SKD = 000062 G
 SKP = 000060 G
 SKPTMK 024610
 SKR = 000061 G
 SLTUSE= 000010 G
 SOER 047615
 SOERRP 002224 G
 SOFT = 000003 G
 SPC = 000050 G
 SPCASC 011346
 SPCOBJ 024664
 SPCREC 024542
 SPO = 000070 G
 SPR = 000071 G
 SRD = 000005 G
 SRER 047660
 SRTBL 031720
 START 037044
 START1 040752
 START2 043154
 START3 043612
 START4 044326
 START5 044706
 START6 045226
 START7 045704
 STATER 012445
 STATUS= 000004 G
 STAT0 035464
 STAT1 035467
 STAT10 036222
 STAT11 036272
 STAT12 036344
 STAT13 036405
 STAT14 036430
 STAT15 036453
 STAT16 036525
 STAT17 036577
 STAT18 036647
 STAT2 035533
 STAT3 035600
 STAT4 035660
 STAT5 035705
 STAT6 035752
 STAT7 036024
 STAT8 036071
 STAT9 036143
 STFPCK 024164 G
 STINIT 037004 G
 ST.ABO= 000002 G
 ST.AVL= 000004 G
 ST.BOT= 000015 G

ST.CMD= 000001 G
 ST.CMP= 000007 G
 ST.CNT= 000012 G
 ST.DAT= 000010 G
 ST.DIA= 000037 G
 ST.DRV= 000013 G
 ST.FNT= 000014 G
 ST.HST= 000011 G
 ST.LED= 000023 G
 ST.MFE= 000005 G
 ST.MSK= 000037 G
 ST.OFL= 000003 G
 ST.ONL= 000400 G
 ST.POL= 000021 G
 ST.RDT= 000020 G
 ST.SEX= 000022 G
 ST.SUB= 000040 G
 ST.SUC= 000000 G
 ST.TM = 000016 G
 ST.WPR= 000006 G
 SUBCNT 003424 G
 SUBITR 023142 G
 SUBSEC 002216 G
 SUC = 000150 G
 SUCCES= 000001 G
 SUNCHR 025162
 SUPRES 025650
 SUW = 000155 G
 SVCGBL= 000000
 SVCINS= 000000
 SVCSUB= 000000
 SVCTAG= 000000
 SVCTST= 000000
 SWR = 000004 G
 SYSFAT= 000000 G
 S\$LSYM= 010000
 S1 = 004000 G
 S1READ= 000040 G
 S1WRIT= 000044 G
 S2READ= 000042 G
 S2WRIT= 000046 G
 TBLBTM= 000132 G
 TBLEND= 003472 G
 TBLFUL 036716 G
 TBLFF 003544 G
 TBLSRT 032162 G
 TBLTOP= 000130 G
 TCNTFL= 000004 G
 TESTPA 002232 G
 TF.BLK= 000010 G
 TF.GCR= 000004 G
 TF.PE = 000002 G
 TF.800= 000001 G
 TIME 020044 G
 TIMER 010556 G
 TIMERR 012312
 TKERR 026474

TKINIT= 111400 G
 TKIP = 000000 G
 TKIPAD 046262
 TKSA = 000002 G
 TKUNIT= 000004 G
 TKUNT 046277
 TMB = 000010 G
 TMCNT 003550 G
 TMER 011771
 TMT 011022 G
 TRK = 000000 G
 TSPA 050134
 TSTMSK 003440 G
 TSTSTP= 000006 G
 T\$ARGC= 000004
 T\$CODE= 026004
 T\$ERRN= 000000
 T\$EXCP= 000000
 T\$FLAG= 000041
 T\$FREE= 115746
 T\$GMAN= 000000
 T\$HILI= 065000
 T\$LAST= 000001
 T\$LOLI= 000000
 T\$LSYM= 010000
 T\$LTNO= 000007
 T\$NEST= 177777
 T\$NSO = 000000
 T\$NS1 = 000005
 T\$PCNT= 000000
 T\$PTAB= 010027
 T\$PTHV= 000001
 T\$PTNU= 000001
 T\$SAVL= 177777
 T\$SEGL= 177777
 T\$SIZE= 000004
 T\$SUBN= 000000
 T\$TAGL= 177777
 T\$TAGN= 010031
 T\$TEMP= 000000
 T\$TEST= 000007
 T\$TSTM= 177777
 T\$TSTS= 000001
 T\$\$AU = 010014
 T\$\$AUT= 010011
 T\$\$CLE= 010012
 T\$\$DAT= 010030
 T\$\$DU = 010013
 T\$\$HAR= 010024
 T\$\$HW = 010000
 T\$\$INI= 010010
 T\$\$MSG= 010003
 T\$\$PC = 000001
 T\$\$PRO= 010004
 T\$\$PTA= 010027
 T\$\$RPT= 010007
 T\$\$SOF= 010025

T\$\$SRV= 010006
 T\$\$SW = 010001
 T\$\$TES= 010023
 T1 040714 G
 T1EXIT 042624
 T1LEOT 042644
 T1ONL 042630
 T1RD1 043050
 T1RD2 043056
 T1RD3 043064
 T1RD4 043072
 T1RD5 043100
 T1RD6 043106
 T1REW 042636
 T1SKD 042776
 T1SKP 042660
 T1SKP1 042754
 T1SKP2 042762
 T1SKR 042666
 T1SKR1 042724
 T1SPC1 042674
 T1SPC2 042702
 T1SPC3 042710
 T1SP01 042716
 T1SP02 042732
 T1SP03 042740
 T1SPR1 042746
 T1SPR2 042770
 T1WR1 043004
 T1WR2 043012
 T1WR3 043020
 T1WR4 043026
 T1WR5 043034
 T1WR6 043042
 T1WTM 042652
 T2 043116 G
 T2END = 004716 G
 T2EXIT 043534
 T2LEOT 043560
 T2RD 043566
 T2REW 043544
 T2SKD 043574
 T2SPO 043602
 T2WRT 043552
 T3 043612 G
 T3EXIT 044220
 T3ON 050201
 T3ONLY 002234 G
 T3RD 044252
 T3REW 044230
 T3SPO 044260
 T3WRT 044236
 T3WTM 044244
 T4 044270 G
 T4EXIT 044620
 T4REW 044624
 T4WRT 044632

PARAMETER CODING
Symbol table

MACRO Y05.02 Thursday 18-Apr-85 13:37 Page 85-12

SEQ 178

T4WTM	044640	T7CMD2	002246	UF.RMV=	000200 G	URDSRG=	000162 G	WR	=	000020 G
T5	044650 G	T7CMD3	002254	UF.VSS=	000040 G	UREEL	012740	WRBYT1=	000120 G	
T5CMP	002235 G	T7CMD4	002262	UF.VSU=	020000 G	UREERL	011262	WRBYT2=	000122 G	
T5CP	050222	T7CMD5	002270	UF.WPH=	020000 G	URSPBF=	000156 G	WRBYT3=	000124 G	
T5EXIT	045146	T7CMD6	002276	UF.WPS=	010000 G	UWEEEL	012760	WRBYT4=	000126 G	
T5RD	045160	T7CMD7	002304	UNDROP=	000032 G	UWEERL	011242	WRERL	031504	
T5REW	045152	T7END	002312	UNDRUN=	000072 G	U1OFF =	005040 G	WRITE	024462	
T6	045170 G	T7EXIT	046222	UNITRV	020662 G	U2OFF =	002420 G	WRKMSK	003442 G	
T6EXIT	045576	T7TBL	002232 G	UNJAM	033242 G	U3OFF =	001540 G	WRTBUF	070552 G	
T6RD	045630	UAM =	000200 G	UNLBIT=	000004 G	U4OFF =	001210 G	WTAPMK	024722	
T6REW	045606	UCDEND=	000170 G	UNTEOT	020256 G	WDSTEL	011222	WTM =	000100 G	
T6SKD	045636	UCDSRG=	000166 G	UNTLOT	020312 G	WMSPOS=	000100 G	XFERST=	000010 G	
T6WRT	045614	UDRNER	012541	UNTSTP=	000002 G	WPRB =	000020 G	X\$ALWA=	000000	
T6WTM	045622	UDROP	003524 G	UNTTBL	003360 G	WPRBIT=	000005 G	X\$FALS=	000040	
T7	045646 G	UEOT	003526 G	URBEND=	000160 G	WPRER	011603	X\$OFFS=	000400	
T7BRFL=	000001 G	UF.CMR=	000001 G	URDEND=	000164 G	WPRT	010722 G	X\$TRUE=	000020	
T7CMD1	002240	UF.CMW=	000002 G							

. ABS. 115746 000 (RW,I,GBL,ABS,OVR)
000000 001 (RW,I,LCL,REL,CON)

Errors detected: 0

*** Assembler statistics

Work file reads: 331
Work file writes: 328
Size of work file: 35448 Words (139 Pages)
Size of core pool: 19714 Words (75 Pages)
Operating system: RSX-11M/PLUS (Under VAX/VMS)

Elapsed time: 00:10:26.01
CZTKBA.BIN,CZTKBA.LST/-SP=SVC40R.MLB/ML,CZTKBA